30-03-2016

SA8T2 Internal Deliverable
WebRTC MEDIA API Service proposal

SA8T2 Internal Deliverable

Contractual Date:	30-04-2016
Actual Date:	30-03-2016
Grant Agreement No.:	691567
Activity:	12/SA8
Task Item:	Task 2 – WebRTC
Nature of Deliverable:	R (Report)
Dissemination Level:	PU (Public)
Lead Partner:	NORDUnet (UNINETT)
Authors:	Frédéric Loui (RENATER)

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 691567 (GN4-1).

Abstract

This document advocates a set of tools (as a JavaScript library) that can achieve interoperability with (and within) already established and coming R&E services. The work was conducted by the Service Activity 8 (SA8, Real Time Communication and Media), Task 2 (WebRTC) team as part of the GN4-1 project. This report should, as such, be read in context of the related work produced by GN4-1 SA8-T2.

Table of Contents

Table of Figures

Executive Summary

WebRTC enjoys growing attention — particularly in regard to how it enables browser-based audio/video conferencing without the need for plugins/extensions. A number of vendors and projects aim to deliver the next alternative to established communication services, such as Skype (actually, even Skype itself will be implementing RTC standards). Audio/video conferencing is indeed a prominent component made possible by the technology standard, and thus an obvious use case. However, WebRTC's concentration on the web browser, standards and technologies provide opportunities that transcend traditional conferencing services. Thus, the notion of a "new Skype" that runs in a web-browser is only scratching the tip of the iceberg.

Key advantages of WebRTC include:

- All components involved in the standard can be implemented on commodity hardware
- Interworking is guaranteed if the implementation is compliant to the standard
- Integrating with other web technologies is straightforward
- Security and management policies are streamlined as communication is part of the data traffic
- Unified communication has a new channel that can, and should, replace legacy telephony and traditional h323 solutions that present inherent constraints
- Last, but not least, in-context application development is made possible — WebRTC enables and improves R&E business application experience

WebRTC presents a very tangible opportunity to change and add value to the way we work, communicate and collaborate in our R&E communities today. Its advantages provide both opportunities and incentives worth careful investigation, with a number of them presenting key differentiators compared to traditional technology. The last point above, in particular, is the main driver of the service development proposal declared in this document.

In a nutshell, "in-context" application development refers to the integration, or embedding, of one service inside another. A novel example to illustrate this would be a classic support ticketing system, where multiple emails are typically exchanged between the helpdesk and the end user. An audio/video/chat service built into the ticketing system facilitates real-time support that practically eliminates the volume of mail exchanges, reduces lead time and complexities — both in getting help as well as providing timely and relevant support.

Our proposed notion of a Media API Service advocates a set of tools (as a JavaScript library) that can achieve interoperability with (and within) already established and coming R&E services (e.g. as in our example above). We put forward an application programming interface (API) to assist development and control of real-time communication applications that can "live" inside another service and integrate with (and extend) its functionalities. A Media API Service achieves this by providing a layer of abstraction that makes implementations more straightforward and reusable (thus stimulating use and reducing time/cost). The implied use-cases are limitless and the benefits potentially priceless.

Further to facilitating easy front-end development, applications will need to leverage WebRTC-compliant middleware for load balancing and bridging. The API will also expose simple routines that permits the developer to connect their application to this middleware.

1 Introduction

1.1 About this document

This document advocates a set of tools (as a JavaScript library) that can achieve interoperability with (and within) already established and coming R&E services.

The work was undertaken as part of the Geant4 Phase 1 project by the WebRTC Task 2 (T2); one of three tasks of the Real Time Communication and Media activity (SA8). This report should, as such, be read in context of the related work produced by GN4-1 SA8-T2.

The WebRTC task ran from 1 May 2015 to 30 April 2016.

1.1.1 Target audience

This document targets technical management and specialists, in particular those working in the fields of real time communications, eLearning and eResearch.

1.1.2 Responsible task members

Frédéric Loui (RENATER) had the lead on this tech scout. Jan Meijer (UNINETT) and Simon Skrødal (UNINETT) were the document editors.

