SDP-based Signalling and OTT Services:
History and Perspective on an Evolving Trend

Jean-Charles Grégoire
INRS-EMT, Montréal, Canada
Email: gregoire@emt.inrs.ca

***Abstract*—Interactive voice and video communications require a signalling mechanism to settle key parameters of the communications. Such operations have always been part of telecommunications, but in the Internet, because of stricter separation of higher and lower level protocols through layering, there have always been some issues left unresolved, mostly with the lower layers. Even the situation with the higher layers leaves room for improvement, and it remains in constant evolution.**

Recent developments in the form of Over the Top services have shed some light on the difficulties we encounter with the standard SIP/SDP model when it is transposed to new service environments, and this leads us to examine where problems can stem from, and how they can be improved. This problem becomes more acute as, in some cases interactive communications have become embedded in more generic (e.g. social media) platforms, rather than remaining standalone applications. This has led to some complications with their integration.

In this paper, we first study the evolution of signalling architecture and exposed the problems it leads to in the Internet age. We then analyse some issues with the Session Description Protocol. We finally lay the foundations for a signalling model more suitable for OTT services.

***Index Terms*—OTT services, VVoIP, SIP/SDP, WebRTC, ORTC, signalling, IETF, W3C.**

I. Introduction

Recent developments in Internet–based platforms have witnessed an increasing use of embedded interactive communications, that is, communication mechanisms based on voice and possibly video which become both part of and added value of an environment, rather than being the prime focus of the application. Facebook, for example, will allow connected users to text-chat, but also talk with other connected "friends". Supporting such trends, we have also seen communication mechanisms become integral part of browsers (e.g. WebRTC [14], standardized by the World-Wide Web Consortium–W3C), thereby setting a much lower bar for potential creators of applications on such platforms since there is no longer a need to develop a separate module to allow communications.

These developments have been done while capitalizing on established practices for voice and video over IP (VVoIP) communications, as set mainly by the IETF, with contributions of other bodies such as 3GPP, which has made VoIP a cornerstone of its IP Multimedia Subsystem (IMS) framework. In the process, however, we see the adoption of practices and models which, while acceptable are not exactly best suited for the context of these new platforms. We specifically refer to the signalling model used in VoIP and its exchange of data.

This paper focuses on several dimensions of VVoIP-based communications. First, in section II, we review the basics of signalling in telephony and show how new connectivity models have emerged and deserve proper recognition. Second, in Section III we look more closely at the SDP framework, show how it has evolved and the challenge its use presents and follow with a discussion of current work in Section IV. In Section V, we discuss how implementations can be simplified for Over the Top (OTT) services, and in embedded circumstances. We follow with a discussion and conclusion.

II. Signalling

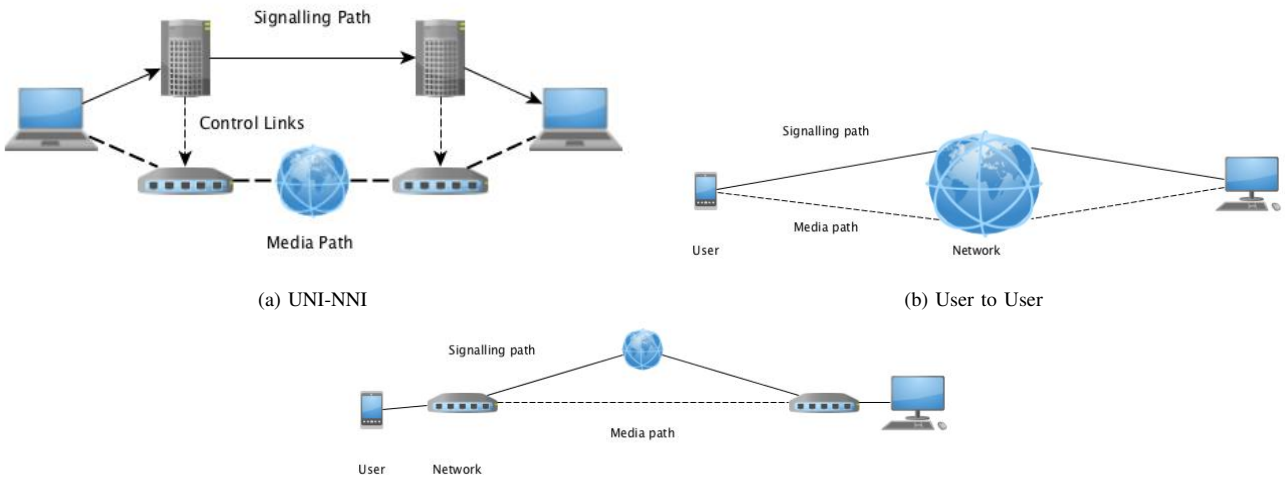
Traditional telephony rests on the complementary notions of *switching* and *signalling*. Established from its inception, these concepts include the creation of a path for the communication as well as the exchange of information required for the set-up of that path. In the scope of this paper, we adopt the traditional view on signalling to be the negotiation of the parameters of a call and the management (acquisition, parametrization, release) of the resources required to support the call. Also, signalling is usually associated with a *call model*, a state machine which captures the different phases of the set-up of a call with their respective operations in terms of resource allocation.

