My Own Communication Service Provider

Sivasothy Shanmugalingam\textsuperscript{1,2}, Noel Crespi\textsuperscript{2}, and Paul Labrogere\textsuperscript{1}

\textsuperscript{1} Alcatel Lucent Bell Labs France, Centre de Villarceaux, 91620, Nozay, France
\textsuperscript{2} Institut Telecom, Telecom SudParis, 9 rue C.Fourier, 91011, Evry, France

Email: sivasothy.shanmugalingam@alcatel-lucent.com, noel.crespi@it-sudparis.eu
paul.labrogere@alcatel-lucent.com

The World Wide Web has become an important platform for delivering services in the last decade. We consider that openness and flexibility are key factors for the success of the Web where end users develop applications freely for their needs. Consequently, end users become producers and consumers of contents and services. On the other hand, end users highly rely for their communication needs on communication platforms (CPs) that are implemented over the Internet by service providers. CPs lack sufficient openness and flexibility for end users due to their own complexity. This paper examines the CPs from the perspective of openness and flexibility for end users, and then proposes a novel concept, My Own Communication Service Provider (MOCSP)—an idea of providing an individual CP for an end user for communication services. To achieve this goal, we design a MOCSP system, based on a top-down and user-centric approach. Moreover, we analyze our approach in terms of benefits for end users.

Key Words—communication services, service provider, Web 2.0, user-centric.

I. INTRODUCTION

The Web has changed the way people perform day to day activities and evolves rapidly (e.g. Web 2.0, Web 3.0) from the invention of the Web in 1989 by Sir Tim Berners Lee [1][2]. “The Web adopts relatively simple technologies with sufficient scalability, efficiency and utility” [3]; we take into the account the factors that contributed to the success of the Web are: its openness and flexibility which serves as a key means for high user participation in creating applications. As a consequence, Web paves the way for innovative applications in the modern Internet era where end users become producers and consumers of contents and services. We assume the meaning of the openness and flexibility as follows; the openness means that users develop their applications or implement their new ideas with less dependability. The flexibility means that developed applications can be easily modified.

Communication platforms (CPs) take a bottom-up approach to provide communication services (CS) for mass users without much personalization, thus becoming as brokers between callees and callers. Service providers (SPs), who implement CPs, have more openness and flexibility in CPs that are hosted in the Internet and/or the Web (such as IP Multimedia Subsystem based) compared to legacy CPs (e.g. intelligent network). Nevertheless, end users highly rely on SP and CP for their communication needs and are more passive in communication service creation. Therefore, for turning the end users into active users at least to some degrees, CPs expose to developers and end users CS related functionality via application programming interface (APIs) (e.g. SIP API or ParlayX) for creating new services using CS. According to programmableWeb, there are around 200 mashups that are CS related out of listed 5100 mashups [26]. From this trend, we argue that CS related functionality that are now defined by SP, should be given to end users rather than providing the related APIs. Therefore, individual needs of end user should be satisfied easily. Existing APIs offer open access to services, but extension to the APIs hard to define in the CP.

Moreover, changes (for new features) in the control protocol of the CS need to update clients in order to maintain interoperability with others. In fact, clients are usually symmetric peers, working in an end-to-end fashion. This approach does not provide enough flexibility for changes. Therefore, end users do not get a reasonable or complete openness and flexibility from existing CPs.

In this paper, we propose a concept, MOCSP—an idea of providing an individual CP for an end user for the communication services. We also present the design of a system based on the MOCSP concept in the Web, following a top-down and user-centric approach. Consequently, we contribute to the vision of Web and Communication service convergence. In other words, MOCSP adds communication resources into the Web which has today diverse resources (e.g. media). Besides, we predict that in the future more users will imagine and implement different unimaginable use cases (e.g. personalized services) with CS, ultimately leading to more innovations due to openness and flexibility to end users.

The rest of the paper is organized as follows: Section II describes existing CPs and compares for openness and flexibility; Section III provides a concept of MOCSP along with an abstract view of CS, communication hyperlink and proposed architecture for a MOCSP system. Later, call flow in the MOCSP system is presented in Section IV and Section V describes the benefits to users from the proposed approach. Finally, Section VI concludes the paper.
II. RELATED WORK

A. Communication Platform

A CP that enables the communication between users is typically dominated by its control protocol. We focus here on CP over the Internet for our analysis because it gives more flexibility for SP than legacy CP (such as intelligent network). We are aware of existing CPs which can be built by either IP Multimedia Subsystem (IMS) based on (Session Initiation Protocol) SIP [4], XMPP/Jingle [5] (like gTalk), Web Service Initiation protocol (WIP) [6] based Service oriented communication (SOC), Service oriented VoIP (SOVoIP) [7], or Web personal communication systems [8].

