WEBRTC FOR BUSINESS PEOPLE

2025 Edition

Unraveling the challenges and opportunities of the WebRTC ecosystem

Sponsored by Ant Media kaleyra NIMBLE APE webRTC.ventures

Tsahi Levent-Levi
tsahil@bloggeek.me

Table of Contents

List of Figures

Introduction

Why WebRTC for Business People?

WebRTC is a transformative technology to the way we think about digital communication. The effect WebRTC has over the way we communicate and how people's expectations from the services they consume is usually understated or dismissed.

The first edition of this report was published when WebRTC was brand new. People asked if this is a real thing that will get widespread adoption or not. Today, that's not the question anymore. We've veered from should we use it, to how do we use it, and then from there to how do we differentiate. Today? We're dealing with how to fuse it with generative AI.

The purpose of this research paper is to explain what WebRTC is, what it can and can't do, outline the ecosystem around WebRTC and provide a healthy dose of vendors who have already taken the plunge with WebRTC and have built a business around it.

At the end of each chapter, you will find links for extra reading material – some from my own writing and some from bright minds from around the web. At times, the same link will appear in more than a single chapter.

Why this 2025 Edition?

We've been in and out of a pandemic. We've had billions of people indoctrinated to how to conduct remote video calls using their phones, tablets and computers. And now we're playing with ChatGPT and thinking how generative AI can be glued to anything and everything.

These changes are affecting also communication technologies and WebRTC. As such, it is important to take them into account and explain them as well. It was high time for a refresh of this report.

In this refresh, some of the vendors profiled have been replaced with a fresh set of vendors, noting the change in market focus, and to an extent in my own focus. I tried to

shy away from most of the large brands (when possible), focusing on niches or on cases where the story of the vendor was important.

About the Author

My name is Tsahi Levent-Levi. I am a developer at heart. I have been working in the telecom and VoIP/UC industries for the past 25 years (and counting) in various roles: from a developer to project manager, product manager, CTO, CPO, co-founder and CEO. Most of that time, I was dealing with signaling and media products that were licensed to other developers who built their own products with them. This gives me a broad view of the market and an understanding of the challenges and opportunities that exist in the domain of VoIP and interactive video.

I came across WebRTC when it was first announced by Google and saw the potential in it. Since then, I have been watching the WebRTC space closely and writing about it on my blog: BlogGeek.me. From a hobby it became a "profession".

Today, I provide consulting services around WebRTC, CPaaS and ML/AI in communications, as well as offering an online course on WebRTC (webrtccourse.com).

I act as Senior Director of Product Management Cyara, after its successful acquisition of Spearline. Spearline acquired testRTC, a company developing a testing, monitoring and support platform for WebRTC applications. I was the Co-founder and CEO of testRTC prior to the acquisition.

Thanking our Sponsors

This new updated version of the report wouldn't have been possible without its sponsors.

I want to personally thank Ant Media, Nimble Ape and WebRTC.ventures for sponsoring the 2025 edition.

Ant Media kaleyra NIMBLE APE webRTC.ventures

Be so kind as to check the sponsor's services by clicking on their logos above 😉

What is WebRTC?

WebRTC stands for Web Real-Time Communication. It is the fusion of two separate branches in technology: VoIP and the web.

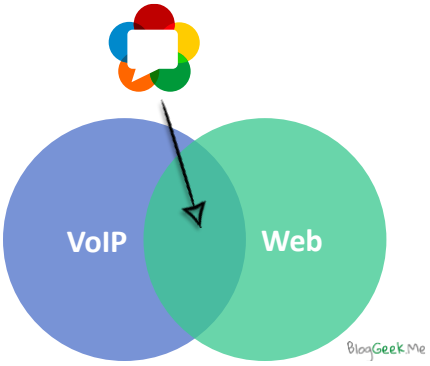

Figure 1: WebRTC sits at the intersection between VoIP and the web

VoIP, short for Voice over IP, is a set of technologies and techniques that enable sending media (usually voice and video) over an internet connection. Up until the introduction of WebRTC, VoIP lived within its own ecosystem silo, next to the booming Internet we are accessing daily via web browsers.

WebRTC comes to connect VoIP into our browsers, and by that, into websites and mobile apps. It does that by offering a thin layer of JavaScript APIs that are implemented by modern web browsers and are part of the HTML5 specification.

This means that now every web developer can add real time communication capabilities to his website or web application.

> ☞ *Further Reading:*
>
> - The history of WebRTC inside Google [Quora]
> - WebRTC - Realtime Communication for the Open Web Platform [ACM]
> - What is WebRTC and what is it good for?

Two important aspects of WebRTC:

- It is free
- It changes the definition of VoIP developers

Together, they lower the barrier of entry for communication services to a level that enables use cases that were always required but never fulfilled by previous technologies.

A standard or a Project

The term WebRTC is a bit misleading, as it refers to two separate things at the same time:

1. WebRTC the specification
2. WebRTC the open-source project

Both notions for WebRTC are used interchangeably most of the time and it is important you know the distinction between the two.

1. WebRTC the Standard Specification

WebRTC is a standard specification maintained and handled by the W3C and the IETF.

The specification deals with:

- What goes “on the wire” – what is sent through the network, so that two WebRTC-capable applications can interact with each other
- What are the APIs that utilize the functionality itself – the defined JavaScript API that is located “on top” of WebRTC

The standardization of what is known as WebRTC 1.0 (the first version of WebRTC) completed only in 2021, about 10 years after this journey began. That said, most vendors ignored that inconvenience and adopted WebRTC in production well before the completion of the standard specification. There are some gaps between the specification and what browser vendors are currently implementing. These gaps are shrinking over time as browsers get updated. Interestingly, at the same time, these gaps are growing as work on new capabilities of the WebRTC specification and implementation are introduced.

> ☞ *Further Reading:*
>
> - WebRTC: The end of an era (and the dawn of a new one)
> - RTC@Scale 2024 – an event summary

2. WebRTC the Open-Source Project

WebRTC is also an open-source project. When Google first announced WebRTC, it was just an open-source project located on its own website. Google made it clear their intention was to standardize WebRTC and embed it into browsers. To that end, they open sourced the code under a permissive open-source license. This piece of code is known as libWebRTC.

As an open-source project, WebRTC is quite a powerful piece of code that anyone can adopt and use in any way he sees fit – with or without relation to the fact that there is a WebRTC specification.

While there were and still are other attempts at offering a complete client-side WebRTC open-source stack, Google's open-source project remains the dominant alternative.

☞ *Further Reading:*

- How to pick the right WebRTC mobile SDK build for your application
- Can a native media engine beat WebRTC's performance?
- Top WebRTC open source media servers on github for 2024

Free

WebRTC is free in every possible aspect.

1. It is packaged as an open-source project and licensed under the permissive BSD license. Past implementations of similar technologies were either provided under a proprietary license that had to be purchased or under more restrictive open-source licenses which weren't palatable for a lot of vendors.
2. It supports and promotes free codecs which don't require any royalty payments for patent licenses. Codecs usually require multiple payments to use, starting from a license fee of a vendor and royalties based on quantities sold. This poses a financial barrier to adoption as well as a hassle when each quarter licensing fees need to be recalculated and paid for.

☞ *Further Reading:*

- WebRTC's Job to be done [UC Strategies]
- WebRTC is FREE. But Developers Aren't
- The future video codec: H.265 versus VP9
- AV1 Specification Released: Can we kiss goodbye to HEVC and royalty bearing video codecs?

This allows developers to repurpose WebRTC and its components in every possible means:

- Porting it to operating systems and environments where similar media processing capabilities are necessary
- Using it in large scale deployments where the number of devices is high and the payment from each customer is low or non-existent (most Over-The-Top solutions today)
- Adopting smaller modules out of WebRTC and using them alone. This is true especially to WebRTC's echo canceller and codec implementations

One interesting thing here is that in recent years we have seen a growing effort of independent developers or small teams to create their own WebRTC implementation. Such implementations are:

1. Often written in languages other than C++ (which is the language of Google's libWebRTC implementation)
2. Open source

Such implementations are partial in their extent and coverage of the WebRTC implementation compared to libWebRTC, but some of them are showing large adoption already. The most prominent of them is Pion, a Go implementation of WebRTC.

One thing to remember though, while WebRTC is free, the actual creation of solutions with it, along with maintaining and operating them, isn't free at all. There are many commercial products, services and offerings on the market for WebRTC. These are meant to reduce risk, effort, development cost and time to market.

WebRTC's Job to be Done

If we look at WebRTC from Clayton Christensen's "Job to be Done" theory from the book "The Innovator's Solution", then what we are aiming for is to think less about market segments and more about the jobs customers want to do.

In the case of WebRTC, the "customers" can be viewed as developers and services who need real-time communication capabilities. For them, the job to be done can be split into two aspects:

1. Reducing their costs – their barrier of entry
2. Reducing the friction for end users using their communication capabilities

Barriers of Entry for New Vendors

As an open-source project with a permissible license, WebRTC offers a huge head-start for developers. It brings them immediate value of around 50,000-200,000 USD – the cost of a commercial media engine.

Coupled with availability through browsers on multiple operating systems and across mobile devices[1], this brings the development and testing costs down to a point where it makes sense to start using it.

This reduction in the barrier of entry brings with it 4 distinct changes in the market:

1. More vendors and developers are looking at using real time communications
2. Use cases that had no reason economically to exist before, now make business sense with an ROI behind them
3. New business models are being experimented with
4. Solutions that were hard or impossible to implement are now created, especially inside web browsers

[1] While the WebRTC open-source project may not offer a nicely packaged mobile SDK, the code itself is actively being used by vendors using it inside both Android and iOS applications.

Reducing End User Friction

As a browser capability, WebRTC enables the web to be much more interactive and capable. Digital storefronts and support services can now include the ability to communicate and "dial" right from within the website. This reduces the friction with the end users and increases their chance of using the service more.

There are many ways in which vendors are exploiting this capability:

- Adding and acting from context coupled with the request to communicate right from the website
- Enabling users to stay in the website without going for their phone to dial for service
- Adding capabilities to social interactions from within the same service, not "losing" the users to other services such as Skype
- Empowering contact center agents to work from homes and café shops without the need for any additional installations or setup
- Connecting users of an application directly with each other with meaningful voice and video interactions as part of the service itself
- Enable ad hoc recording of a person for an online interviewing process, creation of video walls, etc.
- Introducing standardized means of connecting clients to existing services and social networks

This reduction in friction has the potential of increasing the value of the service employing WebRTC.

From a Service to a Feature

This reduction in the barrier of entry for vendors along with the reduction in friction for end users brings with it one more aspect:

WebRTC transforms communications from a service into a feature

Up until now, enterprises and businesses have had to separately think about the services and the communication channels they offer to customers, forcing them to look at communication as a service of its own.

With the introduction of WebRTC these changes and now we can view communications as just another feature in another, larger service. This enables us to stitch communications and embed them as part of the services we create and deliver.

This difference and change cannot be understated.

The Innovator's Dilemma and WebRTC

WebRTC fits into the classic Innovator's Dilemma. Wikipedia sums the Innovator's Dilemma rather nicely:

> First published in 1997, Christensen's book suggests that successful companies can put too much emphasis on customers' current needs and fail to adopt new technology or business models that will meet customers' unstated or future needs; he argues that such companies will eventually fall behind.

Displayed as a graph, the innovator's dilemma is represented as the market's demand versus the progress of a sustaining technology as well as a disruptive technology.

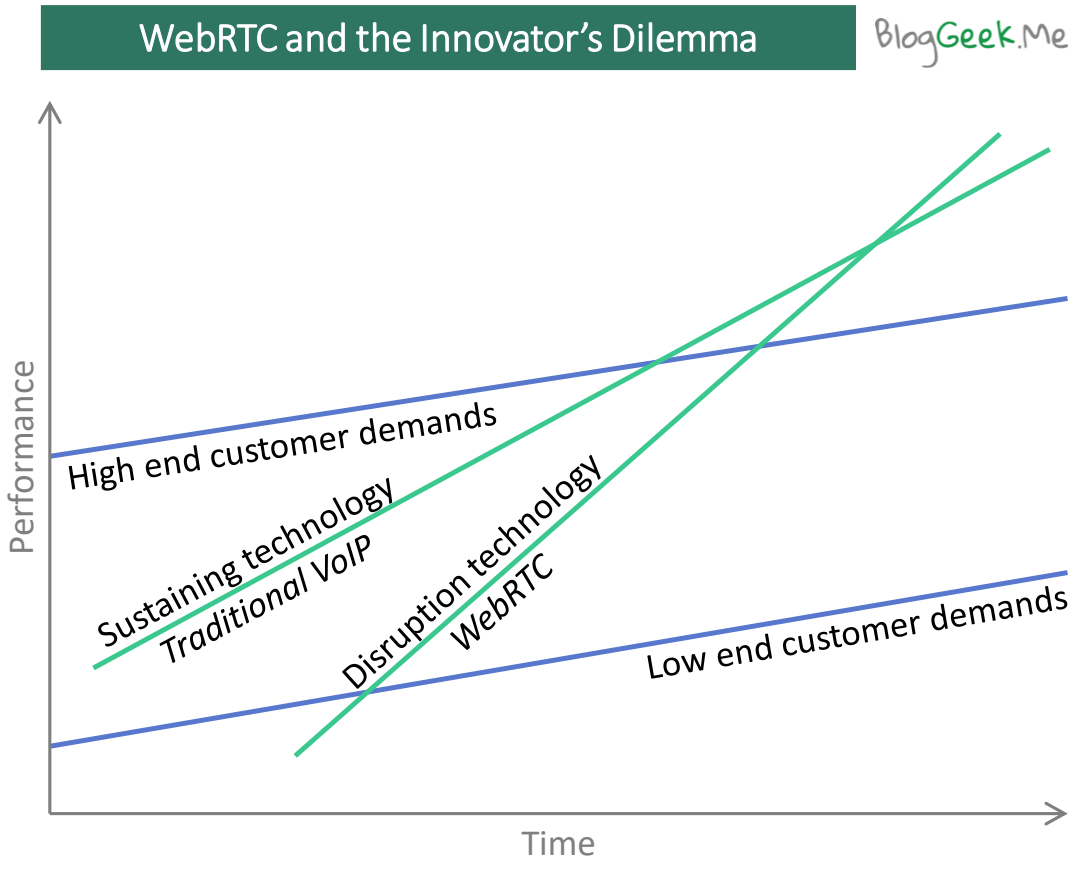

Figure 2: The innovator's dilemma and WebRTC

In our case, WebRTC is the disruptive technology.

