About Home Gateway Mashups

Manfred Schneps-Schneppe, Dmitry Namiot

***Abstract*— This paper discusses Home Gateway Initiative software and telecom mashups. Can we use IMS for mashups and how to do that? What is impact of Home Gateway Initiative decisions to application developers and what can we expect to see on the application market for home devices.**

***Keywords*— home gateway, mashup, telecom, Asterisk.**

I. Introduction

The paper concerns several highly sophisticated areas. At the first hand it is Internet protocol multimedia subsystem (IMS). IMS is defined by 3GPP as a new subsystem, i.e., a new mobile network infrastructure that enables the convergence of data, speech, and mobile network technology over an IP–based infrastructure. IMS was designed to fill the gap between the existing traditional telecommunications technology and Internet technology. The architecture of IMS specifically helps enable and enhance real-time multimedia mobile services such as rich voice services, video telephony, messaging, conferencing, and push services. Although IMS was designed for mobile networks, it can also be used to provide services for fixed networks at the same time, providing unique mixtures of services with transparency for the end user.

At the second it is Home Gateways. Home gateway industry as a rich business area. It is developing now around Home Gateway Initiative (HGI) - a non-profit making organization created to define guidelines and specifications for broadband Home Gateways. Of course, home gateway includes telecom software. Most of the HGI operators are moving towards an IMS oriented architecture (as per HGI papers). But is that choice promising? At this moment most of the application developments in the web area and telecom area could be classified as mashups. Can we see mashups for Home Gateways too?

II. What is a mashup?

In Web development, a mashup is a Web page or application that uses and combines data, presentation or functionality from two or more sources to create new services (Fig. 1). The term implies easy, fast integration, frequently using open APIs (an interface implemented by a software program that enables it to interact with other software) and data sources to produce enriched results that were not necessarily the original reason for producing the raw source data.

Manuscript received Jun 24, 2013.

M. Schneps-Schneppe is with the Ventspils International Radio Astronomy Centre, Ventspils University College, Ventspils, Latvia (e-mail: manfreds.sneps@gmail.com).

D. Namiot is with the Faculty of Computational Mathematics and Cybernetics Lomonosov Moscow State University, Moscow, Russia (email: dnamiot@gmail.com).

The emerging Web 2.0 marketplace presents an important opportunity for Telecom operators to sell their own capabilities and content as services. Operators have a wealth of content associated with their network as well as core network enablers, e.g. call control, presence and messaging, which could serve as potential new revenue streams in a Web 2.0 world. Moreover, with the looming threat from Internet companies, there is an increasing need for operators to make both core and value-added functions reusable and mashable [1].

There are basic things. From the practical point of view mashup (telecom mashup particularly) means some combination of applications. For the telecom mashups it means that we will not or, in the most cases, cannot develop the whole application as a monolithic system. We can count several reasons for that. It could be a complexity of development, lack of the proper description, constantly changed business requirements etc.

In the applications development mashups plays the same role as components on the low level programming. With mashups development we are assembling our system from the individual blocks. We can treat that blocks as programming components but usually, especially in the telecom development they have a much bigger level of integration.

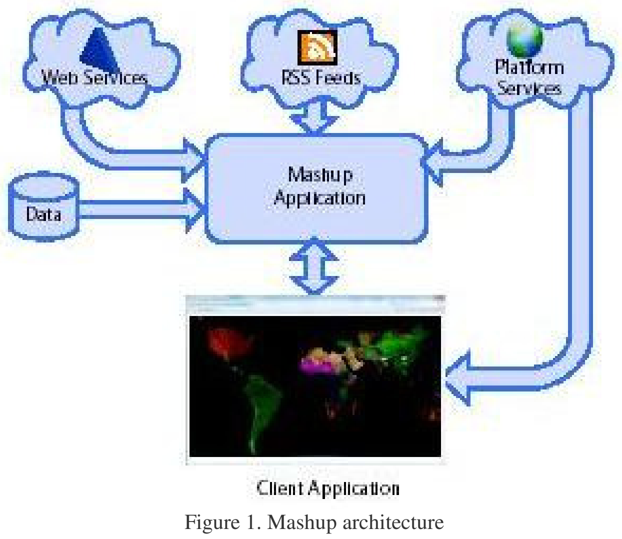

Figure 1. Mashup architecture

To be able to permanently access the data of other services, mashups are generally client-side based applications or hosted online. In the past years, more and more Web applications have published APIs that enable software developers to easily integrate data and functions instead of building them by themselves. Mashups can be considered to have an active role in the evolution of social software and Web 2.0. Mashups composition tools are usually simple enough to be used by end-users. They generally do not require programming skills, they rather support visual wiring of GUI widgets, services and components together. Therefore, these tools contribute to a new vision of the Web, where users are able to contribute.