Communication service providers have gained considerable amount of openness and flexibility thanks to standardized protocols and layered architecture for communication services (transport layer, session control layer and service layer). These benefits are not shared completely with end users, as discussed in the following two sub-sections.

B. Openness

CPs evolves from the walled-garden manner that does not include third parties, to the paradigm of open service marketplace for communication world that is influential for openness in CPs [20]. It means that SPs open their CPs for third party usage, making end users active contributors. In this analysis, we consider only IMS based CP that has adopted many methods to provide openness for end users or developers. The control layer and service layer of IMS can be modeled, based on the roles of caller, callee and SP as shown in Figure 1.

![Figure 1: High level view of IMS based CP. Callees and Callers are User agents. ISC stands for IMS Service Control.](image)

In IMS, SPs build CP and services using open standards, but users can (tech user) create and gain control over their clients, but need to comply highly with the standard (e.g. 200 page SIP standard - RFC 3261).

IMS based SPs can provide APIs to third parties or end users for new service creation. These APIs can be either fine-grained (reflecting different states of a session) or coarse grained (just start the session). In this paper, we classify existing APIs into four groups based on knowledge required to develop applications, and the capability of the offered APIs. Details of the four groups are given below.

1) Protocol level APIs (e.g. SIP):

IMS provides the IMS Service control (ISC) interface to Application servers (e.g. SIP Servlets) where users can implement their applications. In this case, users should be strongly familiar with concepts of SIP and IMS such as dialog, transaction, session or required more knowledge to manage the intricacies of the underlying technologies [9]. In addition, changes to ISC interface need to undergo lengthy standardization process and need proper changes in the SP controlled platform in Figure 1.

2) Web Programming level APIs:

This group consists of ParlayX web service [16] (e.g. third party call control, or click-to-dial) for exposing communication functions using Web standard. These simplified APIs supply less capability than group (1) (i.e. Protocol level APIs), but are more developer friendly thanks to functional abstraction. In the IMS network, Parlay Gateway is newly added to provide abstraction in a HTTP fashion from the signaling protocol. However, to best of our knowledge, there is no real deployment using ParlayX standard to provide fine-grained APIs.

3) Session Data Type (SDT) API

As proposed in [10], SDT model provides primitives with more expressiveness, which is important for reducing the learning curve of advanced CS creation. In fact, developer users should know about abstraction of session, bubble, finite state machine and notification along with more integrated view of multimedia communication. End users can create particular classes of advanced services more easily than in group (1) and (2). For service providers, this feature brings more complexity to the platform. For example, SDT framework manages explicitly the finite state machine that describes the behavior of the involved session’s end points. The SDT prototype reveals that SDT can be deployed to any CP (e.g. IMS based). In fact, SDT focuses on the functional abstraction (session, bubble) for end user service creation (e.g. share experience services), not CP. MOCS Protocol gives more focus on simplicity in creating a unique CP using the top down approach in the Web platform.

4) Domain Specific Languages (DSLs)

Domain specific approach offers high level abstraction that describes the domain. In telephony domain, Domain specific modeling language such as Call Processing Language and VisuCom and domain specific programming language such as Session processing Language is proposed on top of the implementation platform (e.g. JAIN SIP) [11]. Compared to the protocol level APIs, required knowledge to create services (e.g. routing logic) is less in this approach. However, this domain specific approach is bottom-up thinking on the existing communication platform. Our approach is top down and to define the own communication platform based domain specific abstraction.

C. Flexibility

Fundamentally, all the CPs deliver CS to mass users based on the end to end paradigm – less knowledge at the core and more knowledge at the end points, resulting in symmetric peers. For example, SIP end points are in charge for complete message creation and processing. If there is a new header, all clients have to understand this new header to avoid interoperability issues or simply message discarding. This problem can be viewed as a version control issue.

In the end to end paradigm, network protocol design needs strong analysis in order to prevent the race conditions created...
by peers when sending and receiving messages [13] [19]. Therefore, changes to clients require huge amount of time and work. In our architecture, each session can be designed with different semantics (e.g; SIP,XMPP). In addition, changes in one session will not affect other sessions. It means that each session should not follow the way (e.g. SIP) standard defines.