WebRTC is like current day VoIP, but at the same time very different:

- It targets a different type and set of developers
- It reduces the barrier of entry for developers and reduces friction for end users
- It relies on a different set of technologies in the backend
- It exists in all modern web browsers “out of the box”

Incumbents faced with WebRTC won’t see the big difference between it and their current technology. This has the danger of making them complacent.

☞ *Further Reading:*

- A WebRTC JTBD (Job to be Done) Weekend
- WebRTC JTBD: Reducing Barriers of Entry
- WebRTC JTBD: Reducing End Customer Friction
- Who’s a WebRTC Vendor? The BlogGeek.me WebRTC Litmus Test

The Evolution of WebRTC

WebRTC throughout the eras

Another way to look at WebRTC is through its evolution.

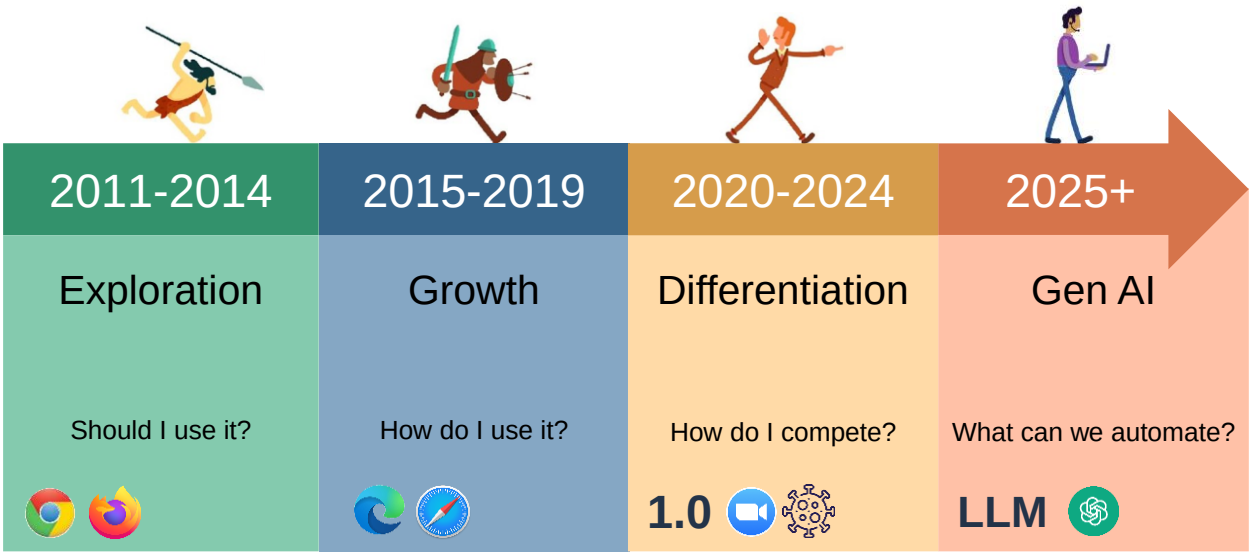

Figure 3: The evolution of WebRTC

The figure above illustrates the different periods since the announcement of WebRTC.

EXPLORATION: In its first phase of exploration, only Chrome and Firefox supported WebRTC. The main question we were asking was should we use it? Is this a real thing? Is this the technology we should rely on moving forward?

GROWTH: During the second phase of growth, that has changed. With the introduction of WebRTC in Microsoft Edge and Apple Safari, along with a better understanding of the WebRTC technology, the question has shifted. We were now more interested in how to use it – which use cases is WebRTC good for? What is the best way to employ this technology?

DIFFERENTIATION: During the third phase of differentiation, which is coming to an end, we focused on how to break through the noise. WebRTC 1.0 got officially released. Zoom showed tremendous growth (without using WebRTC), and we had the pandemic causing an increase of remote communications use, indoctrinating billions of people on

how to make video calls. Here the focus was on how we compete with others in the market? What can we do with WebRTC that will be hard for them to copy?

GEN AI: The generative AI craze that is washing the technology industry found its place in WebRTC and communications as well. OpenAI, with its LLM solutions, is the current forerunner here. Businesses are trying to figure out how to fit generative AI into their offerings. For us, this means connecting LLM with WebRTC and lowering the latency across the pipeline while doing so. AI and Generative AI have a lot of other places in WebRTC. In this era, we are going to see in which of these areas AI is going to shine with WebRTC and where it won't make any real difference.

WebRTC requires an ongoing investment in real time communication technologies for those wishing to lead their markets and differentiate from the competition.

☞ *Further Reading:*

- Generative AI and WebRTC: The fourth era in the evolution of WebRTC
- WebRTC unbundling: the beginning of the end for WebRTC?
- Why CPaaS is losing the innovation lead to UCaaS

Let's review a few more aspects of the evolution that are relevant today. These are:

- The shift from a pre-pandemic reality to a post-pandemic one
- How generative AI is shifting the focus in WebRTC's investments
- Availability of developers skilled in WebRTC

Pre-pandemic to post-pandemic

The differentiation era in the period of 2020-2024 was full of turmoil and change. It started with the COVID pandemic that brought with it worldwide quarantine on populations, forcing on everyone remote communication. These quarantines indoctrinated billions of people around the globe on the use of video meetings, getting everyone more comfortable in using such technologies.

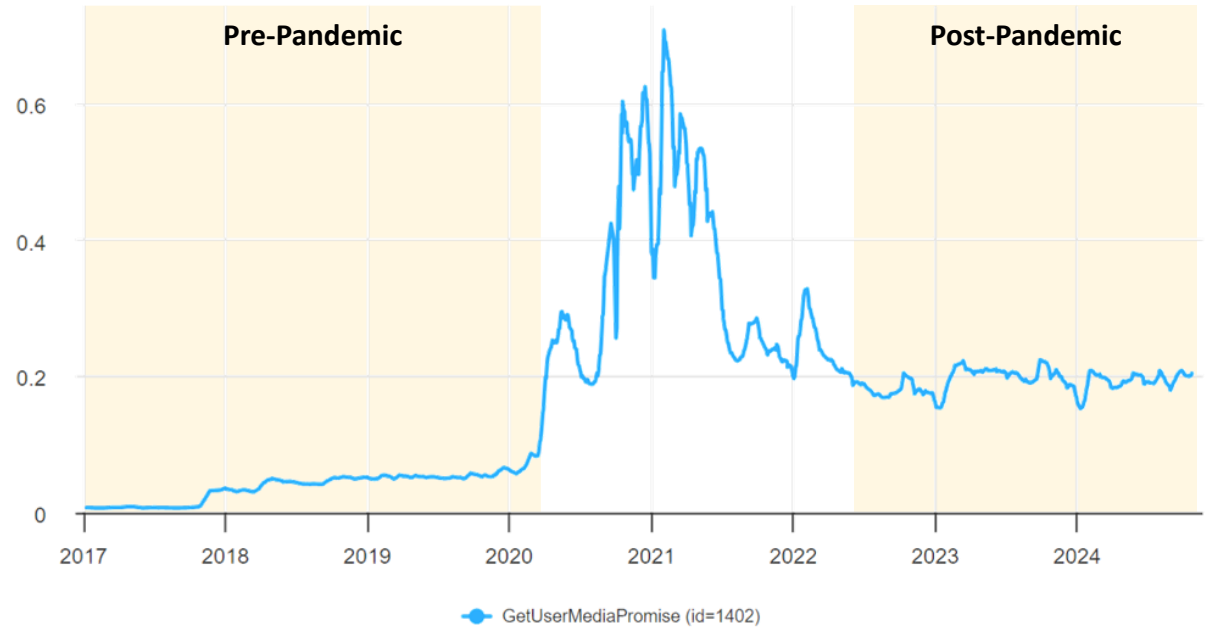

Figure 4: WebRTC use in Chrome page loads in the past 8 years

What the graph above illustrates is a huge increase in use during the pandemic, but the more interesting part is how the stable use of WebRTC has grown x4 from before the pandemic to after it. We are now starting our third year of post-pandemic, and this growth doesn't seem to be abating. On the other hand, we don't see it growing further either.

A shift in the market was imminent at this point, which is why when generative AI started catching attention, it also found its home with companies focused on WebRTC solutions.

Generative AI and WebRTC

OpenAI introduced ChatGPT in November 2022, making LLMs (Large Language Models) popular. ChatGPT enabled users to write text prompts and have "the machine" reply back with answers that were human in nature. The initial adoption of ChatCPT was… instant - faster than anything we've ever seen before.

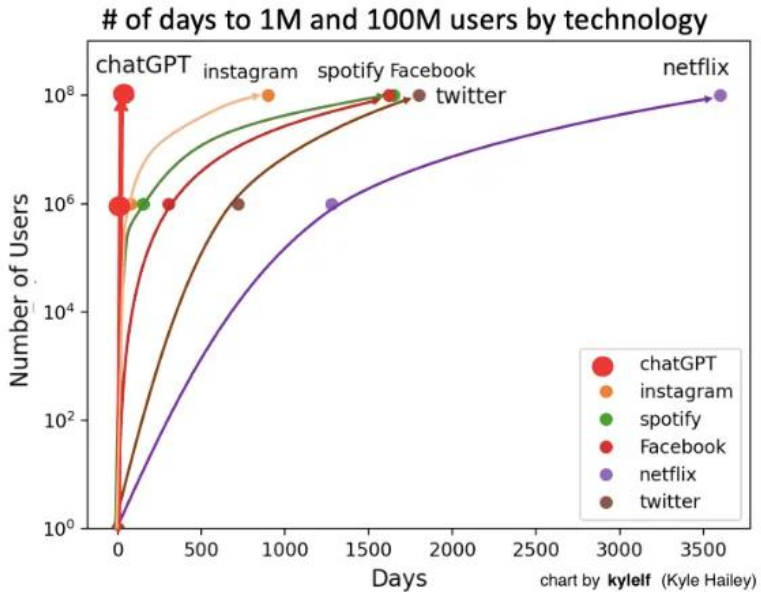

Figure 5: # of days to 1M and 100M users by technology (Source: Kyle Hailey)

This validated the use of AI and Generative AI in a back and forth "prompted" conversation between a human and a machine. From there, the market exploded.

If you look at the WebRTC domain these days, it is in a maintenance mode state. We've had the adrenaline rush of the pandemic, with everyone working on scaling, optimization and getting to a 49-view magic squares layout. The use of WebRTC has gone drastically down after the pandemic. Still a lot higher than pre-pandemic time, but lower than what we had during the quarantine years. This had companies' use going back down and their investments shrinking down with it. The world's turmoils and instabilities aren't helping either, and the inflation is such that stifles investments as well.

So, a new story was needed. One that would attract investment. LLM and Generative AI were then powered by the popularity of OpenAI's ChatGPT.

This is such a strong pull that I believe it is going to last for quite a few years, earning it an era of its own in my evolution of WebRTC view.

WebRTC's "maintenance mode" isn't going to change much. We will see progress in areas such as media processing with the introduction of the AV1 video codec. Much of

the rest will revolve around developers figuring out how to best fit Generative AI with WebRTC.

Some of the time this will be about the best integration points and APIs to use. In other times, it is going to be about minor tweaks to WebRTC itself and maybe even introducing a new API or two to make it easier for WebRTC to work with Generative AI.

From there, the focus and attention will shift towards the use cases themselves – once we figure out the technology side of things, there will be a need to find out what are the winning use cases for Generative AI with WebRTC.

Welcome to the new era of WebRTC (and Generative AI)!

☞ *Further Reading:*

- OpenAI, LLMs, WebRTC, voice bots and Programmable Video

Browser Support

In the distant past, WebRTC support was marred by poor browser coverage. Today, the story is different. WebRTC is available everywhere:

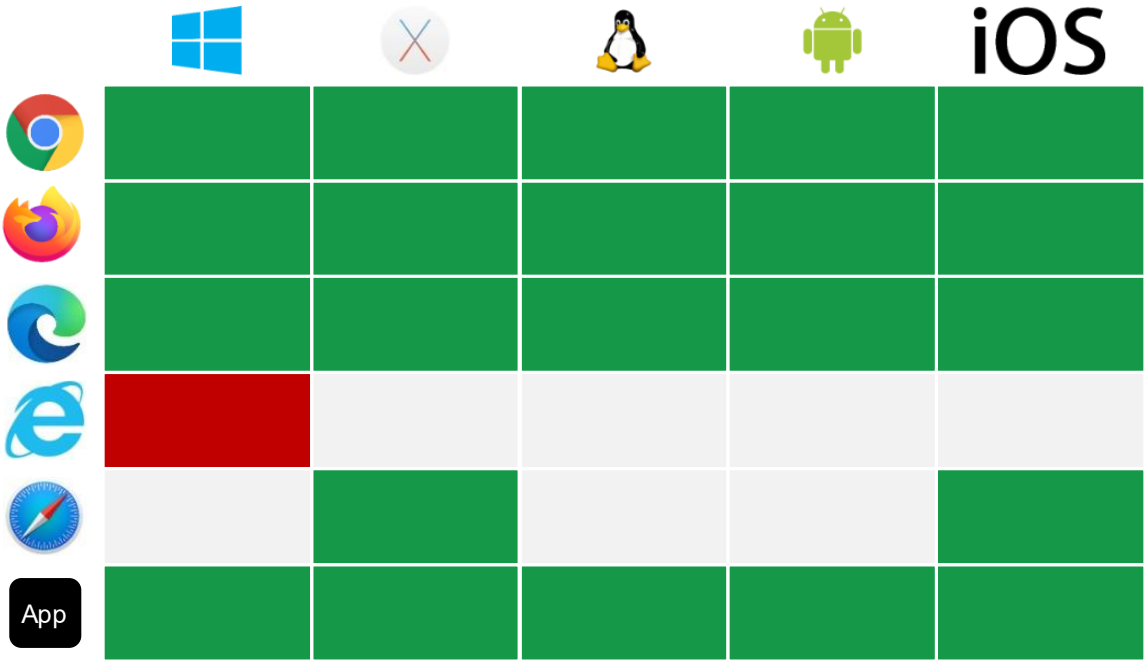

Figure 6: Browsers support for WebRTC

The table indicates a few important aspects of WebRTC adoption:

1. WebRTC is available across all modern browsers in all their relevant operating systems
2. Native applications can be developed everywhere and use WebRTC, without any real need for a web browser
3. Not apparent but important, is the fact that on iOS, all browsers are based on WebKit, sharing the exact same WebRTC implementation (enforced by Apple)

In the next subchapters, I will detail what solutions, if any, are available for developers.

☞ *Further Reading:*

- My WebRTC Device Cheat Sheet
- Is Chrome on its Way to be ONLY Browser out there? (Microsoft throwing the towel on Edge)
- WebRTC iOS 11 Support. Are We There Yet?

