Applications of SNMP in Ubiquitous Environment

Syed Shariyar Murtaza[1], Syed Obaid Amin[2], Choon Seon Hong [3]

[1] shariyar@networking.khu.ac.kr,obaid@networking.khu.ac.kr,cshong@khu.ac.kr

Abstract

Ubiquitous computing is fast becoming a reality. Several ubiquitous devices compliant with the service discoveries like UPnP and JINI are available now. Individually, managing the ubiquitous devices with UPnP is not a problem. But, it is quite cumbersome to manage these devices in the large scale environment especially when the management issues like performance, fault and security management have to be considered. SNMP is a standard for network management and widely deployed. A majority of SNMP compliant devices are already present. We propose that by integrating SNMP with service discovery protocols like UPnP, we could leverage the use of management factors in the business, organizations, and other multi-user ubiquitous environments, while providing a uniform interface to an administrator for the management of UPnP and SNMP based devices.

1. Introduction

Ubiquitous computing is envisaged as a computing paradigm with minimal user intervention, emphasizing detection of environmental conditions and user behaviors in order to maximize user experience.

Up till now, the researchers have mostly focused on making the vision of ubiquitous computing a reality. Service discovery issues, context awareness issues, wireless connectivity, sensors, RFIDs and etc, have mostly been the focus of the researchers. But, with the devices growing in number, embedded ubiquitously around our environment, there is a need to manage these devices. By management we mean, analyzing the performance, diagnosing the faults, caring for accountability and managing the security of the devices.

But, why do we need the management of devices? Service discovery protocols are already here, they allow to control the functionality of each device. The problem arises when large number of devices is deployed in environments and individually determining the fault and diagnosing the performance becomes cumbersome, and if the devices are in a shared environment then the optimum use of shared resources is also the issue of concern. Let us consider a complicated ubiquitous environment e.g. office, organization, manufacturing environment etc, where we have to keep track of all the events occurring in the systems, or if some devices experience havoc, then we have to keep log of the wrecks of incidents, what is effected and when!

SNMP based network management is quite popular. It allows the easy and simple management of network resources, and could be extended to applications as well. There are many reasons for using SNMP. The foremost reason for using SNMP is its wide deployment and it is already a management standard. Large number of SNMP compliant devices is already available and more and more UPnP devices are also emerging. Using our MIB (shown in the later section) any kind of SNMP manager can connect to the UPnP based environment. It is quite simple and several APIs are available for its implementation. With the use of SNMP-MIB we can keep log of the events occurred in the system, which would help in identifying the exact cause of the problems, and would help the administrator in recovery of devices or system. SNMP has the rich feature of extensibility, as MIBs could be defined privately, as well. This is particularly important in ubiquitous environment, where the devices and their interfaces change rapidly.

Once we are able to keep the record of the events occurring in the system then it would be quite easy to perform any kind of analysis on this data. For instance; guessing future trend by complex data mining technique or simply checking the power consumption of an air conditioner in a given day?

In order to have a uniform identification and accessing mechanism for devices we have already proposed a URI scheme [11] for every physical object. The main advantage of using URI for the physical objects is the ability of a person to comprehend it as compared to IPv6, object identifier or any other unique device identification. In our SNMP manager we have also provided this approach. Our manager allows the administrator to query with URI and return the information about the devices. Since URIs are used in UPnP as well, this made our work quite easy. The MIB shown later is developed by considering a UPnP based environment.

Let us elaborate on this, with some examples from management perspectives e.g. consider an air conditioner, which can be controlled by service discovery like UPnP, but to monitor its performance, whether its cooling performance is appropriate or not, is it heavily used! A management protocol like SNMP could be beneficial. Although, it could even be done without even using SNMP and developing a management application at the control point but as mentioned earlier the use of SNMP allows the standardization of the system and homogeneous interface to the administrator for the management of SNMP and UPnP based devices. Similarly, in a multi user environment, misuse of shared resources could be taken

into account e.g. a person's car usage by other persons, use of shared printers. This will also help in detecting the faults in ubiquitous environments, like malfunctioning of sensors or too much traffic from sensors leading to improper results.

