Proceedings of the World Congress on Engineering 2015 Vol I
WCE 2015, July 1 - 3, 2015, London, U.K.

An Agent Framework for Home Energy Management System

Visit Hirankitti, *Member, IAENG*

***Abstract*—A smart home is a promising green technology not only to save people money on their electricity bills but also to alleviate the climate change in a micro-level. In this paper we report on our on-going research project how we are developing a smart home prototype. Here we focus on the essential part of a smart home, i.e. HEMS (Home Energy Management System), by investigating how we develop HEMS using an agent framework. A model of agents for constructing HEMS based on both object-oriented and logic programming approaches is proposed. The agent model is implemented in Python. We also argue for the benefit of combining both approaches into a unified agent framework.**

***Index Terms*— Intelligent Agent, Smart Home, Home Energy Management System, Internet of Things.**

I. Introduction

The world is moving towards a green technology. A technology that can alleviate the climate change. For the electricity domain, in many countries, their governments are now promoting more environmental-friendly ways to produce electricity while encouraging their people to utilize electricity more efficiently.

In this paper we present how we develop a smart home, a green technology that will help a household resident to utilize electricity more efficiently and the home itself can also produce electricity from an alternative source, such as solar or wind energy. The essential part of a smart home is Home Energy Management System (HEMS); this is what we will investigate in great detail in this paper.

When we refer to HEMS, we mean a software to operate a smart home by monitoring and controlling all its electrical devices in such a way to use the electricity in an efficient way as well as to make use of the most from its own generated electricity. To operate a smart home, HEMS requires many hardware components, see Fig 1, and the HEMS itself is running on a home gateway.

In this paper we shall propose an agent framework for constructing HEMS. This is an on-going research conducted under the PEA Smart Home Project funded by Provincial Electricity Authority (PEA) the major utility company in Thailand. In this project we are developing HEMS, a home gateway, smart plugs, and home sensors, whilst all home appliances are commercial products to be acquired in.

To prove our concepts, we will build the real house with all these devices installed and operating in order to test our HEMS in the real-life situation and to evaluate its merit. The project will be completed in November this year.

A lot of research interest now moving towards developing a green technology. How to use energy efficiently in a building and in a home also attracts much attention over recent years. A smart building and smart home with smart energy management system is a trend of recent research.

The intelligent agent approach is one of a few promising techniques which has been applied to develop a smart home and smart building. For example, a recent 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. For us we consider this research as being related to our previous work on intelligent control of traffic-lights [4], [5] where we proposed an agent approach using rules and logic programming for intelligent control.

The remainder of the paper is organized as follows. Section II describes the nature and scope of our smart home and a model of an agent designed based on a smart meter. We then propose our agent framework for HEMS in Section III. In Section IV we point out how our framework can be evaluated. We later discuss related work in section V and conclude this research work in section VI.

II. The Problem description

A. Our Smart Home Hardware Landscape

Our smart home will be equipped with many hardware components as seen in Fig 1. Mainly they are divided into two groups. The first group is to design and develop by us in the smart home project, i.e. a home gateway, smart plugs, and home sensors. The other group is commercial off-the-shelf products, i.e. home appliances, mobile devices, a solar roof-top, a wind turbine, and an energy storage. The last piece of the components is a smart meter which we have completely designed and developed in a previous research project also funded by PEA.

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).