Handling Mobile Devices

In mobile, there are two means of consumption: apps and web.

Applications are a popular way of interaction on devices, but there are times when web and access through a browser make more sense. This includes one-off interactions, click-to-dial on websites and starting an interaction through a link in an SMS message.

While both Android and iOS support WebRTC, they come with their own challenges:

1. Device fragmentation and pre-installed mobile browsers make for variable user experience and performance issues with WebRTC on Android. A sour point is the ability (or inability) to use H.264 hardware video codec in some Android devices
2. Mobile Safari implementation of WebRTC is lacking a bit, causing many developers to refrain from supporting iOS browsers for their WebRTC applications. While this is still true, it is getting better each year

This has led to these main choices available to you:

1. **Build an application.** Deliver your service through a mobile application
2. **Support only when possible.** Enable your service through mobile browsers that support WebRTC properly
3. **Ignore mobile.** Don't invest time in mobile

To make your decision, check how your service gets consumed on mobile. If users are expected to "bump" into it when browsing the internet, through an SMS message or by specifically going to your service via an app. Going via an app means you can use WebRTC in its ported form, wrapped inside an application. Assuming you want WebRTC to be embedded in your app, there are three different options open for you:

1. Port and integrate WebRTC on your own for the mobile devices you plan to support
2. License a ported WebRTC stack from vendors offering such a service. There are a couple of WebRTC outsourcing outfits today, and some of them offer porting services and commercial SDKs that simplify this option
3. Use a 3rd party WebRTC API platform that already have their offering ported to a mobile SDK

☞ *Further Reading:*

- Where is WebRTC positioned on mobile platforms
- The challenges of porting WebRTC to mobile devices

Desktop Applications

While WebRTC is predominantly a browser technology, there is no limit to using it elsewhere. We've seen its adoption taking place on mobile devices. The same is happening, though to a lesser extent, in desktop applications.

Getting WebRTC to work in desktop applications can occur in a few ways:

1. **Using libWebRTC**. The official Google implementation of WebRTC, written in C++ can be ported and integrated into desktop applications
2. **Using other WebRTC libraries**. There are a few alternatives written in Go, Elixir, Rust, Python, etc. Some prefer using these instead of the Google one
3. **Using cross-platform framework based on web technologies**. Here you will find Electron, CEF, and WebView2. All these frameworks enable web development (including WebRTC with its JavaScript API), wrapping and packaging it into desktop applications

Each of these approaches comes with its own advantages and challenges. You will need to figure out what works best for your application and users.

> ☞ *Further Reading:*
>
> - WebRTC Electron Implementations are on 🔥

WebRTC on Devices

An additional opportunity of utilizing WebRTC is in other devices – set-top boxes, video room systems, IP cameras, and Chromecast-like dongles are some examples.

In this case, a device or component that needs to be able to process and send/receive media can achieve that by porting WebRTC and embedding it within the device.

There are three main reasons to select WebRTC for these devices:

1. **Google's libWebRTC implementation** has a very permissive open-source license, making it easy to use as the basis of a generic media processing component. It has also been proven to be portable numerous times already.
2. There are a few **open source and commercial alternatives** to Google's libWebRTC implementation if that's not to your liking for some reason.
3. By using **WebRTC embedded in the device,** making it easy to interact with these devices from other mobile devices, laptops and browsers.

☞ *Further Reading:*

- Chromecast as a predicate of M2M use of WebRTC
- Motion detecting baby monitor with WebRTC [webrtcH4cKS]
- Comcast Getting it Right with WebRTC: Opts for Video Streaming over Video Chatting

WebRTC Hype

There is no hype when it comes to WebRTC.

Pundits have been debating this for quite some time now, with varying opinions. I believe there isn't and there wasn't any hype around WebRTC whatsoever.

When I started blogging about WebRTC, I was alone in the field. Very little information and thought leadership could be found about WebRTC on the web. Since then, that has changed with individuals, vendors and media outlets now covering this space.

Any apparent hype being felt can be attributed to many reasons:

1. **WebRTC in a new technology**. As such, the process of understanding it that takes place these days in public on blogs looks like hype for some: you will find few observations these days about where SIP belongs within the domain of VoIP, but a lot of writing about WebRTC's place in VoIP
2. **WebRTC deals with VoIP and web**. While VoIP belongs to a small set of developers within the communication market, WebRTC opens that market to a distinctly larger set of developers. This increase in availability of communication technology seems to some like hype
3. **WebRTC reduces the barrier of entry**. Use cases that had no ROI up until WebRTC are now being introduced, enhancing the ways in which VoIP is used

Here are a few data points about WebRTC and how they have fared in the past couple of years.

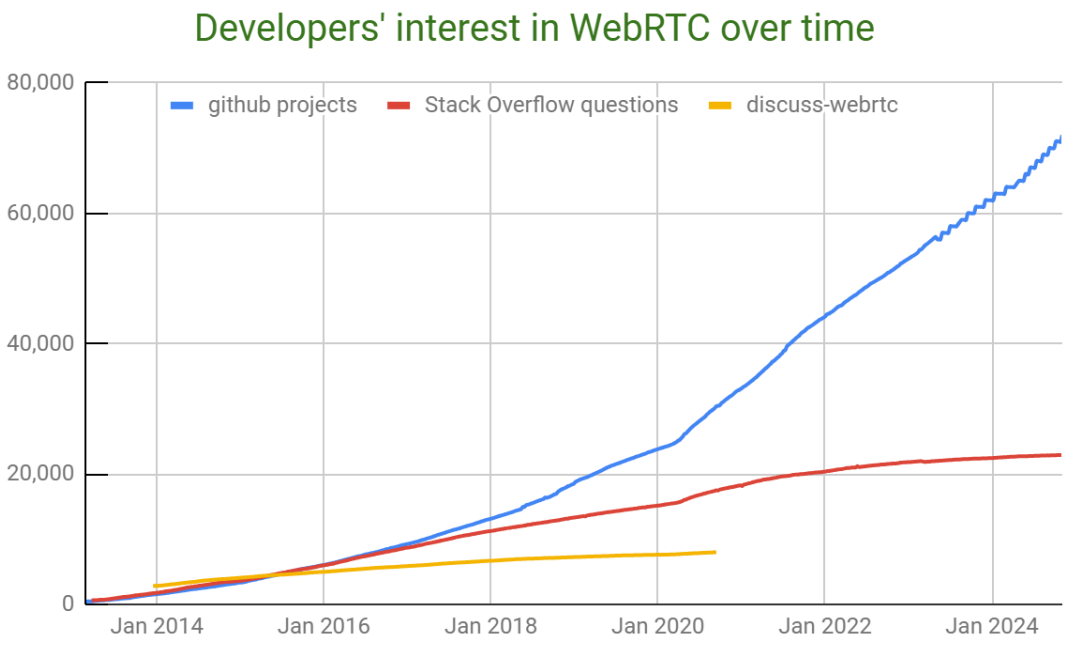

Figure 7: Developers' interest in WebRTC (2013-2024)

The graph above illustrates three domains that are prime destinations for developers interested in WebRTC:

1. The Google group called discuss-webrtc
2. The number of github projects related to WebRTC
3. The number of questions around WebRTC on StackOverflow

Over the last 10 years, we have seen a steady increase in interest in WebRTC. The increase is linear in nature so far.

A few thoughts here:

- The github projects line switched its growth angle at the beginning of 2020. This is mainly due to the quarantines and focus on remote communications. The change in the rate of adoption is still linear but the increased angle of adoption we've seen since 2020 is consistent to this day
- There is no growth in discuss-webrtc members. discuss-webrtc indicates developers who are actively working with WebRTC and are in the thick of the technical conversations conducted around it. Google has modified permissions to this group since 2020 so we can no longer track the number of members. It is highly probably that the number has stayed flatlined
- StackOverflow questions aren't growing quickly. This also indicates the number of questions revolving around WebRTC technology. It would have been expected to see more of these since the pandemic, and while there is a slight bump during 2020, it has since flatlined quite a bit

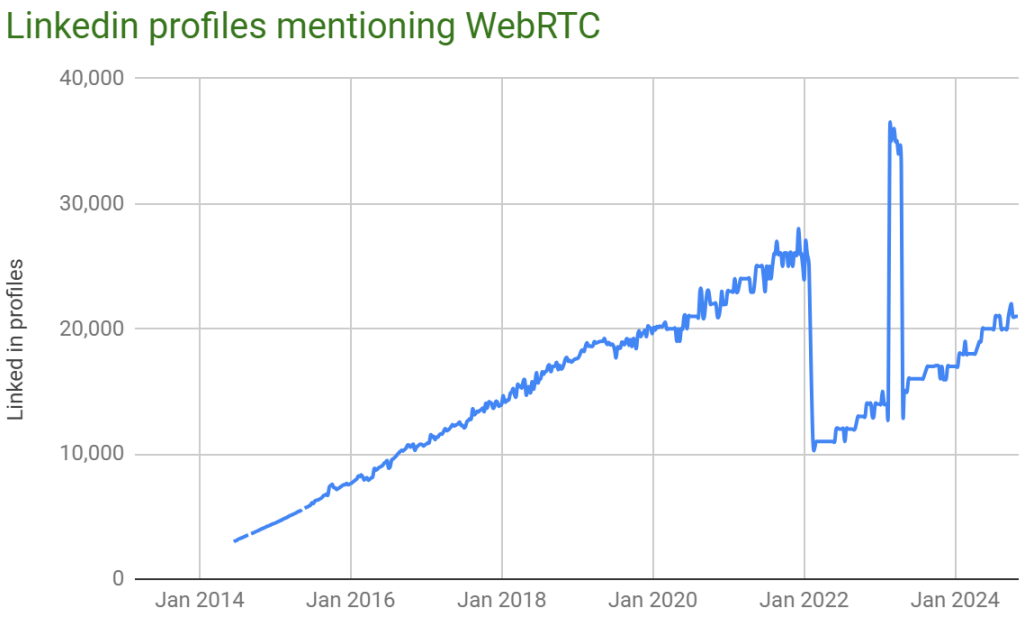

Figure 8: LinkedIn profiles related to WebRTC

The graph above details the number of LinkedIn profiles that have WebRTC somewhere in their description. While this number is rather low, it is easy to see an upwards trend. The weird spikes in 2022 and then 2023 indicate a change in the way the search query returns the value. Ignoring it, the results over the years are quite consistent: the graph shows a linear increase in interest – this is an indication that we are either before the real hype of WebRTC – or that no hype will happen with WebRTC at all. Our bet? There is no hype and there will never be one with WebRTC.

☞ *Further Reading:*

- There is no hype around WebRTC [NoJitter]
- WebRTC hype check [NoJitter]
- WebRTC hype: recheck [NoJitter]
- Gartner Hype Cycle, WebRTC Not Included [The New Dial Tone]
- WebRTC on the Rise? Geek Questions Hold the Answer [NoJitter]
- Data Nerding with WebRTC GitHub Data [webrtcHacks]

The WebRTC Ecosystem

WebRTC is only a technology. It was designed to create a developer ecosystem around it by defining the APIs and making them widely available via Java Script inside web browsers.

This created a vibrant ecosystem of vendors who use WebRTC in different ways. This ecosystem can be categorized into 5 broad categories, outlined below.

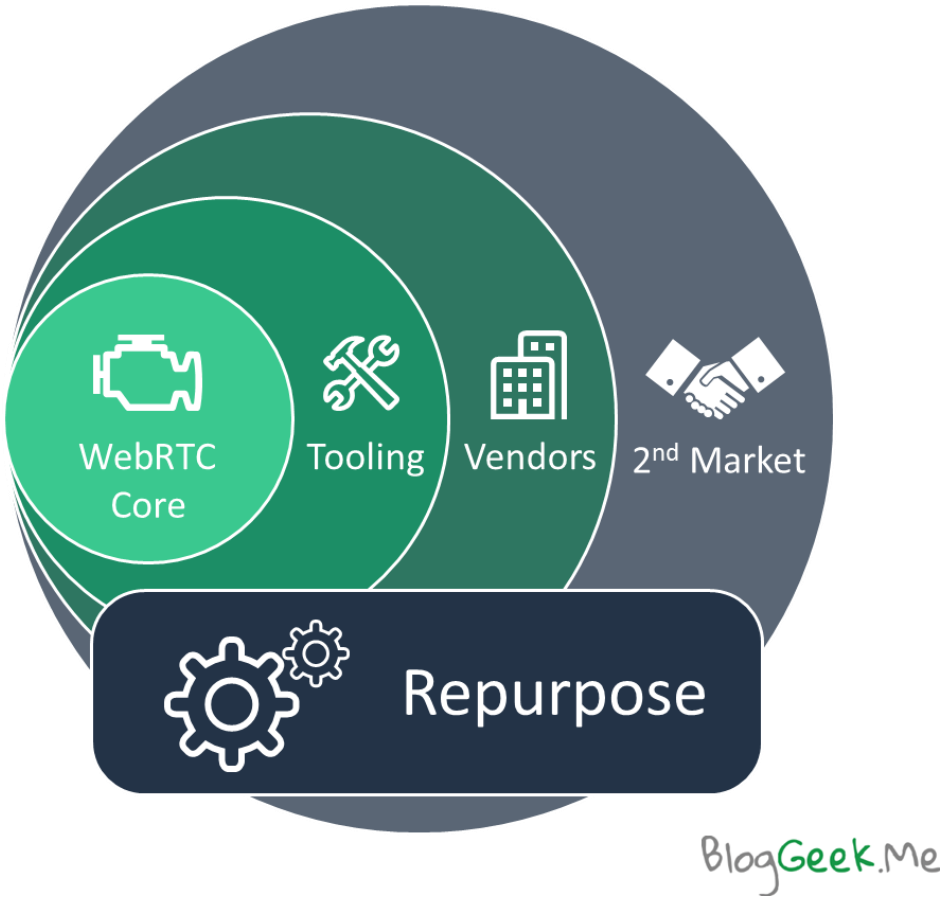

Figure 9: Types of vendors in the WebRTC ecosystem

Before I go into detail about the categories there are a few things to keep in mind:

- A vendor can be in multiple categories at the same time
- Similar vendors chose a different category to belong to. The choice changes their core competencies and basis of competition to some extent
- The existence of such an ecosystem is what makes the decision of developers in an approach a difficult one, while at the same time opens interesting opportunities in the make-up of your architecture and employee mix

WebRTC Core

The supporting role at the heart of WebRTC belongs to vendors who contribute to it directly. Here you will find Google, Mozilla, Apple and Microsoft who are embedding WebRTC into their browsers. Most just take Google's libWebRTC almost "as is" and integrate it into the browser. Sometimes, other vendors would become important contributors to the WebRTC core. It happened a bit more in the past, but not much in recent years. A past example includes Cisco who contributed the openH264 codec and paid royalties for it.

