Efficient Residential Consumer Device Interaction With Network Services

Eric William Burger, *Senior Member, IEEE* and Ophir Frieder, *Fellow, IEEE*

Abstract — *Consumer access networks are low bandwidth, especially in the up-stream direction. This asymmetry is heightened for real-time multimedia applications. The web wait is not acceptable when in a conversational interface. We describe the use of KPML, a protocol for transporting user stimulus. We illustrate the application of KPML to a typical consumer activity, accessing information sources over the telephone. We include results measuring KPML against other protocols for transporting user stimulus.*

Index Terms — Consumer network services, tele-control, telephony.

VII. Introduction

Consumers access the Internet through various transport mechanisms. Some regions have advanced, high-speed access networks, such as Japan and Korea, where consumers commonly have multi-megabit per second downlinks and megabit per second (Mb/s) uplinks. However, much of the world has significantly less consumer bandwidth. Common digital subscriber line (DSL) speeds in the U.S., for example, are at best on the order of a Mb/s downlink and tens of Kb/s uplinks.

There has been a steady introduction of new consumer voice services using the Internet Protocol (VoIP). These services allow consumers to use their existing home telephone wiring and devices by using an Integrated Access Device (IAD) to gateway the consumer's telephones to the IP network. Consumers think of their telephones as operating as they always had on the traditional switched telephone network. They dial numbers and receive calls just as they always have done.

A popular use of a telephone is to access consumer information services, such as banking-by-phone, traffic and weather information, and the purchasing of goods. These services use interactive voice response (IVR) systems. Consumers interact with these systems by pressing the number buttons on the telephone keypad. Pressing a button on the telephone keypad generates a dual-tone multi-frequency (DTMF) signal that the phone transmits in the voice, or bearer channel established from the phone to the IVR system. A common term for such user-entered information is user stimulus.

Consumer VoIP introduces new problems with transporting DTMF signals in the bearer channel. Unlike VoIP on a LAN, there are a number of constraints imposed by the consumer environment. Because of the limited bandwidth of the upstream connection from the consumer's residence, IAD's often compress the voice. A common coder/decoder (codec) for consumer voice compression is G.729. However, G.729 distorts DTMF such that the receiver of the stream cannot reconstruct the DTMF tones. Named tone packets, as defined by RFC 4733 [1] address G.729's loss of DTMF. Rather than sending packets representing the waveform generated by the DTMF, such as the combination of a 697 Hz and 1209 Hz sine wave to represent the "1" key [2], named tones send packets in the bearer channel that indicate the name of the key pressed, such as "1", "2", "*", and so on. RFC 4733 sends packets in the bearer channel, as its design goal was simply to transport audio between two endpoints, particularly when using a codec that cannot transmit DTMF. The expectation is the receiving endpoint regenerates the DTMF. However, as we demonstrate, RFC 4733 is inherently unreliable and is not bandwidth efficient. Our concern is with consumers transmitting key press commands from their residential phones to servers in the network. As noted above, bandwidth efficiency is important for typical consumer situations.

Another issue is that the user key press entry is not audio transport, but signaling. The existing VoIP transport protocols transport voice from one endpoint to another. For example, the common protocol for transporting voice and video is the Real Time Protocol (RTP) [3]. Design trade-offs made by RTP include leveraging the human ability to interpolate audio and to favor consistent packet inter-arrival times over reliable packet delivery. Unreliable packet delivery is acceptable, and often required to meet the human aural requirements for relatively consistent packet inter-arrival times. However, losing signaling is not acceptable for many IVR applications. For example, when a consumer presses a button on their keypad indicating the execution of a transaction, it is not acceptable for the network to lose the key press.

While the bearer channel is unreliable, the signaling channel is reliable. As hinted above, DTMF entered by the consumer is, in fact signaling. Thus, it is most appropriate to transmit the DTMF signaling in the signaling channel.

One possibility is to have the IAD detect the DTMF digits and transmit the signaling in the signaling path. A method for this is to send the key presses as H.245 UserInputIndication (a signaling packet that indicates a DTMF key entered) [4]. However, such messages are on a per-key-press basis, which wastes network resources and precious processing resources at the IAD.

VIII. Network Services Topology

In Fig. 1, we show a typical network services topology. The consumer's conventional phone connects to the IP network through an IAD. For an IVR service call, the network routes the call to an IVR platform. The media gateway performs a

E. W. Burger was with Cantata Technology, Inc., Salem, NH 03031 USA. He is now with BEA Systems, Inc., Burlington, MA 01803 USA. (e-mail: eric.burger@bea.com).