A. Traditional Telephony

Signalling was done manually in the early days, through an operator, but quickly became automated and, in the digital age, a large infrastructure supporting a variety of services culminating with the Signalling System No 7 (SS7) and the (Advanced) Intelligent Network.

The notion of signalling, being so fundamental, has made its way into other networks. Large-scale architectures for, say, cellular or optical networks, will isolate signalling into a *plane* of dedicated functions, distinct from switching. In the process, the notion of what signalling exactly is has been blurred with related concepts such as control, and, while the separation of different functions into planes is a given, terminology may not always strictly agree between bodies such as ITU-T and 3GPP. This however, is not an issue per se. Of greater concern to us here is the distinction between the communicating entities for the purpose of signalling: In networking, there has been a clear distinction between communications between terminal and network–the User to Network Interface (UNI)–and the network to network interface (NNI).

Figure 1 presents variants of signalling model. Figure 1a shows the traditional UNI and NNI interfaces of telephony. It is worth insisting on the notion of interface and what it



(a) UNI-NNI

(b) User to User

(c) User to Platform

Fig. 1. Different signalling models

entails: In standards bodies, the interface is a support for a communication between two entities, here user device–network and network–network, which is characterized by a protocol. We should note that there are multiple reasons why UNI and NNI must be differentiated, e.g.

- the UNI interface can be less secure;
- the scope of operations is different, the NNI allowing aggregations not relevant for the UNI;
- signalling and data traffic can be multiplexed on the UNI, but disjoint elsewhere, e.g. for issues of QoS.

B. IP Telephony

The Internet has introduced a different perspective on signalling. In the spirit captured by the seminal "rise of the stupid network" paper [1], which essentially gave embodiment to the end-to-end founding principle of the Internet [2], focus has moved from infrastructure to terminals, which arguably had the full responsibility for the execution of applications, with a matching focus on end-to-end signalling which we will call the user to user interface, or UUI, as shown in figure 1b. In the process, the whole notion of network has disappeared, since it was supposed to be application neutral and therefore transparent.

The application-level signalling protocol of the Internet, the Session Initiation Protocol (SIP) [6] embodies this model, but also its challenges. While being end-to-end, it is associated with a "trapezoidal model" of communications which allows (optional) routing of signalling messages through relays, while the media will follow a direct path. Relays are necessary for a number of reasons, but their main benefit is to provide indirection: the same way a phone call is routed between different providers, an IP phone call can be routed from one domain to another, hiding the way a user can be reached. This model natively supports value added services, and multi-connectivity among other features.

Routing calls is only part of the problem, however, and an offer-answer model [7], [13] is included for the purposes of negotiating the parameters of the call. We will come back on this issue in detail later, but examples of such parameters would include the media used (e.g. voice only, or voice and video or text) or the nature of the media (i.e. which codec is used, and which parameters). Otherwise, SIP has a simple call model based on requests, responses and acknowledgement.