The number of vendors in this area is small, as they require extensive knowledge into the inner workings of WebRTC as well as good relations with Google. This role comes with little upside in the way of direct revenue but gives a lot of power in ensuring technical superiority versus other vendors:

- Browser vendors may use their support of WebRTC to further their own applications (Google Meet; Microsoft Teams; Apple FaceTime) or to try and gain market share from other browser vendors due to superior user experience in their WebRTC implementation
- Cisco made its contribution to advance their own technologies and solutions in browser environments

These vendors usually do not intend on banking from WebRTC directly and have indirect business models for WebRTC; at times, the business model isn't easily apparent.

Tooling

Vendors in this position offer tools that make the development of WebRTC-based services much easier. These vendors have varying business models focused around B2B, where the target market is other vendors who wish to deploy communication services.

There are many tooling vendors. Some of them are not even vendors, but rather open-source projects that are hosted on github and are quite popular.

To put some order into the Tooling vendors, I have created the WebRTC Developer Tools Landscape Infographic:


BlogGeek.Me
WEBRTC DEVELOPER TOOLS LANDSCAPE
Browsers
Chrome
Edge
Firefox
Safari
Standalone Plugins & SDKs
aiortc
gstreamer
pipe
PION
Signaling
OPEN SOURCE
matrix
PEER JS
simple-peer
HOSTED
ably
Firebase
PubNub
PUSHER
scaledrone
stream
VoIP
CLIENTS
JsSIP
SIP.js
SERVERS
Asterisk
FreeSWITCH
NAT Traversal
ON PREMISE
coturn
PION
HOSTED
CLOUDFLARE
ExpressTURN
Metered
twilio
XIRSYS
Media Servers
OPEN SOURCE
OPEN WEBRTC TOOLKIT
JANUS
jitsi.org
KURENTO
Licode
LiveKit
mediasoup
Medooze
openVidu
PION
COMMERCIAL
Ant Media
ceeblue
compunetix
Dolby.io
Flashphoner
phenix
REDS PRO
WOWZA
Testing & Monitoring
LOADERO
testRTC
CPaaS
100ms
Jitsi as a Service
agora
aws
apiRTC
Azure
bandwidth
CLOUDFLARE
/daily
digitalsamba
dyte
enablex
Enghouse Video
eyeson
infobip
kaleyra
LiveKit
liveswitch
Metered
]pexip[
plivo
SignalWire
sinch
SKYWAY
stream
telnyx
twilio
videosdk
VONAGE
voximplant
Whereby
zoom
www.bloggeek.me

These vendors can be split into several domains:

- Browsers
- Standalone Plugins and SDKs
- Signaling
- VoIP
- NAT Traversal
- Media Servers
- Testing and Monitoring
- CPaaS

> ☞ *Further Reading:*
>
> - WebRTC Developer Tools Landscape

Browsers

Browser vendors sit at the middle of the WebRTC ecosystem. They enable WebRTC to work at all in the browser. Google also offers the WebRTC implementation that gets integrated with the Chrome browser as an open-source project. This opens a lot of interesting opportunities and use cases for other vendors.

As browsers offer slightly different approaches to the way they implement WebRTC, knowing which ones are more popular with your target audience is important.

Standalone Plugins and SDKs

Besides browsers, developers often want to be able to wrap WebRTC inside native mobile applications, PC applications, legacy browsers and embedded devices.

While developers can port the open-source code of Google's WebRTC project on their own, often they would resort to using a third-party supplier that adds support for these devices through a commercial SDK or plugin.

> ☞ *Further Reading:*
>
> - Temasys' WebRTC Plugin: One More Thing & Where is this Headed?
> - WebRTC Plugin? An Electron WebRTC app is the only viable fallback
> - WebRTC Electron Implementations are on 🔥

Signaling

WebRTC does not implement signaling, so developers either implement this on their own or use third parties. There are various implementations on GitHub offering such support and a few more popular than others.

The other alternative is to opt for a managed service that can offer signaling capabilities. There are a few of these listed in the landscape infographic.

VoIP

WebRTC is VoIP, and some VoIP solutions are WebRTC. That said, if you need to connect to SIP or PSTN, you will usually end up using a VoIP backend infrastructure. This group of vendors lists the popular client and server frameworks that are used with WebRTC for that purpose.

NAT Traversal

With WebRTC, developers need to deploy NAT Traversal servers. These can either be open source or commercial; and today they can also be managed services, where you direct your WebRTC clients to STUN and TURN servers of a third party and pay for the traffic they handle.

> ☞ *Further Reading:*
>
> - WebRTC TURN: Why you NEED it and when you DON'T need it
> - Why Doesn't Google Provide a Free TURN Server?

Media Servers

There are many use cases where a media server is needed. This can be to support group chat scenarios, recording or connectivity to external systems.

Media servers come in different shapes and sizes, some open source and some commercial. There is no single product here that fits all use cases.

Testing and Monitoring

There are a few vendors who are focused on offering testing and monitoring services and capabilities that are targeting WebRTC specific products. Their purpose is assisting vendors throughout the lifecycle of developing and releasing WebRTC products to the market.

CPaaS

Communication Platform as a Service encompasses all the needs a developer has with regards to communications within his application. This report is focused on this domain and covers these vendors in detail.

> ☞ *Further Reading:*
>
> - Why CPaaS is losing the innovation lead to UCaaS
> - Future of CPaaS; a look ahead
> - What should CPaaS providers do today to prepare for the "post pandemic"?
> - Cloud giants joining the WebRTC API game. How is that changing the CPaaS landscape?
> - Twilio Signal 2020. I expected more from the leading CPaaS vendor

Projects Vendors

While not listed in the WebRTC Developer Tools Landscape, it is worth mentioning the projects vendors. These are companies who offer outsourcing and consulting services to the WebRTC market.

As with any vibrant community, there are those who offer their services to assist companies in building whatever it is they require. WebRTC include large and small software houses of this type – ones that take on development tasks related to WebRTC.

Some of these vendors offer a breadth of capabilities coming either from a broad outsourcing and system integration heritage, others are more focused into the VoIP and video processing space and the rest are starting off with WebRTC or can be considered as "heads for hire", usually focusing on projects to local vendors.

WebRTC being a new domain, there is a real need for such vendors as well as a rise in the number of vendors who are publishing their expertise in WebRTC and even investing in showcasing it in.

When considering hiring project vendors, freelancers and outsourcing companies, be sure to check their existing experience with WebRTC. Many of the vendors have little or no experience with this technology and will be happy to be selected by you to develop your project. In most of these cases, the results are not satisfactory to say the least. Having experience with WebRTC development before going into a project reduces failure risks drastically.

Open-Source Projects

WebRTC is building around it a large and healthy set of open-source projects. At the time of writing these lines, there are over 70,000 github projects related to WebRTC. These numbers are growing consistently, and they show a vibrant community.

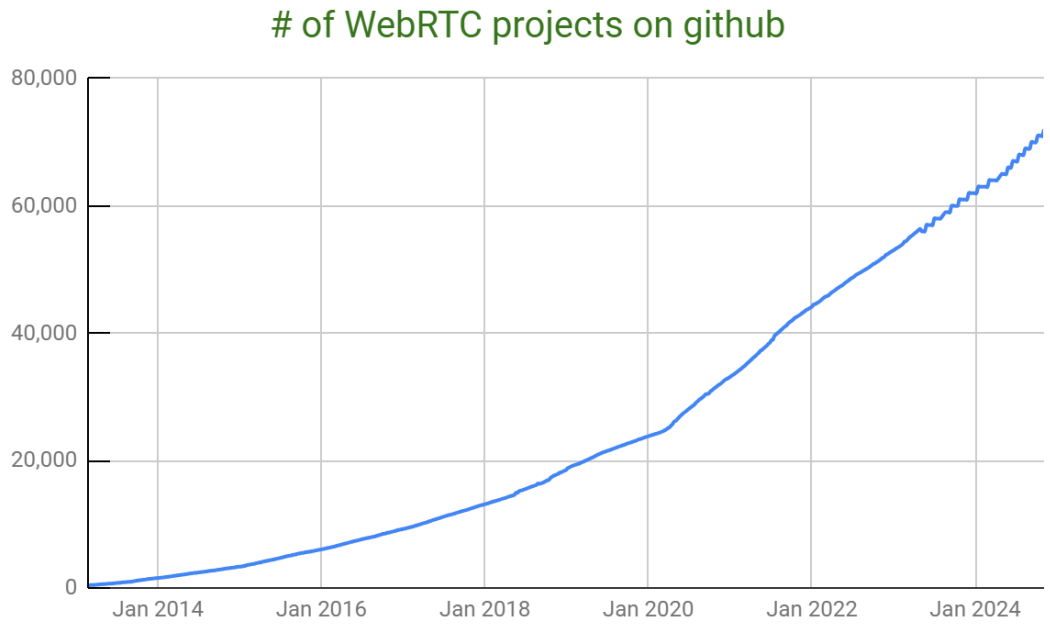

Figure 10: WebRTC related projects in github

These projects usually sprout for the following reasons:

1. A single person, student or other, researching technology and just introducing the results online for everyone to use and share. This is usually done to attract a potential employer
2. Someone with a pain point that can be solved with WebRTC. Twelephone was such an example of a service built around social communications
3. A vendor who believes in the open-source movement and contributes components back to the community while building his own services and offerings

How much these projects will be used as the baseline of future projects and products outside of the scope of their original developer is an open question. The rate of introduction of new projects has grown since the pandemic came to our lives and the quarantines began at the beginning of 2020. Since then, the rate of growth has stayed stable.

☞ *Further Reading:*

- and the WebRTC Open Source Popularity Contest Winner is… [webrtcHacks]
- Examples of WebRTC open source modules:
    a. rtcstats: a module to collect peer connection statistics
    b. Janus: open source gateway for WebRTC
    c. Pion WebRTC: a Go implementation of WebRTC

Vendors

Vendors are those who end up offering WebRTC-based communication services to the end users. Their market may be B2B or B2C, depending on whom they serve.

It is hard to cluster these vendors together, as they cater to different verticals and use cases. It is also where business models diverge greatly.

Vendors in this category may use a Tooling vendor to rely on or just build everything from scratch on their own. Reasons for selecting different routes here depend on a large variety of parameters, unique to each case.

The use cases chapter will try to take a closer look as some of the markets to which these vendors belong to.

Repurpose

Repurposing WebRTC can be viewed as the gray market of WebRTC. There are active vendors in this space, where most of them are not widely known – their use of WebRTC is such that is left behind curtains.

WebRTC, the open-source project, as opposed to the specification, is a state-of-the-art media engine. It is the result of GIPS' commercial offering and On2's proprietary video codec, coupled with Google's ongoing investment in its quality.

To put things in perspective, when GIPS got acquired by Google, many vendors who were GIPS' customers immediately started searching for alternatives, which were hard to find at that level of quality and capability. Among GIPS customers you could have found IBM, Google, Yahoo, WebEx, Nortel, AOL, Citrix, Avaya, Samsung and QQ.

When Google open sources WebRTC under a very permissive BSD license, they in effect provided anyone who needed a media engine free access to a high-quality solution – only thing missing was an SLA.

Vendors have taken note, and some have pried bits and pieces of WebRTC that fits their needs and embedded them into their own products and services. This requires little publication of the fact, so most of these activities are left unknown in the market.

☞ *Further Reading:*

- Changes in the media engine market landscape
- The WhatsApp RTCP exploit – what might have happened? [webrtcHacks]
- An interview with Jan Linden of GIPS prior to its acquisition [Wirevolution]

2nd Market

The 2nd Market of WebRTC is something quite rare, but its existence is important to note.

WebRTC fuses VoIP and the web. This brings two important traits together:

1. VoIP gains an open and modern delivery system – the web browser
2. The web gains P2P capabilities and real-time communication

The web is based today on loosely coupled web sites and technologies that are meshed up together to build new solutions and services. Today, simple websites end up connecting to multiple online APIs of various suppliers.

This trend of mashup and open API publication is also something you see with WebRTC, and not only with the API vendors: many of the WebRTC services players are starting to offer APIs of their own for various purposes:

1. Integration into enterprise directory services
2. Integration with CRM/ERP systems of enterprises
3. Widgets and call buttons to place on websites
4. White labeling and branding via APIs

To some extent, this creates a 2nd market of companies who integrate with these services vendors and still see themselves as part of the WebRTC ecosystem.

☞ *Further Reading:*

- TNW Academy announcement [TheNextWeb]
- WebRTC 2nd Market: The Story of Intuitive Solutions

WebRTC Use Cases by Verticals

This chapter holds the main body of this research paper. It consists of various verticals and use cases of real vendors within those verticals who are using WebRTC in one way or another.

The purpose of it is for entrepreneurs and product managers to get acquainted with the market dynamics of WebRTC and to see how different vendors are tackling the technical and business issues associated with the introduction of WebRTC.

Things to Remember

- The vendors listed in this chapter were selected not due to their technical merit or the type of support they provide – these were not part of the analysis done. The vendors listed here were selected as examples of the different services and business models that vendors are offering in the WebRTC ecosystem.
- Some of the vendors in this section have been interviewed for BlogGeek.me in the past. You can find more about them here: https://bloggeek.me/webrtc-interviews/

The Vendors

Projects Delivery

NIMBLE APE

webRTC.ventures

Tooling

kaleyra

PION

testRTC powered by Spearline

Customer Service & Support

AVATOUR

HELP LIGHTNING

O2

talkdesk

Enterprise Communications

webex by CISCO

dialpad

miro

slack

Virtual & Hybrid Events

Airmeet

StreamYard

vimeo

Healthcare & Education

doxy.me

Teladoc HEALTH

Topia

Vedantu YOUR PERSONAL TEACHER ONLINE

Social

FaceTime

Messenger

Streaming and Content Delivery

Ant Media

Microsoft eCDN

WebTorrent

Projects Delivery

Projects Delivery vendors are outsourcing vendors who specialize in WebRTC and real time communications or media processing applications. They offer an important service that enables businesses to introduce and add interactive capabilities to their own product offerings without the need to build a workforce that is highly skilled in WebRTC technology.

While there are many outsourcing vendors, not many of them focus and specialize on WebRTC. In this domain, we will be featuring two of our sponsors for this report:

- **Nimble Ape** – a professional services business specializing in real-time media applications and media delivery
- **WebRTC.ventures** – a custom design and development agency specializing in real-time communication applications

Nimble Ape

Solution Type	Projects Delivery	**Website**	nimblea.pe
Target audience	Enterprises, Startups	**Country**	United Kingdom
Business Model	Project based		

Nimble Ape is a professional services business operating from the UK and managed by Dan Jenkins.