One other motivating factor of this project is to reduce or eliminate "*standby power consumption"* of electric or electronic devices. Standby electricity is the energy consumed by appliances when they are not performing their main functions or when they are switched off. Most people are ignorant of the fact that modern electrical and electronic appliances, consumes energy even when they are not in use, for example energy required to display clock, detect remote control signals of audio players etc. Recent studies showed that most of the electricity used by home appliances is consumed in standby mode. Although, our paper doesn't elaborate much about it but with the help of our proposed system an administrator would be able to save the power consumed by devices in standby mode and can control his electricity bills.

In the rest of this paper, we'll elaborate on the relevant work, system overview, MIB, emulation, example application and finally conclude our paper.

2. Relevant Work

SNMP refers to a set of standards for network management, including a protocol, a database structure specification or Managed Object definition and a set of data objects. SNMP encompasses a manager/agent model. The manager and agent interact with each other using Management Information Base (MIB) and a set of commands.

UPnP (Universal Plug and Play Protocol) enables discovery and control of networked devices and services, such as network-attached printers, Internet gateways and consumer electronic equipments. UPnP encompasses two main objects, control point and a device, and use XML, SOAP as the communication protocol. Both communicate using UPnP protocol.

In the past, certain approaches for network management have been proposed. XML based network management is an important area of research in the realm of network management. The authors [8, 9] mentioned certain advantages of XML over SNMP, like HTTP header compression for large data, use of HTTP over TCP, SSL for reliability and security, and more data types in XML Schema. Taek Ju et al [10] proposed embedded web server with XML based management. For embedded devices, Hong et al [7], proposed the integration of SNMP and Web servers into the embedded devices. A work in some ways similar to our work has been carried out in [3]. In this paper author proposed a proxy based architecture with SNMP, concerning only the EIB (European Installation Bus). They didn't consider about the ubiquitous environments or the UPnP based environment. The MIB designed was also static, supporting only EIB systems.

Although, several advantages of XML based network management have been articulated by authors, but still the most important fact is that SNMP has been deployed widely and has proved itself as a standard. As mentioned earlier many SNMP based devices are already available. SNMP's ability to measure data, to support fault and performance management is efficient. SNMP v3.0 also provides good security features. But, the configuration and remote management suffer at SNMP. Our objective here is to integrate UPnP with SNMP and provide the homogenous interface for the management of both SNMP and UPnP based devices. Use UPnP to discover and control devices for UPnP compliant devices, while employ SNMP to perform the management activities, like accounting, fault tolerance and performance monitoring for both type of devices.

3. Purview of the System

The system's overview can be comprehended by Fig. 2. Here we manage one dedicated UPnP control point with the SNMP agent to monitor the system. The dedicated control point is used because in multi user environments, multiple users can enter and leave the system, with personal or different control points. By using a dedicated CP with SNMP agent, we can monitor the activities and find out the utilization of shared resources. This dedicated control point and the SNMP agent act as a proxy agent between UPnP and SNMP.

This proxy agent subscribes to different events in different devices, as requested by the SNMP manager. It then acts as an eavesdropper to the ubiquitous environment. Whenever, the services of the devices are invoked by other control points, and then the devices send the appropriate event notifications to proxy agent. The proxy agent's UPnP control point gathers information from the devices and its SNMP agent keeps them in the MIB format.

The manager whenever needed can query the agent, as per MIB or set specific traps, converting the data into a visualized form, showing statistics and performing other required analysis.

