An Intelligent Agent for Home Energy Efficiency

Visit Hirankitti, *Member, IAENG*

***Abstract*— Using energy efficiently will help alleviate the climate change, one major purpose of smart home development is to achieve that by employing Home Energy Management System. In this paper we propose an agent framework based on an object-oriented and logical approach for HEMS to achieve home energy efficiency. The framework embraces demand-side management and efficient use of energy in a smart home.**

***Index Terms*—Intelligent Agent, Smart Home, Home Energy Efficiency, Demand-Side Management.**

I. Introduction

The world is moving towards energy efficiency ranging from houses to buildings and to factories. In this paper we propose an agent framework to attain home energy efficiency. It is a framework for constructing a Home Energy Management System (HEMS) for smart homes.

When we refer to HEMS, we mean a software operating a smart home by monitoring and controlling all its electrical devices in such a way as to use the electricity in an efficient way as well as to make use of the most from its own generated electricity if available. To operate a smart home, HEMS requires a hardware environment, see Fig. 1.

This research investigation is conducted under the PEA Smart Home Project funded by Provincial Electricity Authority (PEA), the major utility company in Thailand. In this project we have developed HEMS, a home gateway, smart plugs, and home sensors, whilst all home appliances are commercial off-the-shelf products. To prove our concepts, we build the real house with all these devices installed and operating in order to test our HEMS in the real-world scenarios.

The intelligent agent approach is one of promising techniques which have been applied to develop a smart home and smart building. For example, a research [1] applied defeasible logic for intelligent control of appliances in a building, such as light bulbs and air conditioners, in order to save energy. For the same purpose [2] adopted BDI agent model and used finite state machine to model agents' behaviors. Another direction of smart home research was to develop an agent to assist the elderly and disabled [3] by proposing a logical framework based on Event Calculus to model events and actions. In recent years, we have developed an agent framework for a smart home [4][5], and this paper extends our previous works.

Visit Hirankitti is with the Department of Computer Engineering, Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, Ladkrabang, Bangkok 10520, Thailand (phone: +662-329-8341; e-mail: v_hirankitti@yahoo.com).

The remainder of the paper is organized as follows. Section II describes the nature and scope of our smart home and home energy efficiency. Section III introduces our agent framework. We then elaborate our HEMS agent in Section IV where we apply our agent framework to handle demand-side management and efficient use of the energy in homes. We later discuss related works in section V and conclude this research work in section VI.

II. The Problem description

A. Our Smart Home Hardware Environment

Our smart home is equipped with several hardware components as seen in Fig. 1. Each component and its functionality to operate the smart home is described below:

- Home Gateway

HEMS is the software operating the smart home and it is installed and runs in the home gateway, hence the home gateway is the central computer to monitor and control other hardware components in the smart home.

- Smart Meter

The smart meter monitors characteristics of the electricity supplied to the home, including the home energy consumption. This information is essential for HEMS.

- Smart Plug

To allow an ordinary home appliance without any smart capability to control by HEMS, such a home appliance needs to connect to a smart plug. Our own designed smart plug is a stripped-down version of our smart meter, so it has an ability of advanced metering. The plug can be controlled by HEMS to turn on/off the home appliance via ZigBee.

- In-home Display (a mobile phone/tablet)

A home resident can access to and operate HEMS using our mobile phone application and our web application running on a tablet. Each of these software applications connects to the home gateway via the internet. Once the user gets logged-in to HEMS, firstly the user can add, remove, and configure their home rooms and devices. Secondly the user can monitor the device activities and control them to serve his/her purposes. Thirdly the user can program his/her HEMS scripts, control rules and preferences on HEMS to automate these devices due to some conditions on the current home state and some emerging events.