O. Frieder is with the Illinois Institute of Technology, Chicago, IL 60616 USA (e-mail: ophir@ir.iit.edu)

Manuscript received January 12, 2007

similar function to the IAD, converting the VoIP packets into the Time Domain Multiplex (TDM) realm, also known as the switched telephone network. Legacy IVR platforms, which connect to the switched network, connect to the VoIP network through a media gateway. We will focus on the interaction with an IP IVR server, which performs the functions of the traditional IVR platform, but with native VoIP support. Once the network routes the call to the IVR platform, the IVR platform answers the call and responds to the caller's requests.

In VoIP, the bearer channel, where the actual media (e.g., voice) traverses the network, uses the real-time protocol, or RTP. RTP can transport the actual tones for DTMF. In addition, in VoIP, one can send named tones (RFC 4733) where the codec distorts the DTMF waveforms.

One might consider RFC 4733 packets to be signaling, rather than media. However, RFC 4733 packets have a number of drawbacks when used for signaling. First, the packets physically travel in the bearer stream. This means that any application that has interest in the signaling must be in the bearer path. In the analog world, this was not a problem, as the only way to detect DTMF was to establish a bearer-channel connection to the phone, usually over the public telephone network. However, it is an excessive burden to place on an application to have to terminate and interpret the bearer channel if all it is looking for is key press user input.

A second drawback of using RFC 4733 is that RTP, the transport RFC 4733 uses, is not a reliable delivery mechanism. RTP trades off reliable delivery for real-time stream delivery, where it is more important that the packets come at regular (or predictable) intervals, even if that means dropping a packet. Whereas this is an acceptable trade-off for the delivery of multimedia streams, it is not acceptable for the delivery of signaling.

In our system, we make only a signaling connection from the IAD to the IVR platform. This has a number of advantages. First, the IAD directly tells the IVR platform the exact key pressed. Thus only the signaling need traverse the access network. Given the typical rate of packets for signaling are on the order of 1 packet/second and the typical rate of packets for voice media are on the order of 50 packets/second, this is a significant savings. In addition, there are well-known problems with establishing media connections through a network address translator [5], or NAT, which most residential network gateways and IADs are. For example, many signaling protocols, including the Session Initiation Protocol (SIP) [6], insert the IP address of the media endpoint into some signaling messages. However, a NAT silently changes the endpoint's IP address, making the address in the signaling incorrect. Moreover, there are many network services scenarios where multiple applications wish to receive the signaling from the phone. Transmitting DTMF in the bearer channel means that all applications must either receive and relay the bearer channel to the next application, which means the applications have to be aware of each other, or the IAD must send multiple streams over the bandwidth-constrained consumer uplink. Transmitting the user input in the signaling layer eliminates these problems.

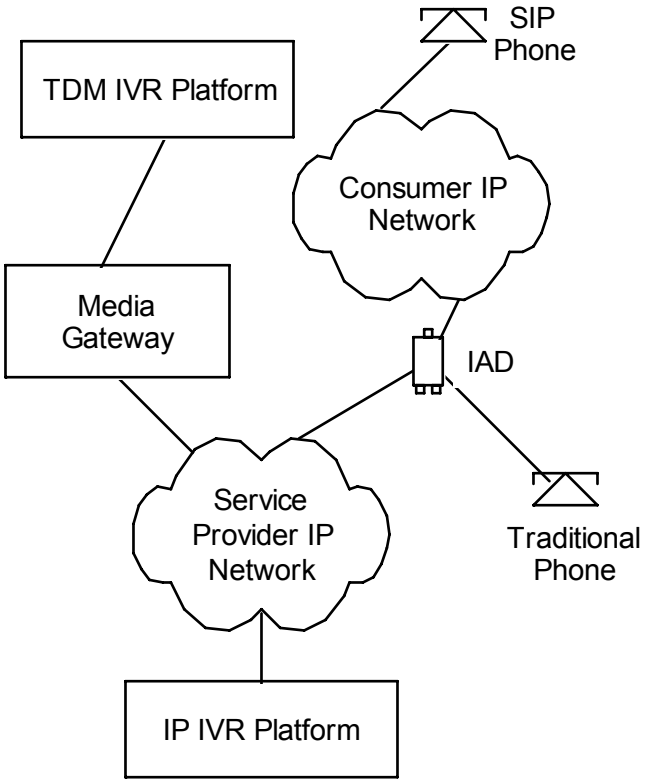

Figure 1 - Network Services Topology

IX. KPML

IADs use the KPML protocol [7] to transmit key presses in the signaling layer. The model of KPML is the IVR platform subscribes to the key press state of the phone. The IAD reports key press events. KPML enables the IVR platform to request the IAD to look for specific key press patterns, and the event report can indicate which pattern matched. The IAD monitors the analog bearer channel to the phone, performing signal processing on the channel to detect and interpret DTMF tones. The following is a brief overview of KPML.