SIP gathered only marginal interest, competing with the already established, ITU-T promoted H.323 [4] suite of protocols, until it was adopted by the 3GPP as the foundation for IMS. Since then it has gained legitimacy and evolved steadily to meet the requirements of an evolving world (over 30 updates since its inception, at the time of this writing). For example, its call model has introduced sub-states (e.g. through the use of preconditions [8]) which are required for resource management.

We must note that the ideal view of an end-to-end, even with relays, has had to make room for evolved platforms to support a diversity of communication services, as well as ancillary services such as directories. While, from the terminal's perspective, we may keep an end-to-end view of the call, in practice, platform elements become intertwined (e.g. for authentication support) and platform to platform interfaces have emerged, certainly in the form of domain to domain communications in IMS.

C. Communication platforms

Various forms of Internet platforms have, through time, integrated support for interactive communications, from text to audio and video, as witnessed by Google Mail or Facebook. Unlike previous models, interactive communications become an added value to a platform which already provides core services of authentication and presence for other purposes. Information exchanges are first and foremost non-interactive,

supported by a push model of various media (text, images, ...), but interactive communications are integrated as a possible means of communications should both parties be simultaneously available.

In this model, we argue that interactive communications are embedded, rather than being the essential feature of the platform; in other words, we are no longer dealing with an enriched telephony environment, but with an environment enriched with (video)telephony. This goes even further as client applications are no longer necessarily native, but can be embedded in a browser which can offer A/V communication primitives through the WebRTC environment. Finally, these platforms provide a monolithic, all-encompassing environment, with no need to interoperate as peers with other platforms – only breakout functions in some cases, e.g. to connect to a traditional telephony environment.

In this context, communications no longer follow a U2U but rather U to Platform (U2P) model, see Figure 1c: they are created and controlled through the platform and, in some instances such as eCommerce, established by the platform itself.

Under such circumstances, we can wonder what if the tools of the U2U model (SIP/SDP) are equally suited to the task. We will explore this issue further in the next section.

D. The network dimension

Moving from network to platform, it can be argued that we have lost track of an important dimension, the network itself, for the purpose of connectivity and quality.

Connectivity is a given in the Internet, except for a couple of issues. User equipment can be assigned link or site-local addresses which cannot be routed in the Internet. This typically happens when NAT boxes are present. Even if that is not the case, it is possible that some firewalls block some forms of traffic. In some cases, or arguably all cases, quality of service is also an issue. In the U2U perspective of a transparent Internet, this raises the question of where the responsibility of handling these issues lies: the platform, the user, or elsewhere.

1) Negotiated approach: In the SIP/SDP offer-answer model, elements of connectivity and quality are included in the negotiations. In some cases, it is possible for the User to communicate with its access network node and negotiate some of its features itself. However, since quality tends to be the responsibility of the network provider, it is also possible that the offer-answer be examined by operator-owned relays which will communicate with the appropriate network elements to execute the proper actions, be they to remove restrictions or to allocate the proper channels.

This piggybacking of U2N negotiation on top of an otherwise U2U model breaks the e2e model, and leads to issues of privacy. It also puts some complexity on the User side to identify what its needs are. Nevertheless, this approach has been chosen by 3GPP [15] for cellular network deployments. The lack of consensus on a network layer signalling model at that time may have contributed to this choice.

2) Non-negotiated approach: In the non-negotiated approach, these issues are not resolved, but rather sidelined, or even, in the worst case, ignored. Different approaches can be used to identify restrictions and circumvent them, if necessary. The ICE/STUN/TURN protocol suite [12] allows the User to discover and bypass restrictions in its connectivity, and mechanisms are used to adapt media quality to network fluctuations.

This approach is widely adopted for U2P models, in situations where services run "over the top" (OTT), without provider supported quality of service. It is, for example, built into the core of WebRTC [14], or applications such as Skype.

E. Discussion

We have shown so far how the move from traditional telephony to IP-based telephony has led to changes in the signalling support model, how a new model has emerged, and how some of their limitations in terms of network interactions are managed. We have underlined that some issues we are facing in the current limits of the signalling approach are related to layering, and the will to enforce a proper e2e model, in keeping with the core philosophy of the Internet. In the process, a clear notion of a signalling infrastructure has been lost and other mechanisms have had to fill the void.