- Home Sensors

A home sensor monitors an ever-changing state of the home, e.g. temperature, and also detects an internal/external event of the house, e.g. intrusion. The sensor is equipped with a Bluetooth module so that it can communicate

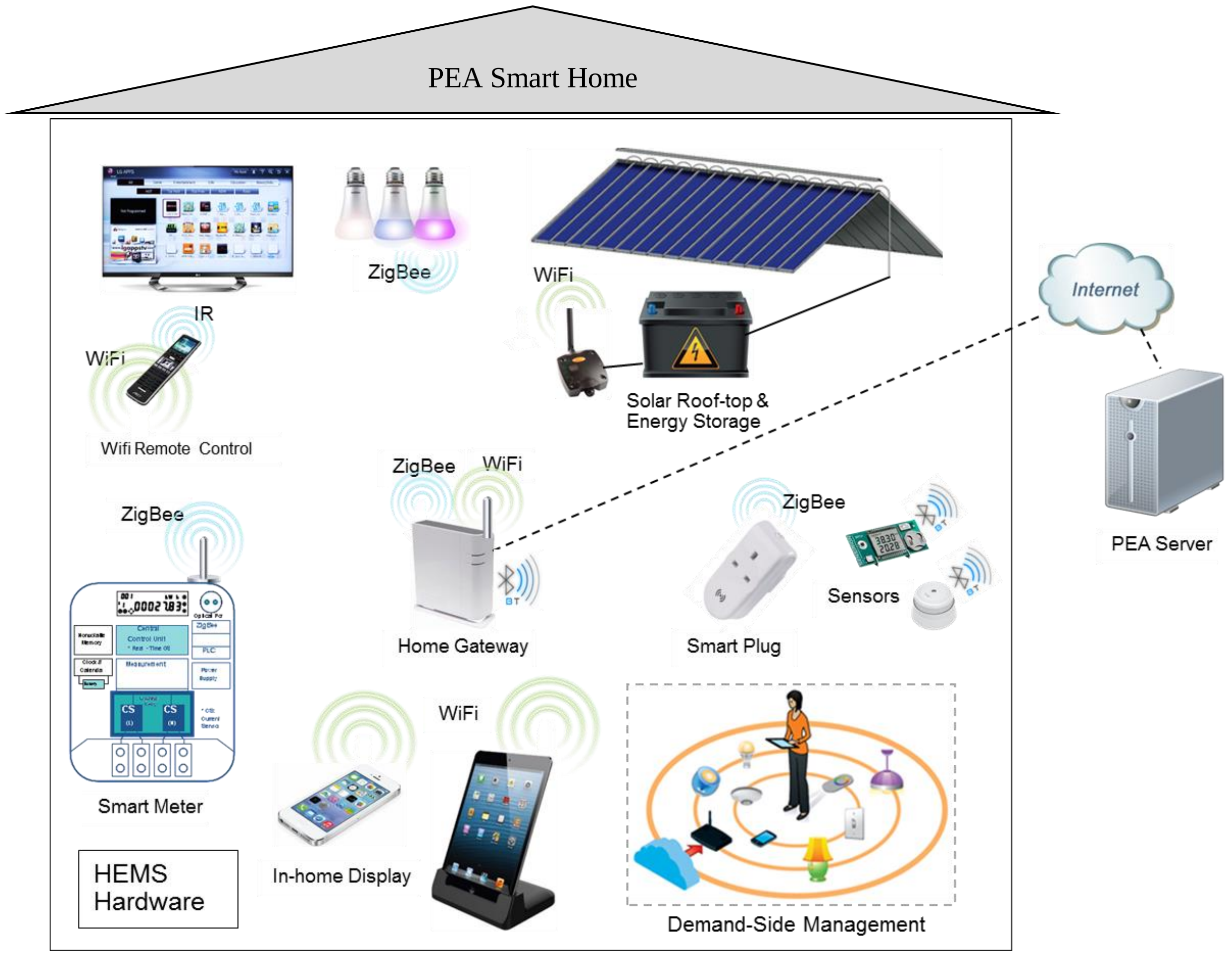

Fig 1 Our smart home hardware environment.

with HEMS wirelessly. Our home sensors are, for example, thermometer sensors to report temperature, and light sensors to detect a lighting condition, infrared sensors and Bluetooth Beacons to detect human movement and position respectively.

- Home Appliances

Home appliances provide functionalities to a home resident. Each device's activities are monitored and controlled by HEMS wirelessly over Wi-Fi or ZigBee. The way to control each home appliance could be as simple as turning it on and off, or it could be more advanced control of many settings depending on how smart that home appliance is. For example, for a smart plug it can be turned on and off, but for a light bulb it allows us to dim the light and change its color as well.