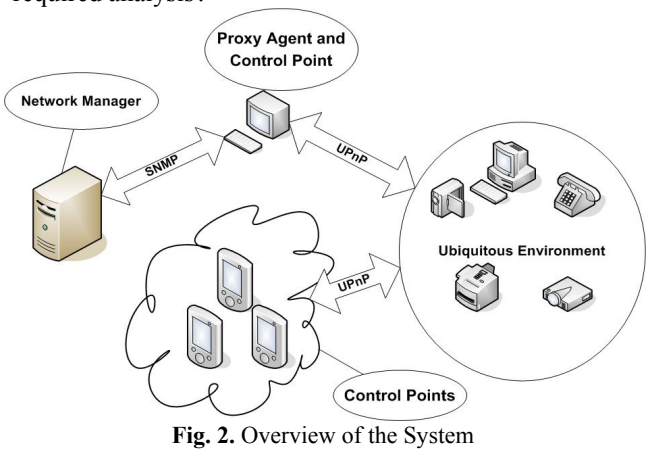

Fig. 2. Overview of the System

4. MIB

In SNMP based system MIBs are static, they don't allow the addition of dynamic contents. But, in ubiquitous environment appliances enter and leave the system dynamically. Also, the interfaces of the appliances are liable to change, either because of the updates by

manufacturer or by the replacement of manufacturer by a user.

Due to this, we have created a generic MIB. This MIB is like a relational database, in which unique device name (UDN) is acting as the primary and foreign key in different MIB tables. These MIB values are filled by the agent, working together with the UPnP control point. MIB is demonstrated in figure 3.

This MIB constitutes five different tables. Most of the fields constituted by this MIB are mapped from the UPnP Device and service description schema while other fields have been added for the management purpose. The detail of each of the table is as follows:

'Device Profile Table' keeps the record of general information about the devices. Its index keys are URL Base and Unique Device Name. Check in and check out time fields are used to determine when the particular device entered or left the system. Since, each device could have embedded devices therefore the parent device field is used to detect whether a particular device has any parent or it itself is the root device. Similarly we have the embedded device field. Both of these fields are used to search recursively for embedded or parent device. Rest of the fields re self explanatory and their functionality can easily be determined from their name.

'Device Service Table' provides information about the services offered by each device. It also has two indexes unique device name (UDN) and service Id. It contains two important fields Subscribe and Trap. These fields are set by the manager. Subscribe is used to subscribe to service state variables by UPnP control point while the trap is used to issue traps to the manager by agent if state variables are modified .

'Service Action Table' encompasses the information about each action and their arguments, direction of each argument and the associated state variable name.

'State Variable Table' contains information about the state variables of a device, mentioning whether the state variable is evented, and its minimum and maximum value. This subscription field is set by the manager.

'Event Info Table' stores information about the events to which the proxy agent has subscribed. It contains the state variable name, date time and the control point Id which caused the event to occur and to which value has the state variable been set.

5. Implementation

In order to analyze the performance of our system we have emulated our system using the following simple scenario. We considered a single person apartment having the following 10 devices installed in the system: Main door sensor, fridge, TV, air conditioner, wash room bulb, 2 Tube lights, Microwave oven, phone and PC. We used two control points, one as a user control point, and other as an eavesdropper to listen to the events. An SNMP manager was also deployed.