1.2 Background

An application programming interface (API) provides the developer with building blocks to make service development faster and easier. Faster, because the API typically provides well documented and well defined routines to achieve frequently requested functionalities. Easier, because the API abstracts away many complexities of the underlying systems. Since APIs define a standard way of operations (e.g. routes, protocols, data structures) they implicitly promote reuse.

SA8 Task 2 proposes the development of a Media API Service that advocates a set of tools (as a JavaScript library) that makes it trivial to add high level WebRTC components to any web application. The API service can be integrated with other central, national or institutional infrastructure components, e.g. recording services, authentication and authorisation, logging, captioning, presence, legacy infrastructures (SIP), etc.

The proposed service will allow for the creation of a standard set of well-functioning virtual lecture rooms that are easy to embed in any virtual learning environment. It will also facilitate standard components for multi-party videoconferencing to be embedded in the context of any other web based tool. The service will lower the cost for contextual communication as well as to increase the quality of such integrations.

By allowing a deconstruction of virtual room functionality the service can, in a simple and affordable manner, be used to select and add feature and UI combinations/subsets inside any web application. The implied use-cases are limitless and the benefits potentially priceless.

1.3 Rationale

WebRTC may be used to enable in-context communication (i.e. embedding RTC capabilities inside a non-RTC web application) at low cost. The proposed Media API Service facilitates domain-specific in-context communication to virtually any web-based service.

Requirements for contextual communication in R&E are normally quite similar, thus offering the potential for reuse. The Media API Service proposition promotes high-level components to assist web developers in European R&E and spare them from "re-inventing the wheel".

In addition, a Media API Service would allow the R&E community to integrate key value-adding services at a central point. Note that this does not necessitate the services themselves to be centralised.

A GÉANT Media API Service will reduce integration time and cost for developing contextual communication in web applications, as well as increase the quality of the individual components.

1.4 Tech scout objective and methodology

The objective of this work was to describe the idea for a shared Media API service for R&E in sufficient detail to allow a collaborative activity for further investigation to be planned.

The ideas presented in this document are based on Task 2 work with Jitsi functionality, Task 2 team discussions and observations in both the WebRTC market and our R&E user communities.

2 Proposed service functionality

The idea for the Media API Service was originally inspired by Jitsi-Meet; an open source (WebRTC-powered) video conference application, which exposes an API to embed meeting rooms within 3rd party web applications. The Jitsi-Meet API documentation for a future version illustrates well some aspects of what our proposed Media API should cover:

- https://github.com/jitsi/jitsi-meet/blob/lib-jitsi-meet/doc/API.md

Another free (though not open source) video calling service, appear.in, also provides an API (albeit very basic) for embedding meeting rooms that may readily be tested:

- https://developer.appear.in/

Operations that should be supported by our Media API Service include:

- Room operations, e.g.:
  - creation, deletion
  - alter ROOM's behavior, e.g. display participant's thumbnail or not, organize predefined thumbnail layout, full screen mode
- Participant operations, e.g.:
  - invite/add participants to an existing room
  - kick/remove participant from room
  - dynamically display Last-N thumbnail per room
- Participant streams operations, e.g.:
  - mute/unmute participant audio
  - mute all participant audio except for the speaker
  - disable/enable video for one participant or for all participant
  - force the quality selection of the media
  - mute all
  - display participant's avatar instead of the video
  - audio/video recording
- Reporting operations, e.g.:
  - expose API to provide quality KPI per participant (Mean Opinion Score, BW (down)/(up)load, packet loss, jitter) so the application can be aware and react accordingly to inform the user about potential issues degrading the user experience. But information about the duration of a communication room can be useful.

To the best of our knowledge, there are no products/services that provide these functionalities today, though this will need to be investigated further, preferably in a collaborative action by the GÉANT community.

In-context application development is made possible by WebRTC and related web-technology standards. While it is not possible to achieve the same with traditional services (e.g. you can't embed a Skype desktop client into a website), some web-based systems do indeed provide live chat as a feature (think customer support sites). The Media API Service differs in that it provides real-time communication (audio/video/data) as a service (front-end and middleware routines) that may be plugged into *any* web-based application. Hence, it is not built for one specific purpose or system only, but provides API components that may be mixed, matched and reused in services that may benefit from having inbuilt communication components.