III. SDP

The Session Description Protocol was created in the mid 1990s in the context of the study of multicast over the Internet with the MBone project [3], [5]. It was designed as a descriptive format for media flow broadcasts and it is only later that its use in a unicast interactive context was explored, in connection with the Session Initiation Protocol (SIP) [10]. Whereas originally information was about a fixed media source and hence static, things changed in an interactive environment where not only information required to set up media exchange, such as addresses, had to be known about both destinations but also some other elements had to be negotiated, as part of the offer-answer model. Further evolution has led to the inclusion of different parameters to support infrastructure-level parameter negotiation. This has led to a complex, rather obfuscated and not necessarily efficient mechanism. Let us consider the following example 1, from webrtchacks, to illustrate our point. We have deliberately chosen an example from WebRTC, an OTT platform.

A. Structure Analysis

The example 1 is an SDP offer for a WebRTC session with three media flows: audio, video and data, each announced with an "m-" line: 7, 30 and 51. Each section, that is the lines between an "m-" line and the following one, has specific codec and/or protocol offerings.

We can identify different parts with distinct purposes.

- *protocol information*: essentially the revision of the protocol, on the first line.

Listing 1. SDP Offer Example for WebRTC

```
1  o=- 8717899652504307731 2 IN IP4 127.0.0.1
2  s=-
3  t=0 0
4  a=group:BUNDLE audio video data
5  a=msid-semantic: WMS
6  m=audio 1 RTP/SAVPF 111 103 104 0 8 106 105 13 126
7  c=IN IP4 0.0.0.0
8  a=rtcp:1 IN IP4 0.0.0.0
9  a=ice-ufrag:4t22CxThPocZSIBa
10 a=ice-pwd:wKJ0LyPB8DAD+r8bNpSS1riU
11 a=ice-options:google-ice
12 a=mid:audio
13 a=extmap:1 urn:ietf:params:rtp-hdrext:ssrc-audio-level
14 a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
15 a=recvonly
16 a=rtcp-mux
17 a=crypto:1 AES_CM_128_HMAC_SHA1_80 inline:LQP8XKpMskWEeWCsByH2ZXUwEoRZ3wdA68Pkxerb
18 a=rtpmap:111 opus/48000/2
19 a=fmtp:111 minptime=10
20 a=rtpmap:103 ISAC/16000
21 a=rtpmap:104 ISAC/32000
22 a=rtpmap:0 PCMU/8000
23 a=rtpmap:8 PCMA/8000
24 a=rtpmap:106 CN/32000
25 a=rtpmap:105 CN/16000
26 a=rtpmap:13 CN/8000
27 a=rtpmap:126 telephone-event/8000
28 a=maxptime:60
29 m=video 1 RTP/SAVPF 100 116 117 96
30 c=IN IP4 0.0.0.0
31 a=rtcp:1 IN IP4 0.0.0.0
32 a=ice-ufrag:4t22CxThPocZSIBa
33 a=ice-pwd:wKJ0LyPB8DAD+r8bNpSS1riU
34 a=ice-options:google-ice
35 a=mid:video
36 a=extmap:2 urn:ietf:params:rtp-hdrext:toffset
37 a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
38 a=recvonly
39 a=rtcp-mux
40 a=crypto:1 AES_CM_128_HMAC_SHA1_80 inline:LQP8XKpMskWEeWCsByH2ZXUwEoRZ3wdA68Pkxerb
41 a=rtpmap:100 VP8/90000
42 a=rtcp-fb:100 ccm fir
43 a=rtcp-fb:100 nack
44 a=rtcp-fb:100 nack pli
45 a=rtcp-fb:100 goog-remb
46 a=rtpmap:116 red/90000
47 a=rtpmap:117 ulpfec/90000
48 a=rtpmap:96 rtx/90000
49 a=fmtp:96 apt=100
50 m=application 1 RTP/SAVPF 101
51 c=IN IP4 0.0.0.0
52 a=rtcp:1 IN IP4 0.0.0.0
53 a=ice-ufrag:4t22CxThPocZSIBa
54 a=ice-pwd:wKJ0LyPB8DAD+r8bNpSS1riU
55 a=ice-options:google-ice
56 a=mid:data
57 a=sendrecv
58 b=AS:30
59 a=rtcp-mux
60 a=crypto:1 AES_CM_128_HMAC_SHA1_80 inline:LQP8XKpMskWEeWCsByH2ZXUwEoRZ3wdA68Pkxerb
61 a=rtpmap:101 google-data/90000
62 a=ssrc:1092095154 cname:YuFlsTZUYKl5tFHm
63 a=ssrc:1092095154 msid:sendDataChannel sendDataChannel
64 a=ssrc:1092095154 mslabel:sendDataChannel
65 a=ssrc:1092095154 label:sendDataChannel
```