- Solar Roof

The solar roof is the only power generator of our smart home. It works autonomously, when there is sufficient sun light it will keep supplying electricity into a hybrid inverter which then feeds the generated electricity to the house, to charge it to the energy storage, and/or to sell it to the grid.

- Hybrid Inverter

This device takes electricity from the grid, from the solar roof, and/or from the energy storage, then supply that to the home. The hybrid inverter we employ in our smart home can operate in one of the following modes:

- Prefer-to-Load mode. When the inverter is set (by HEMS) to this mode, it will supply the solar power to the house load (to supplement the power from the grid) as the first priority, if solar power is not sufficient, the energy storage will supply the power to the house in addition. In the case when the solar power exceeds the house load, it will charge the energy storage simultaneously until the energy storage is fully charged; and if that is so the exceeding power will be sold to the grid.
- Full-Match-Load mode. In this mode the solar power will never be sold to the grid, it will supply only to the house load first, and to charge the energy storage when the power exceeds the load.
- Full-Sell mode. The inverter will sell all the solar power to the grid only, not to supply any to the house or to the energy storage.
- Backup-UPS mode. The inverter will charge the energy storage with its solar charger controller until it is fully charged and remaining power will supply to the house load together with the power from the grid.

This hybrid inverter provides a Modbus serial interface for our home gateway to communicate with for both monitoring and control purposes. Hence HEMS can monitor the energy storage status, such as percentage of being charged, and control it to switch between the charging and releasing mode via the inverter, and this inverter also provides many options for HEMS to monitor and control it.

- Energy Storage

The electricity generated from the solar roof will be stored in an energy storage, a type of long-lasting rechargeable battery. The decision whether the generated electricity is to be stored in or released from the energy storage is under control of the hybrid inverter as we described earlier.

- The PEA Server

To complete the whole picture of the PEA Smart Home, the PEA server (to be operated by PEA) is necessary. This server will serve PEA many purposes in order to manage the energy efficiency in a large scale.

Firstly, the server will collect load profiles, at the appliance/device level, and power generation profiles from all smart homes. This information is vital for PEA to use for big data analysis of load forecast, assessment of home micro-grid capability, and plan for demand-side tariffs.

Secondly, via PEA server the company can issue a demand response (DR) event to a HEMS of every smart home to response in order to alleviate peak loads of a region.

Thirdly PEA server can broadcast a dynamic tariff and/or real time pricing information to every smart home in real-time. This means PEA can influence the load shapes of their customers with some assistance from HEMS.

Fourthly the collaboration between the PEA server and HEMS will allow demand-side management to be viable, and this will lead to realistic energy efficiency of the demand and supply.

B. Demand Response

Demand response (DR) is a program a utility company offering to its customers to help reduce or shift their electricity usage during peak periods in response to time-based rates or other forms of financial incentives. Working with the PEA server, our smart home can accommodate DR in following ways:

- Time-based or real-time pricing. Our smart meter can calculate billing for any tariff rate chosen by the resident. HEMS will be sensible to the tariff rates broadcasted by the PEA server, i.e. it will apply the tariff dynamically as it is broadcasted by the server.
- Automatic DR response. HEMS will response promptly to the DR events issued by the server. HEMS will apply the DR event-response rules to handle those events. Once the DR alert returns to normality the DR-disable event will be issued by the server, and HEMS will reset the house back to normal.

C. Demand-Side Management

Demand-side management (DSM) concerns all management of energy usage at the demand side, i.e. the customer. In this paper, our *objective* of DSM is the home's efficient use of electricity from the grid and the home's benefit from the electricity generated by the solar roof. Therefore, our DSM objectives are:

- Cost saving from the electricity bill
- Energy saving
- Energy efficiency
- DR enabling
- Satisfaction of some degree of comfort on living

The agent framework for HEMS to elaborate next aims to achieve these objectives.

III. An Agent Framework

A. A Generic Agent

Recently we have developed an agent framework for HEMS which is based on an object-oriented and logical approach [4][5]. The generic agent of our framework is an object-oriented class. In this case State Variables, State Historical Records, Events, Event Historical Records, and Event–Response Rules are its predefined attributes, while the State Updaters, Event Listeners, and Event Handlers are the methods to manipulate on these attributes. For the Operational and Scheduled Tasks, Provided Services, and Communication Protocols, they are also implemented as methods.