Software developers struggle with the complexities involved in providing public switched telephone network integrations (PSTN) for their business applications, such as CRMs, trading solutions and on-call center software applications. The Media API Service intends to provide an interface that will not only simplify such integrations, but also provide further value-adding qualities. The API provides easy to use building blocks that makes extension and integration development simple and flexible. By invoking the Media API Service, each organization can tailor their applications to their business needs and extend them with audio/video capability with ease. While the service is handling the underlying communication complexities, the application would be able, via the API, to control the media behavior. Our proposed API provides a solution that has been sought after for more than a decade.

Since the service middleware (e.g. load sharing and video bridges) can run on commodity hardware, the cost of entry and scalability are significantly lower. Take for example the Jitsi Video Bridge, an open source WebRTC compatible SFU (Selective Forwarding Unit) that allows for multiuser video communication. The Jitsi SFU can handle 1000 video streams using 550Mbps of bandwidth with only 20% CPU:

- https://jitsi.org/Projects/JitsiVideobridgePerformance

Assuming the service is inherently ready to work in a load sharing fashion, the service can work at a very high scale at low cost, allowing thousands upon thousands of simultaneous video streams within and between our communities. Load sharing across multiple Jitsi Video Bridges, for example, is enabled by Jitsi Jicofo.

Irrespective of middleware being used for bridging and load sharing, our Media Service API would abstract away from the application developer any complexities involved with communicating with this layer.

3 Value proposition for a GÉANT WebRTC Media API Service

In this section we provide details of our thoughts on the value proposition of a shared WebRTC Media API service for R&E.

3.1 Offering

The Media API Service offers:

- A turnkey Media API solution dedicated to high quality multimedia unified communication
- For a high number of users leveraging if necessary NREN community cloud IaaS
- In a multi-domain environment
- A unique R&E community driven service
- A very affordable price compared to the existing traditional H323 based services
- A service that can be seamlessly integrated within the existing NREN infrastructure

The Media API Service makes it possible to create and integrate in-context applications, while hiding the underlying complexities from the developer. Web applications from any area can be extended to embed a module using the Media API Service, thus enabling real-time audio/video communication between people currently collaborating inside the application.

