ASP perspectives on Networked Appliances

Abhrajit Ghosh, David Marples and Stanley Moyer
Telcordia Technologies, Inc.
445 South Street
Morristown, NJ - 07960 USA

***Abstract*-ASPs today provide a means for businesses to outsource their applications thus saving on setup and maintenance costs. This paper presents an example of an ASP providing a service to a specialized networked device: a Networked Alarm Clock. This example is used to illustrate the issues ASPs would need to deal with to provide services to specialized networked devices within a home network.**

I. INTRODUCTION

An ASP (Application Service Provider) is a business entity that deals with the delivery and management of applications and computer services from remote data centers to multiple users via the Internet or a private network [1]. Application support can be provided via different means. An ASP can provide back end computing services, provide technical support, integrate with other applications, customize the application based on customer specific needs and meet availability requirements of the customer.

Current day ASP services target the PC centric desktop. As applications become more specialized based on user requirements, the PC oriented one-size fits all approach will not always be the best solution. Netpliance [2] for example provides a dedicated function web browser/email device that works well for users who do not need a full-blown PC for their communication needs.

Most ASPs have been focussing their efforts on commercial enterprises that wish to outsource applications to save on setup and maintenance costs. Today's domestic market is a strong target for future ASPs for various reasons. Among these is the increasing availability of always on broadband access to the home. Another driver is the fact that a variety of specialized home appliances with the ability to access communication networks have emerged recently.

While a PC based platform is highly customizable and programmable it is prone to errors and outages. It is not an attractive platform for an ASP when it is owned and operated by a domestic consumer for two principal reasons:

i) Several applications provided by different ASPs could interact and cause service delivery problems if run on the same PC platform.

ii) A PC based computing platform, while being very flexible and customizable, is typically too complex for the average consumer to operate and maintain.

Providing a specialized service to a specialized device (for example, a home appliance) is an attractive proposition since there is typically no sharing of multiple ASP applications at these devices. In addition, these devices have significantly lower configurability than a PC, making them less prone to user-related errors and outages. They provide a dedicated service, which can be supported by value added services provided by the ASP. Such an appliance would typically be a cheaper investment than a full-blown PC since the processing infrastructure would be minimal: the ASP would handle most of the computing and data access needs of the device. An ASP service provider of the future would thus be rendering a value-added service to a specialized device.

Since specialized devices would typically have limited computing power, processing would need to be offloaded from the device to the ASP's platforms as much as possible. There would be a trade off between network bandwidth accessible to the device and computing power available on the device. In some cases the home network may have additional computing support (e.g. a PC, a residential gateway) in which case some of the processing could be distributed here as well.

The ASP would need to be aware of the rendering capabilities of each device in terms of: display area sizes, display area type, audio capabilities, input capabilities (number of buttons available, for example) and so on. There would be a need for the ASP to customize parts of their application service on a per device basis. In some cases, the ASP would need to provide translation capabilities from one format to another. (e.g. Unified Messaging services where the same message can be picked up in either text or audio format depending on the type of accessing device.) Additionally, some ASPs may render their applications to not one but to a set of devices. In this case the ASP needs to coordinate the actions of these devices to provide a meaningful service to the user.

Most networked appliances would be intended for tasks specific to a certain user or to a certain set of users. It will thus be necessary for the ASP to maintain a profile of the user that would store service options requested by the user, alternative devices available for the same service to the user and the capabilities of these devices. Moreover, such profiles should lend themselves to easy configuration by the user. While many configuration tasks would be best performed via a web browser client running on a PC, some of the more frequent configuration tasks should be accomplishable via the appliance itself since it is probably

more accessible to the user than the PC running the web client. Alternative ASP provided configuration solutions could be based on IVR or audio response systems.

The ASP could act as an information service mediator since it could provide access to third party information or services depending on the preferences and capabilities specified in a user's profile. This capability would allow an ASP to outsource some of its functionality and handle Service Level Agreements with third party service providers on behalf of the user of the appliance.

Many of the issues mentioned above were identified based on our experiments with a Networked Alarm Clock appliance. This paper provides a brief overview of the Alarm Clock Service rendered to this device and illustrates the issues an ASP providing such a service would have to deal with. We next identify other types of services that could be provided by ASPs to various Networked Appliances and conclude with a brief discussion of the role of an ASP in this context.