A Generic Agent
- State **State Variables** **State Updater** **Historical Record of the State**
- Events **Event listener**s **Historical Record of the Events** **- Events–Response Rules** **Event handlers**
- Operational and Scheduled Tasks
- Provided Services
- Communication Protocols **- Ports**
- Goals
- Policies
- Constraints

Goals are the ultimate mandate for an agent to carry out. In the context of HEMS, Goals are, for example, how to reduce home energy consumption, how to gain the most from the in-home produced energy. While the Goals are the agent's mandate, Policies will shape up them. Policies are abstract conditions to put upon the properties of the environment state in order to satisfy an agent's individual preference. For example, HEMS could be given a policy stating that whatever the home state that turned out from its action should be, it must be comfortable for the resident, i.e. a room temperature should be 25 Celsius on average, a lighting condition should suit for working, and so on.

We could say Policies are high-level conditions to put on the agent's course of actions. Constraints on the other hand are the low-level ones, i.e. they are conditions to put upon the properties of the environment state to ensure those properties are acceptable to the agent in any stage of its actions.

For our agent, to solve a goal alternative candidate plans will be inferred using backward reasoning and these plans will be tested by forward reasoning against the Policies and Constraints. All the survival plans will be considered as possible action plans for the agent to carry out.

When an agent operates, it performs many processes for the following tasks simultaneously, these tasks are:

- timer task

- continuous state reading, updating, recording of the historical data
- event listening, recording, and event handling
- routine tasks
- scheduled tasks
- waiting to run a service task in response to a request
- solving the pre-defined goals to satisfy the constraints and policies

B. *Actions*

In order to preserve the correctness of an action execution in our agent framework. Every action to be taken by an agent must be coupled with both its pre-condition and post-condition. That means before an agent can execute an action it must test the pre-condition of that action first to ensure that pre-condition has be met before the action will be carried out; otherwise no action will be taken. Additionally, after the action has been taken successfully by the agent the post-condition must be put in place too. For example, for an agent to turn on any home appliance, that agent needs to be ensured that appliance is in the "off" state, and after the appliance has been turned on, there will be an update of its status on the agent to the "on" state as well.

C. *Event-Response Rules*

One important part of this agent framework is how the framework deals with events and performs the event handling. This issue involves some form of logical reasoning. To do logical reasoning, the framework needs to accommodate logic. In this stage we shall explain how our object-oriented framework can be enhanced with logic.

Objects and classes are the central concepts of an OO approach, and this is not so different in logic where objects (including persons), their properties and relationships together are expressed in terms of logical sentences; and these sentences are to be reasoned by logical inferences.

The rule we employ in our agent framework serves for two purposes, one is for defining a predicate which is in a form of a Horn-clause, and the other is for expressing an Event–Response rule, which is in a form of a conditional statement. An Event–Response rule is in the form:

<Program Statements> if <Events and Some Conditions>,

where <Program Statements> is a sequence of program statements and <Events and Some Conditions> is a conjunction of logical atomic sentences.

An event is expressed by an atomic predicate with at least one argument specifying a time stamp to signify when the event occurs. An event will be raised to be observed by the event listener which will then hand it to the event-handler to response with some matched Event–Response rules being fired.

Here is an example of Event–Response rules:

```
inverter.set_mode("fullsell") if
 fully_charged(enstorage,T1) and
 7.00<=T1<=17.00 and
 unoccupied(home,T2) and T1<T2<=17.00.
```

This rule says if the energy storage is fully charged during 7 am - 5 pm and the residents leave the house after that, set the inverter to "fullsell" mode to sell the solar power to the grid.

Events, rules, and predicates are attributes of an agent in our framework; they are treated as objects at the meta level and they will be handled by their associated methods accordingly.