D. Web Personal Communication system

[8] discusses many Web personal communication systems, enabling communication services from web pages. But in the proposed solution “Contact Me My Way”, users get more benefits: user (i.e.callee) control the way caller can call them, without disclosing the real identity, usability (caller does not need an account, configuration and installation) based on the concept of “communication hyperlink”. We advance further using communication hyperlink (see section 3.C) to give openness and flexibility for end users.

III. MY OWN COMMUNICATION SERVICE PROVIDER (MOCSP)

A. Concept

MOCSP is a concept which allows end users to create (and own) themselves their communication platforms for communication services. We propose a top-down, Web-based and user-centric approach for realizing the MOCSP concept. For the top-down approach, our goal is to enable a programmer (or end users) to concentrate on defining what the service should do first, and then dealing with every single detail of how to implement it. It means that end users are facilitated to define the services rather than defining or putting more focus on control and media flow.

Alignment of the MOCSP system with the Web platform is strongly motivated by three factors:

- Predication of the blendness of Future of the Web and the future of the human society [2]. It means that end users accomplish their needs in the Web in the future also.
- Openness and flexibility of the Web platform.
- Simplicity of the Web architecture based on HTTP, URL and HTML.

In the rest of this section, we provide a formalization of communication services at the high level, and then map these services into a Web concept (e.g. URI) since our realization is based on the Web platform. Then, we provide architecture of the MOCSP system. Finally, we show that MOCSP system is a realization of an I-centric communication system for communication services.

B. Definition of Communication services

We take a top-down approach to outline the CS instead of designing the control protocol. Thus we formalize the CS at high level and in the social context. It means that users want to define how they can be reached by others in different situations [8]. The formalization as shown in Figure 2 gives importance to user session (not like session in SIP composed of transactions and dialog) and becomes a basis for deriving meaningful abstractions on CS. A user session typically means person-to-person communication (personal communication) and should have one callee, one or more callers and one or two medium. Each user session hides the control session that define and control media session end points; therefore, it is viewed at very high level where caller, callee and medium properties (audio/video/text) are important parameters.

![Figure 2: A formal definition of communication services](image)

Unlike other CPs, we do consider identity for users (e.g. SIP address), authentication, control protocol and media protocol in the context of communication services/session as supplementary details. In addition, the high-level abstraction hides the protocol and technology specific details, therefore, ensuring easy-to-develop applications.

C. Communication Hyperlink (CH)

The World Wide Web (WWW, or simply Web) is an information space in which the items of interest, referred to as resources, are identified by global identifiers called Uniform Resource Identifiers (URI) [3]. Hence, we consider communication services (e.g. user session) as resources, in which callee, caller and corresponding medium (or user session) are considered as a resource and identified by global identifier (URIs), called communication hyperlink.

Communication Hyperlink is initially defined in [8] for providing client applications to users (i.e. callers) on demand but we formally map resource and communication hyperlink as explained in the previous paragraph.

In WIMS2.0 [12], an IMS session is modeled as a resource, consisting of signaling protocol information; it enables easy access of session based IMS capability using the HTTP protocol. We argue that there is no technical necessity to define all the information within control session as a resource for our proposal.

D. Architecture

MOCSP architecture follows the principal of signaling and media separation in order to be a flexible architecture. Based on this best practice, we propose a high-level architecture of the MOCSP system as shown in Figure 3, consisting of control plane and media plane.
MOCSP control plane is instantiated as a Web application and deployed in a Web server, enriched by communication hyperlinks. Caller and Callee are in two Web browsers connected to the MOCSP Web server. Media server is proposed (optional entity) for the transcoding purpose.

MOCSP Web application (in the control plane) executes three functionalities for callee:

1. Support to create different communication hyperlinks. When a callee creates hyperlinks, service logic for control session is developed for MOCSP Web server, callee side and caller side and placed in the MOCSP Web server.

2. Provide relevant user agent to callee and caller. At the run-time, MOCSP provides relevant user agents dynamically (kind of client software to manage a session) to caller and callee, therefore caller does not worry about user agent, configuration and installation, but depends on the Web browser thanks to this simplicity for users.

3. Create and manage control session when a caller clicks on communication hyperlinks. MOCSP is a key element compared to callee and caller clients for the session management, so not following a pure end to end paradigm. More details of control session are given in section IV.