- *legacy information*: session information, time . . . some of this information is redundant, as it can be provided by other means, or obsolete, on lines 2-4.
- *application transport information*: which protocol variants are used for transport, and security parameters, e.g. lines 17, 18, 43–47..
- *connectivity information*: here the use of ICE to connect to the Internet, and security parameters, lines 10–12, 33–35, 54–56.
- *media information*: all the codecs offered, for each media; lines 19–29 for audio, 47–50 for video, 62 for data.

We propose that this format suffers from a number of problems.

a) Format efficiency: This format is neither machine efficient, nor human efficient. It lacks hierarchy or clear structure, which makes it hard to understand. Attributes (the right hand side of "a=") are reused for multiple interpretations, leading to an embedding of keywords which only implicitly reflect a substate (e.g. rtpmap, fmtp, rtcp-mux, etc.) and whose interpretation may depend on previous lines.

b) Unnecessarily Verbose: Most items are repeated through the stages of the offer-answer process, analyzed in terms of delta between what was offered, and what has been accepted. In the process, the value of some parameters may have changed, which can lead to failure to recognize the document.

c) Abstraction: There are no abstractions in this model; it is not possible to reference parts of the text, everything must be quoted verbatim.

d) Separation of concern: Some elements are access-specific, while others are end-to-end; some will remain fixed while others will change. The concerns are different yet it is not possible to distinguish them.

e) Redundancy: SIP, SDP and even RTCP, the control companion protocol of the Real Time Protocol (RTP, used to transport media) all allow similar declarations, e.g. for bandwidth requirements. This complicates developers' work in the creation of applications or standard libraries, as well

as network application developers, who may need to monitor different ways to communicate the same information.

Other forms of redundancy come from legacy: different profiles of RTP adding enriched feedback and/or encryption such as RTP/AVP, RTP/SAVP, RTP/AVPF and RTP/SAVPF were created before an option to negotiate capabilities was introduced. It is now possible [11] to either explicitly name the profile, or negotiate its characteristics.

f) Self Awareness: How terminals figure out their characteristics and create an SDP document is always an "out-of-scope" issue. Ultimately, it is related to the computing platform and the features it provides, but this is hardly straightforward to discover, and this leads to strong coupling between platform and application, or redundant implementations.

g) Manipulation Challenge: As part of the offer-answer process, or for other reasons, it is necessary to modify the SDP document. The following is an example of how it can be done on a WebRTC platform, but it is easy to see how brittle such an approach–pattern matching and substitution–is.

```
// audio bandwidth 50 kilobits per second
sdp = sdp.replace(/a=mid:audio\r\n/g,
                  'a=mid:audio\r\nb=AS:50\r\n');
```

h) Incomplete: SDP's evolution is largely a story of features which have been added, or corrected–the following quote is a rather explicit example, which actually refers to line 5 of the example 1, and denotes how something that was possible in some implementations later became standardized.

> As defined in RFC 4566 [RFC4566], the semantics of assigning the same port value to multiple "m=" lines are undefined, and there is no grouping defined by such means. Instead, an explicit grouping mechanism needs to be used to express the intended semantics. This specification provides such an extension.