There are three primary goals of KPML. The first is to present a compact, application-level representation to reduce the processing burden of application clients. This is particularly important for consumer electronic devices. The second is to reduce the number of messages required to transfer application-level state. The third is to reduce network traffic and reduce the application processing burden by sending messages only to applications interested in a given user input pattern.

Applications send subscription messages to a device. The subscription message for each application identifies a pattern of user input for the device to notify the application. KPML uses the SIP SUBSCRIBE/NOTIFY mechanism [8]. This mechanism provides a means for handling multiple, independent requests.

The device monitors input from the user to identify the occurrence of the patterns identified in the subscription messages. When the pattern occurs, the device notifies the corresponding application. The device only notifies the particular application that provided the subscription message, conserving processing and communications resources.

Subscription messages can also contain tags associated with patterns. When the device detects a match and reports it to the application, the device also returns the tag, enabling the application to determine easily exactly what response to the input is appropriate without needing to maintain a large amount of internal state information.

Some applications care to monitor the stream continuously for a particular pattern. However, other applications look for only a single occurrence of a particular pattern, at which time the application finishes monitoring or may register a different set of patterns. The first type of request is a "persistent" request, whereas the second type is a "one-shot" request. KPML provides for the requesting application to specify the nature (persistent or one-shot) of the request.

Indicating the nature of the request in the subscription reduces the protocol overhead for terminating the subscription. For many interactions, the subscription termination messages may be a significant portion of the overall number of messages and bytes transferred.

Fig. 2 shows a sample KPML request from the IVR platform to the IAD. Here the key sequence 1 indicates a command to get movie information and 2 indicates a command to get banking information.

By using a tag, one can abstract the control program from the actual interface. That is, rather than have IVR code looking for the string "1", the code looks for the tag "movies", as depicted in Fig. 3. This makes it easier to introduce new modalities, such as HTML on a personal digital assistant (PDA), with only minor changes to the control program.

Note the IAD will notify the controller only when the user enters a matching pattern. This is useful if multiple applications are looking for different patterns. In addition, it reduces network traffic by eliminating the sending of signaling packets to uninterested applications.

```
<?xml version="1.0" encoding="UTF-8"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-
request"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
 xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-
request kpml-request.xsd"
 version="1.0">
    <pattern>
        <regex tag="movies">1</regex>
        <regex tag="banking">2</regex>
    </pattern>
</kpml-request>
```

Figure 2 - KPML Request

```
<?xml version="1.0" encoding="UTF-8"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-
response"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
 xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-
response kpml-response.xsd"
 version="1.0" code="200" text="OK"
 digits="1" tag="movies"/>
```

Figure 3 - KPML Response

X. Protocol Comparison

We have implemented KPML in a network IP gateway [9] , and others are implementing it in a SIP Phone, IAD, and network media gateway. Using KPML resulted in significant bandwidth and packet reduction. In addition, KPML enables one to build low power consumer SIP Phones that do not require elaborate browsers or displays to access legacy IVR systems. At the same time, the KPML application model makes it easier to build network applications that offer both DTMF interfaces to phones and rich, graphical user interfaces to PCs and PDAs.

As an example of the efficiencies of KPML, consider a typical consumer retrieving information from an IVR service. Here the consumer calls the service, by dialing a phone number or entering a SIP URI that the network routes to the IVR platform. The service we examine here is a local services application. The service will give the caller the location of the nearest bank or movie theater, with what is playing at the movie theater if that is the consumer's choice.

In the following analysis, we consider the approach of transmitting the user's input as in-band tones and as named tones, compared to the KPML approach.

A. Call Flow

In Fig. 4, we illustrate the general call flow. First there is a call from a IAD to the IVR platform. Here the user calls the IVR platform. This can occur by translations from a dialed number. In this case, the user dials a number, like 800 555 1212, which routes over the PSTN to the gateway, which then maps the number to the SIP address of the controller. Instead of calling over the PSTN, the user could connect directly to the controller from a SIP phone by entering the Request-URI (address) of the controller. Likewise, an IAD can do the mapping from a number to the controller Request-URI. In this example, the Request-URI is "sip:service@ivr.example.com". Details of the session establishment protocol are in the SIP specification, RFC 3261.

In all scenarios, one establishes the session with the SIP handshake in messages **(401)**, **(402)**, and **(403)**. The handshake establishes the session by exchanging and negotiating capabilities.