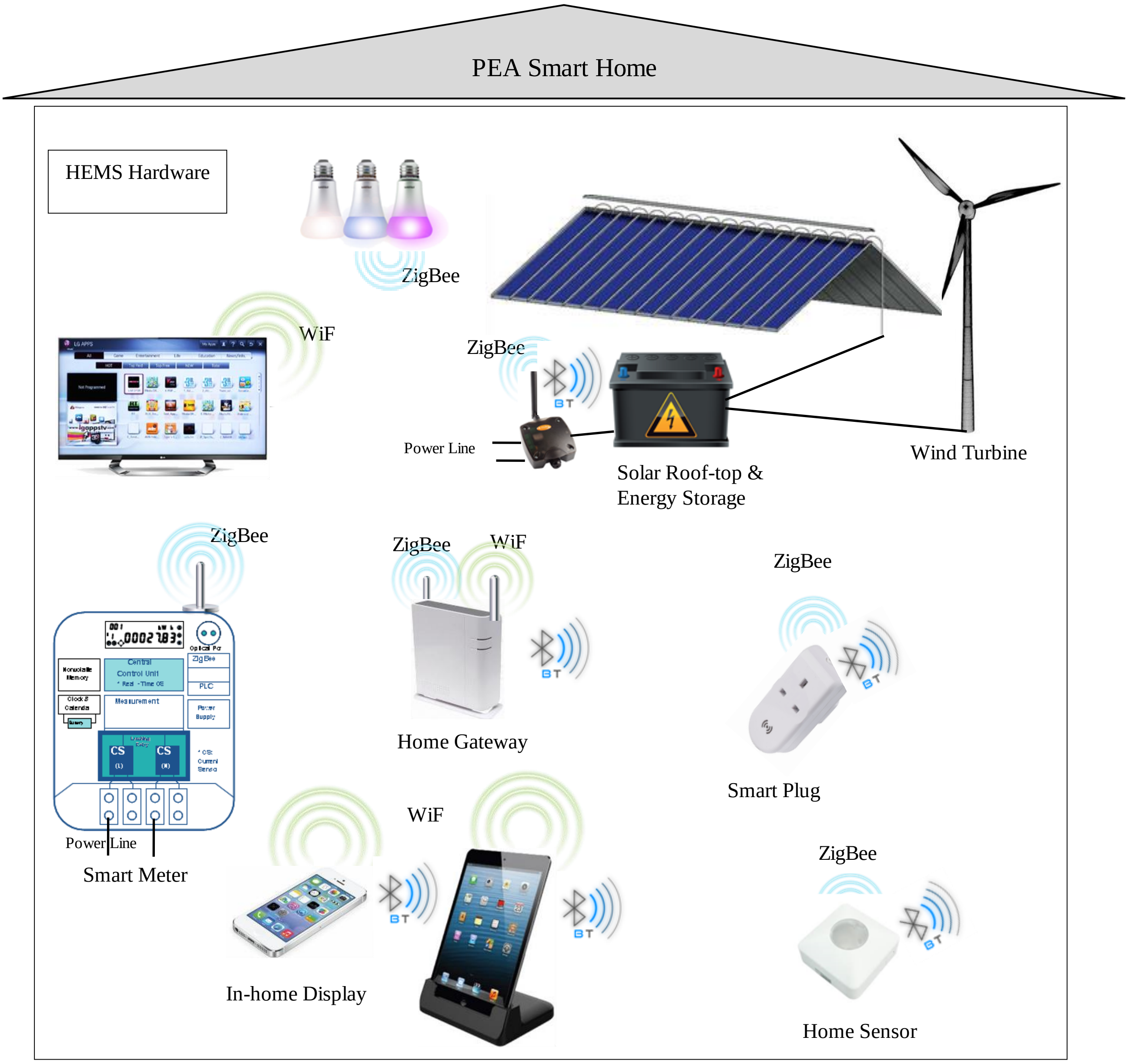

Fig 1 Our smart home hardware landscape.

Next we shall explain what hardware components are doing in our smart home:

- Home Gateway

HEMS is the software operating the smart home and it is installed and runs on the home gateway, so 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 consumed in the house. It can notify some abnormality of the electricity, such as over voltage/current. Its measurement of home energy usage will be used to calculate a tariff. The meter is so-designed to be able to disconnect or connect loads of the house if necessary that can be controlled by the utility company who supplies the meter.