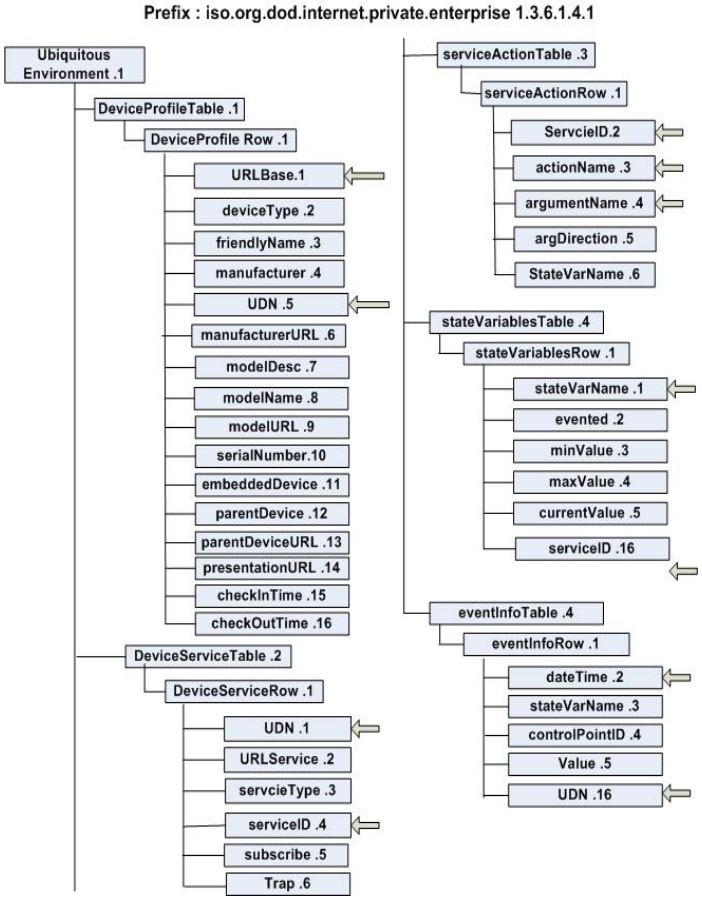

Fig. 3 MIB Tree

. In order to simulate a person's behavior we used the simple algorithm in a user's control point, its pseudo code is shown here.

```
Step1:  Initialize the virtual clock and schedule
the event for every 5 min.
Step2:  On every timer event
  Step 2.1: If clock>= (7) and clock< (9)
    Step 2.1.1 Generate a random number for a
               device
     Step 2.1.2 Generate a random number for an
     action of a service in device
     Step 2.1.3 Hold (10 * Random number)
     Step 2.1.4 Invoke action
```

We divided the time into 4 parts to simulate the peak hour usage or no usage e.g. breakfast and wakeup timings (7 to 9), office timings (9 to 6), dinner and relaxation time (6 to 11) and sleeping time (11 to 7).We also used different time intervals (hold) between invocation of different actions for a device to simulate a natural timing difference between the use of devices. The above code manifests the pseudo code only for the break fast and wakeup timings.

We have developed the system using C#, .NET and Intel's UPnP SDK .Architecture of our SNMP agent with control point (Proxy agent) is shown in the figure 4. It includes both UPnP control point and SNMP agent, thus acting as a gateway between UPnP environment and SNMP manager. The translation code is demonstrated in figure 5. It is quite simple for both Receive-getRequest and Receive-getNextRequest. In case of Receive-setRequest,

after validating the OID , control is transferred to the Control Point (UPnPHandler), where it retrieves the UPnP record related to that OID and subscribe to the corresponding service. The Control is then transferred back to the Receive-setRequest, where appropriate entry is done in the MIB_DB. But, if control point couldn't subscribe to the device, then error message is sent to the manager.

After subscribing to different services, control point receives modification information in their state variables via a call back function, OnModifiedStateVar. In this function UPnP Record is saved in the MIB Db and if the trap value is set for the corresponding OID by the manager, then a response message is sent to the manager.

Figure 6 shows our SNMP manager, which could load MIB files and listen to SNMP traps and perform different SNMP commands and operations. It stores the listened data in a database and helps in performing different type of analysis