The team specializes in projects that necessitate the use of real time communications and other forms of media delivery.

Nimble Ape is comfortable in delivering projects where open source components are used as well as when CPaaS and Programmable Voice and Video by third party vendors need to be utilized.

In recent years, Nimble Ape introduced a few distinct services, orthogonal to its consulting efforts. The most notable being:

1. CommCon – an event for real-time and open media folks to get together, share their knowledge and celebrate all things open source and open standards
2. Broadcast Bridge – a managed A/V studio for creators to broadcast their sessions live over the internet
3. ICEPerf – an online service that monitors and measures different managed TURN service providers

Things to Remember

- Many of the notable project delivery vendors will have their own additional services and development projects in parallel to their professional services business
- These vendors usually specialize in both open source and managed CPaaS technologies

WebRTC.ventures

Solution Type	Projects Delivery	**Website**	webrtc.ventures
Target audience	Enterprises, Startups	**Country**	United States, Panama and Colombia
Business Model	Project based, Staff Augment		

WebRTC.ventures builds custom web and mobile apps. They provide a complete UX, design, and development services. WebRTC.ventures can also provide production support and DevOps when needed, taking care of all the hosting and deployment as well.

WebRTC.ventures is a part of AgilityFeat, who builds nearshore teams of Latin American developers for US companies. The focus on WebRTC got AgilityFeat to “spin off” its WebRTC activities into WebRTC.ventures in 2015.

Although WebRTC.ventures does build applications on top of CPaaS platforms, many of the projects that WebRTC.ventures takes upon itself require the use of open source frameworks and an “on premise” approach where the end result gets installed on the customer’s own data center or his cloud IaaS account.

Things to Remember

- Companies are sometimes looking to deploy their own services and not run on top of a CPaaS vendor
- WebRTC.ventures fills that gap by offering their inhouse developing expertise in a market with high demand for skilled WebRTC experts: Developers, DevOps, UI/UX, and QA

Tooling

Tooling vendors are at the heart of the WebRTC ecosystem. They are the enablers filling the gaps that the WebRTC specification doesn't solve – this essentially mean a lot of backend capabilities.

These vendors come in different shapes and sizes. I have selected 3 such vendors:

- **Kaleyra** – a Prebuilt solution in the Programmable Video domain enabling the creation of seamless customer experiences
- **Pion** – an open source WebRTC implementation in the Go language
- **testRTC** – a testing and monitoring service for WebRTC applications

Kaleyra Video

Solution Type	CPaaS	**Website**	kaleyra.com
Target audience	Developers	**Country**	United States
Business Model	Pay as you go		

Kaleyra offered mobile communication services for large enterprises. In 2021 it acquired Bandyer to expand its services towards video communications. Bandyer were using WebRTC to build such a solution.

Post acquisition, Kaleyra repackaged its video capabilities as part of its CPaaS offering, making Kaleyra Video into a Prebuilt solution in the Programmable Video space. Kaleyra was acquired in 2024 by Tata Communications.

This shift from a communication service towards CPaaS and APIs brings with it some differentiated features and concepts into the Programmable Video domain, such as the notion of unique, identifiable users in the system.

Things to Remember

- Different CPaaS vendors offer different feature sets and different approaches to their platform
- Kaleyra took the approach of offering a ready-made Prebuilt low-code solution aiming to reduce the development and integration effort to the bare minimum
- At times, companies in this space acquire certain startups for their technology, but not necessarily for their existing business and customers

Pion

Solution Type	SDK	**Website**	pion.ly
Target audience	Developers	**Country**	United States
Business Model	Open source		

Pion is an open-source implementation of WebRTC in the Go language.

Its goal is to create a community around it and make it easier to develop and deploy WebRTC applications. It started with a client and TURN implementations but has grown since to include media servers as well.

Pion is one of the most notable examples where an open-source implementation of the WebRTC protocol stack for a client implementation has succeeded in attracting enough developers to make it interesting. It is currently growing in use and popularity for certain use cases.

Things to Remember

- Many developers of WebRTC applications rely heavily on open-source frameworks
- Picking and choosing the right open-source alternative isn't a simple task. Ease of use, feature richness, support and future stability are often the main aspects one should look at
- Pion is one of the few frameworks that has gained enough interest and is growing

testRTC

Solution Type	Testing & Monitoring	**Website**	cyara.com
Target audience	Enterprise developers	**Country**	United States
Business Model	Subscription, Pay as you go		

testRTC is a testing and monitoring cloud-based service focused on WebRTC applications. It is part of Cyara.

Due to the nature of WebRTC, traditional VoIP or web testing and monitoring tools are insufficient for WebRTC developers.

testRTC offers self-service tools to meet WebRTC application developer needs, be it in regression, performance and stress testing; or users connectivity and quality tracking and troubleshooting.

This approach enables customers to pick and choose the specific tools they need based on where they are in the product development lifecycle.

Things to Remember

- Testing and monitoring is often neglected or left for the last moment. Developers should give more thought to these requirements in their applications
- Some vendors today build their own in-house testing and monitoring for their WebRTC applications. Others prefer a ready-made solution and end up using testRTC
- The author of this report was also the CEO and Co-founder of testRTC, now Senior Director of Product Management, testRTC at Cyara. This makes him somewhat biased 😊

Customer Service & Support

One of the areas where many see the value of WebRTC is customer service and support. The main example given is usually the ability of a customer to surf a website, click a call button on the website and get connected to an agent. That is only the tip of the iceberg.

WebRTC has many different uses in the domain of customer service and support – some of which may prove to be a better fit for many enterprises.

Use cases in this section include:

- **Avatour** – 360° remote site meetings
- **Help Lightning** – AR-enabled remote assistance
- **O2 Czech Republic** – O2 in the Czech Republic, using WebRTC as part of their contact center for their wireline services
- **Talkdesk** – a cloud-based contact center solution provider making use of WebRTC

☞ *Further Reading:*

- Where Amazon Mayday's service is headed [Disruptive Wireless]
- What to Expect when Deploying WebRTC in Contact Centers?

Avatour

Solution Type	Contact Center	**Website**	avatour.co
Target audience	Enterprises/Industrial	**Country**	United States
Business Model	Unknown		

Avatour offers 360° remote site meetings.

With Avatour, a person can join remotely to a place, enabling him to inspect that location, join a virtual tour or remotely join an on-site training.

To do that, a host needs to use off the shelf 360° hardware or use an Avatour kit. Guests can join from web browsers or using VR headsets. Since the host shows the location using a 360° camera, the guest can change the focus of what he is looking at. This makes the experience extremely useful when the main purpose is to inspect a remote location and its surroundings.

Things to Remember

- Taking the classic video conferencing experience and redefining it in interesting ways can lead to new experiences and use cases
- Off the shelf hardware can lead to these new experiences and use cases

Help Lightning

Solution Type	Contact Center	**Website**	helplightning.com
Target audience	Enterprises/Industrial	**Country**	United States
Business Model	Unknown		

Help Lightning offers AR-enabled, remote assistance software for field services – mainly to support and train technicians.

Unlike regular call centers, Help Lightning's focus is on "see what I see" solutions, where the technician on the field shares his view with a remote technician that then places his hands virtually on that remote view to assist with figuring out how to solve the issue. This kind of merged video experience is rather unique.

Where complex equipment needs to be handled, this makes perfect sense. It enables Help Lightning customers to reduce truck rolls and have the most experienced engineers available to more customers due to the reduced need of physically sending them to every location that needs their assistance.

Things to Remember

- Even within contact centers, WebRTC isn't limited to the traditional user-agent type of interactions
- WebRTC enables field services and similar types of use cases, where innovating beyond the simple video call or "see what I see" interactions can be achieved

O2 Czech Republic

Solution Type	Contact Center	**Website**	www.o2.cz/osobni/en/
Target audience	Consumers	**Country**	Czech Republic
Business Model	Unknown		

O2 CZ is a fully converged service provider. When users check different products and services on O2 CZ website, they can leave their phone number and have a contact center agent call them back.

Once the call takes place, if the browser they use supports WebRTC, then the voice call with the agent can be enhanced with a one way video stream from the agent to the user and along with collaboration capabilities, where the webpage gets “replaced” with a whiteboard the agent can write on.

This approach reduces the average handling time while increasing customer satisfaction for O2 CZ. So much so, that O2 ended up acquiring the outsourcing vendor who developed the service for them.

Things to Remember

- Telecom operations isn’t only about deploying WebRTC services that consumers can use between each other. It is also about improving the operator’s contact center itself
- WebRTC can be used in cases where we only want it to share video – and in a single direction – while letting audio flow through legacy networks
- There is more than one way to use WebRTC in a contact center

Talkdesk

Solution Type	CRM, Contact Center	**Website**	talkdesk.com
Target audience	SMB, Enterprises	**Country**	United States
Business Model	Seat based		

Talkdesk is one of the leaders in cloud contact center solutions.

Part of the philosophy of Talkdesk is to provide the contact center agents with a single pane of glass – a single screen where they can see everything about a contact and interact with him without the need to switch across windows or applications. Towards that end, voice calls conducted between agents and customers make use of WebRTC, as part of the customer view for the agent.

This enables businesses using Talkdesk to provision their agents with a single device where all interactions take place from within the web browser itself. That approach reduces the installation and maintenance time for agents, especially in today's reality of work-from-home agents.

Things to Remember

- Remote work is here to stay after the pandemic as well. This means simpler installation and deployment of agent software is desirable and WebRTC is part of the solution
- Embedding the calling functionality as part of the CRM and call center software makes it easier for agents to use the application

Enterprise Communications

Enterprise Communications has been focusing on Unified Communications – mainly – a way to communicate across an organization in a multinational enterprise.

In recent years, companies have moved to offer Cloud based video conferencing offerings, in an attempt to shift IT spending from CAPEX to OPEX and to start selling video services instead. WebRTC is now part of this trend and can be viewed as an accelerator of it.

Today, WebRTC in the enterprise is a lot more than a feature in Unified Communications offering.

The vendors provided in this subchapter as example use cases were selected due to their varying use of WebRTC within the enterprise space. I've decided not to go with the "tried and true" services such as Google Meet, Cisco Webex or Microsoft Teams and instead go with slightly different vendors, that aren't considered as Unified Communications solutions per-se.

- **Cisco Webex** – a leading web communications service provider
- **Dialpad** – AI powered communications
- **Miro** – collaboration board with communication capabilities
- **Slack** – the Enterprise Messaging posterchild

Cisco Webex

Solution Type	Unified Communications	**Website**	webex.com
Target audience	SMB, Enterprise	**Country**	United States
Business Model	Per seat		

Webex started as a web conferencing solution that was acquired by Cisco in 2007.

It then became the cornerstone of Cisco's video conferencing service, expanding towards contact centers as well.

When WebRTC was introduced, Webex was already a full-fledged service. Adding WebRTC to it wasn't a simple task. Cisco identified the need and the opportunity of supporting WebRTC within its services and joined wholeheartedly.

Cisco took part in and contributed to the discussions around the standardization of WebRTC and contributed the free implementation of its H.264 video codec known as openh264 to the community.

Today, Cisco Webex has their own proprietary communications implementation that interoperates and works with their WebRTC implementation in the browsers. Users can join any Webex meeting directly from their browser.

Things to Remember

- Vendors using other standards or proprietary solutions tend to take a longer time to adopt and migrate towards WebRTC. This is mainly due to existing products and services
- Such vendors still gain from the use of WebRTC. Especially when it comes to the ability to connect to their services from a browser without the need for an installation on the user's device

Dialpad

Solution Type	Customer communications	**Website**	dialpad.com
Target audience	SMB, Enterprise	**Country**	United States
Business Model	Per seat		

Dialpad defines itself as an AI-powered customer communications platform.

It started as a unified communication solution and on the get go had a WebRTC interface for users in web browsers. It was one of the first to also merge unified communications with contact center software into a single application and one of the first to embrace AI as an integral part of the solution (via its acquisition of TalkIQ in 2018).

Today, WebRTC is used at Dialpad for both voice and video communications, powering its client applications across devices and operating systems. In one way, WebRTC is a core competency of Dialpad, enabling their service. In another way, it isn't the leading technology and focus today – that is probably reserved for AI.

Things to Remember

- Communication vendors are adopting WebRTC to speed up their development process and to enable communication in web browsers
- WebRTC is just one of multiple core technologies vendors now need to excel at to be leaders in their market

Miro

Solution Type	Remote collaboration	**Website**	miro.com
Target audience	SMB, Enterprise	**Country**	United States
Business Model	Per seat		

Miro offers a collaboration canvas that is interactive.

People using Miro canvas can remotely interact and see each other's mouse cursor and any change they make to the canvas in real time. It made sense for Miro to add live voice and video communications for its users; and it did so by integrating a third-party CPaaS Programmable Video vendor.

Miro wanted more control over this technology, seeing it as core for its platform. As such, it acquired Around in 2022. Around built a remote communication service. Miro was interested in the technology – embedding it into its Miro platform replacing the third-party with it and owning its own future and roadmap.

Things to Remember

- As we convert our various collaboration activities to the virtual world, we get into such solutions as Miro. A platform where the focus isn't the communication piece but rather the document or canvas we collaborate around. In these cases, the ability of WebRTC to turn communication services into features in other services is priceless
- There's more than a single way to implement communication with WebRTC. You can self-develop, use a third party, rely on open source, etc. It is also possible to switch from one alternative to another, like Miro did with its acquisition of Around

Slack

Solution Type	Enterprise Messaging	**Website**	slack.com
Target audience	SMB, Enterprise	**Country**	United States
Business Model	Free / Subscription		

Slack is the fastest growing enterprise messaging service. Its success has made it the target of many vendors, especially from the Unified Communications space.

In the beginning of 2015, Slack acquired ScreenHero, later folding it into its Slack service and introducing voice calling to Slack. This has since been enhanced to offer video calling capabilities as well. In 2020, Slack decided to halt its internal development of its own WebRTC infrastructure in favor of using Amazon's Chime SDK, probably as part of a larger cooperation between the two companies.

While the main Slack experience revolves around asynchronous text messaging and collaboration, users are now able to call each other or call a Slack channel to conduct a real time call using WebRTC.

Slack offers its voice and video communication services via the web browser and its mobile and PC applications.