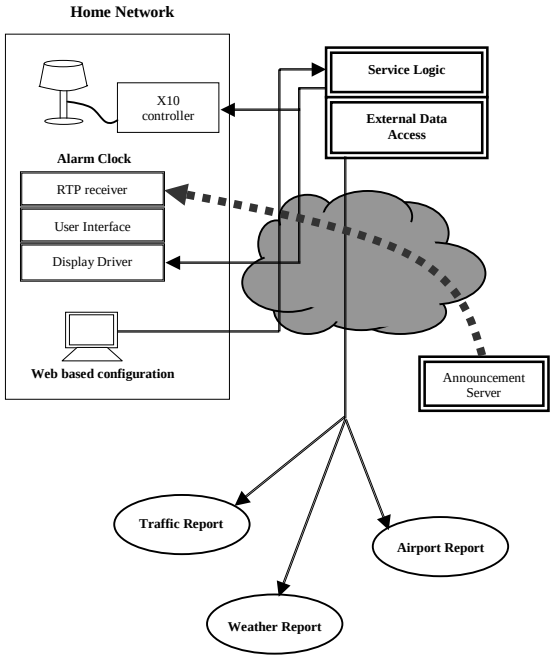

Fig. 1. Functional components

II. ALARM CLOCK SERVICE CASE STUDY

Our implementation of the Alarm Clock Service makes use of an Alarm Clock device with a network connection. It is described in more detail in [3]. The service logic resides on an HTTP server that is network accessible from the Alarm clock. An alarm can be configured to be dependent on a set of external factors. In our experiments, these external factors were one or more of weather conditions, traffic conditions and airline departure delays. Some time before an alarm is scheduled to sound the Alarm Clock device contacts the Service logic, which in turn contacts service providers that give information about the external factors pertinent to that alarm. Once the service logic for an alarm has the required information, it computes an adjustment (if required) for the current alarm time and sends the adjusted alarm time back to the Alarm Clock Device. Once the alarm actually sounds, an announcement is played to inform the user about the reasons for the adjustment of the alarm time. In addition, any ancillary devices (e.g. bedside lamp, coffee maker) are activated if so specified in the user profile. Our Alarm Clock service is thus a value-added service provided by an ASP to a specialized device.

A. Location of processing functions

Executing the Alarm Clock service requires 8 distinct functional components. (1) a Display Driver module that provides low level primitives for controlling the display of characters on the clock's LCD screen, (2) a User Interface (UI) module that uses the Display Driver to provide a menu based display to the alarm clock user, (3) a Service logic module that that is invoked via an HTTP server, (4) a set of scripts that is executed by the service logic module to retrieve and parse data from external sources, (5) an X10 based light/appliance controller used by the service logic to activate ancillary devices, (6) an Announcement Server that composes and streams (using RTP) announcements to the user in case an alarm time was adjusted, (7) RTP receiving capability on the user side and (8) a web based configuration interface for the clock and ancillary devices. Depending on the computing power and network bandwidth available to the clock, certain subsets of (1) through (8) could be placed in the clock or could reside on ASP operated platforms. In a typical configuration, the Display Driver, UI module and RTP receiving capability could be located on the Alarm Clock, the Service logic, scripts and the Announcement Server would be placed on the ASP side and the web configuration could be performed on a PC that could be on either side. Such a configuration is shown in Fig. 1. Alternately, the UI module could be on the ASP side if the Clock does not have enough processing power. A third variation would be to move the service logic on to the clock so that alarm time adjustments are computed locally while the information mediation is handled on the ASP platforms.

B. Rendering the same service to different devices

The Alarm clock device had a 4 x 20 character display and 3 buttons available for interacting with the user. Thus it was possible to have a menu based UI system. In

addition, it was possible to communicate with the service logic using HTTP from the Alarm Clock. We implemented a modified version of the Alarm Clock service for rendering to a pager. Since the pager we used had a much smaller display than the Alarm Clock and did not provide any buttons for user interaction we did away with the UI module and added an alarm proxy module that communicated with the pager via email and required configuration via a web page. This proxy module would need to reside on an ASP platform since the pager was not capable of hosting this application. Some of the service components implementing the Alarm Clock Service thus needed to be replaced but the back end service logic remained the same. An ASP providing this Alarm Clock service would thus need to customize some of its service components based on the type of device to which the service was being rendered.

C. Configuration and Customization

It is possible to configure alarm times using the Alarm Clock device via the UI module. However, more complex configurations need to be made via a Web based client since such tasks would be very cumbersome using the Alarm Clock's UI. Such complex tasks include the identification of ancillary devices available at the user's end, for example. An ASP would need to provide UI capabilities for tasks that are frequent in nature, at the device itself, but tasks that that are infrequent and cumbersome should be accessible via a web based client or some other means.