Terminal information and capabilities are still rather limited in the current instance of SDP. Terminal orientation, for example, was only recently added.

i) Archaisms: Finally, we can mention archaic constructs which have required workarounds, for example the size limit of the codec table (rtpmap) has led to an escape mechanism and increased verbosity.

B. Further observations

In many ways, it is a tribute to human creativity and ingenuity that SDP has managed to evolve to fill so many requirements, so far away from its original purpose, from declarative to interactive. At the same time, we have shown that the result, for all its legitimacy, is rather messy. Evolution has been pushed by different groups with requirements of their own, matching specific purposes. From anecdotal evidence, it seems that many terminals are not fully compliant with SDP or RTP, a sign that new features are not necessarily easy or straightforward to integrate.

Standard groups, such as 3GPP, provide strong guidelines on the use of SDP features. Nevertheless, they still leave some leeway for implementations, and we end up with statements such as the following, from [16]:

> If AVP is to be used then the MTSI shall not indicate any SDPCapNeg attributes for using AVPF in the SDP answer.

This is a clear indication that the protocol stack is not completely specified and some freedom is left. But it can be argued that it is the bane of standards.

IV. Discussion and related work

Attempts by IETF to deeply revise SDP have largely failed, as described in [11]. Rather, extensions have been provided to meet the various demands of different groups–industrial or standard bodies, including 3GPP–and the use of SDP has permeated in other environments, such as Jingle/XMPP [18] and WebRTC [14].

SDP meets static and dynamic purposes. On the static side, we have capabilities for communications, both in terms of media and protocols. Further, the media capabilities can be grouped in configurations, with specific parameters. A negotiation process helps resolve which actual configuration is used for the communication. SDP supports that process, but being only declarative, relies on a different protocol, typ. SIP, to execute the negotiation process. It is not possible, however, to isolate the parameters of the negotiation from an SDP document.

We must also not forget that other protocols such as SIP and RTP keep on evolving in harmony with SDP–or not, as their use does not necessarily rely on one another.

Work on WebRTC has opened another front, by trying to define a programmatic interface (API) to access an embedded communication engine. This model requires the existence of a bridge for signalling, but essentially encapsulates SDP documents generated internally - i.e. it is the only abstraction available for negotiation, which is not always suitable, hence the kind of contortions that were presented above.

In a typical (?) swing of the pendulum, work on a lower level API, the Object Real-Time Communications (ORTC) [19] has been initiated and is currently being studied by different groups (ortc.org), including Google; it has recently gained recognition from W3C. It aims at giving more flexibility to programmers by creating an object model offering abstractions of the low level mechanisms, sidestepping the use of SDP but in the process losing track of the signalling dimension and essentially focusing on media manipulation, ignoring U2U or U2P interfaces. Applications have to create their own exchange mechanisms, including interoperability, if required. As it is based on WebRTC, it also does not integrate some elements of the offer-answer model, and indeed claims that this is a bonus, for their purposes.

Such work only reinforces the trend of a loss of a clear signalling model. It puts more emphasis on local control and simply works around connectivity issues, with limited e2e concerns: it is up to applications to become aware of potential network issues.

This also shows that the attempt to keep a "one size fits all" approach around SIP/SDP has reached some limits and, certainly for Web communities, the temptation to move to different paradigms is strong, although moving from a protocol model to an API model does not resolve the issue of the representation of information for the purpose of sharing, and establishing consensus. We focus on that problem, orthogonal to WebRTC/ORTC trend, in the next section.

V. A Different Perspective

We propose a new look at the organization of media information, with the goals of:

- structuring call information;
- reducing the amount of information exchanged at any time;
- supporting other forms of call establishments;
- help with the creation of APIs.

Note that we present here requirements for the purpose of this discussion, rather than a full implementation.

A. Initialization

Whether it is an application or dynamic (downloaded) code, there is always a first step in which a user will connect and authenticate or register with a service platform, thereby creating a context for future communication sessions. Terminal capabilities can be established at that time, as well as connectivity and quality management models.

Note that even in the case of a peer-to-peer model, there is always some support infrastructure, if only for presence management, which can play that role.

B. The Static Context

This information is provided during initialization.