The typical architecture of a mashup is divided into three layers [5]:

- Presentation / User interaction: this is the user interface of mashups. The technologies used are HTML/XHTML, CSS, Javascript, Asynchronous Javascript and Xml (Ajax).
- Web Services: the products functionality can be accessed using the API services. The technologies used are XMLHTTPRequest, XML-RPC, JSON-RPC, SOAP, REST.
- Data: Handling the data like sending, storing and receiving. The technologies used are XML, JSON, KML etc.

It is worthy to mention the Open Mashup Alliance (OMA) [6]. It is a consortium of individuals and organizations dedicated to the successful use of Enterprise Mashup. technologies and adoption of an open language that promotes Enterprise Mashup interoperability and portability. The best known members of OMA are HP and Intel. Enterprise Mashups combine and remix data from databases, spreadsheets, websites, Web Services, RSS/Atom feeds, and unstructured sources that deliver actionable information for better decision-making. The Open Mashup Alliance has been chartered to steward an open, free-to-use Enterprise Mashup Markup Language (EMML) that can reduce the risk and cost of enterprise mashup implementations, improve mashup portability of mashup designs, and increase the interoperability of mashup solutions.

III. TELECOM MASHUPS AND IMS

According to the HGI papers [7], Home Gateway should be IMS based. The IMS enabler should support services based on an IMS platform and interoperate with the Next Generation Network (NGN). IMS, which abstracts the service level control functionalities in a network and makes application development easier, is starting to help Telecom Mashup development. IMS defines the Application Server (AS) in its service architecture to provide the value-added services. The AS's in the network represent capabilities, which are system components that are used presumably with other components (e.g., content servers) to implement a service to the user. The signalling protocol SIP is the IMS Service Control interface used between the core network Call Session Control Function (CSCF) and the service capabilities implemented in the AS's.

Figure 2 shows the various blocks described below.

The main sub-block is IMS Interworking, mapping external IMS messages to internal (home) non-IMS devices and vice versa. It is based on the signaling protocol SIP.

SIP proxy/server supports local registration of SIP devices (it includes the SIP registrar function) and proxy functionalities in cooperation with the identity management block.

Identity management maps the IMS identity stored in the HG to a local identity. Devices should first register locally with the HG and then the IMS core.

Device management stores device capabilities on the HG. This function can be implemented using protocols like SIP or UPnP.

Self provisioning enables the user to modify some service configurations through a (Web) User Interface (UI) on LAN side, at the Application Server (IMS).

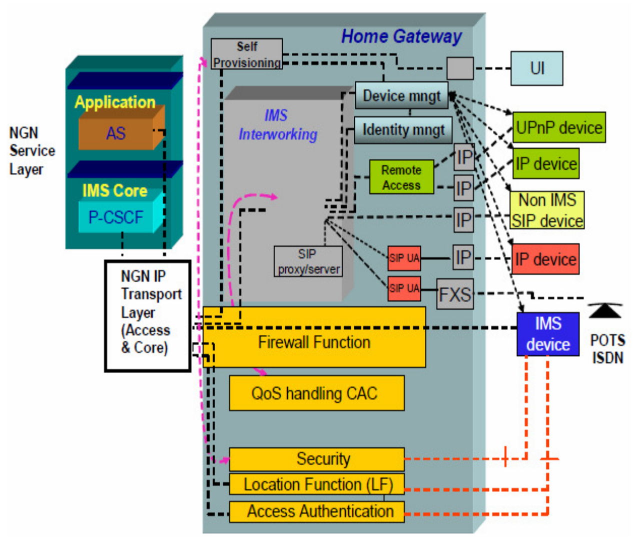

Figure 2. IMS internetworking

Unfortunately, the decision of telecom operators to deploy IMS is more a strategic decision than a technological decision [9]. The end-to-end model adopted in IMS introduces several technical challenges, for example concerning QoS, privacy and billing. The main technological issue is related to interoperability. IMS mixes the points of view of IP, wireline telephony and mobile network operators. Moreover, it introduces new networking paradigms and provides specifications, not implementation-ready solutions. Finally, it uses some recent protocols like Diameter that have not been widely deployed. For all these reasons, interoperability may be difficult to achieve in IMS networks.