- Smart Plug

To allow an ordinary home appliance without any smart feature to be able to control by HEMS, such a home appliance needs to connect to take electricity from a smart plug. Our smart plug is a stripped-down version of our smart meter, so it has ability of advanced metering. The smart plug can be controlled by HEMS to turn on/turn off a home appliance via a wireless communication using either ZigBee or Bluetooth. In addition, the plug is also programmable to be turned on/off automatically according to a pre-programmed schedule uploaded from a smart phone via a ZigBee or Bluetooth communication.

- In-home Display

Home residents can access to HEMS using the In-home Display. These users will be provided with a mobile application and a web application running on it. The in-home display is a smart phone and a tablet. It can connect to the home gateway either via Wi-Fi or Bluetooth. Once it gets connected to HEMS on the home gateway, the users can manage or configure their home devices, monitor the home activities, and even write a program/script to control these home devices.

- Home Sensors

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

A home sensor monitors an ever changing state of the home and also detects an internal or external event of the home. The sensor then feeds this information to HEMS for a control purpose. Each home sensor is equipped with a ZigBee module or a Bluetooth module so that it can be connected wirelessly. The home sensors deployed in our smart home are, for example, pyroelectric infrared sensors and Bluetooth Beacons (sensors) used to detect human movement and position, thermometer sensors used to report temperature, photo sensors used to detect a lighting condition, etc.

- Home Appliances

They provide functionalities to the residents to serve their personal purposes. The device activities can be monitored and controlled by HEMS wirelessly over Wi-Fi, ZigBee, or Bluetooth. The way to control each home appliance could be as simple as turning on/off it (using a smart plug) or an advanced control depending on how smart that kind of home appliance is.

- Solar Roof-Top and Wind Turbine

They are the power generators of the smart home. The two devices are autonomous, they can work on their own and keep charging electricity into the home energy storage.

- Energy Storage

All the electricity generated by the two generators is stored in the energy storage, a kind of a long-lasting rechargeable battery. The current it directly supplies is DC, so it needs a DC-to-AC inverter to convert that into AC to be able to supply to the loads in the home. HEMS can decide and control when the home should get the electricity from the energy storage and/or otherwise from the source from PEA by only controlling another smart meter equipped with the energy storage. Essentially, this smart meter will connect/disconnect the electricity supply from the energy storage as well as can measure how much energy being supplied by the energy storage to the home.

B. *A Lesson Learned from Smart Meters*

In 2012 KMITL was funded with one and half year research project by PEA to design and develop industrial prototypes of (single-phase and poly-phase) AMI smart meters. According our complete industrial smart meter prototypes, we think they are complicated enough and can serve well as intelligent agents, so we apply an object-oriented approach to model them as follows.

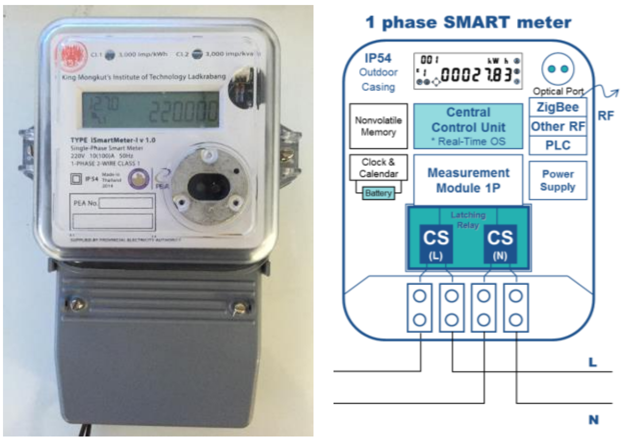

Fig 2 One of our completed smart meter prototypes and its structure.