D. Single device accessing diverse services

The alarm clock service blends diverse network based applications to provide a coherent service to the user. These network-based services include different information sources (weather reports, traffic reports, airplane schedules), Service logic for alarm time adjustment and network support servers such as the Announcement Server. An ASP would need to establish SLAs with such information service providers so that they make the required information accessible in a well-known format.

A Network Support server like the Announcement Server is itself an ASP platform that provides a specialized functionality. Our implementation of the Service logic stores the information obtained from external service providers within the alarm clock device. When the alarm sounds this information is passed to the Announcement Server that in turn uses it to compose an announcement that is sent via an RTP stream to the user. Thus the Service logic needs to make sure that sensible information is passed to the Announcement Server for it to compose the appropriate announcement.

We have used the Alarm Clock Service to exemplify an ASP service. Various issues that an ASP would need to deal with when providing such a service were identified. The next section discusses other possible ASP services and issues regarding their execution.

III. OTHER ASP SERVICES

An ASP could provide an integrated network based security solution using various devices (e.g. motion sensors, microphones, speakers, cameras, and authentication devices). The ASP would be responsible for running the service logic using authentication databases that the user could configure. Such configuration could be done by means of a web client. The ASP could make use of an Announcement Server to generate audio announcements based on information collected via motion sensors and authentication devices. In this case the ASP would need to decide the amount of processing to be delegated to each device. For example, should the authentication device be able to access the authentication databases directly or should it just pass on the identity of a user to a back end server that would carry out the required database accesses and comparisons. Further, the playing of announcements would need to be a back end service since it may not be feasible to provide this capability at each speaker in the system. The entire back end of the service may reside on the ASP's platforms while the front-end devices could have minimal processing logic on them. This would make them cheaper to buy and easier to install and maintain.

Another ASP service to a networked device is an Internet based picture frame service. Such a service involves a network connected picture frame that is updated by an ASP's application. The ASP needs to maintain a database of users that is allowed to update the picture frame and resolve conflicts if they occur. The only processing required in the device itself is the rendering of the graphical images that arrive via the network. Since a picture frame would be likely to have a sizeable display area, the configuration and customization of the service could be doable from the device itself via a touch screen. Once again, the ASP runs the application logic. In fact, this service is a specialization of the Netpliance [2] device mentioned earlier in this paper.

IV. CONCLUSION

When providing application services to specialized devices the principal decision that needs to be made is that of distribution of processing functions. Processing functions could reside at the device, on a processor that is within the user's local network or on the ASP platform. Since a typical network device would prefer to dedicate most of its hardware to the specialized function that it performs, it would be preferable to locate most of the processing either in a user's local network (if it exists) or

on the ASP platform. An ASP would be responsible for maintaining the state of a device as well so that in case the device malfunctioned or was lost, another could easily (and cheaply) replace it.

For a mobile user it makes sense to not be dependent on a local network's processing power. The ASP would need to thus perform additional functions to track the location of the user and provide the service of choice. Most Networked Appliance services will benefit from wirelessness since many Networked Appliances are intended to be portable. The ASP service needs to deal with issues of lossy, low bandwidth connections to wireless devices. Very often the network itself can provide the required service by storing messages for retransmission after recovery from a connection loss or by providing compression services.

Services provided to a user may vary based on the location of the user. ASP services need to be location aware and need to work with Network layer services to determine geographical location as well as service capabilities and requirements at a particular location. In addition the ASP should store session states so that a mobile user can continue a suspended or interrupted session at a later time from a possibly different device.

Finally, secure access to networked devices will determine the consumer's willingness to accept an ASP service into their home. Security issues for home appliances have been dealt with in [4]. [4] Illustrates how SIP [5] could be used to securely access networked home appliances.

REFERENCES

[1] The ASP industry consortium FAQ. http://www.aspindustry.com/faqs.cfm

[2] http://www.netpliance.com

[3] Lessons from the internet alarm clock. Telcordia Technologies internal white paper. Moyer, S., Marples, D. and Ghosh, A.

[4] Framework draft for networked appliances using the Session Initiation Protocol. IETF draft. July 2000. Moyer, S., Marples, D., Tsang, S., Katz, J., Gurung, P., Cheng, T., Dutta, A, Schulzrinne, H.

[5] SIP: Session initiation protocol. RFC 2543. Handley, M., Schulzrinne, H., Schooler, E., Rosenberg, J.