One of the main motivations for IMS is to enable the delivery of real time multimedia services using IP related technologies, but IMS has to manage the different access related constraints imposed by heterogeneous access technologies (e.g. handover in radio access networks). In particular, this makes the establishment of end-to-end QoS guarantees quite difficult. Moreover, mobile wireless devices have limited functions, are usually related to a single user and are controlled by the network operator, whereas fixed wired terminals are powerful and controlled by the user. These differences have to be taken into account for authentication (e.g. using SIM-based secure access) and security related operations, for example. IMS aims at providing multimedia services with unified network architecture. However, IMS does not yet include several promising technologies (e.g. P2P, VPNs, SMS and IPTV).

Additional issues are currently or need to be addressed in the standards, for example: universal service obligation, number planning, lawful intercept, number portability, reliability and voice quality, emergency services, inter-carrier compensation and data protection [8].

IV. Telecom mashups and technical issues

Now let us ask the simple question? Is there any valuable set of telecom mashups based on IMS? This technology is not new already and technically we should see some visible development in this area. But we can not. Why? The same issue has happened as with Parlay. Over engineered technology stack, lack of the implementations, lack of the web protocols. See the picture above? Can you see web there? After being widely proclaimed as the telecom industry's Next Big Thing, the immutable force of gravity has pulled IMS back down to earth a bit. IMS continues to develop, but at a painfully slow pace, considering the rapid changes going on outside the telecom world [9].

There's general agreement that SIP is a good, accessible protocol for redeveloping core telco services, as BT and other leading operators are doing. But IMS SIP is not, on its own, enough to enable an operator to deliver a rich set of revenue-generating services to the market. While each telco core service has an intrinsic value, operators will make more money out of them in a next-generation IP world if they can blend their core services in innovative ways with other functions to create value-added services. In other words, they need to make these SIP-based core services available for mashups.

Unfortunately, traditional telco application vendors are building a next generation of SIP-based applications in the "black box," replicating the manner that has characterized the telco industry for so many years. Ironically, it was this stovepipe mentality that IMS was intended to tear down [10]. Each black-box application has the vendor's own instance of a core service locked inside it. The core service has been implemented using IMS standards and protocols, and it runs on a SIP application server. But it is not available to anyone else for mashup, which restricts what an operator can do with the core service once the service is deployed.

Our own experience can only conform that. We can mention also rapidly growing cloud telephony API's like Twilio [11]. Twilio allows your web application to easily make and receive phone calls and SMS text messages using a simple web service API and basic web programming. That is the key issue - well defined functionality: voice call becomes reusable and mashable via the standard web programming API. No new paradigm for the web developers, a lot of development tools, fast deployment in the various hosts, etc.

V. Telecom services and Home Automation

Now let us describe the place for the telecom services within the Home Automation projects. We can highlight two possible areas: voice responses - e.g. using text to speech service and explain measured data by voice as an answer to user's call and opposite task – using call as a switch on/switch off command for our equipment. From our vision it covers all the areas (the rest is simply a combination of two methods).

As the next step let us describe our vision to computer telephony and appropriate services. Technically we are speaking of course about one simple thing – the ability to catch the call and process it programmatically. Actually, it is all. Let us see what we can do next. By our vision any (ok, almost any, to be polite) telecom service could be presented as a combination of the finite set of basic services. Let us count them:

1) application accepts a call and hangup. For example all the voting services (or switch in/switch off) services look so. Just get a call (get A-number), do some action on the server side and drop a call;
2) call redirection. Applications accepts a call, get a new number by the own (e.g. some database lookup) and redirects the incoming call to the new destination;
3) media play. Just got a call and play some media file (static or dynamic) in the answer
4) record media file. Just got a call and write the voice to some file
5) DTMF recognition. Just get and recognize tone signals from the line

On practice, it is all do we need. The vast majority of the services are actually just a combination of the above 5. It is not a big list actually, so we may expect that we will be able to pickup an appropriate development tools for them. And telephony API's like Twilio are perfect fit for this. There is simply no need (read – no place) for IMS.