IV. THE HEMS AGENT FOR HOME ENERGY EFFICIENCY

A. *HEMS: The Smart Home Agent*

A smart home, or the HEMS agent, can be modelled by our object-oriented agent framework. According to an OO concept, the smart home agent contains other objects; it contains floors and stairs; each floor contains rooms and spaces; and each room and each space contains home appliances and sensors. Clearly smart meters, home appliances, home sensors, inverters, and the energy storage are components inside the home, whilst the residents are considered to be external agents not being contained in the home but rather interacting with the home.

The HEMS agent accommodates many component agents which are working on their own simultaneously. Each of these agents maintains its own states, observes and handles its own events. Only the events referred to in HEMS's Events–Response Rules will be notified by those agents to HEMS. A class diagram illustrating class HEMS and its related classes is displayed in Fig. 2.

The HEMS's Events–Response Rules aim to cover most central control of the home. When a required event is notified (to HEMS) by a component agent, HEMS will handle it with its appropriate Events–Response rule. Some events are obtained from monitoring of the home residents' behaviors, e.g. human movement are notified by pyroelectric infrared sensors and Bluetooth Beacons; HEMS will handle them with some related Events–Response rules accordingly.

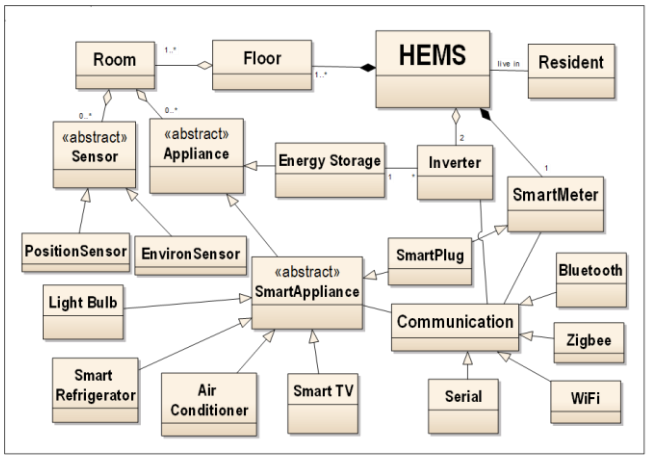

Fig 2 A class diagram showing class HEMS and its related classes.

The HEMS agent based on our framework is as follows.

A HEMS Agent
- **Aggregation** HEMS contains floors; each floor contains rooms and spaces; and each room (or each space) contains home appliances and sensors. It also contains a smart meter, inverters, and the energy storage.
- **State** **State Variables** represent a home state, which can be

requested by-demand from its component agents, e.g. home temperature (read from a temperature sensor), home power consumption (read from a smart meter), home energy reserved (read from the energy storage). A home state includes a status of each home appliance, e.g. a TV is currently turned on. The agent will try to maintain its state and sub-states to be the most up-to-date as well as possible. **State Updater** keeps revising the home state variables periodically by listening to each component agent to update the latest reading. **Historical Record of the State** stores historical data of these State Variables.
- **Events** The event is either **internal** or **external**. An event received from the PEA server is regarded as an external whilst the other others are internal as they are issued from the component agents of the home. Each agent including a resident can notify an event to HEMS via a communication port of the home gateway which has been set up in advance. On each port there is **an event listener** of HEMS monitoring an emerging event. **Historical Record of the Events** stores historical data of the events. - **Events–Response Rules** These are the central rules for control managed by HEMS; it requires interoperation with the component agents, the PEA server, and the residents to notify HEMS with emerging events and inform it of their current states.
- **Operational and Scheduled Tasks** This is the task the home always operates as its main task. The Scheduled Tasks is a central control schedule of operations for the agent components.
- **Provided Services** These are the services provided by the component agents accessible by the home and the services provided by the home itself. For example, a resident can remotely turn on/off a home appliance and can view the current state of his/her own house or even check whether an inverter is still working.
- **Communication Protocols** Our smart home uses ZigBee/Bluetooth/WiFi to communicate with its component agents, and Bluetooth/WiFi to communicate\interact with an in-home display—the home resident. - **Ports** We can categorize communication ports into three different purposes. The first is for reading a home state, the second is to listen to an event, and the third is to take a request from a home resident and return a response.
- **Goals** This is a goal to be solved by a dynamic construction of an action plan to satisfy the constraints/policies.
- **Policies** High-level constraints the home must satisfy during its operation, e.g. human comfort, cost saving.
- **Constraints** Low-level constraints the home must comply during its operation, e.g. while an air conditioner is working, each room temperature should not be higher than 25 Celsius, the energy storage must not supply the electricity to the home if its capacity is below 20%.