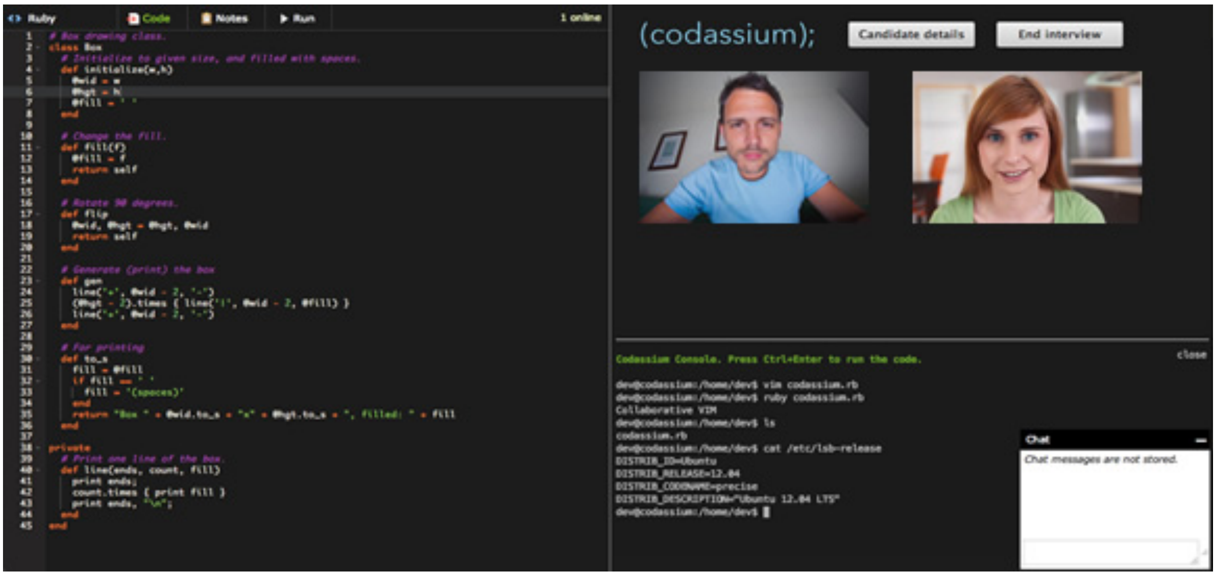

Figure 1: Codassium — "Hiring a coder" as a Service (https://codassium.com)

Use cases close our R&D communities include, but are by no means limited to;

- LMS (e.g. open meeting room on a subject page)
- MOOC (real time communication with/between students from around the world)
- University website (Q&A sessions for prospective students)
- Tutorial apps (guidance and collaboration on any subject)
  - e.g. WebTut (https://wiki.geant.org/display/WRTC/GN4-1+WebRTC+Roadmap)
- Examination (practical examination on any suitable hands-on subject, e.g. programming)
- Survey/questionnaire applications (e.g. for implementing mixed-methods research — quantitative/qualitative, interviews), which could also implement speech recognition for on-the-fly translation/transcription.
- Research applications (e.g. researchers in different locations may get together and interpret results in real time without having to exchange emails)

More general use cases include implementations for for tele-medicine (e.g. doctor-patient meetings or, indeed, diagnostics/treatment if combined with sensor-technology), marketplace/marketing, surveying, help/support portals, surveillance, social media platforms, etc.

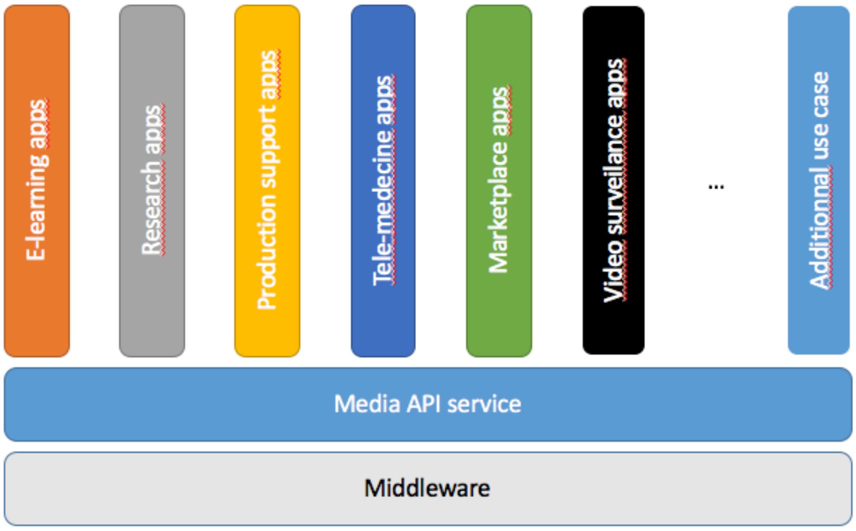

Figure 2: Sample use-cases for the Media API Service

3.2 Service delivery model

An important aspect of the API service is to simplify negotiation and communication with middleware infrastructure, which involves firewall/NAT traversal (e.g. see relevant work by SA8T2 on STUN/TURN/ICE, https://wiki.geant.org/display/WRTC/GN4-1+WebRTC+Roadmap), video bridges and load balancing. This middleware comprises enabling/supporting services that must be in place for the API to serve any purpose. Identity Federation is also an important facet in this context, which needs to be investigated further.

The Media API Service should cater for any middleware configurations, be it one or many SFUs, SFUs of different brands, or even a full-mesh configuration. NRENs can, in a joint effort, make multiple SFUs available (even from different brands). The API would then have access to an unmatched amount of video conference/media resources and thus scalable to any magnitude.

