Supporting conversational voice and video in the web browser/runtime

By Stefan Håkansson (stefan.lk.hakansson@ericsson.com), Hubert Przybysz (hubert.przybysz@ericsson.com) and Piotr Kessler (piotr.kessler@ericsson.com) of Ericsson AB

Introduction

Enabling the web browser/runtime to support conversational voice and video would be a big step forward in the process of making the web browser/runtime a complete application platform. Since many users hesitate to download and install applications for different reasons but use and trust the browser for all kind of services (from social networking to banking transactions) this would increase the device reach for conversational voice and video services. Another factor is that much of the service innovation happens in the web domain. Combining existing services to a new service (a.k.a. mashup) is a well known way to innovate, and adding voice and video to the toolbox would likely generate new inventive cross platform services in the browser environment. This will in turn increase the use of new web services enriched with voice and video communication.

There are at least four factors that must be fulfilled to make web browser/runtime supported conversational voice and video a success:

- It must be convenient and feel safe to use for the end user
- It must be easy to use for the web author
- It must be accepted and implemented by web browser/runtime vendors
- Interoperability between implementations must be secured

Use case aspects

When adding voice and video communication capabilities to the web browser/runtime several use case scenarios where it would be advantageous, or essential, to add conversational voice and video communication can be envisioned (and hopefully even more will be invented by web authors!).

We would like to discuss use case aspects in different dimensions.

End-user service type

The end-user service type can range from advanced collaboration services via situations where voice and video is added as a component (or spice!) to another end user service (a social networking service, an online game) to a service that is similar to telephony (but presented and controlled through the web browser/runtime) and interconnects with a telephony system.

Another aspect is the number of concurrent sessions (in the device) that are able to use conversational voice and video. In many cases the user will be active in several concurrent services, perhaps displayed in different browser tabs.

Yet another aspect is that the user can be concurrently on-line in the same service on several devices.

Device and deployment

The device type can range from a desktop computer to a smartphone, with vastly different capabilities – from audiovisual rendering and capturing capabilities to processing power and energy resources (battery capacity is an important aspect of handheld devices). Current sales figures indicate that smartphones will be the most common device type.

The web browser/runtime can range from browsers that are downloaded and installed on the device post-sales to web runtimes that are deeply integrated in the device platform. This in turn implies different capabilities and performance. In the first case the codec implementation could be part of the browser while in the latter case hardware optimized implementations of e.g. codecs would usually be reused. Since there is a continuous development of media compression methods different generations of implementations can support different codec sets (and it may not be possible to upgrade older ones due to e.g. processing performance)..

Different device types, equipment and situations also generate different user behavior. In some cases the device is held against the ear, in other cases a headset is used, and in yet other cases the device is placed at a distance and just talked/listened to.

Another aspect is application interoperability where applications, including voice and video, can be provided by different sources and have differences in functionality.

Access technology

Different access technologies have different capabilities and characteristics. It can range from giga bit Ethernet to different radio technologies.

Another factor is that the access availability can change – e.g. you can move out of WiFi coverage (but remain in cellular coverage).

Requirements

Based on the use cases above and with some reasoning, the following requirements can be derived:

Session handling

Apart from the obvious task of setting up a session the sessions handling mechanism be able to handle a lot of other tasks. It should be able to

- Handle negotiation of media formats as well as adding and removing media components
- Handle changes of access network (e.g. moving from WiFi to cellular networks or vice versa) on the fly
- Handle several concurrent services with voice and video capabilities (i.e. there must be a mapping between the session and the application)
- Handle several concurrent voice and video sessions within the same application
- Interconnect to telephony systems

Security and privacy

It is very important that the high levels of security and privacy can be guaranteed. The area can be broken down into at least the following sub areas:

- Authorization of application usage of input resources (cam and mic)
- User and server authentication
- Protection of signaling and media

Media related functionality

Set-up and modification of media formats

It must be possible to negotiate media formats during set-up and during the session to be able to use the most efficient format for device form factor and network capacity, and to be able to re-use device implementations of media functionality.

Media and transport formats

To guarantee interoperability there should be at least one audio and one video format that is mandatory to support. Given that smartphones will be the largest terminal segment the codecs (optimized for battery efficiency and network load) used for their default communication services should be part of the mandatory codec set.

Long media path delay affects the conversational quality negatively, therefore media and transport formats that reduce the delay should be selected when possible.

Media processing

For a satisfactory user experience it is important that there is support for

- Acoustic Echo Cancellation (to enable use without head set)
- Jitter and loss management (to handle transport imperfections)
- Noise reduction (enhance the user experience when the other party is in noisy environments)
- Media Rate adaptation (to handle varying network capacity, to avoid starving other traffic)
- Audiovisual synch
- Gain Control

NAT and FW traversal

There will always be cases where one or more NATs or FWs are present in signaling and media paths. Therefore it is important to support traversal of such elements in order to enable voice and video conversations. It should be transparent for the application.