An Agent: A Smart Meter
- Mater State **State Variables** represent a meter state, e.g. time, current voltage, real-time current, power factor, etc. **State Updater**, the measurement module, keeps updating the state variables continuously. A real-time clock keeps an accurate time. **Historical Record of the State** stores historical data of these State Variables.
- Events Event data is notified by **an event listener** (or a sensor), e.g. removal of meter cover will trigger a pedal switch which in turn creates an anti-tampering event. The event could be **internal**, e.g. an event of over current, or **external**, an event of anti-tampering. **Historical Record of the Events** stores historical data of these emerging events. **- Events–Response Rules** The smart meter has simple Event–Response Rules, e.g. if the meter cover is removed it will record this event and notify the utility company; if a power outage happens it will send the current meter status to the utility company and shut down the meter (the meter has a power back-up battery, so it can operate for a short period of time after a power outage happens). When an event gets notified, **an event handler** will fire the appropriate rule and execute the corresponding Response action.
- Operational and Scheduled Tasks This is the task the meter always operates as its main task, e.g. instant reading and displaying of various metering data. Some tasks are performed in a timely manner (i.e. according to **a pre-defined schedule**), e.g. storing a snap-shot of metering data on every 15 minutes into the meter load profile.
- Provided Services These are services to serve the outside world. They are **stored procedures** to be executed once **a request** is issued from the outside and the meter responses to the request by executing the matched procedure.
- Communication Protocols Our smart meter deploys many communication protocols, i.e. HDLC, ZigBee Pro, G3-PLC (Power Line Carrier), and DLMS/COSEM. A smart meter is connected via some **communication ports**, i.e. serial ports.

From what we learned from a smart meter, we can use it as a basis to design a general framework of an intelligent agent for HEMS.

III. AN AGENT FRAMEWORK

A. *A Generic Agent*

An agent in our design is in a form of an object-oriented class. In this case State Variables, State Historical Records, Events, Event Historical Records, and Event–Response Rules are predefined data attributes, while the State Updaters, Event Listeners, and Event Handlers are methods to manipulate on these data attributes. For the Operational

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

and Scheduled Tasks, Provided Services, and Communication Protocols, they can also be implemented by methods in OOP.

To make it more extensive we add Goals, Constraints, and Policies which are high-level commitment an agent wants to achieve at the global level.

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
- Constraints
- Policies

When an agent operates, it has to perform 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

For a low-level embedded-system agent, like a smart meter, this multi-tasking jobs can be implemented in C with a support by an RTOS (Real-Time OS), but for a high-level agent, we can employ multi-threading programming supported by the software and OS platform; and for our agent implementation Python copes with this quite well.

The essential part of this agent framework is how the framework deals with events together with event handling, and similarly how to deal with goals, constraints, and policies; both of these two issues certainly involve some form of logical reasoning. Due to the limited length of this paper, we shall reveal only the former and leave out the latter to be investigated elsewhere.

To cope with logical reasoning in this agent framework, the framework needs to accommodate logic. In the next section we will explain how logic can be merged into our object-oriented framework.

B. Enhancing Object-Oriented based Agent with Logic

Objects and classes are the central concepts behind an object-oriented approach, and this is not so different in logic where objects (including persons), together with their properties and their relationships are expressed in terms of logical sentences; and these sentences are being reasoned with using logical inferences.

Apart from objects, other formalism, like numbers, mathematical expressions, and functions, both object-oriented approach and logic all have these in common. However, what are missing from an object-oriented approach are predicates (predicate symbols to be precise) which describe objects' properties and relations between objects. Provided with objects, numbers, variables, predicate symbols, function symbols, and logical connectives, one can form a logical sentence.

Taking an object-oriented approach for granted, we can enhance it with logic by introducing predicates and logical sentences, in terms of facts and rules (similar to those in logic programming). We could add a new syntax in an object-oriented language, such as Python the following:

```
preddef father(X:Human,Y:Human):

  Rule(father(X,Y) if male(X) and
                           parent(X,Y))
```

This is a definition of predicate `father` which takes two object variables `X` and `Y`, both belong to class `Human` (a variable name begins with a capital letter). A predicate definition is so-designed that it can be asserted with more rule later, for example

```
class Human(object):
  def __init__(self,objname):
    self.__name = objname

john = Human("john")
mary = Human("mary")

father.assert(Rule(father(john,mary)if True))
```

The predicate definition is added with a fact, a rule with a body `True`.