For the HEMS agent to serve energy efficiency purposes, we define certain Event-Response rules, policies, constraints, and goals to achieve that.

B. Event-Response Rules for Energy Efficiency

Due to a limited length of this paper, we can give only some examples of such Event-Response rules:

```
hems.setmode("DR")if
 current_mode(home,M,T) and M≠dr and
 dr_event(pea,dr,T).

inverter.set_mode("fullsell")if
 current_tariff(pea,current_pakage,Tar,T) and
 energy_selling(pea,Rate,T) and Rate > Tar
 and battery_level(enstorage,L,T) and L > 80.
```

The first rule says whenever a DR event is issued by PEA, HEMS will set the house to the DR mode. The second says when the energy selling rate is higher than the home current tariff rate at any time and the energy storage has stored enough power, let the inverter sell the power to the grid.

C. HEMS's Policies and Constraints

The policies govern the way the home behaves. We define the policies of our smart home to be four operation modes and they are represented by some desirable (satisfiable) conditions of the current state of the house, e.g.

- Normal (no conditions): `none`
- comfort: `room(R,Temp,T)and human(H)and in(H,R,T) and 25<Temp<26`
- economy: `room(R,Temp,T)and human(H) and in(H,R,T) and 26<Temp<28`
- max-saving: `imposing the goal max_saving(.)`
- demand-response: `aircon(A)and in(A,home,T)and turned_off(A,T)`

where in the 2nd, 3rd, and 5th conditions all the variable are universally quantified. The comfort policy can be read as "any room which has a human in, its temperature must be between 25 and 26 Celsius". Interestingly, the max-saving policy is to impose solving of a HEMS goal `max_saving(.)`, this will be described next.

D. Solving Goals for Energy Efficiency

Our HEMS provides an important service of energy saving for its residents, and this is what goal in our agent framework will play its vital role.

Let us model the energy saving knowledge first. The idea is that to know how to save energy is to know how much energy is consumed by each device and even by each individual function of that device. Hence we need to construct our energy consumption knowledge base. In the smart home context, this knowledge base will contain a fact of the form: `consumes(F,D,E)` which states that a function `F` of a device `D` consumes energy amount of `E` W, for example:

```
        consumes(on,tv,148).
        consumes(standby,tv,10).
                .
        consumes(on,ledBulb,15).
        consumes(dim75,ledBulb,12).
        consumes(dim50,ledBulb,9).
                .
        consumes(settemp24,aircon,4900).
        consumes(settemp25,aircon,4800).
        consumes(settemp26,aircon,4650).
```

Once we have created this energy consumption knowledge base, and we know all the current active devices, let they be {d1,d2,d3,…}, in the house, and also know which of their functions currently operating, i.e. {f_{d1}, f_{d2}, f_{d3},…}, we can easily explore alternative ways of how we can save energy by comparing the current amount of energy consumption of each f_{di} with other function of di which consumes less energy consumption than f_{di} does.

For example, let's assume there is an active `ledBulb` device which is currently performing function `on` and this consumes 15 W; obviously to save energy for this device we could change this light bulb to any function, such as `dim75` or `dim50` which consumes less energy than 15 W. Alternatively, we could even change from one active device to another lower energy consumption device if their functions are exchangeable.