Notification

It is important that the user can be notified when someone is inviting her/him to a voice/video session. For some use cases, such as telephony-like services, it could be argued that it should be possible to notify the user even if the browser is not running. It is important that the web author can control how the user is notified (pop-up, visual signal, audio signal etc.)

QoE monitoring

In some cases it is important for the service provider to have knowledge about the quality of service that the end user experiences. W3C recently started a new working group that will propose solutions for the User Agent to report data like page load times. We think the solution should contain ways for the service provider to monitor quality related to real-time voice and video. Important such metrics are codec, bit-rate and packet loss rates.

Capability discovery and exchange

As terminal capabilities in terms of e.g. camera, microphone and screen size will vary there may be a need for capability exchange between endpoints. This implies also that the web browser/runtime must be able to find out the capabilities of the device. Similarly applications should be able to express their capabilities (for instance voice, video or both).

Web Author related requirements

From the web author perspective there are at least two requirements.

One relates to the design of the API. To make it simple to develop basic voice and video supported services the API should be aligned with existing web APIs and easy to use. The API should also be flexible enough and expose enough detail to allow for innovation – the web author's ambitions should not be hindered by a cramped API. A specific requirement is that media components should be easily enabled/disable to enable adding and removing media during a session.

The other relates to control of the rendering. The web author must be in complete control of the rendering (and be able to e.g. apply CSS transforms to it).

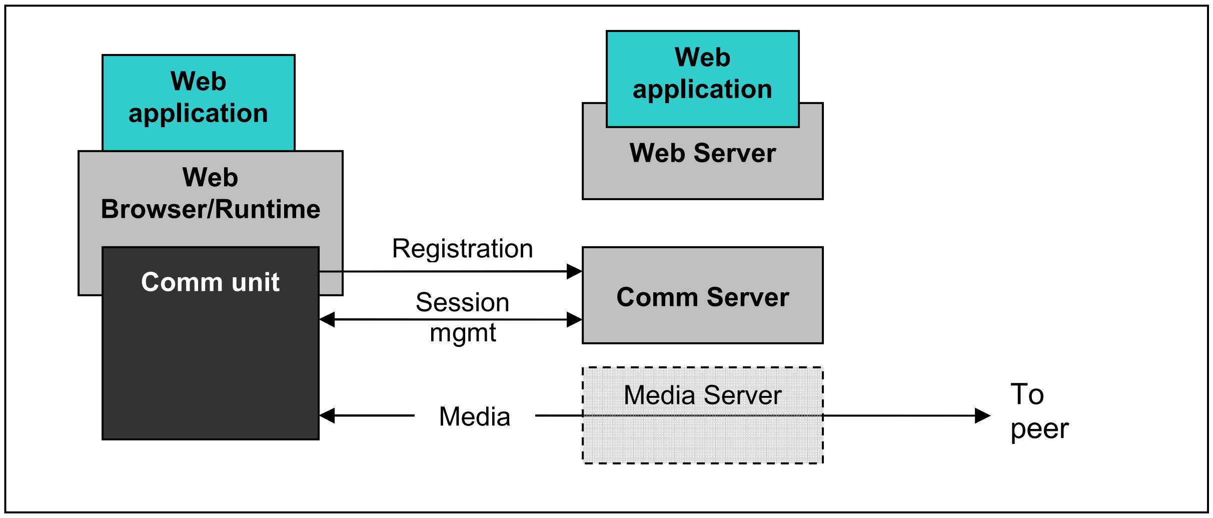
Interoperability

To enable interoperability between web applications developed by different authors the solution should provide an interoperability mechanism.

Proposed solution

Overview

We think it makes sense to have a design that builds on the concept of a rather self contained unit that manages the communication services. The web author would interface with the self contained unit via an API (available on a few different levels to suit different scenarios). Media would be presented to the user via the already available html elements <audio> and <video>, thereby giving the web author full control over the presentation. Corresponding solutions should be used for controlling media capture. In the picture below this is outlined. The application is naturally served from the web server. It would (via the API) use the self contained unit to register with a communication server in order to announce its capabilities and its ability to receive session requests. When starting to communicate with voice or video the self contained unit uses the communication server to set up a peer-to-peer media connection (in special cases the media may have to pass a media server).



Properties of the self contained unit

Choice of protocols

SIP and SDP are standardized and widely used protocols for handling conversational sessions. When adding the capability to initiate and terminate audiovisual sessions in the web browser/runtime, SIP and SDP should be supported as protocols for session handling. There are several reasons for this. One is (as indicated) interoperability with existing services and equipment. SIP with SDP does also offer a well developed solution for negotiation of media formats, encryption key exchange, adding/removing/manipulating media components during a session etc.

A lot of effort and energy has gone into solving different aspects of audiovisual communication using SIP/SDP. This means that it is tried and proven to work. While there may be other ways to implement the required functionality they would in the end have to implement similar functionality as provided by SIP/SDP. SIP/SDP is already available for fast deployment.