The envisioned service providers could be GEANT and any NRENs prepared to deploy a media resource (at least 2 for redundancy purpose). In special cases, where an NREN provides a community cloud service, even more video bridges could be hosted (thus ensuring not only redundancy but also elasticity property to the service). As a highly scalable service, video conferencing can be made accessible from any web-application and provides the opportunity for ALL staff and students at schools, colleges, universities and research institutes to simultaneously access the resource.

WebRTC components are maintained using a Cloud orchestration technique able to deal with self-contained application mechanism, such as the Docker Cloud application distribution format. It is therefore a cooperative model proposal within the NREN and GEANT community.

The service is instantiated inside a specific multi-domain WebRTC VPN or WebRTC MD-VPN. Considering the traffic matrix and the potential security concerns that might be raised there are two MD-VPNs:

MD-VPN #1: Any-to-any user traffic

VPN encompassing WebRTC application traffic. Essentially, it is the VPN hosting multimedia traffic between the NREN users inside the NREN community (network). This VPN is having Internet gateways so that participants outside the WebRTC VPN can access the service. In case of security issues from the Internet, focusing only on these gateways is easier in order to mitigate the attacks.

MD-VPN #2: Any-to-any out of band management

VPN used to administer and maintain all the WebRTC components. Software upgrade and maintenance must reside in a secured VPN dedicated to specific and identified hosts. This VPN will also have an Internet gateway, but will be restricted to reach only the identified resources in order to ensure maintenance of the WebRTC software and operating system security patches.

Internet gateways for GEANT WebRTC Media Service

When all the participants are on-net (i.e. located inside NREN networks) the traffic is flowing via the user traffic MD-VPN. External users (off-net users) need to connect to an existing room from the Internet. This introduces traffic from outside of the NREN networks, thus the requirement for Internet gateways.

4 Compelling reasons to develop the service

Traditional standards lend themselves to different interpretations by commercial vendors. This makes it difficult, and in some cases impossible, to achieve interoperability between proprietary RTC services. Further, these services typically run on *expensive* proprietary hardware that severely counteracts scalability. Few organizations can justify the investments demanded to make an audio/video conferencing service available to a large number of users. More overheads are again introduced in the procurement cycles, which organizations in the public sector must typically handle every 2-3 years.

The point should also be made that proprietary solutions demand specific training for engineers to properly understand and support the technology supporting the service. Consequently, these engineers are confined within the boundaries of the proprietary technology.

WebRTC and related web technologies, on the other hand, represent quite the opposite, where acquired skills are easily transferable and made relevant in other contexts. Actively standardized by the W3C and IETF, WebRTC components may run on low-cost servers without the need for dedicated proprietary hardware. Service deployment models may also leverage cloud environments that inherently provide elasticity property, meaning that the service can be made accessible to an infinite number of users encompassing our entire R&E community. The Media API Service is tied to a traffic matrix well identified and that can be secured based on existing IT security policy already in place within the organization.

For CIO's facing cost reduction pressure, this service is thus a tremendous opportunity to provide a high quality – and scalable – collaborative platform at a very reasonable cost when compared to the traditional audio/video conference technology.

Based on web technology, integration within existing IT contexts and integrations of extensions within the service is possible without having to purchase northbound or southbound proprietary interface/connector. Support for legacy systems ensures interoperability and that prior investments do not go to waste.

The API service can support any system, commercial or open source. Within the GEANT context, the NREN community has the advantage of being able to take active part in a community-driven open source service utilizing innovative technology.

In the long run, Unified Communication will be ubiquitous and our proposed Media API Service will scale with the rapid changes in the technology and services landscape. It will provide building blocks essential to enable straightforward and cost effective development of collaborative components for new and existing applications. Every arrow points (and has been pointing for a long time) in the direction of open and standards-based web technology, interoperability and a substantial shift away from the traditional closed legacy and proprietary paradigm. It is a fact that our R&E community already has to relate to, but even more so in the future. To embrace these shifts sooner rather than later is a matter-of-fact that should take a pragmatic approach.