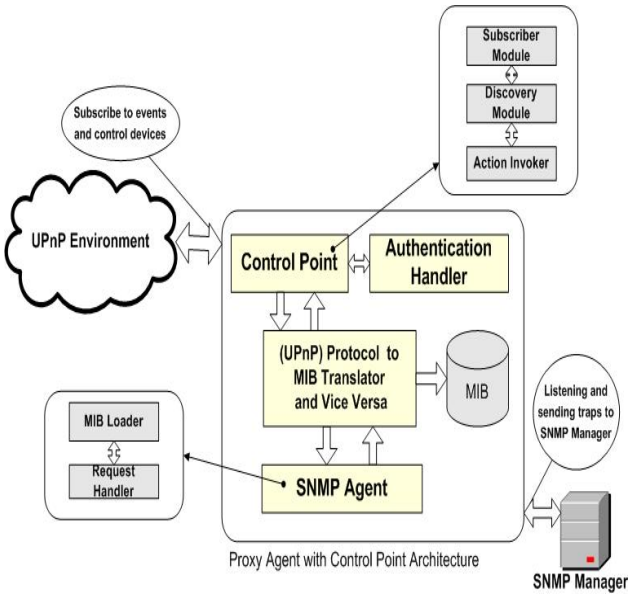

Fig.4. Gateway Architecture: Control Point with SNMP agent

6. Sample Application

Certain applications could be built on the top of this architecture. To check the effectiveness of our system, we decided to calculate the power consumption of different devices. Therefore, after the accumulation of data in our repository, we applied the following primitive billing equation to calculate the electricity bill based on the power consumption by different devices. The equations are given below. We assumed all the devices power consumption is already known and its unit is kilowatt.

Since Energy=Power*Time .

Electricity Cost= Energy*(Cost Per Kilo-watt-hour)

Let cost= $0.10 kwatt-hour

SNMP Or From SNMP to UPnP

```
Receive-getRequest:
   Step1: ResponseId=LookUpMIB_DB( objectIdentifier)// validate the Object
            //Identifier by looking it in the MIB DB
  Step 2: if ResponseId is NULL  or other Error Occurred
     Step 2.1 :  Issue getResponse( genErr(ResponseId),objectidentifier)
  Step 3 : else
      Step 3.1 :   Issue getResponse(value , objectidentifier)
Receive-getNextRequest:
  Step1:Objectidentifier= NextIdentifier(ObjectIdentifier)// Calculate
            // OID number of the next identifier
  Step2: ResposneId=LookUpMIB_DB( objectIdentifier)
  Step 2: if ResponseId is NULL or other Error Occurred
     Step 2.1 :   Issue getResponse(genErr(ResponseId),objectidentifier)
  Step 3 : else
      Step 3.1 :   Issue getResponse(value , objectidentifier)
Receive-setRequest:
   Step1: ResponseId=LookUpMIB_DB(objectIdentifier)// validate the Object
            //Identifier by looking it in the MIB DB
   Step 2: if ResposneId is NULL or other Error Occurred
     Step 2.1 :  Issue getResponse(genErr(ResponseId) ,objectidentifier)
   Step 3 : else
     Step3.1.ResponseId= UPnPHandler(objectIdentifier)// Call the
            // Control Point Handler
     Step3.2. if ResponseId==TRUE
       Step 3.2.1 :  Issue getResponse(value , objectidentifier)// Send to
            // Manager
       Step 3.2.2 :  Issue SetUpMIB_DB(objectIdentifier, value) //Save in
            // DB
     Step 3.3: else
       Step 3.3.1:Issue getResponse(genErr(ResponseId),objectIdentifier)

UPnPHandler
   Parameter: ObjectIdentifier
   Step1: UPnP_Record= LoopUpMIB_DB(objectIdentifier,"UPnP")// Overloaded
          // function to retrieve the UPnP record, such as device URL,
          //service name, whether Control Point Subscribed to it or not
          // and the UPnP state variable corresponding to the Object
          // Identifier. (Fields are same as shown in MIB).
   Step 2: If UPnP_Record==NULL return  NULL
   Step 3:  if (UPnPRecord.DeviceService.ServiceId.IsSubscribe == False)
       Step 3.1.1:  If (Subscribe_To_Device(UPnP_Record)==TRUE)
          Step 3.1.1.1:      return TRUE
          Step 3.1.1.2: else return NULL
   Step 4:  else return NULL
```

From UPnP to SNMP

```
OnModifiedStateVar
   Description: Call Back function

   Step1: SetUpMIB_DB(UPnpRecord,"UPnP")// Again,Overloaded function to
            //Save UPnP Record  in MIB DB
   Step 2: If (UPnPRecord.DeviceServcie.ServiceID.Trap)==TRUE// If
            // the Trap field is set , send the response to manager
       Step 2.1:  Issue getResponse(value , ObjectIdentifier)
```

Fig.5. Translation Code

Our system's current version doesn't provide any optimization suggestion i.e. how to optimally utilize the resources, but instead, it does provide the evaluation of resources. By using this one can easily get the idea which resource is used more and at which time and how to effectively use the system.