The IVR platform then plays a prompt **(404)**, asking the consumer what they would like information for: movie listings or bank locations. The consumer enters a "1" for movies and a "2" for bank locations. In our call flow, the consumer enters "1" for movies **(405)**. At this point, the IVR platform asks the user for their ZIP Code **(406)**. The consumer enters their ZIP Code **(407)**. The IVR platform responds with the nearest theater, as well as what is playing there **(408)**. Then the IVR platform asks the consumer for another request **(409)**. The consumer wants bank locations, so they enter a "2" **(410)**. The IVR platform asks for the consumer's ZIP Code **(411)**, which the consumer enters **(412)**. The IVR platform reads the locations of the nearest bank branches **(413)**. The consumer hangs up **(414)**, which the IVR platform acknowledges **(415)**.

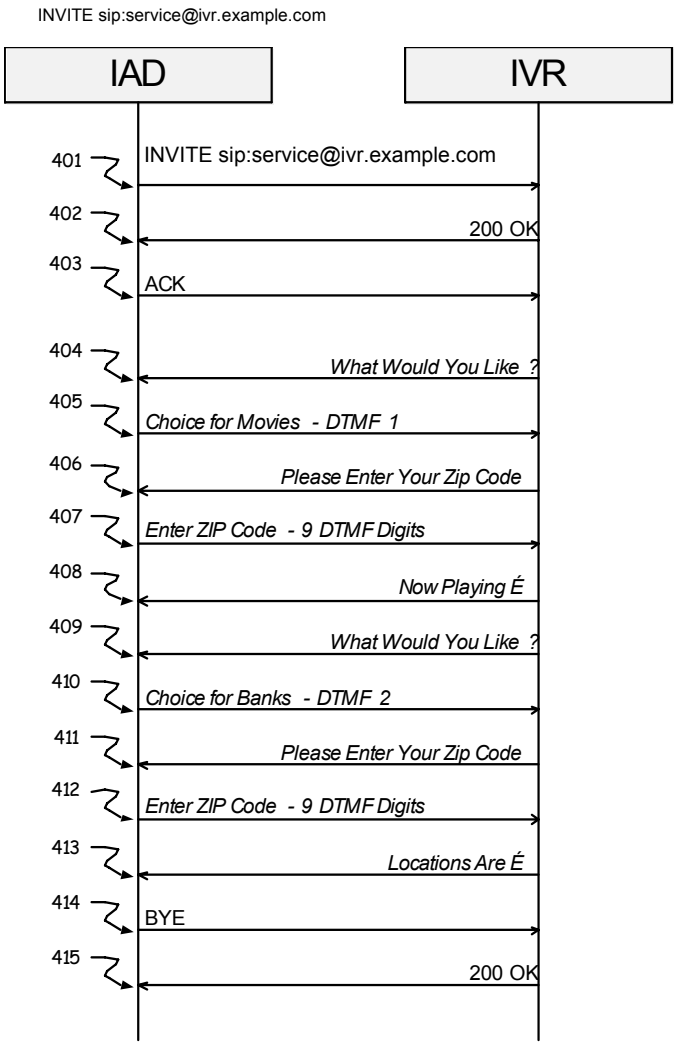

Figure 4 - Call Flow

B. Named Tones

For the named tones case, the digit signaling **(405)**, **(407)**, **(410)**, and **(412)** from Fig. 4 become expanded by the redundancy protocol of RFC 4733. The redundancy protocol of RFC 4733 is to repeat a given digit some number of times, usually 5. If possible, the device sends redundant digits in the same packet with new digits, saving on per-packet overhead. If the caller does not enter a new digit within the repeat period, the device emits a packet with the remaining redundant digits.

Note this is an example of "reliability by hope." RFC 4733 assumes there is a loss of connectivity for no more than the redundancy period. The redundancy period is the time from when the gateway sends the first copy of the digit to the last redundant copy. RFC 4733 "hopes" the network will deliver at least one copy of the digit during the redundancy period. Although statistically sound, this method is inefficient, in that the device sends many more packets than necessary when the network is working, yet does not provide reliable packet delivery when the network has congestion or other interruption.

In Fig. 5 we show the flow for the call flow using RFC 4733 to encode the consumer's digit entry. Steps **(501)**, **(502)**, and **(503)** in Fig. 5 is the SIP session establishment protocol handshake: INVITE, 200 OK, and ACK, respectively. In step **(504)**, the IVR platform streams the What Would You Like prompt to the IAD.