4.1 Technological Environment

WebRTC works on just about any end-user device, and just about any end-user has a device. It also runs on just about any server hardware (as opposed to proprietary systems), which allows for open, low-cost and incredibly extensive use cases.

WebRTC is designed to work in a cross-platform web environment on hardware that is accessible to virtually everyone in our community, e.g. smartphones, tablets and personal computers. The hardware found in these devices typically have the capacity to several codec streams at the same time. Since the consumer hardware is powerful, extremely competitive in price and available in such incredible numbers, WebRTC enjoys market penetration on just about anything that runs on electricity. The market shows no signs of slowing down, and various implementations of “the Internet of things” are also starting to leverage WebRTC. Put simply, the current technology context favors the development and adoption of WebRTC.

The network may also be configured to take into account multimedia traffic, with an awareness of various application criteria, and this is stimulating the evolution of current network designs to make them more programmable and application aware.

Good and affordable audio quality is re-enforced by one keyword — convergence. This is changing the working practice by enabling multimedia conference, not only as per usual in the office within the organization, but also in a high mobility context. Further efficiency in the organization is now possible by integrating WebRTC-based services with the existing security mechanisms. Authentication, authorization and accounting via the eduGAIN Identity Federation is the stepping stone for a secured R&E unified communication service.

As such, a shared GÉANT WebRTC Media API Service is providing a ubiquitous way to provide multimedia and communication means to every single person of the R&E community by leveraging identity federations.

4.2 Benefits

By subscribing to the GÉANT WebRTC Media API Service, direct benefits to the NRENs are:

- Low-cost and easy access to a highly scalable unified communication service able to cope with any number of users.
- This service can may be used to extended multiple web service applications.
- NRENs are not only customers of the GÉANT WebRTC Media API Service, but also welcome to play an active role in the service offering (e.g. by hosting resource components to support

resiliency and scalability). Elasticity can be enabled by the NRENs' private cloud or provided by a third party community cloud from other NRENs.

- The service does not aim to replace traditional video conferencing services; WebRTC based services are complementary to, and may be integrated with, existing NREN traditional services.
- Last but not least, the GÉANT WebRTC Media API Service is one piece of the unified communication puzzle proposed by GÉANT and NRENs. As a community, it is possible to address a greater challenge and at some point influence the industry by providing a high quality service and extending our user base.

4.3 Cost

GN4 Cost Model Working group would address this topic. In a nutshell, the objectives are to find a model where cost recovery is guaranteed to all NRENs and for GÉANT to participate with the proper level of service level support. (SLS)

4.4 Schedule / timeline

A shared GÉANT WebRTC Media API Service could be available (ready for subscription) in about 3 years..

4.5 Engagement

As a community-driven service under the umbrella of GÉANT Project, users can provide feedback to their local NREN that in turn can relay the suggestions (bug reports, request for improvement etc.) to the GÉANT Association. At that stage, all of these requests are fed back to:

- GÉANT NOC if this is a service issue related to the infrastructure supporting the service.
- The PARTNER NREN responsible for the service maintenance in case of bug report
- GÉANT WebRTC media service steering committee working group in charge of evaluating the RFI.

The proposed service is therefore managed by an existing streamline supervision and operational process.

5 Conclusions

This document described and proposed a Media API Service essentially comprised of easy to use (and understand) web code that could be integrated with existing web applications. The API suggests to abstract away the complexity of setting up and handling communication between peers, and to provide developers with the means to embed and control media communication in their applications. The API promotes a new form of communication made possible directly inside business applications by enabling an innovative way of collaboration. The underlying technology allows the service to run on commodity resources, such as virtual machines within the NREN's IaaS infrastructure, and be accessible through the R&E/GEANT networks.

The joint effort of NRENs putting together IaaS and network resources under the coordination of GEANT would build a gigantic heap of media resources to serve applications development. The Media API Service depends on other "enabling services", such as STUN/TURN, federated authentication and authorization and "presence management". Combined, these services provide a base upon which a unified communication platform, able to support R&E applications, can be built. Recent WebRTC advancements, combined with other favorable technology improvements, are propitious to build value-adding in-context applications not possible with traditional videoconferencing systems.