Things to Remember

- In a way, Slack's acquisition of ScreenHero has been a talent acquisition. The original service (screen sharing) wasn't added to the official Slack offering that got rolled out in its voice calling
- Messaging services are adding voice and video communication capabilities via WebRTC. They are doing that to be able to compete and as a way of having their users interact with their service more
- Developing WebRTC infrastructure may seem easy to begin with, but has its own set of hurdles, some of which cause even larger vendors like Slack to fold and opt to buy instead of build

Virtual & hybrid events

When you simplify and dilute it enough, virtual and hybrid events platforms are glorified webinar solutions. The shift in importance and focus on the events platforms started due to the pandemic and its quarantines. The main difference from the webinar platforms can be seen in three areas:

1. Scope of the solution – handling subscriptions, running multiple sessions, creating conversations outside of the specific session, etc.
2. Increased focus of interactivity with the audience
3. Powerful studio production capabilities

I've decided to pick a few different vendors here, each with a different focus to events:

- **Airmeet** – virtual and hybrid events platform with a social lounge feature
- **StreamYard** – a livestreaming and webinars platform
- **Vimeo** – video hosting platform extending to live events

Airmeet

Solution Type	Events	**Website**	airmeet.com
Target audience	SMB, Enterprise	**Country**	India
Business Model	Subscription		

Airmeet is another events platform that started life a few months prior to the pandemic. It specializes in virtual and hybrid events. Airmeet includes all the bells and whistles found in many other webinars and virtual events platforms.

One of the interesting features originally promoted by Airmeet is its social lounge, where a visual view of virtual tables and chairs enables attendees to join any table for side conversations before and after the event takes place. This is an interesting attempt to bring back the interactions taking place in physical events and bring them to the virtual world.

Things to Remember

- The virtual events space is still at its infancy, as such, many vendors are trying different interaction modes to mimic real world events
- In the case of Airmeet, this manifests itself in their social lounge feature

StreamYard

Solution Type	Events	**Website**	streamyard.com
Target audience	SMB, Enterprise	**Country**	United States
Business Model	Subscription		

StreamYard came to be by offering a simple platform to record and stream live broadcasts and webinars. StreamYard popularized the concept of using WebRTC to create online content and stream it live to third party platforms such as YouTube Live, LinkedIn Live, Facebook Live, etc.

During the pandemic, Hopin acquired StreamYard as part of a bigger attempt to win both in person and virtual events. After the pandemic, with the weakening of the market, Hopin sold most of its assets, leaving only StreamYard as an independent company. This shows the attraction and usefulness of this tool beyond the days of quarantines.

Things to Remember

- Webinars was a well-established market. WebRTC injected some color, interactivity and differentiation in it
- Many similar platforms have since launched, showing that in many ways, the barrier of entry when using WebRTC is quite low
- The usefulness of StreamYard lives beyond its use for just webinars or events. The author of this report uses it for example when recording course lessons and other online video materials

Vimeo

Solution Type	Events	**Website**	vimeo.com
Target audience	SOHO, SMB	**Country**	United States
Business Model	Subscription		

Vimeo is known for its independent video hosting platform. In recent years, it has expanded to become a collection of tools to help create, manage, and share high-quality videos.

This includes live streaming capabilities that enable hosting live events; or the ability to screen record via a Chrome extension.

What Vimeo did was embed and integrate WebRTC in its newer services, to augment and improve its allure to potential users. You create and conduct your events on the Vimeo platform and then host and share it through it as well.

Things to Remember

- Vimeo's live events are focused on high production capabilities with controls of screen graphics and overlays. These weren't available in most webinars but are becoming more important in the "events" space
- Established companies such as Vimeo are finding ways to use WebRTC and expand their market offering and reach by introducing new products and services with it
- When this happens, usually new experiences are created in existing market segments and solutions, offering synergies with what the vendor has on offer already

Healthcare & Education

Healthcare and education are two interesting market segments when it comes to WebRTC. The challenges here are dealing with regulations and liability as well as with different interaction models.

These two markets have distinct requirements that are specific to each one of them and the vendors in this space are usually razor focused on the industry itself, often adding communications (and WebRTC) as an additional feature to a larger service offering.

WebRTC is uniquely positioned as a very suitable technology. The reasons for that include:

- WebRTC is secured and private by default, whereas other systems can optionally be made secure
- WebRTC is suitable for easy incorporation into business processes – and Telehealth is all about business processes (from a regulatory point of view)
- Development of these niche markets requires a lot of domain specific knowledge and experience– something that is hard to come by in the VoIP domain, so having something like WebRTC that makes it easy to develop VoIP use cases, makes it possible to develop such targeted services

The vendors discussed in this section provide different Telehealth and education solutions that are built differently and have very different business processes:

- **doxy.me** – an easy to use, freemium telehealth solution
- **Teladoc Health** – virtual care provider for a wide range of physical and mental healthcare services
- **Topia** – an online collaboration service who won mindshare in the education market
- **Vedantu** – an online 1:1 tutoring service in India

☞ *Further Reading:*

- WebRTC in telehealth: More than just HIPAA compliance
- Zooming in on remote education and WebRTC
- What to Expect when Deploying WebRTC in Contact Centers?

doxy.me

Solution Type	Healthcare	**Website**	doxy.me
Target audience	Vertical	**Country**	United States
Business Model	Freemium, Subscription		

doxy.me is a telehealth solution that uses a freemium model for its growth. Doctors can use it freely and later can sign up for a professional paid accounts for additional features. Monetization itself also supports clinics and larger organizations.

The free offering is limited to its feature set (no HD video for example), but it does offer unlimited minutes and sessions. This makes it just suitable enough to be used by doctors and enable them to experience telehealth but acts as a nice starting point for the ability to upsell paid plans to them.

At its heart, doxy.me strives to be easy to use, globally available and accessible from the web browsers. There's nothing to install, thanks to its use of WebRTC.

Things to Remember

- Offering a freemium model to healthcare is not an obvious choice. It seems to be paying off to doxy.me as it was their engine for growth
- The approach of going directly to doctors and clinics instead of through larger organizations is one that fits well with this freemium model

Teladoc Health

Solution Type	Healthcare	**Website**	teladochealth.com
Target audience	Vertical	**Country**	United States
Business Model	Licensing		

Teladoc Health offers virtual care to a wide range of physical and mental healthcare services. Teladoc Health focuses on employers, health plans and hospitals in the US and elsewhere as their clients.

The solutions and services provided by Teladoc Health that make use of WebRTC are varied in nature. Some coming from different acquisitions made by Teladoc throughout the last several years.

Things to Remember

- Telehealth has been booming in the last few years due to the pandemic
- This is true the world over
- The pandemic has shown us that a lot of medical care can be done remotely quite efficiently without the need for in person visits. This reduces stress in physical clinics and hospitals as well as reduce travel and wait time for patients
- Operating a telehealth service in multiple countries is tricky as there is a need to meet the regulatory requirements in each country separately

Topia

Solution Type	Education	**Website**	topia.io
Target audience	Vertical	**Country**	United States
Business Model	Subscription		

Topia offers a kind of a virtual 2D world experience where users can roam around the “world” and interact with each other based on their proximity with them. Once you get close enough to a user, audio and video channels to that user opens up for direct communication.

These types of solutions have been introduced during and after the pandemic, with an intent to model office interactions, social events and similar settings. In the case of Topia, they managed to attract education institutions as their customers, where students and teachers can interact online to promote learning.

Such experiences are usually a bit more casual and less rigid than the usual unified communication solutions, offering more flexibility and creativity in the setting of the interactions themselves.

Things to Remember

- 2D virtual worlds offer unique experiences that can be modeled in many ways when it comes to WebRTC media architecture
- This is the first step towards a metaverse. All that is missing is that third dimension 😊

Vedantu

Solution Type	Education	**Website**	vedantu.com
Target audience	Vertical	**Country**	India
Business Model	Revenue sharing		

Vedantu is an education marketplace developed in India for the market in India.

Vedantu enables teachers to join the service as online tutors, and students to join the service and take private lessons from teachers they select.

Lessons on Vedantu are private, connecting a single teacher to a single student, enabling them to interact using voice, video and a whiteboard.

The service started its life when no CPaaS vendor had a footprint in India, requiring Vedantu to build its own WebRTC infrastructure. As part of the service, Vedantu handles the low network quality in India by adding on-screen connectivity signal, being able to rollback to PSTN to keep the lesson going and by being able to charge by the minute, based also on the network connectivity throughout the lesson.

Things to Remember

- Teaching online varies between countries. India poses a challenge when it comes to handling network quality. Vedantu as a local player can focus on the needs of its local market better
- Online private tutoring is happening in different mediums and for different age groups. Vedantu is targeting school related curriculum, but WebRTC isn't limited to this only

Social

Social apps are all the rage. If anything, they accumulate the most amount of time people in certain age groups spend on their mobile devices. Most social apps need to gather over 100 million monthly active users (known as MAU) to register as interesting.

With the trends of selfies and social sharing, comes the improvement and focus of smartphone manufacturers on the front facing camera. This in turn means that each one of us is carrying a very capable recording device, and social apps are making use of it.

Social apps tend to offer a free service to its users, in most cases, profiting through ad delivery or selling virtual goods.

In this section, we will see how these vendors make use of WebRTC:

- **Apple FaceTime** – Apple's built-in video calling service, using WebRTC for guest access
- **Messenger from Meta** – one of the world's largest messaging services, making use of WebRTC for its voice and video calling

Apple FaceTime

Solution Type	Video chat	**Website**	Apple App Store
Target audience	Consumers	**Country**	United States
Business Model	None		

FaceTime is Apple's built-in video calling service for its devices, be it iPhone, iPad or Mac. When FaceTime was announced, it promised to support open standards, but was always kept proprietary.

It 2021, Apple enabled connecting to a FaceTime call from web browsers by using an invitation link. The intent was to allow guest access to FaceTime by non-Apple users. This approach and solution make use of WebRTC for the guests as they connect to the FaceTime infrastructure.

Things to Remember

- Social networks are free to use. The monetization behind them lies in having a big enough audience. For that to happen, they refrain from interoperating and connecting directly with other social networks
- Allowing guest access to a video calling service is desirable to increase the utility of a social network. Doing so using WebRTC, as Apple has done, is quite a common approach
- It is interesting to see how Apple makes use of WebRTC where it makes business sense to them, while dragging their feet in properly supporting WebRTC in its mobile Safari implementation

Messenger from Meta

Solution Type	Messaging	**Website**	messenger.com
Target audience	Consumers	**Country**	United States
Business Model	None		

Messenger from Meta boasts over 1 billion monthly active users.

In 2015, Facebook (Meta) introduced video calling to its Messenger service. The feature was made available on the desktop as well as in its mobile apps for Android and iOS – to that end, it was built with WebRTC. By the end of 2017, Facebook indicated that over 400 million people are making voice and video calls every month in its Messenger service.

Later, Meta added group video calling, limiting the number of participants in a single session to 50.

This is most probably one of the largest video calling services in existence today.

In 2018, Meta also launched Portal, a smart display focused on video calling.

Today, Meta invests in WebRTC across its social products (Messenger and Instagram). It hosts a yearly RTC@Scale event, sharing its own learnings and experiences with WebRTC.

Things to Remember

- WebRTC makes sense for a variety of services
- Even for closed-garden services such as Meta, it is beneficial to adopt WebRTC as part of the technology stack to support voice and video calling
- By integrating WebRTC, Messenger from Meta was able to enhance its service and offer more communication options to its users instead of having them communicate on other dedicated services such as Skype

Streaming and Content Delivery

While many view WebRTC as a video calling technology, it can be used for other types of services as well. This can be done by using its media streaming capabilities or employing its data channel, which enables sending arbitrary data directly between browsers.

In this section, I'll introduce four different vendors, each making use of WebRTC in different ways – either by employing media streaming or by using the data channel – to create streaming and content delivery types of use cases.

The vendors are:

- **Ant Media** – a streaming media server provider
- **Microsoft eCDN** – a service specializing in peer assisted data delivery, focused on CDN and file transfer for enterprises
- **WebTorrent** – A torrent client based on web technology, built on WebRTC's data channel

☞ *Further Reading:*

- Fitting WebRTC in the brave new world of webcams, security, surveillance and visual intelligence

Ant Media

Solution Type	Media servers	**Website**	antmedia.io
Target audience	Enterprises, Developers	**Country**	United States
Business Model	Subscription, Pay as you go		

Ant Media offers scalable real time video streaming solutions. These solutions include all the traditional streaming protocols such as HLS, LL-HLS and LL-DASH. Ant Media also added support for WebRTC early on compared to other media streaming server platforms.

Their solution revolves around the Ant Media Server which can be deployed in cloud service providers or on premise locations.

Ant Media's focus on live and low latency content makes is suitable for use cases such as live auction and bidding, remote control of drones and vehicles as well as other IoT (Internet of Things) use cases.

In 2024, Ant Media introduced WHIP protocol support to its server, opening up the ability for any client device supporting WHIP to broadcast live through its servers.

Things to Remember

- Media streaming was an established market before WebRTC. WebRTC made it relevant for live streaming and expanded its reach and scope
- In the domain of streaming, WebRTC is but one out of quite a few streaming protocols that need to be supported
- WebRTC in the domain of collaboration is usually built around the ecosystem of the browser with a single vendor offering the service end to end. In the media streaming domain, there is higher likelihood of more vendors in the ecosystem, which is where protocols such as WHIP come into play

Microsoft eCDN

Solution Type	Distributed CDN	**Website**	microsoft.com
Target audience	Enterprise	**Country**	United States
Business Model	Subscription		

Microsoft eCDN started as Peer5, an Israeli startup that has focused from the start on the data channel. The entrepreneurs behind Peer5 came with background of networking web but no background in VoIP.

What they found in WebRTC was the ability to connect browsers directly, and they have built on top of that basic function the ability for multiple browsers to cooperate directly amongst themselves when trying to access similar content.

That capability has been used to build an eCDN (Enterprise CDN) that reduces the load from video streaming servers and congestion from internet connections into offices.

Peer5 was acquired by Microsoft in 2021. It got renamed to Microsoft eCDN and wrapped under the Microsoft Teams service. The main purpose of it is to deliver townhalls and organization wide meetings and events to large enterprises.

Things to Remember

- Reduce load on the servers who hold the original content
- Reduce bandwidth requirements from the server farm, thus reducing costs
- Reducing the occurrence of rebuffering on peak loads, due to availability of peers accessing the same content - the more people try to access the same content at the same time the better the system can reduce the load on it

WebTorrent

Solution Type	P2P networking	**Website**	webtorrent.io
Target audience	Developers	**Country**	United States
Business Model	Open source		

WebTorrent is a torrent client that works directly inside the browser. It is built on a similar protocol to the BitTorrent protocol. WebTorrent creates a dynamic decentralized network of browsers that can download a file by obtaining chunks of it from one or more other browsers that has that store them.