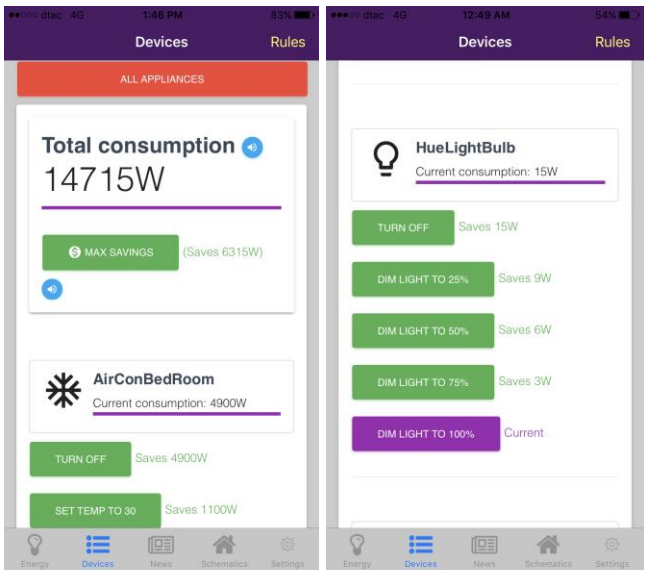

Fig 3 The Energy saving feature in the HEMS mobile application.

Therefore, we pose the energy saving `max_saving(.)` as a goal to HEMS and let HEMS solve it by employing backward reasoning to explore all possible less-energy consuming functions of the current active devices while applying constraints to prune the search space. The constraints may rule out some function of some device which violate the constraints. For example, for a fridge to turn `off`, this is an alternative way of saving energy as this function consumes only 0 Wh, but we do not allow that as our food will be spoiled, so we shall assert a constraint

```
not_allowed if fridge(F) and in(food,F,T) and
turned_off(F,T)
```

We could also use the economy mode (policy) as this kind of constraints. To be flexible, our agent framework allows these constraints to be programmable.

Not only that HEMS can explore all possible alternative ways of saving energy, but it can also find the maximum saving solution for the resident, see Fig. 3; this is our mobile application's screen, where HEMS reports the home current consumption and advises the maximum saving if the resident chooses it by just pressing the "MAX SAVINGS" button; the right screen displays alternative ways of saving energy by each possible function of an active device.

We can also apply the same mechanism of solving a goal to save electricity bill by comparing current user payment based on the current tariff rate with other alternative rate to find a cheaper one based on the current load shape of the house. There will be also other kinds of goal for HEMS to explore further in the future using this general mechanism of problem solving.

V. Related Works and Future work

The logical reasoning in our problem solving is rather similar to that of LPS [6][7], i.e. the way our agent solves a goal by backward reasoning and matches rules with emerging events by forward reasoning, we will pay more attention to the logical aspect of our agent framework in the future study.

VI. Conclusion

The agent framework we proposed in this paper employs event-response rules to handle emerging events, the HEMS agent can reason about home states in order to satisfy the energy-efficiency policies and is able to solve goals to achieve home energy saving.

Acknowledgment

We gracefully acknowledge the funding for this research by Provincial Electricity Authority under the PEA Smart Home Project (Project Code B-037-57).

References

[1] T. G. Stavropoulos, E. S. Rigas, E. Kontopoulos, N. Bassiliades, and I. Vlahavas, "A multi-agent coordination framework for smart building energy management", In *Proc. of 25th International Workshop on Database and Expert Systems Applications*, pp. 126-130, 2014.

[2] Q. Sun, W. Yu, N. Kochurov, Q. Hao, and F. Hu, "A multi-agent-based intelligent sensor and actuator network design for smart house and home automation", *Journal of Sensor and Actuator Networks* ***2***, pp. 557 – 588, 2013.

[3] L. Chen, C. Nugent, M. Mulvenna, D. Finlay, X. Hong, and M. Poland, "A logical framework for behaviour reasoning and assistance in a smart home", *Int. Journal of Assistive Robotics and Mechatronic* ***9*** *(4)*, pp. 20 – 34, 2008.

[4] V. Hirankitti, "An Agent Framework for Home Energy Management System," In *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2015*, WCE 2015, 1-3 July, 2015, London, U.K., pp. 184-189.

[5] V. Hirankitti and T. Makee, "An Object-Oriented Agent Framework for HEMS," in *Proc. of SAI Intelligent Systems Conference 2016*, London, pp. 810-818, 2016.

[6] R. Kowalski and F. Sadri, "A logic-based framework for reactive systems", In *Proc. of 6th International Symposium RuleML 2012*, France, pp. 1-15, 2012.

[7] R. Kowalski and F. Sadri, "Reactive Computing as Model Generation," New Generation Computing, vol. 33, issue 1, pp.33-67, 2015.