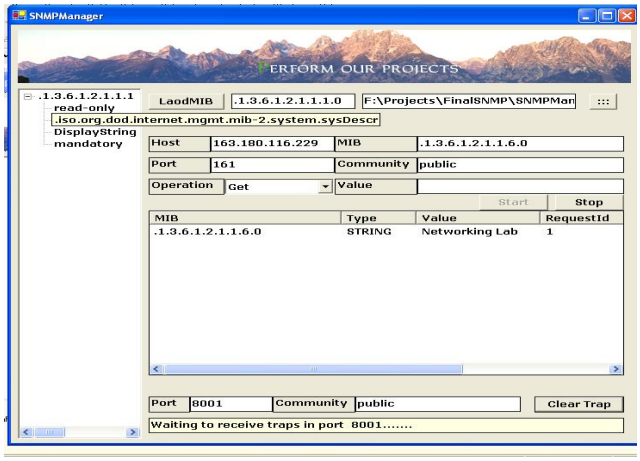

Fig.6. SNMP Manager

The power consumption graph for a single day for 4 different devices is shown in the figure 5 and the hourly power consumption of the system in a whole day is shown in the figure 6, along with the cost of electricity incurred in a day.

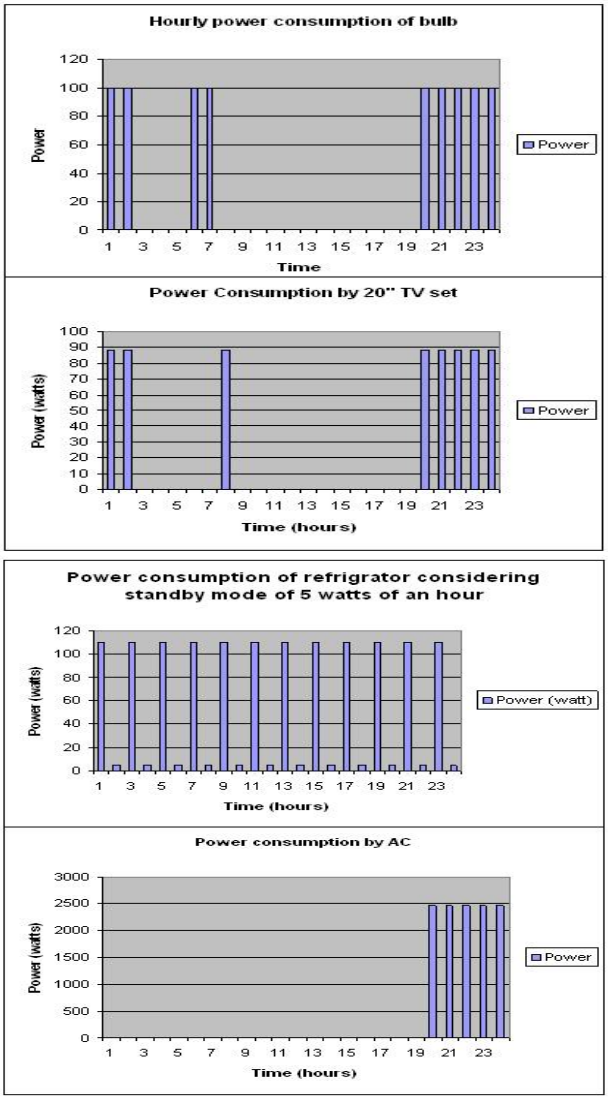

Fig.5. Power Consumption graphs for individual devices

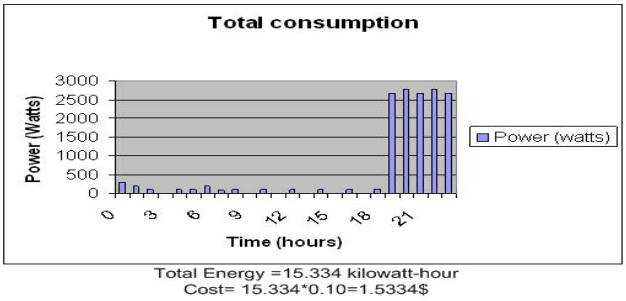

Fig.6. Total Power Consumption in a day

The scattered plot for showing the power consumption in thirty days is shown in the following figure. Using this data we could estimate the future power consumptions.