As an open-source project, it started by implementing the protocol in JavaScript to make it workable inside the browser. Later, it grew by scope, size and the various uses people found for it.

Today, multiple torrent clients support a hybrid mode that enables the orchestrated use of both BitTorrent and WebTorrent. This means that you can open up a torrent from a browser and stream data directly to it.

Things to Remember

- WebTorrent makes use of WebRTC's data channel, which enables direct communication across web browsers without any need for a server
- In a way, WebTorrent's challenge is bigger than that of BitTorrent, as its clients join and leave torrents dynamically a lot faster, based on the sites they visit

The Missing Use Cases

The list of use cases, market segments and verticals is by no means exhaustive. There are many more verticals that weren't covered here. Some of them traditional, such as telecom vendors, who provide SBC, gateways and other equipment to carriers and enterprises; and others, are markets of their own, such as finance, gaming, conferences and events, etc.

The use cases were selected because each tells a story – usually different than the ones that existed prior to WebRTC.

The ecosystem of WebRTC today covers a large and growing number of vendors and services. Some with no clear business model (or goal) while others are commercial in production services. The ecosystem has been rather dynamic in the past 10 years, with vendors disappearing, popping up, getting acquired or pivoting in other direction. This trend is expected to continue for the foreseeable future.

In some ways, WebRTC has been relegated into the background already, with services using it without announcing that fact at all – in the end, technology is just technology, it doesn’t speak about the end customer’s benefits. This shift indicates a more focused view on delivering the use case than on the technology.

A technical look at WebRTC

Now that we've understood the main concepts of WebRTC and seen some of the vertical and use cases, it might be good to dig a bit deeper into the technical characteristics of WebRTC itself. I will try doing this with a business focus and simple words – staying away from the technical jargon where possible.

WebRTC API Components

WebRTC APIs are designed around 3 main objects. These are:

- **MediaDevices**, enabling access to the cameras, microphones and the display of the device
- **PeerConnection**, taking care of all media transport activities in WebRTC
- **DataChannel**, dealing with proprietary data transmission between clients

MediaDevices

The MediaDevices APIs enable capturing the input devices: camera, microphone and the display, giving access to them via JavaScript. To get over privacy issues, when these APIs are called in a website, the user is asked for permission to give such access.

This API can be viewed as external to WebRTC – in a way, it doesn't deal with communication at all. That said, MediaDevices opens possibilities for a wide variety of services – from photo-booth experiences to capturing a user's photo when onboarding a new service (this is what MailChimp does when you register a new account or edit your profile).

> ☞ *Further Reading:*
>
> - Face detection with GetUserMedia [Raymond Camden]
> - What if Microsoft was Really Serious about WebRTC

PeerConnection

At the heart of WebRTC lies the Peer Connection. This is where most of WebRTC's VoIP implementation lies.

The PeerConnection API holds inside it everything: SDP negotiation, media sending and receiving, handling network issues such as packet losses, codec implementations, NAT traversal, etc. The API itself wraps it all in a simple package that enables adding media streams into a peer connection, negotiating capabilities and NAT traversal.

Most vendors focus on adding value by using the PeerConnection APIs.

> ☞ *Further Reading:*
>
> - An example of using PeerConnection [Big Nerd Ranch]

DataChannel

The Data Channel can be considered the "jack in a box" of WebRTC. It is a generic interface that can carry any type of data – media or otherwise – between 2 WebRTC devices.

While doing that in applications is rather straightforward, WebRTC offers a solid infrastructure that usually isn't available for developers, along with the fact that it offers it inside a web browser in a real P2P fashion:

1. Ability to send data either reliably or unreliably, giving the flexibility for developers to choose what fits their needs without adding additional applicative layers
2. Same security and NAT traversal capabilities used for sending media is extended to the data channel as well
3. P2P support within the web browser for any generic data
4. Adheres to the network characteristics by limiting or extending bandwidth to what is available dynamically

These aspects make the Data Channel an unknown factor in what developers can do. Some of the early implementations that have adopted the data channel include:

- Assisted P2P video streaming, similar to how BitTorrent is working
- CDN (Content Delivery Network) P2P, where multiple users browsing the same website share the resources and images, they download from web pages amongst themselves directly
- Online file sharing
- Gaming and chat applications, where low latency is required

☞ *Further Reading:*

- Introduction to broadcasting over IP
- Sending files using the DataChannel
- WebRTC's secret weapon is the data channel [Chris Kranky]
- 7 Creative Uses of WebRTC's Data Channel

Networking and Supported Features

WebRTC deals with networking much in the same way that other VoIP solutions treat them. The main difference you will find is that WebRTC leaves a lot less options for developers in certain areas; and in the case of WebRTC, fewer options mean better solutions.

VoIP standardization has suffered for over a decade from over-complexity. Put simply, if something can be done with a specific VoIP protocol, it can probably be done in multiple ways, making development, testing and interoperability a real headache.

WebRTC took the route of as little options as possible when it came to issues of how to implement the basics. This chapter covers several areas where such decisions were made, providing a glimpse into the mindset behind WebRTC.

While WebRTC is based on existing specifications, it did divert from the "tried and true" of VoIP. In some cases, it was selected to support specifications that weren't popular or in wide use in the industry. In other cases, it tweaked and improved the existing specifications to fit its own needs.

The topics covered here are:

- **P2P** – the type and nature of the peer to peer that WebRTC provides
- **Signaling** – what role does signaling take in WebRTC
- **Firewall and NAT Traversal** – the mechanisms of NAT traversal supported by WebRTC
- **Security** – what is the security paradigm employed by WebRTC
- **Media processing** – how WebRTC handles media processing
- **Interoperability** – where interoperability fits into the WebRTC story
- **Extensibility** – focus areas enabling differentiation

P2P

WebRTC is said to be a P2P protocol, by which it means that 2 peers connected via WebRTC get connected directly to each other. While that may well be true, there are several aspects of WebRTC's P2P nature that need clarification:

1. WebRTC is no different than most VoIP solutions

WebRTC provides only the media path, and many of its predecessors use much the same techniques to send media directly between peers. In a way, there's nothing new in WebRTC's P2P nature when compared to SIP-based deployments or other proprietary deployments. As with other VoIP protocols, WebRTC can work P2P or via servers – it is up to the implementer (and the network conditions) to decide which architecture is used.

2. WebRTC is the only P2P technology in web browsers

The main difference of WebRTC's P2P nature is within the web browser itself. Web browsers have been designed to allow communication between the web browsers to a server. Up until WebRTC, there was no way for one web browser to talk directly to another web browser without the intermediation of a server. With WebRTC, that becomes possible

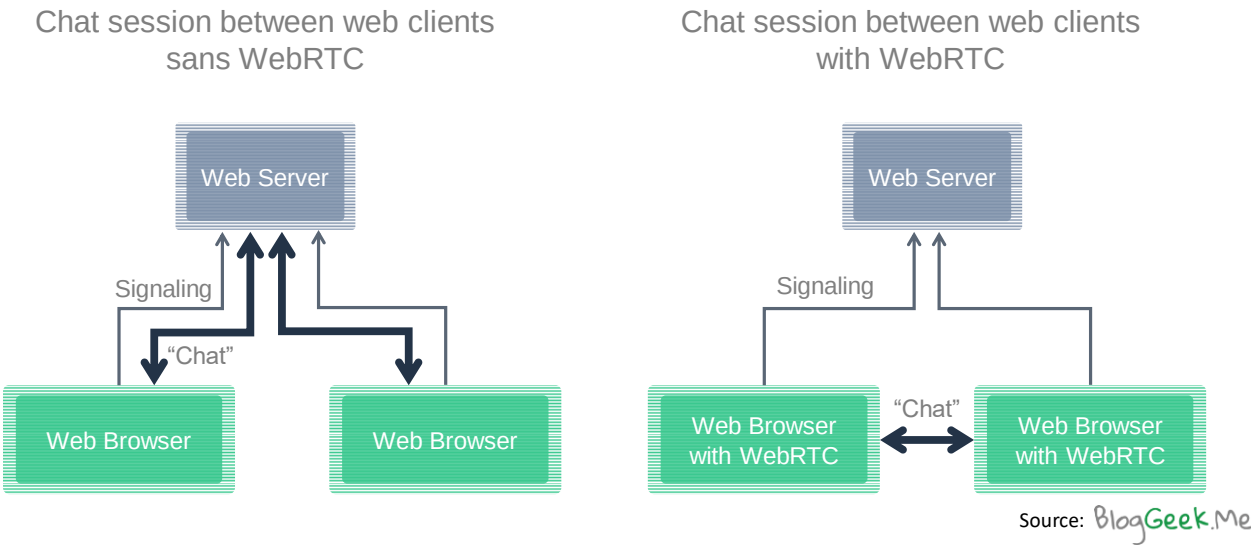

Figure 11: Chat session between Web browsers with and without WebRTC

3. WebRTC may require media relay

While WebRTC offers a P2P solution, at times, it requires relaying the media itself via intermediary TURN servers. More on that in the next subchapter on NAT Traversal.

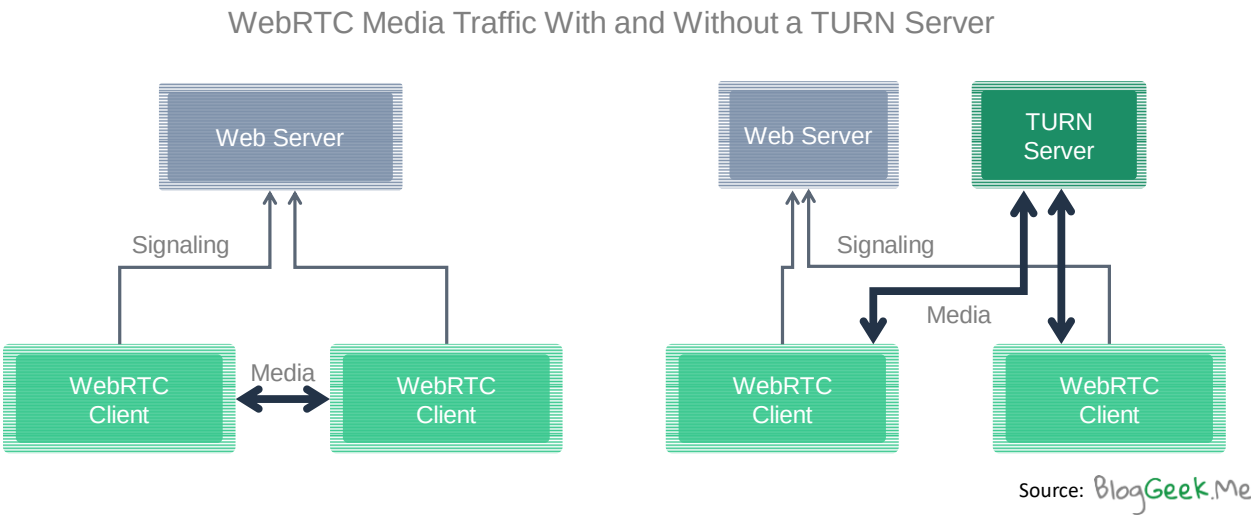

Figure 12: WebRTC media traffic with and without a TURN server

4. WebRTC requires external signaling, which isn't P2P

While WebRTC offers a P2P solution, to reach another peer, there needs to be some centralized decision-making processes that occur on the external signaling protocol that each service chooses to adopt.

☞ *Further Reading:*

- 4 facts about WebRTC and P2P
- The role of the server in a WebRTC P2P service [vLine]
- What is WebRTC P2P mesh and why it can't scale?

Signaling

Signaling was specifically left out of WebRTC. There are those that see this as a positive decision while others believe that signaling needs to be added under the fold of WebRTC.

The reasons for not having signaling as part of WebRTC are various:

1. Adding signaling to the WebRTC specification would drag the process through standardization – there are more existing options for VoIP signaling than there is for media processing
2. WebRTC can be hooked up to use different signaling protocols today. Selecting one mechanism, either existing or new, would mean reducing the use cases and deployments where WebRTC can bring value
3. VoIP signaling is good for VoIP. WebRTC adds a web component that was always missing in VoIP, where a lot of services already have some form of signaling. Being able to accommodate these services made sense

The only component within WebRTC that does deal with signaling is SDP. SDP stands for "Session Description Protocol" and is used for describing multimedia session capabilities and negotiating them. It is a critical first step towards connecting media streams.

SDP is used in SIP, and as with other VoIP related issues, there are those that believe it isn't suitable for WebRTC while others see it as sufficient.

SDP can be considered as a "necessary evil" which will be part of WebRTC for years to come until a better/different solution is negotiated in the standardization bodies.

A more detailed discussion on the various signaling options can be found in the next chapter.

☞ *Further Reading:*

- The death of signaling as we know it
- Why signaling isn't important anymore
- Node.js at the service of WebRTC
- Signaling options for WebRTC applications [webrtcH4cKS]
- SDP: Session Description Protocol RFC [IETF]

Firewall and NAT Traversal

To get media flowing from one browser to another, the media packets may need to pass through NAT devices. NATs are Network Address Translators. They are deployed everywhere – in our home DSL boxes, enterprise boundaries and in our service providers' backend. They have various roles and they come in different shapes and sizes. To that, enterprises and home devices add Firewalls, preventing unauthorized access to the network – these are unaware of VoIP and WebRTC related traffic and block it by default.

In most daily scenarios, our devices have local IP addresses – ones that cannot be used globally. When we access the web, the NAT device replaces the local IP address of our requests with a public IP address. This process is done in different ways by different NAT devices. This causes headaches for those who need to pass NATs to communicate – especially when that communication is bidirectional in nature.

The main challenge with NAT traversal in WebRTC is the fact that it is a new challenge for web developers, who are accustomed to having data pass through NATs and firewalls seamlessly for web traffic. They are unaware of the intricacies, challenges and cost of dealing with NATs in the real-time communications domain.

The selected solution for WebRTC in NAT traversal is the use of ICE, which in turn makes use of STUN and TURN. These are the same three protocols used by SIP. As with other similarities, here too WebRTC takes the approach of using the latest possible drafts and ideas and mandating them for use with WebRTC.