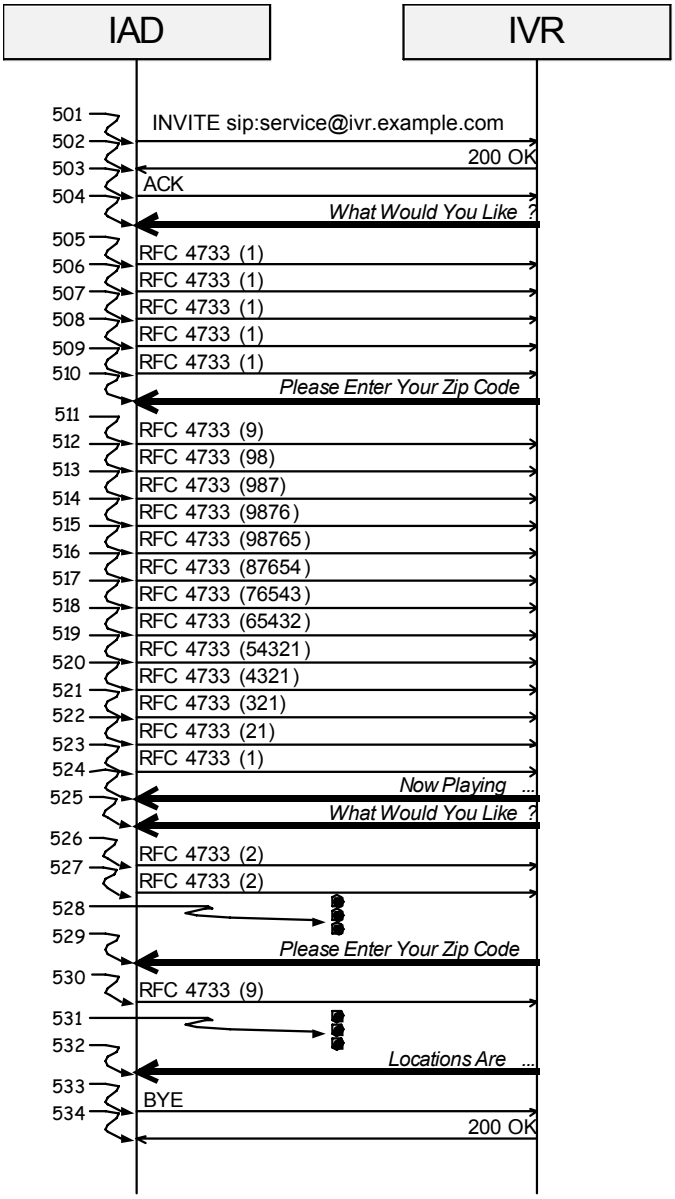

Figure 5 - Named Tones Call Flow

The consumer enters their selection by pressing the "1" key. The IAD uses RFC 4733 to encode this as a "1" named tone packet, and transmits it in message **(505)**. Messages **(506)** through **(509)** are the redundant copies of the key press. The IVR then asks the consumer for their ZIP Code. Note that as the user enters more digits, RFC 4733 allows the IAD to piggyback the redundant digits with a new digit packet. For example, message **(511)** contains the first entered digit, a 9. When the consumer enters the second digit, an 8, the IAD sends both the first redundant copy of the 9 and the first copy of the 8 in a single packet **(512)**. When the user is not entering

digits, the IAD completes the redundancy algorithm by sending the redundant copies of the digits, as needed in messages **(516)** through **(523)**. The platform then plays the requested information **(524)** and again asks the consumer what they would like **(525)**. The consumer enters the "2" key **(526)**, which the IAD sends 4 redundant copies **(527** and **528)**. The IVR platform streams a prompt requesting the ZIP Code **(529)**, to which the consumer enters a 9-digit ZIP Code **(530** and **531)**. The ZIP Code here expands as in messages **(511)** through **(523)**. We do not repeat all of the message flows, but the figure represents them in ellipsis **(531)**. The IVR platform streams the locations **(532)**. The consumer hangs up **(533)** and the platform acknowledges the disconnect request **(534)**.

C. KPML

In Fig. 6, we show the call flow using KPML to encode the consumer's digit entry. Messages **(601)**, **(602)**, and **(603)** in are the SIP session establishment protocol, INVITE, 200 OK, and ACK, respectively.

The IVR platform subscribes for the tone detection events **(604)**, which the IAD acknowledges **(605)** and sends an instant notification **(606)**, which the IVR platform acknowledges **(607)**.

The IVR platform then streams the menu prompt to the IAD **(608)**. The user enters the digit, and the IAD sends the collected digit to the IVR platform **(609)** which the IVR platform acknowledges **(610)**.