Offer: It must be possible to extract from the terminal platform and organize the capabilities, that is:

terminal
: terminal characteristics (e.g. screen size, rotation modes);

media
: codecs supported, standard parameters and preferences, organized per category of codecs;

Internet reference
: protocol stack available, for IPv4 and/or IPv6

Reachability
: ICE-related information.

All information uses standard names, e.g. such as maintained by IANA [20]. Furthermore, each capability is separately identifiable. The following is an example of a typical syntax for capabilities declaration.

```
<offer>
    <audio label=a>
        <opus/48000/2 minptime=10>
        <ISAC/16000>
        <ISAC/32000>
        ...
    </audio>
    <video>
    ...
</offer>
```

This information must be provided in text form from the platform, or the application itself, to allow self identification. In some instances, we could imagine that this information would be supplied by the creator of an application (e.g. a browser), accessible from an identifier obtained from the application.

Selection: The selection of a specific set of capabilities is associated with a session. Each capability selected is named by reference, possibly with parameters added. Again, a traditional syntax can be used, using labels for references.

```
<selection> SID
    <a:1>
    ...
</offer>
```

A session is created in connection with a specific capability.

Protocol stack: Possible variants on the protocol stack must be part of the capabilities. There will be issues to resolve such as variants of the IP stack (v4 vs. v6), for RTP/RTCP (AVP, AVPF, SAVPF), whether bundling and/or multiplexing is allowed on UDP flows, etc.

These elements are mostly discoverable through systems APIs, or are part of application libraries (e.g. RTP).

Connectivity Model: The presence of middleboxes is a fact of the Internet and must be managed. The ICE mechanism allows to circumvent them, but its use is contingent on the service architecture:

- it may not be required;
- media relays can be imposed;
- use of ICE may be imposed, with specific relays.

Several connectivity models may be supported and declared as part of the capabilities.

Quality Management Model: Similarly to connectivity, quality management is part of the capabilities. Recall that we have different approached to quality.

Managed, locally
: Network-level QoS mechanisms, such as marking or the allocation of a dedicated channel for cellular communications, are used, under the control of the terminal.

Managed, remotely
: Similar to the previous, but under the control of the infrastructure.

Unmanaged
: In this case, we depend on adaptation mechanisms, based on feedback from the receiver (ECN or RTCP) to manage quality.

We should note several tbings:

- other mechanisms, such as FEC, can be added;
- different approaches can be adopted for each flow;
- unmanaged mechanisms can be used even in the case of a managed connection.

Note that there is a need to understand the performance characteristics of the codecs used to perform suitable resource allocation and/or adaptation.

Finally, the receiver should have the possibility of setting an upper bound on incoming bandwidth.

C. The Dynamic Context

All previous elements have established capabilities for the different dimensions of the context: media, transport, connectivity and quality. The session will focus on a specific coordinated choice for each media, and once it is instantiated we can execute operations such as: *start*, *stop*, *add*, *remove*, *pause* or *resume*, with the usual semantics, on each flow.

These operations require end-to-end coordination and a communication channel, which ideally should be a direct one, for example supported by DCCP [9].

D. Further Issues

There are some issues which warrant further investigation.

a) Limits of self awareness: We have already evoked the problem of discovering which codecs are available, especially if they are built into the OS, as well as the parameters they support. This cannot be done dynamically in general.

A similar point can be made about support for QoS at the network level, or elements of the protocol stack itself. This is an acknowledged problem of the current internet protocols [21], largely related to the existence of middleboxes and the impairments they induce.

The temptation can be great to provide implementations for all protocols that the platform requires on top of UDP, at the risk of duplication of features, and loss of efficiency.

b) "Ignore what you do not understand.": Any exchange format for signalling has to be open ended: there may be elements which the receiver will not understand, yet has to accept as legitimate input.

c) Establish profiles: In our view, capabilities should refer to profiles, that is, specific combinations of elements, which could be referenced with several benefits: increased compatibility, reduced verbosity, ease of implementation. This is in line with work done by some standard bodies to help implementers, as "first line" standards such as produced by, say 3GPP or IETF tend to be open ended, or underspecified, often for political reasons.