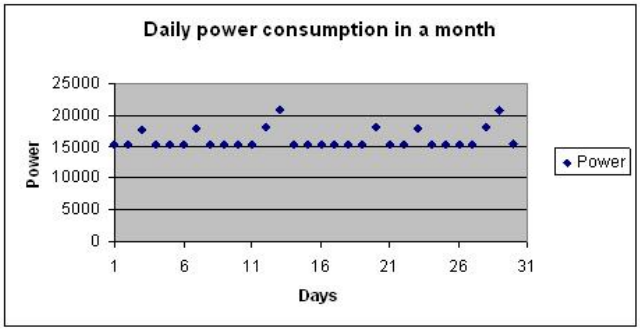

Fig.7. Scattered plot for daily power consumption

In order to do so we applied we applied the linear regression to the above data to calculate the cost for the next month.

Energy= 490.002 kWh
Cost= 490.002*.10= 49.0$

The above data can be extrapolated further to estimate the electricity bills of whole year and if any of the actual billing cost deviate from our estimated one then by using this we can detect abnormality or fault in devices like extra power consumption and can take steps to reduce the cost and consumption. The reason for applying regression analysis over here is to demonstrate one of the applications of our system. This type of analysis could be done without even using our system, but in case of large environments this would be far too much daunting task to individually analyze each device. Our system provides the centralized control to all the devices in the system either they are SNMP or UPnP based and let the administrator perform the management tasks and different analysis from one central location by using the same standard. Thus, providing the uniform interface and hiding the heterogeneity of the system.

7. Conclusion

We have observed that UPnP control point with the SNMP agent works quite well, but it consumes more memory, which is not suitable for small devices. In order

to demonstrate the viability of our system we demonstrated the simple scenario, but we are yet to analyze its result in a highly complicated environment, where many services move to and fro, in and out of the system.

We envisage that by using SNMP for management of UPnP based devices, we could easily manage heterogeneous UPnP and SNMP based devices from one single platform, particularly in environments with large number of devices, where use of shared resources is of concern.

References

[1] William Stallings, SNMP, SNMP v 2, SNMP v3, RMON1 and RMON2, third edition,1999.

[2] UPnP : www.upnp.org

[3]Ran Gladi, "SNMP for home automation", International Journal of network Management, 2004

[4] JINI:www.jini.org.

[5] Michael Jeronimo and Jack Weast, UPnP Design by Examples, A software developer's guide to Universal plug and play,2003

[6] MSDN, http://msdn.microsoft.com

[7] Liu hong, Bai dong, Ding Wei ,The integration of SNMP and Web in embedded devices, Info-tech and Info-net, Proceedings ,IEEE,2001.

[8] Frank Straub, Torsten Klie,"Towards XML oriented Internet Management", Integrated Network Management, 2003. IFIP/IEEE Eighth International Symposium on Integrated Network Management (IM 2003), March 24-28, 2003, Colorado Springs, USA

[9] Mi-Jung Choi*, Hong-Taek Ju** and James W. Hong*, Towards XML and SNMP Integrated Network Management, proceedings of Integrated Network Management, IFIP/IEEE ,2003

[10] Hong-Taek Ju; Mi-Jung Choi; Sehee Han; Yunjung Oh; Jeong-Hyuk Yoon; Hyojin Lee; Hong, J.W, "An embedded Web server architecture for XML based network management", proceedings of NOMS, IEEE/IFIP ,2002.

[11] Syed Shariyar Murtaza,Choong Seon Hong," A Conceptual Architecture for Unifrom Identification of Objects", Proc. of the Fourth Annual ACIS International Conference on Computer and Information Science (ICIS' 05), IEEE CS Press, 2005, pp 670-675.