The IVR platform streams the ZIP Code prompt to the gateway **(611)**. In response, the user enters the ZIP Code, which the IAD sends to the IVR platform **(612)** and the IVR platform acknowledges receipt of the digits **(613)**. The IVR platform streams the information to the consumer **(614)** and then streams menu prompt again to the consumer **(615)**. The consumer enters a "2" for bank locations **(616)** and the IVR platform acknowledges receipt of the digits **(617)**. The IVR platform asks the consumer for the ZIP Code to look up **(618)**. The consumer enters the ZIP Code **(619)**, which the IVR platform acknowledges **(620)**. The IVR platform then streams the requested information to the consumer **(621)**. The consumer hangs up **(622)**, which the IVR platform acknowledges **(623)**.

XI. Network Utilization

We set out to compare the efficiencies of using various digit signaling schemes, namely sending named tones (RFC 4733), and using KPML.

RFC 4733 sends media packets during prompt playing, but sends only named tone packets and redundancy packets in the media stream when the caller enters digits. KPML sends media packets during prompt playing, but sends only KPML events (and handshakes) in the signaling stream when the caller enters digits. RFC 4733 is an update to the older RFC 2833 [10].

To determine what the associated cost is, if any, to use a reliable protocol in the signaling channel (KPML), as opposed to sending signaling in the media channel (RFC 4733), we conducted an experimental study.

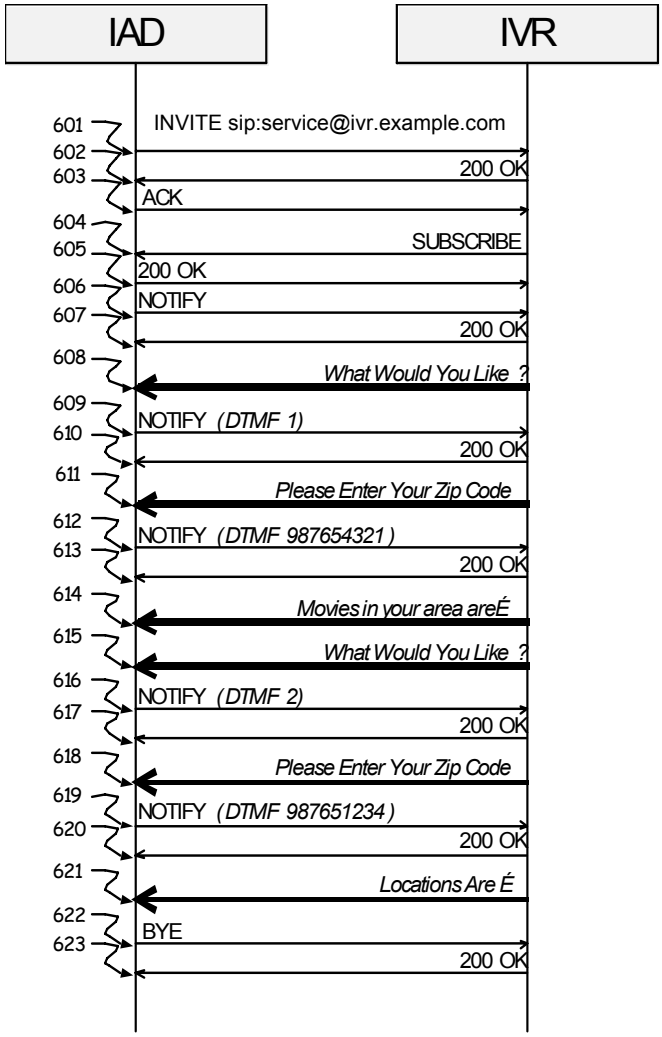

Figure 6 - KPML Call Flow

A. RFC 4733

In Table 1, we enumerate the packet and byte counts for the message and RTP exchanges for the transaction described in *Protocol Comparison* above, using RFC 4733 for signaling transport. The numbers in the "#" column refer to the message number.

B. KPML

In Table 2, we enumerate the packet and byte counts for the message and RTP exchanges for the transaction described in *Protocol Comparison* above, using KPML for signaling transport. The numbers in the "#" column refer to the message number.