In addition this predicate can be proved as a query by a method `solve`, which returns a set of answers.

```
father(A,B).solve()
> {father(john,mary)}
```

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. For example,

```
send_message(M,"The cover is opened.",pea) if
unauthorized_at(M,T1) and
meter_cover_is_opened(M,T2) and T2 >= T1.
```

An event is expressed by an atomic predicate with at least one argument specifying the time stamp to signify when the event occurs. The event will be raised to be observed by the event listener which will then hand it to the event-handler to

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

response with some matched Event–Response rules being fired.

Events, rules, and predicates are data attributes of an agent in our framework; and they are treated as objects at the meta level and they will be handled by their associated methods accordingly. In this context we allow objects, object variables, functions and methods of an objected-oriented approach to be used in a logical context and conversely a logical sentence to be used in the context of object orientation quite naturally.

C. A Smart Home Agent

Considering a smart home, or HEMS to be precise, as an agent, we can model it using our object-oriented based agent model introduced earlier. However, as a smart home contains other objects, it is considered to be a container object instead, that is, a smart home contains floors and stairs; each floor contains rooms and spaces; and each room contains home appliances and home sensors. Precisely smart meters, home appliances, home sensors, energy generators, and the energy storage are components in the home, whilst the residents are considered to be external agents not being contained in the home but interact with it.

The HEMS agent is complicated as it accommodate 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 issued by those agents to HEMS.

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

Taking all these into consideration, we can now define an agent framework for HEMS as follows.

A HEMS Agent
- Aggregation HEMS contains floors and stairs; each floor contains rooms and spaces; and each room (or each space) contains home appliances and sensors. It also contains two smart meters, generators, 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 has been turned on or off. **State Updater** keeps revising the home state variables periodically by issue a request to each component agent to obtain the recent reading. A request is sent and a response is received via a communication port. **Historical Record of the State** stores historical data of these State Variables.
- Events The event could be **internal** or **external**. An event received from a resident 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 a coming event. **Historical Record of the Events** stores historical data of the events. **- Events–Response Rules** These are central rules of control managed by HEMS; it requires interoperation with the component agents and the residents to notify HEMS with events and inform HEMS of their 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 house 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 the CCTV is still working.
- Communication Protocols Our smart home uses ZigBee Pro, HDLC, and DLMS/COSEM protocols to communicate with a smart meter; it uses ZigBee Pro or a Bluetooth mesh protocol to communicate with its component agents; it uses a Bluetooth mesh protocol to communicate with an in-home display (a home resident). - Ports We can categorized communication ports into three different groups. 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 a proof to satisfy the constraints and policies.
- Constraints High-level constraints the home should comply during its operation, e.g. while an air conditioner is working, each room temperature should not be higher than 28 Celsius, the energy storage must not supply the electricity to the home if its capacity is below 10%.
- Policy human comfort, cost saving, energy saving

Now we give some examples of HEMS's Events – Response rules:

```
Tv.stand_by() if unoccupied_for(Room,T1,T2)
  and working(Room.Tv,T2) and T2 – T1 > 10
Tv.resume() if unoccupied_at(Room,T1) and
  human_entering(Room,T3) and
  is_standby(Room.Tv,T2) and T3 > T2 >= T1
```

```
storage_meter.on() if is_off(storage_meter,T)
and tariff(pea,Tar,T) and
energy_selling(Rate,T) and Rate > Tar and
energy_storage.battery_level(L,T) and  L > 80

storage_meter.off() if is_on(storage_meter,T)
and tariff(pea,Tar,T) and
energy_selling(Rate,T) and Rate > Tar and
energy_storage.battery_level(L,T) and  L < 10
```

The first rule says when a room is unoccupied more than 10 minutes and there is a television in there still working, then HEMS will turn it into a stand-by mode. The second rule will be required to turn the TV back to its normal operation. The third rule says when the energy selling rate is higher than the tariff rate at any time, let the energy storage supply the power to the home by connecting the smart meter of the energy storage to the power line, and the final rule will disconnect it when the battery level at that time is rather low.