Another big issue in this context is a practical domination of Web APIs in the modern world. There are several our papers devoted to the integration web and telecom [12, 13]. System based aspects of mashup covered in our paper devoted to Smart Cities software [14].

VI. Open source telecom gateways for Home Automation

Here we can mention our own development – telecom web API for Asterisk [3]. Historically, by the way, it was a first implementation of web API's for telecom.

As a basic telecom-enablement tool for the Home automation projects we choose Asterisk. Asterisk is software that turns an ordinary computer into a voice communications server. Asterisk powers IP PBX systems, VoIP gateways, conference servers and more. It is used by small businesses, large businesses, call centers, carriers and governments worldwide. Asterisk is free and open source.

As per official site Asterisk is often referred to as "the open source PBX" and it's true that you can use Asterisk to build a PBX. But a PBX is only one of many applications you can build with Asterisk. Asterisk also is gateway, voice mail, IVR etc. One big advantage for the production is the fact that Asterisk is quite popular, so it is not a niche solution used with one or two projects only.

The way we've decided to go is a new layer in the software development architecture. We've suggested a new layer for Asterisk development. But instead of introducing our own API we've decided to go with HTTP. In other words we've created HTTP gateway, that lets developers talk with Asterisk via HTTP requests/responses. And Asterisk related applications (scripts at the original) becomes in this model just ordinary CGI scripts. All the complexity for the above mentioned API's, protocols, managers, etc. is hided within our gateway. Application developer now is an ordinary web developer. He/she can simply see a list parameters for incoming request (parameters in HTTP request) as well as the format (list of the parameters) for the response.

We propose to integrate a new component (proxy) into the Asterisk platform. The main functionality of the proxy is to translate telecommunication calls into HTTP requests to external web services. Telecommunication services are located separately from the PBX, while the information they receive from Asterisk is presented as a HTTP-request.

Technically, HTTP GET/POST request is a request, in which external telecommunications service passed information about the subscriber's name - CallerIdName, caller's number - CallerIdNumber and called number - Extension. Upon receiving necessary parameters, such as (calling/called number) a web service produces and forwards its instructions to the proxy. The latter receives and translates them into Asterisk instructions. The development of such services under the architecture described above is similar to a conventional CGI-script, for which there is a plenty of programming tools. As a result a programmer doesn't need to be familiar with the Asterisk API. The following picture illustrates the common schema (Fig. 3).

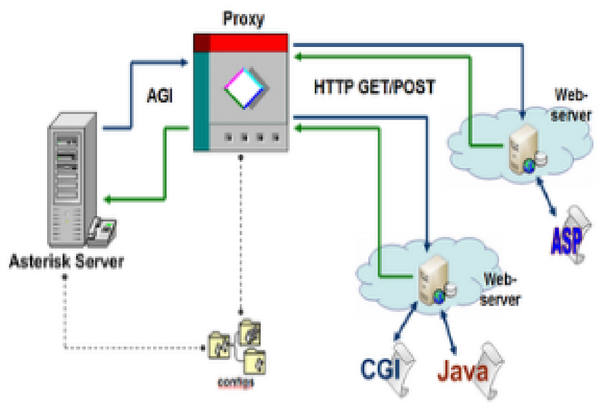

Figure 3. Asterisk gateway

As seen, the AGI-proxy component is at the heart of the model. The AGI-proxy is a Java-based application implemented on the basis of FastAGI, an open source library. The AGI-proxy is installed directly on the PBX Asterisk side.

In fact, the Asterisk represents the same thing for the AGI-proxy, as the J2EE container does for a Java-servlet. All calling messages produced by the Asterisk come to the proxy within the method service with two parameters: interfaces AgiRequest and AgiChannel. Through the first parameter one can get the information about the calls (caller name / number, called number, context of the call, feed settings, etc.) and through the second one the interaction with the Asterisk is carried out (call termination, transfer mode and etc.).

Subscriber's name (CallerIdName), caller's number (CallerIdNumber) and called number (Extension) are wrapped within a string-parameter by the AGI-proxy. Then the AGI-proxy executes an HTTP request on its behalf to an external web service, whose URL is set in the configuration. The response from the invoked service is seen as an indication to what to do with the call. The call may be terminated or redirected to a specified number; a media file may be played etc. Under this architecture the sequence of calls is reduced to a very simple diagram (Fig. 4).