TABLE I
RFC 4733 NETWORK USE

#	Msg.	Bytes	I/O	Packets In	Packets Out
501	INVITE	571	O	0	1
502	OK	530	I	1	0
503	ACK	235	O	0	1
504	RTP Intro	112336	I	472	0
505	RFC 4733 1	86	O	0	1
506	RFC 4733 1	86	O	0	1
507	RFC 4733 1	86	O	0	1
508	RFC 4733 1	86	O	0	1
509	RFC 4733 1	86	O	0	1
510	RTP ZIP	26894	I	113	0
511	RFC 4733 1	86	O	0	1
512	RFC 4733 2	95	O	0	1
513	RFC 4733 3	103	O	0	1
514	RFC 4733 4	111	O	0	1
515	RFC 4733 5	119	O	0	1
516	RFC 4733 5	119	O	0	1
517	RFC 4733 5	119	O	0	1
518	RFC 4733 5	119	O	0	1
519	RFC 4733 5	119	O	0	1
520	RFC 4733 4	111	O	0	1
521	RFC 4733 3	103	O	0	1
522	RFC 4733 2	95	O	0	1
523	RFC 4733 1	86	O	0	1
524	RTP Now Playing	180404	I	758	0
525	RTP Main Menu	75922	I	319	0
526	RFC 4733 1	86	O	0	1
527	RFC 4733 1	86	O	0	1
528.1	RFC 4733 1	86	O	0	1
528.2	RFC 4733 1	86	O	0	1
528.3	RFC 4733 1	86	O	0	1
529	RTP ZIP	26180	I	110	0
530	RFC 4733 1	86	O	0	1
531.1	RFC 4733 2	95	O	0	1
531.2	RFC 4733 3	103	O	0	1
531.3	RFC 4733 4	111	O	0	1
531.4	RFC 4733 5	119	O	0	1
531.5	RFC 4733 5	119	O	0	1
531.6	RFC 4733 5	119	O	0	1
531.7	RFC 4733 5	119	O	0	1
531.8	RFC 4733 5	119	O	0	1
531.9	RFC 4733 4	111	O	0	1
531.a	RFC 4733 3	103	O	0	1
531.b	RFC 4733 2	95	O	0	1
531.c	RFC 4733 1	86	O	0	1
532	RTP Locations	132566	I	557	0
533	BYE	246	O	0	1
534	OK	264	I	1	0
	Bytes in	**555096**			
	Bytes Out	**4682**			
	Packets In	**2331**			
	Packets Out	**39**			

TABLE II
KPML NETWORK USE

#	Msg.	Bytes	I/O	Packets In	Packets Out
601	INVITE	571	O	0	1
602	OK	530	I	1	0
603	ACK	235	O	0	1
604	SUBSCRIBE	797	I	1	0
605	200 OK	299	O	0	1
606	NOTIFY	328	O	0	1
607	200 OK	299	I	1	0
608	RTP Intro	112336	I	472	0
609	NOTIFY	664	O	0	1
610	200 OK	299	I	1	0
611	RTP ZIP	26894	I	113	0
612	NOTIFY	662	O	0	1
613	200 OK	299	I	1	0
614	RTP Now Playing	180404	I	758	0
615	RTP Main Menu	75922	I	319	0
616	NOTIFY	664	O	0	1
617	200 OK	299	I	1	0
618	RTP ZIP	26180	I	110	0
619	NOTIFY	662	O	0	1
620	200 OK	299	I	1	0
621	RTP Locations	132566	I	557	0
622	BYE	246	O	0	1
623	OK	264	I	1	0
	Bytes in	557,388			
	Bytes Out	4,331			
	Packets In	2,337			
	Packets Out	9			

XII. ANALYSIS

We examined the message flows and bandwidth usage of various representative methods of transmitting signaling information from a user with a telephone connected to a Voice-over-IP network. Namely, we examined sending named keypress packets that are representative of the DTMF tones (the named tones or RFC 4733 case) and using a signaling-level protocol (the KPML case).

Using signaling instead of the actual digital waveforms for transporting the user input clearly is a benefit to network resource consumption. For comparison, a summary of the resources consumed to carry the entire audio path in G.711 is included in the Table 3. Both the named tones and KPML have two orders of magnitude fewer outbound bytes and one to two orders of magnitude fewer outbound packets than the continuous RTP stream (G.711). In Table 3, we compare the results.

TABLE III
SUMMARY

	Bytes		Packets	
	Inbound	**Outbound**	**Inbound**	**Outbound**
G.711	555,096	145,042	2,331	608
RFC 4733	555,096	4,682	2,331	39
KPML	557,388	4,331	2,337	9

The prompts dominate the inbound byte count.

Examining Table 3, we find that the byte counts for KPML and RFC 4733 are on the same order of magnitude. We see that RFC 4733 uses approximately 0.4% fewer inbound bytes and packets than KPML. However, it uses approximately 8% more outbound bytes and over four times the packets than KPML.

Most important, RFC 4733 does not have the protocol property of reliable delivery. Moreover, the KPML model enables stateless servers, which the named tone model does not. KPML provides the properties of an easier-to-program model and reliable delivery at only a small premium over RFC 4733. In addition, the more digits captured and interpreted by KPML, the smaller the difference in network utilization becomes on the outbound side. KPML has a more pronounced benefit when considering the difference in network utilization on the inbound side.