What we can see as the benefit of an object-oriented approach for the HEMS agent is that it supports the concepts of software components, i.e. the agents of home sensors and home appliances can be added to and removed from HEMS as will. They can be categorized into sub-classes and super-classes. These software components can be published on the web, and will be shared and used by the public globally.

D. A Home Resident Agent

The last agent in the smart home context is a home resident. As the only way a resident can communicate with the home, i.e. HEMS, is via a mobile application installed in the in-home display, an agent mobile application running on the in-home display hence represents the human resident; its agent model is as the following.

A resident agent may not possess all elements of the agent framework referred to in Section III, but the agent maintains some states of a human resident, like updating his/her activity, e.g. walking, sleeping, waking up, etc. The agent application may keep observing some events while a resident is staying at home, e.g. tracking the resident movement using Bluetooth signals. The resident may do some services for the home when being asked, like please turn off the fire alarm; and of course the resident can issue commands to HEMS directly to control home appliances. So more or less a resident can be modelled as an agent in our agent framework.

In the near future, with an advent of wearable computers, such as a smart watch, this small computer can provide useful information such as personal movement, preference, and even emotion to the smart home. This can make HEMS serve the resident even better.

IV. Evaluation of Our Framework

The agent framework for HEMS just proposed, when it is fully implemented, will be put into test with its hardware components (which are under development) in the real-living environment, where the real smart home will be built (in three month time) with all hardware and software components be installed and operated.

The merit of our agent model is the enhancement of logic upon a framework of object orientation, which has been a well-proven concept in software engineering of this decade. HEMS requires logical reasoning of intelligent control and efficient energy management. We believe this is the direction to go and it will be a good testbed for logic to prove its usefulness in a real world application.

One key element of our HEMS is the smart meters and the smart plugs, these devices help us greatly in assessing how well our HEMS is in terms of the energy saving and efficient energy management based on an intelligent reasoning.

V. Related Works and Future work

The agent framework we proposed in this paper employs intelligent control using rules to handle emerging events. This is the same approach we had investigated elsewhere in other domain, i.e. intelligent control of traffic-lights based on an agent approach [4] and also on a multi-agent approach [5]. In those works we employed the logic programming approach, but now the problem we are solving involves many hardware devices which should be considered as objects or things as recently being known as the Internet of Things.

For the future work we will investigate how our agent framework can be combined with Production System Language (LPS) [6]. According to LPS our Events–Response rules are “Reactive Rules” in this language.

VI. Conclusion

We have presented an agent framework for HEMS based on an object-oriented approach enhanced with a logic reasoning. The agent model can be applied on very level of a smart home context ranging from a smart home (HEMS), smart meters, home sensors, home appliances, and so on.

Acknowledgment

We gracefully acknowledge the funding for this research by Provincial Electricity Authority under the PEA Smart Home Project (Project Code B-037-57) and in this paper we make use of some result from Research and Development of AMI Smart Meters Project (Project Code E-025-54-KMITL) which is also funded by PEA.

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*, 2014, pp. 126-130.

[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 and J. Krohkaew, “An agent approach for intelligent traffic-light control”, In *Proc. of Asian Modelling Symposium AMS2007*, 2007, pp. 496-501.

[5] V. Hirankitti, J. Krohkaew, and C. J. Hogger, “A multi-agent approach for intelligent traffic-light control”, In *Proc. of 2007 International Conference of Computational Intelligence and Intelligent Systems*, London, 2007, pp. 116-121.

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