From the concept and architecture mentioned above, we argue that MOCSP promotes Web and communication services convergence which is necessary due to Smartphone’s enhanced Web browsing capabilities (e.g. iPhone and Android) and technical evolution of Web. Our paradigm adds communication related resources (i.e. user session identified by URI) into a remarkable information space of the Web that grows across languages, cultures, and media [3].

E. Comparison of MOCSP vs I-Centric Communications

I-centric communications, putting the individual user (“I”) in the center of service provisioning rather than offering inflexible services that are unaware of actual customer needs or situations, is proposed in [21][22]. This paradigm argues to give more importance to individual communication needs, not to specific technologies. In the I-centric vision, I-centric services adapts to end users according to the ambient awareness, personalization and adaptation.

Compared to this generic model of I-centric communication in [22], MOCSP instantiates uniquely the I-centric vision for communication services. At this point, I-centric services (CS) in the MOCSP system are not concerned with ambient awareness, and adaptation, but it will be integrated later. Callee designs I-centric services (CS) in the MOCSP system, based on the context (who can be reached and how (media type)) and encapsulates the session control within it. However, MOCSP and I-centric communication take the top down approach for realizing the services.

IV. CONTROL SESSION IN THE MOCSP SYSTEM

As we have mentioned in Section III, each user session hides the control session that define and control media session end points. Main purpose of the control session is to control the media flow, to allow media change and codec negotiation. We choose the protocol for control session from [23], because:

- Compared to SIP, proposal [23] brings many benefits; it is unilateral protocol, based on TCP reliability; it adopts simple negotiation mechanism for codec choices; control session can manage and change the media flow in both directions separately and independently.
- Based on Proposal [23], MOCSP Web Server gains intelligence about media behavior of callee and caller. This intelligence is very useful to make decisions in forwarding messages; it can also help to reduce the race conditions in the control session.

The Web Socket is “TCP for the Web”, allowing bidirectional communications [24][25]. Therefore, we propose to use Web Socket for connecting callee and the MOCSP Web Server, and caller and the MOCSP Web Server. The MOCSP Web Server identifies the callee and caller based on the established Web Socket connections.

We merge the piecewise approach protocol (Proposal [23]) in the MOCSP Web Server and propose a call flow for a single user session/control session. Main intention of the call flow is to reduce latency in the call setup. All the steps needed are shown in a sequence diagram in Figure 4.

Initially, callee logs in to the MOCSP Web Server (MWS); it is shown as REGISTER message in Figure 4. The REGISTER message may have many request and response messages, but for the simplicity, it is not shown. In this phase, Callee Web Browser (Callee WB) creates a Web Socket and waits for receiving a call. Once caller clicks on the communication hyperlink (provided by callee), Caller Web Browser (Caller WB) sends an HTTP POST message to the MWS; it is shown as CLICK message in Figure 4.
Once MWS receives the HTTP POST message, it sends the open (medium, nomedia) message to the Callee WB over the established Web Socket. It means that caller is willing to send the medium (e.g. video or audio), but no description of the media flow. Callee WB indicates the call arrival and once callee accepts the call, Callee WB sends the oack(desc1) message to the MOCSP Web Server, indicating that callee is willing to receive the media on particular address, port number and priority-ordered list of codec that it can handle.

After the open (medium, nomedia) message to Callee WB, MWS sends the response message to caller (shown as ACCEPT message in Figure 4). Then, Caller WB creates Web Socket with MWS and sends the open(medium, desc2) message.

Later, if oack(desc1) message is received by MWS, and open(medium, desc2) message is also received by MWS, oack(desc1) message will be sent to the Caller WB. At the same time, MWS sends the describe (desc2) message to the Callee WB.

Then, Both – callee and caller WB sends select messages (select(sel2), select(sel1)); it means which codec they use to send the media. When callee and caller WB sends select messages, they start the media transmission. When Caller WB receives select(sel2), it is able to receive the media from other end. Similarly, when Callee WB receives the select(sel1), it is able to receive the media from other end. In any case, if caller, callee, or MOCSP Web Server wants to stop the session; they can issue a close message and stop it.

We can use 3PCC call flow (RFC 3725 flow 1) for establishing session in the MOCSP system, but it will increase the latency of call setup compared to the proposed call flow in Figure 4. Because 3PCC works on transactional mode; therefore, Caller WB responds to session once it receives the offer of Callee WB. In Figure 4, while MWS receives the descriptor from the Callee WB, Caller WB sends descriptor to the MWS.