The programming flexibility of KPML over RFC 4733 is an important feature. Consider how the program logic at the IVR platform changes when one changes the menu order or ZIP Code length. In the RFC 4733 case, one must modify the program logic at the IVR Platform to expect a different number of digits. In the KPML case, one only needs to modify the KPML markup. For either five or nine digit ZIP Codes, the controller receives a NOTIFY with the “zip” tag, informing the IVR Platform that the digits represent a valid ZIP Code. This logic is much simpler than having to parse out each and every digit individually.

Note too the additional bandwidth used by KPML is in the inbound direction. Typical consumer Internet access technologies, such as cable modem and DSL, are asymmetric, providing considerably more inbound bandwidth than outbound bandwidth. Thus KPML’s conservation of outbound bandwidth, to the order of one-fourth the bandwidth needed by RFC 4733, and the slight (0.4%) increase in the inbound bandwidth, which is plentiful, is acceptable for consumer applications.

VII. CONCLUSIONS

We previously described a new environment which easily and efficiently controls devices in the home environment remotely, without the need for specialized hardware in the home devices [11]. This environment, KPML, imposes no requirements for complex line sharing or specialized control hardware. The ubiquitous plain old telephone with a 12-digit keypad is all that one needs to interact with network services, even in light of new Voice-over-IP networks..

We have shown how KPML provides an efficient, reliable protocol for consumer devices, possibly connected to plain old telephones with 12-digit keypads, to efficiently interact with network services using Internet transport technologies. More importantly, KPML enables device developers to create dynamic user interfaces, which are much easier to create and maintain.

Further contributions not explored in this paper include the ability of the KPML approach to allow multiple devices to simultaneously get input from a single controlling device, map to the web model of application development, and reduce the number of SIP messages for single, stand-alone user interface interactions.

VIII. ACKNOWLEDGMENT

The authors would like to thank Martin Dolly for his editorial contribution to the IETF KPML specification.

IX. REFERENCES

[1] H. Schulzrinne and T. Taylor, "RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals," IETF, RFC 4733, December 2006.

[2] ITU-T, "Technical Features of Push-Button Telephone Sets," International Telecommunications Union, Geneva, ITU-T Recommendation Q.23, November 1998.

[3] H. Schulzrinne, S. L. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," IETF, RFC 3550, July 2003.

[4] ITU-T, "Control protocol for multimedia communication," ITU, Geneva, Recommendation H.245, July 2003.

[5] T. Hain, "Architectural Implications of NAT," IETF, RFC RFC 2993, November 2000.

[6] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," IETF, RFC RFC 3261, June 2002.

[7] E. W. Burger and M. Dolly, "A Session Initiation Protocol (SIP) Event Package for Key Press Stimulus (KPML)," IETF, RFC 4730, October 2006.

[8] A. B. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification," IETF, RFC RFC 3265, June 2002.

[9] E. Burger, "A New Inter-Provider Interconnect Technology for Multimedia Networks," *IEEE Communications Magazine,* vol. 43, pp. 147-151, June 2005.

[10] H. Schulzrinne, S. L. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," IETF, RFC 1889, January 1996.

[11] E. W. Burger and O. Frieder, "A Novel System for Remote Control of Household Devices Using Digital IP Phones," *IEEE Transactions on Consumer Electronics,* vol. 52, May 2006.

Dr. Eric W. Burger (M'84-SM'00) is Deputy Chief Technology Officer at BEA Systems, Inc. Prior to BEA, he was Chief Technology Officer of Cantata Technology, Inc. and its predecessor firm, Brooktrout Technology. He founded SnowShore Networks, Inc., where he invented the SIP Controlled Multifunction Media Server. He held various research and management positions with MCI, Cable & Wireless, ADC/Centigram, The Telephone Connection, Valid Logic Systems, and Texas Instruments. He holds degrees from the Massachusetts Institute of Technology, K.U. Leuven, and the Illinois Institute of Technology. He currently serves as the Chair of the SPEECHSC, LEMONADE, and MEDIACTRL Work Groups in the IETF and is active in signaling and applications protocol development in the IETF and W3C. He is a member of the ACM, AAAS, and ISOC. His research interests focus on real-time multimedia protocols, architectures for large-scale multimedia systems, and the application of web services to real-time, interactive communications.

Dr. Ophir Frieder (SM '93, F '02) is the IITRI Chair Professor of Computer Science and the Director of the Information Retrieval Laboratory at the Illinois Institute of Technology. His research interests focus on scalable information retrieval systems spanning search and retrieval and communications issues. He is an AAAS Fellow, ACM Fellow, and IEEE Fellow.