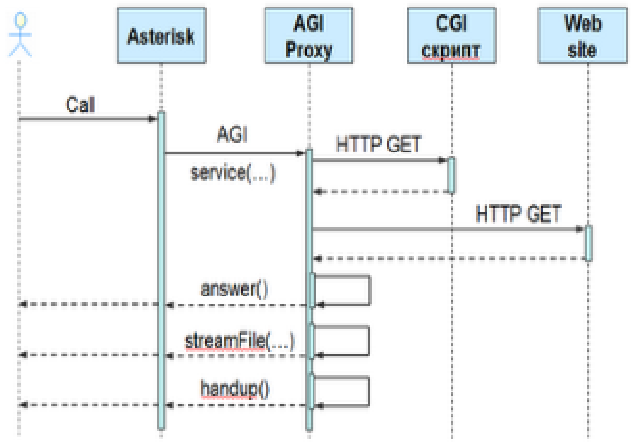

Figure 4. Call flow

Source code and files necessary to install the demonstration examples are available via Google Code [15]. It is a part of Wired Smart Home architecture [16].

Why it is good for Home gateway? Asterisk is open source PBX and we could add it as a telecom part of Home gateway. For example, one of the most used products in this area (Transwitch Atlanta 2000) follows this way. The telecom part of their product is Asterisk [17]. And as we noted above, they do not need IMS stack for services.

VII. Conclusion

This paper provides a quick analysis of the software offering for the Home Gateways, compares IMS suggestions with web mashups and proposes an alternative way for developing telecom services for home gateways: telecom mashups based on the existing web technologies. This new approaches simplify the development and deployment for new services.

REFERENCES

[1] Banerjee N., Dasgupta K., Mukherjea S., "Providing Middleware Support for the Control And Co-ordination of Telecom Mashups", In Proceedings of Middleware Workshop on Next-Generation Converged Networks and Applications (MNCNA), Nov. 2007.

[2] Roberts S. Essential Jtapi. – Prentice-Hall, Inc., 1998.

[3] “Web gate for Asterisk”. Available: http://asterisk.linkstore.ru/

[4] Clarkin, L., Holmes J., “Enterprise Mashups”. Available: http://msdn.microsoft.com/en-us/architecture/bb906060

[5] Koschmider, Agnes, Victoria Torres, and Vicente Pelechano. "Elucidating the mashup hype: Definition, challenges, methodical guide and tools for mashups." Proceedings of the 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web at WWW. 2009.

[6] http://www.openmashup.org/

[7] http://www.homegatewayinitiative.org/

[8] Ericsson, “IMS - The value of using the IMS architecture,” Ericsson Tech. Rep., 2004.

[9] Bertrand, Gilles. "The IP Multimedia Subsystem in Next Generation Networks." Network, Multimedia and Security department (RSM)-GET/ENST Bretagne (2007).

[10] http://www.heavyreading.com/servsoftware/document.asp?doc_id=125310/

[11] Chudnovskyy, O., Gebhardt, H., Weinhold, F., & Gaedke, M. (2011). Business Process Integration using Telco Mashups. Procedia Computer Science, 5, 677-680.

[12] M. Sneps-Sneppe and D.Namiot. (2012, April). About M2M standards and their possible extensions. In Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on (pp. 187-193). IEEE. DOI: 10.1109/BCFIC.2012.6218001

[13] M.Sneps-Sneppe and D.Namiot. (2012, April). About M2M standards. M2M and Open API. In ICDT 2012, The Seventh International Conference on Digital Telecommunications (pp. 111-117).

[14] D. Namiot and M.Schneps-Schneppe. (2013). Smart Cities Software from the developer's point of view. arXiv preprint arXiv:1303.7115.

[15] Asterisk Web Gate code: http://code.google.com/p/asterisk-web-gate/.

[16] Schneps-Schneppe, M., Maximenko, A., Namiot, D., & Malov, D. (2012, October). Wired Smart Home: energy metering, security, and emergency issues. In Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on (pp. 405-410). IEEE. DOI: 10.1109/ICUMT.2012.6459700

[17] Transwitch: http://www.transwitch.com/products/product/page.jsp?product=190/