V. IMPACT ON END USERS

The MOCSP system can be modeled as shown in Figure 5. This model offers an individual platform (merging the service layer and session control layer in IMS). Initially, we analyze the benefits MOCSP system brings more openness and flexibility. Later, we discuss two additional advantages from our approach such as end users’ privacy and scalability provided for creation of network centric services.

A. Openness

The end users gain complete openness in the MOCSP system since end users control the platform. Functionally, end users are freely able to define/change the control session for each user session without adding the complexity to the architecture. It means that fine-grained APIs can be easily defined and consequently develop service composition. End users need the knowledge of control session and Web programming thanks to simple and individualized platform. As a result, MOCSP system will help users to define new applications for their needs.

In this paper, we have not defined standard primitives that can help end users in reducing programming complexity with control session.

In fact, this openness can be concretely validated when many users implement different use cases or innovating applications in their own using this proposed approach. This user testing is important to define the protocol agonistic primitives. For achieving this goal, communication related resources at the component level (e.g. abstraction) for the MOCSP system are needed for reaching a large user base; it reduces the knowledge required to set up this system.
B. Flexibility

We present our design considerations that enhance the flexibility. Intelligence of the service is placed in callee’s side, callee’s side and MOCSP Web server. Conceptually putting more intelligence in the network (MOCSP Web Server) is the main design consideration. This consideration enables to reduce the work at the Web browser and to prevent the race conditions in the signaling protocols. Importantly, MOCSP Web server is not a back-to-back user agent in the signaling path (as in SIP), but behaves like a master element for session control.

Another design consideration is that when a service is developed, all the service logic resides in the Web server. Service clients are delivered dynamically to users upon start of a session. Therefore, if service logic is modified for a new feature, it is easy to implement only in one place.

In addition, the MOCSP system can implement different semantics for the control session for different communication hyperlinks (or user session). It means that all the communication session is private, only understandable by MOCSP web server, and user agent (callee, and caller) or semantics of a session need not to be globally understandable. Therefore, control session (semantic of the session) for each user session can be different. These changes are easily implemented. This flexibility in the MOCSP system supports end users to experience personalized communication services.

In Section IV, we present the general model for the control session which can be used for all the user sessions, but it is not mandatory to use one control session model for all the user sessions.

C Privacy

Today, Web 2.0 era, while users are inspired by Web 2.0 applications, they are seriously concerned of their privacy on the internet [15]. In fact, Web 2.0 applications do not concern with users privacy, thus leaking user privacy information easily or Web 2.0 application does profiling, analyzing and exposing of the personal information. Similarly, SP adopted many similar methods associated with user privacy: applying the data mining technique to extract the patterns from call detail data and inspecting and filtering the packets in the communication session to gather both historical and real-time information [16][17]. All these situations show privacy as a serious problem. Technically, MOCSP system addresses this issue for callee. It means that MOCSP system can only profile and share call detail data (i.e. session details) with others thanks to overall control to end users, thus overcoming the major privacy concern.

D. Scalability

In the IMS architecture, core network (a set of proxy) mainly performs the rendez-vous function. On the other hand, symmetric peers are key entities to manage the communication services. Therefore, network scalability during many simultaneous sessions is not an issue. However, some services should be developed in the network in order to enrich user experience. In fact, creating network centric services (e.g. network initiated session transfer) is a nightmare due to scalability issue. MOCSP system is an individualized platform, and therefore, it is easy to develop network centric services.

VI. Conclusion

This paper has introduced a new concept for inter-personal communications called MOCSP which provides more openness and flexibility for end users. We designed the MOCSP system, based on a top down approach for communication services in the Web platform. This individualized platform is callee centric because callee gains complete control of the communication platform. In addition, we are able to conclude that emerging new paradigm called “Web and communication services convergence for the users” is a reality in near future.

We have identified two future works. First, we will implement the system as designed in this paper for the validation purpose. In addition, MOCSP system can be today developed by Web developers or tech-savvy users; therefore we need to focus on more user friendly or usability aspect of the MOCSP system.

Moreover, Future Web will be designed based on social interactions, application needs and infrastructure requirements [2]. Personal communication can be considered as social interactions; MOCSP implementation is an application; therefore, it will create some new requirements at infrastructure level. In the future, MOCSP model will be considered as an example to provide inputs for defining the future Web infrastructure (i.e. defining communication building block/component) at W3C.

REFERENCES