VI. Conclusion

The SIP/SDP model for telephony has become the de-jure reference for all forms of Interactive communications, but in the process has had to evolve to accommodate multiple requirements, to the point that it has become unmanageable. The trend in embedded communications is to completely bypass the model to move to purely ad-hoc solutions.

In this paper, we have presented different deployment contexts for interactive communications, studied some limitations of the SIP/SDP call model and discussed elements of a more flexible alternative. We have tried to put back a more traditional signalling perspective on current work done on APIs for embedded communications. At the same time, we have shown that elements of the breakdown of e2e communications as well as lack of transparency of the protocol stack are unavoidable in the current Internet.

This is still the early stages of this work and we are in the process of refining the definition of capabilities as well as the exchange mechanism.

Acknowledgments The author wishes to thank Ms. Greene, Ms. Ouhmama and M. Duhamel for discussions on some aspect of this work, as well as M. L. Reyero and Dr. A. Vakili from Summit Technologies for their support of this research.

This work has been jointly funded by the Natural Sciences and Engineering Research Council of Canada and Summit Technologies through a Collaborative Research and Development Grant.

References

[1] Isenberg, D., "Rise of the stupid network." Computer Telephony, 5(8), pp. 16–26, (1997).

[2] Saltzer, Jerome H and Reed, David P and Clark, David D, "ACM Transactions on Computer Systems (TOCS)", Vol.2, No.4, pp. 277–288, 1984.

[3] Macedonia, Michael R., and Donald P. Brutzman. "MBone provides audio and video across the Internet.", Computer Vol. 27 No. 4, pp. 30–36, 1994.

[4] Thom, G., "H. 323: The multimedia communications standard for local area networks," Communications Magazine, IEEE, 34(12), pp. 52–56, 1996.

[5] M. Handley, V. Jacobson. "SDP: Session Description Protocol." April 1998. RFC 2327, IETF, (DOI: 10.17487/RFC2327)

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

[7] Rosenberg, J., Schulzrinne, H. "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, IETF, June 2002.

[8] Camarillo, G., Marshall, W and Rosenberg, J., Ed. "Integration of Resource Management and Session Initiation Protocol (SIP)," RFC 3312, IETF, October 2002.

[9] Kohler, E., Handley, M., and Floyd, S. "Datagram Congestion Control Protocol (DCCP)," RFC 4340, IETF, March 2006.

[10] M. Handley, V. Jacobson, C. Perkins. "SDP: Session Description Protocol", RFC 4566, IETF, July 2006. (DOI: 10.17487/RFC4566)

[11] Andreasen, F. "Session Description Protocol (SDP) Capability Negotiation", RFC 5939, IETF, September 2010.

[12] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols" , RFC 5245, IETF, April 2010.

[13] Okumura, S., Sawada, T., and Kyzivat, P. "Session Initiation Protocol (SIP) Usage of the Offer/Answer Model", RFC 6337, IETF, August 2011.

[14] Jennings, Cullen, et al., "WebRTC 1.0: Real-time communication between browsers." W3C, Editors Draft, Aug (2014).

[15] 3rd Generation Partnership Project "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3 (Release 13)", 3GPP TS 24.229 V13.1.0 (2015-03).

[16] 3rd Generation Partnership Project "IP Multimedia Subsystem (IMS); Multimedia Telephony; Media handling and interaction (Release 12)", 3GPP TS 26.114 V12.9.0 (2015-03).

[17] GSM Association "IMS Profile for Voice and SMS" Official Document IR.92, Version 9.0, 8 April 2015.

[18] XMPP Standards Foundation "XEP-0293: Jingle RTP Feedback Negotiation", Draft Standard, 1.0, 11 August 2015.

[19] Raymond, R. Ed., "Object RTC (ORTC) API for WebRTC", W3C Community group, Draft Community Group Report, 05 October 2015.

[20] Internet Assigned Numbers Authority (IANA) "Session Description Protocol (SDP) Parameters", www.iana.org, 2015.

[21] Trammell, B., Hildebrand, J. "Evolving transport in the Internet," Internet Computing, IEEE, Vol. 18, No 5, pp. 60–64, 2014.