Trying not to dive into details, here is a short explanation of each of the 3 specifications used in WebRTC for NAT traversal:

1. **STUN** – provides means for the WebRTC client to find its own public IP address simply by asking an external STUN server. Knowing this information in many cases is enough to get a WebRTC session to connect
2. **TURN** – when STUN fails, TURN can be used to relay all media between the WebRTC clients. This requires more bandwidth and processing power from the backend, but it gets more calls to connect
3. **ICE** – ICE is in charge in orchestrating this whole show – deciding which route and NAT traversal mechanism best fits a given moment in time for the session

At the end of the day, STUN and TURN servers need to be deployed, which is a hassle. Developers use multiple solutions for such a case:

1. Decide to deploy only with STUN support, and use publicly known STUN server addresses (hoping they will not be moved or removed)
2. Deploy with STUN and TURN, installing and maintaining such servers as part of the operation
3. Use third party SaaS provider for NAT traversal, who provide paid STUN and TURN server access globally and at scale

It should be noted that there are cases where none of the techniques used by WebRTC can successfully get a session to connect. This usually happens when the endpoints in question are within enterprises with strict security rules. There are multiple drafts proposed to the IETF to address this issue – mainly by tunneling media traffic over web sockets.

☞ *Further Reading:*

- Introduction to NAT Traversal in WebRTC [webrtcHacks]
- An overview of NAT Traversal by AudioCodes [SlideShare]
- WebRTC TURN: Why you NEED it and when you DON'T need it
- ICE always tastes better when it trickles [webrtc Hacks]
- ICE: Interactive Connectivity Establishment RFC [IETF]
- Why Doesn't Google Provide a Free TURN Server?
- Introducing ICEPerf.com [Nimble Ape]

Security

Security is part and parcel of WebRTC. Where other VoIP protocols have taken the route of allowing security as an optional feature (usually as an add-on years after the original standard was specified), WebRTC has no means of sending media without having it encrypted.

This approach reduces the number of options and decisions developers have to make about the way they use WebRTC and results in a robust solution suitable for modern communication.

There are several important aspects to remember when dealing with WebRTC security:

1. WebRTC uses SRTP and AES encryption for the media being sent. These are both widely used and mature specifications
2. While media communication is secure with WebRTC, the signaling part isn't. It is up to the developer doing the integration with a specific signaling solution to take care of protecting that path – it can be as "easy" as placing that communication over HTTPS/TLS
3. WebRTC in the browser have an additional factor of security – any security holes found within WebRTC are handled by the browser vendors. While this reduces the control of implementers, browser vendors have been very responsive to date in closing known security holes much faster than vendors in other domains
4. With the current awareness around security and privacy due to the Snowden Affair and talks about the NSA practices with both carriers and web vendors, WebRTC is sometimes touted as a possible solution due to its secured nature. That is usually overstated as WebRTC is a technology that can be used in decentralized scenarios but also in centralized ones

Never take security for granted, but keep in mind that WebRTC brings top notch capabilities in security that are not readily available elsewhere.

☞ *Further Reading:*

- Everything you need to know about WebRTC security
- How security and privacy may be threatened by WebRTC
- The Best WebRTC Security is Prone to the Stupidest Developer

Media Processing

WebRTC enables media processing in both the client-side as well as the server-side. This isn't achieved directly by WebRTC but is enabled by WebRTC.

Client-Side Media Processing

On the client-side, WebRTC offers several powerful capabilities for media processing:

1. Ability to pass captured video to WebGL and WebAssembly, for further client-side manipulation and processing of the video
2. Ability to pass captured audio to WebAudio, for further client-side manipulation and processing of the audio
3. Ability to place captured video on a canvas element, manipulate the canvas element, and then use the canvas as a video input back into WebRTC
4. Ability to capture and record media locally
5. Ability to fork/forward media across peers

Client-side media processing is still in its infancy with WebRTC. Most of the use cases currently being experimented with are around video background replacement and audio noise suppression. For most use cases, you will need to rely on server-side media processing instead. That said, you should explore client-side media processing to see if they fit the task at hand.

Server-Side Media Processing

Server-side media processing isn't covered by WebRTC from a specification point of view, or from an official implementation one. The assumption is that developers need to take care of this part on their own.

The main scenarios/features that are covered by server-side media processing include:

1. Multiparty conferencing – having multiple participants join a single session together and being able to interact in real time. Group sizes vary, but are usually kept below 50 active participants
2. Recording – the ability to record and archive a session, either with a single stream or multiple streams
3. Broadcasting – sending a stream or even a multiparty conference to a larger audience of passive viewers/listeners

Server-side media processing is a popular architectural element in many of the commercial offerings that end up using WebRTC. In many cases, just the need to record a long 1:1 session is enough to warrant the use of a media server.

Interoperability

WebRTC can live in a flux, where a specific use case can live in its own island. In such cases, there is no external ecosystem you need to connect with. In some other use cases, connectivity to other domains is mandatory – things like integrating with an existing PBX system or to the global PSTN are examples of this.

Such a requirement around connectivity is usually referred to as interoperability. Interoperability in WebRTC happens in three separate domains, each with its own challenges:

1. **Signaling** – adding signaling on top of WebRTC isn't always possible and has its own nuances. SIP and H.323, for example, are challenging as WebRTC signaling protocols:
    a. Accommodating SIP isn't supposed to be hard, but it does have its nuances**:** the SDP is managed slightly different, and many of the RFCs and features that are mandated in the media streams for WebRTC are considered new and optional for SIP. Bridging the gap requires careful planning and awareness of the problem
    b. H.323 cannot be glued on top of WebRTC within the browser itself. This may be true with other signaling protocols as well. Providing connectivity between WebRTC clients to an H.323 deployment is not trivial
2. **Networking** – WebRTC comes with mandated security as well as support for the latest RFCs (ability to send multiplex media on the same network port, etc.). It means that existing and legacy deployments can't directly interoperate with WebRTC without some mediation in-between
3. **Codecs** – Some of the codecs selected for WebRTC are different than those used in other VoIP deployments. This means that connectivity can occur only by transcoding the media between WebRTC and external networks

These challenges are usually solved by mediation mechanisms, where the most common ones used are gateways and session border controllers (SBC). There are numerous vendors offering this equipment, tooling and services in different shapes and sizes, so finding one that fits your needs without the hassles of developing it from scratch is possible.

Further Reading:

- Multiple vendors announce gateways for WebRTC
- The necessity of gateways

Recommendations

This chapter can be seen as a shortlist of recommendations of the various insights found throughout this research paper. They are general guidelines to those who wish to build services that use WebRTC.

1

Start from a business case

- Identify a challenge or a problem you wish to tackle, preferably with a pain-point applicable to a specific market niche
- Understand who will be willing to pay for what to solve this problem

2

Research the competition

- There are many WebRTC vendors out there providing solutions. Make sure you know who provides similar capabilities to what you plan today
- Google to find who they are, check on WebRTC blogs and news outlets to see who is out there
- Play a bit with WebRTC services online – see how the interactions feel like, what have people done with it

3

Validate that WebRTC can cover your needs

- Make sure it is available on the web browsers that your target audience uses
- Decide what your fallback strategy to browsers that don't support WebRTC is (options are: none, plugin, Electron, native app)
- Don’t forget to decide on how you wish your service to run on mobile devices: which ones, in an app or on the web, etc.

4

Analyze the ability to use existing tooling solutions

- Are you comfortable with outsourcing the work for external vendors that specialize in it?
- Can you use an API vendor to deal with the whole WebRTC part?
- Should you self-host and manage everything or use CPaaS offerings for things like NAT traversal, signaling and/or media?
- Are there open-source modules available that you can integrate into your solution?

5

Select technologies based on existing skillsets

- What kind of backend do you need for your service?
- Which frontend capabilities are required?
- Do you have the resources in your company to handle this?
- Be sure to have a web developer on board
- Depending on the use case, you might want to consider having a veteran VoIP engineer or architect on board

☞ *Further Reading:*

- Best practices for WebRTC POC/Demo development

Appendix A: Online Resources

This appendix provides some links to other online resources out there, where the main focus isn't a technical one. If you are looking for thought leadership, business insights or market related information about WebRTC, then this can serve as an initial reference list.

Blogs

There are many blogs who are now covering WebRTC; some of them are large news outlets while others bring deep thinking as well as user stories.

Below is a hand-picked list of such blogs. You can consider them mandatory reading list when it comes to understanding WebRTC.

BlogGeek.me

My very own blog, where I write a lot about WebRTC and try also to give room for experts by way of guest posts.

https://bloggeek.me/

WebRTC Weekly

A weekly newsletter providing roundup of posts and articles from around the web about WebRTC. Curated by Chris Koehncke and me.

http://webrtcweekly.com/

WebRTC Glossary

Not exactly a blog but bear with me. This is a glossary site for all terms related to WebRTC.

http://webrtcglossary.com/

webrtcHacks

This site deals more with the technical aspects of WebRTC, highly curated and relevant topics. Very recommended.

http://webrtchacks.com/

WebRTC for the Curious

An open-source online book created by WebRTC implementers to share their hard-earned knowledge with the world. It's written for those who are always looking for more and don't settle for abstraction.

https://webrtcforthecurious.com/

Communities

There are several online communities where conversations around WebRTC happen. Most of these revolve around three main places:

1. discuss-webrtc mailing list, where developers ask questions and (sometimes) Google may answer. This mailing list is maintained and moderated by Google
2. reddit's webrtc page has questions related to WebRTC, with quite a few people replying them
3. Specific discussion boards of open source frameworks – each has its own unique discussion board for questions

Meetup Events

At the time of writing, there were over 50 different WebRTC related meetup groups around the world. These are localized groups that conduct local events about WebRTC. The target audience as well as the content of each meeting vary widely, but if there is a meetup located close by to where you live, and it is active, then join it to gauge the local community that is hacking its way around WebRTC.

Appendix B: Finding out More

Other than this report, the following services are available:

BlogGeek.Me Blog

My blog is the place where the bulk of my writing exists. It is available freely and can be subscribed via both RSS and email. See https://bloggeek.me

WebRTCweekly Weekly Newsletter

The WebRTCweekly is a weekly newsletter. It lists articles from around the web about WebRTC from the past week. This service is offered freely and is curated by me and Chris Koehncke. See https://webrtcweekly.com/

Reports

This is my second report about WebRTC. To find out about other paid reports I have written, see https://bloggeek.me/reports/

Online Course

I now offer an online course on Advanced WebRTC Architecture. To learn more see https://webrtccourse.com

Consulting

I provide consulting services to vendors, especially around VoIP, video conferencing and WebRTC. See https://bloggeek.me/consulting/

Testing and Monitoring

In 2015, I co-founded a company called testRTC, which got acquired by Spearline in 2021.

testRTC offers testing and monitoring services for those who develop and deploy WebRTC based service. See http://testrtc.com

Appendix C: Our Sponsors

Ant Media

Ant Media Server is an innovative and highly scalable platform that delivers **ultra-low latency** and **adaptive streaming** for a wide range of applications, from live broadcasting and video conferencing to gaming, e-learning, and telehealth. It's developed by Ant Media Inc. a leading technology company specializing in live video streaming solutions.

Ant Media Server supports multiple streaming protocols, including **WebRTC, RTMP, HLS, RTSP, SRT**, **Zixi, LL-HLS / LL-DASH,** and provides seamless integration with popular cloud providers such as AWS, Azure, and Google Cloud. Our platform is built to be flexible, offering on-premise and cloud-based deployment options, and scalable to handle up to **millions of viewers**.

With advanced features like **adaptive bitrate streaming, recording, security tools**, and **rich SDKs/APIs** for mobile and web applications, Ant Media Server empowers organizations to deliver reliable, high-quality streaming experiences to their audiences. Trusted by thousands of customers in over 120 countries, Ant Media continues to push the boundaries of live-streaming technology to help businesses and developers achieve their real-time streaming goals.

- Customer Success Stories
- Free Video Conferencing Tool – Circle

Tata Communications Kaleyra Video

Kaleyra Video is the real-time video and audio collaboration API for businesses that want to ship integrated and branded experiences to interact with their customers provided by Tata Communications.

Easily add video calling and collaboration to the interactions with customers always keeping the brand at the core without the need to deal with the complexity of maintaining a WebRTC infrastructure allowing seamless scalability and focus only on “making your beer taste better”.

Kaleyra Video provides

- **Flexible APIs:** Prebuilt customizable UI, APIs, and tools to allow different use cases including telehealth, customer support, video banking, and more
- **Support and resources:** Top notch documentation, access to solution engineers that can help design integration, and sample code for all kinds of customers
- **Global with multi region infrastructure:** 99.9% uptime, 100% compliant HIPAA with a BAA, GDPR, SOC-2
- **Call data and events:** Gain insight into usage with usage dashboard and webhook events to help you monitor and identify any customer issues

Product

- Add calls capabilities into any app or website
- Screen sharing, files sharing, chat, snapshot, whiteboard, recording, transcription, e-signature
- Global reach and scale with multi regional infrastructure
- Technical support

Benefits

- Enterprise-ready infrastructure and support
- Faster go-to-market and time-to-value
- Move fixed CAPEX to scalable OPEX
- Top-class call quality and bandwidth optimization
- Built to simplify developer experience
- High integrability

Use cases

- Video banking
- Customer support
- Telehealth
- Claim & damage assessment
- Insurance

If you're interested in having more information or a demo please visit https://www.kaleyra.com/video/ and contact us.

Nimble Ape

Ideas are easy. Expertise is hard to find. At Nimble Ape, our aim is to take your Real Time Communications and Media Delivery concepts and turn them into reality.

We are passionate about building the right outcome for you, so we work hard to deliver bespoke, robust, long-lasting solutions that are entirely tailored to your needs.

We've worked across a wide spectrum of industries, and with a wealth of global clients, so we're confident that we will be able to help you, no matter your needs. Whether you're a single person start-up or large corporation looking for support on a new project, we'll support you no matter your size.

Founded in 2013, Nimble Ape has seen many changes in how we deliver media to end users and we pride ourselves in being driven by innovation and make sure we stay at the forefront of the industry so that we are best-placed to deliver world-class projects.

If all of that sounds exciting and you want to find out more about how we can help you with your project, contact us at contact@nimblea.pe or get in touch on our socials. We look forward to speaking with you.

WebRTC.ventures

WebRTC.ventures is one of the few software development agencies in the world dedicated to WebRTC. We build, assess, optimize, test, and manage real-time communications applications. WebRTC.ventures also specializes in AI integrations and contact center modernization.

Founded as AgilityFeat in 2010, we began to focus on WebRTC in 2015. We are a global company headquartered in Virginia (USA) with a QA/testing center in Panama City, Panama and an additional development office in Colombia. Our team of developers, designers, testers, DevOps engineers, and project leads hail primarily from North and South America.

We can work independently or side-by-side with your team to augment their capabilities. We can also provide technical services beyond just video, voice, chat, and AI to help you scale your application and manage it in production.

WebRTC.ventures is proud to produce WebRTC Live, a monthly webinar series with industry guests about the latest use cases and technical updates for WebRTC.

- Read our blog
- Subscribe to our YouTube channel
- Follow us on LinkedIn
- Submit a contact form