Secure layer protocols for signaling must be supported using TLS or IPSec.

RTP/UDP, with support for SRTP, is the natural transport format for the media components.

Registration

In order to be addressable and reachable by peers and to be able to send and receive voice and video the application must register in the self contained unit. Many applications may concurrently be registered. SIP REGISTER should be used to register in the communication server. The user and the communication server would be authenticated as part of the registration procedure. A secure connection should be created for the SIP/SDP signaling.

Session handling

SIP/SDP provide mechanisms for flexible and dynamic session handling. Those mechanisms are exposed to the author via the API to enable creation, modification and termination of conversational sessions. The API should provide a callback when a conversational session is initiated by another peer. The self contained unit identifies the application to callback based on registration data.

Changes of access networks should be managed by the self contained unit. SIP/SDP should be used to handle session handover between access networks.

Media plane

While there are no standardized solutions for echo handling, noise suppression, jitter management, packet loss concealment of gain control there are effective implementations available. The self contained unit should include support for these functions.

Other parts can be solved using standardized technologies, and some of those will be discussed:

Codecs

In addition to supporting the (or at least one out of a limited set of) default codec(s) for voice and video, other codecs (either added as software plugins or being part of the device platform and potentially HW accelerated) must be possible to use.

A large number of mobile devices include HW accelerated support for coding of voice and video in formats that are currently used for voice and video communication. Interoperability with these devices should be provided on a codec level. These codecs are also optimized for cellular networks.

In addition the self contained unit must be able to use codecs that are supported by the device platform.

Rate adaptation

A lot of work has gone into rate adaptation for media streams (not using TCP for media transport). In IETF algorithms like TFRC and TFRC-SP and protocols such as DCCP have been developed for this purpose. We think a solution based on existing IETF technologies for RTP/UDP flows should be used. The rate adaptation should be handled by the self contained unit and not be possible to manipulate by the web author (except possibly the possibility to prioritize between flows of the application in question).

Audiovisual synch

Audiovisual synch can be achieved by using RTP time stamps in combination with RTCP.

NAT and FW traversal

We think that solutions for NAT traversal for RTP/UDP-based voice and video media and for SIP signaling should be based on existing IETF technologies like ICE (RFC 5245) and Outbound (RFC 5626). Such complexities should be made transparent to the web author and should therefore be handled by the self contained unit. The communication server and the media server will help the self contained unit in establishing signaling and media paths through NATs as needed. Care must be taken to minimize keep alive traffic for battery preservation.

Authorization of use of mic/cam

This topic has been discussed in W3C DAP and in WhatWG. Different solutions (PowerBox, device element, …) have been proposed. We think that the proposed device element (opening a dialogue where the user must select camera and microphone to use) could be a starting point, but it must be combined with solutions that clearly indicate that the cam/mic is in use (perhaps by icons in the browser chrome).

QoE monitoring

In WhatWG a discussion on reporting metrics from the video element has been discussed. There are past and ongoing activities in IETF. We think that these solutions could be combined into a solution for this purpose.

API

We think it would make sense to have several levels of API. High level APIs should be the base for quick and easy implementation of audio and video conversational services. Additionally the details of SIP/SDP and media management should be exposed on lower level APIs.

Notification

Presumable the work by the W3C WebNotification WG can be used for user notification. Persistent SharedWorkers could be a tool to enable notification when the browser is not running.

Ways of working, what to produce

The proposed technical solutions in this document essentially fall into the categories protocols, codecs, algorithms and APIs. The protocols proposed are all standardized by IETF. Similarly, standardized codecs are available. The algorithms in question (such as for rate adaptation, gain control, echo cancellation) are usually not standardized. Finally, the APIs are yet to be defined.

Our proposal is to agree on the protocols, default codecs and default algorithms in an appropriate forum. A way to speed deployment in web browsers/runtimes could be to carry out the work as an open source project. Assuring interoperability is an important part of such a project. The software should be designed in such a way that new functionality is easily added and that device functionality (e.g. HW accelerated functions) could easily be incorporated.

The APIs should be standardized in W3C since that is an organization recognized by web browser/runtime vendors and by web authors.

Summary

This paper has discussed factors for success, use cases and requirements for real-time voice and video communication in web browsers/runtimes. A solution where the related processing is carried out in a self contained unit that the web author interacts with via APIs is proposed. Technology choices for the self contained unit are proposed, the main ones are:

- SIP (with SDP) should be supported for session handling
- It should be possible to negotiate media formats between end points
- It should be possible to re-use platform support for codecs and other blocks
- RTP/UDP should be used (when possible from a network topology perspective) for media transport (SRTP for media protection).

Most components of the self contained unit are readily available (either as standards or proprietary algorithms), but how they are used and combined must be agreed on.

The APIs for the web authors should be available on different levels and should be standardized in W3C.

A way to speed up development and to ensure interoperability could be to jointly develop an open source implementation of the self contained unit.