A KelCor Thought Leadership Paper

# Transforming Customer Engagement Using WebRTC APIs and Frameworks

## How Extending Websites and Mobile Apps into the Contact Center Using WebRTC Enables Rich, In-Context Customer Interaction

**September 2015**

Sponsored by

# Table of Contents

# Executive Summary

WebRTC is an emerging communications standard available on billions of PCs, smartphones, and tablets. It enables real-time voice, video and data sharing in a web browser without the need for browser plugins or downloads. When WebRTC-enabled applications are coupled with contact centers, customer engagement can be transformed because customers and prospects can communicate with a contact center agent via voice or video from within the context of browsing a web page or from within a dedicated mobile app. This user context can accompany the call to the contact center agent, thus giving agents more information about user intent, which allows the agent to satisfy the user's needs faster and better.

This paper provides a brief description of how WebRTC works followed by examples of three real-world applications that have integrated WebRTC-enabled engagement strategies with the contact centers: Amazon, American Express, and Plantronics. It discusses how organizations can rapidly develop and deploy WebRTC-enabled applications using WebRTC APIs which provide significant functionality not found in the standard alone, including WebRTC gateways to the public switched telephone network (PSTN) or SIP trunking connections. We describe the AT&T Enhanced WebRTC API and show how the unique features of this framework can be used. We conclude with planning suggestions and important points to consider when preparing a WebRTC-enabled customer engagement strategy.

# A Short Description of WebRTC

WebRTC is an emerging standard that enables real-time voice, video and data sharing in a web browser[1] without the need for browser plugins. Potentially billions of devices supporting a browser – PCs, laptops, smartphones, tablets and a host of new devices – from a variety of manufacturers will be real-time communications-enabled. Although third-party programs like Skype have been around for a long time, and some browser-based plugins have been available for limited communications interactions, the implications WebRTC brings to organizations of all types and sizes are enormous. Ubiquitous voice, video, and data for gaming, customer service, communications, and personal and group engagement opens a new world of possibilities for innovation and disruption. The transformative power behind WebRTC is that ordinary web developers using just JavaScript Application Programming Interfaces (APIs) can craft fully functioning voice, video and data collaboration applications or embed these capabilities within other applications with minimal coding and without the need for understanding the underlying communications protocols and networks.

## Powered by a Triangular Peer-to-Peer Architecture

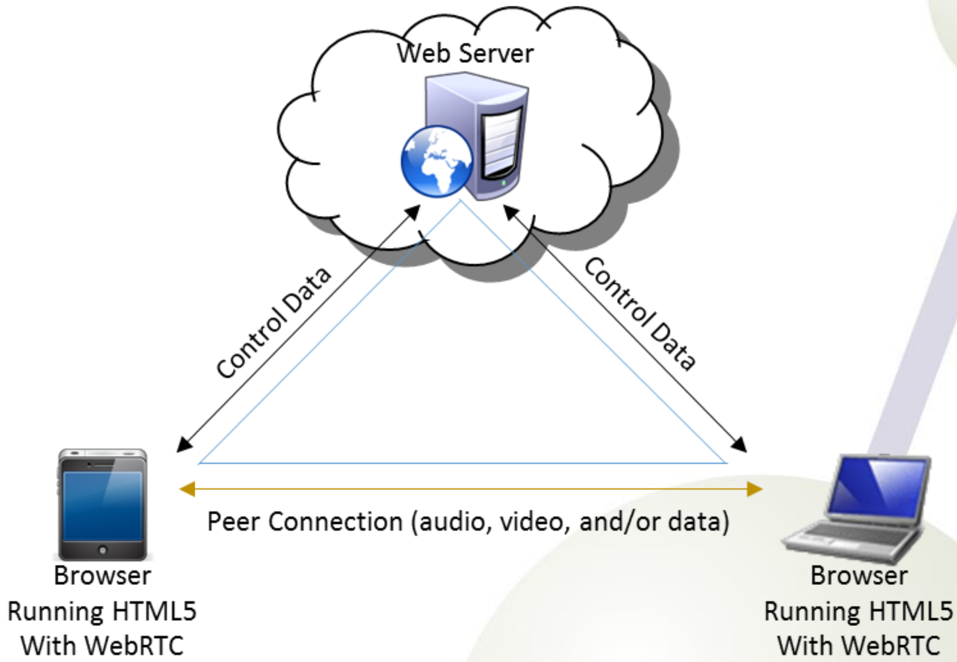
The WebRTC architecture involves web servers and browser clients[2]. The server "serves up" web applications with embedded JavaScript, and the browser clients (PCs, tablets, smartphones) run the JavaScript application. Traditionally, browsers have communicated only with web servers. What is unique about WebRTC is that the web application can now enable peer-to-peer (P2P) communications between two browser clients (See Figure 1).

---

[1] WebRTC is also being used in numerous mobile applications. When used in mobile apps, WebRTC audio, video, and data sharing software is embedded within the app directly or through third-party software libraries optimized for the particular mobile platform the app runs on.

[2] Because the WebRTC audio and video source code has been provided by Google at no charge, some organizations are also embedding WebRTC into native apps on smartphones and tablets.

**Figure 1. WebRTC's Triangle Architecture**




While the control data flows between the browser client and the web server, the audio and video streams flow directly between the browsers. Transmitting media directly between browsers is very useful because voice and video are very sensitive to network latency and jitter[3]. In addition, WebRTC media is encrypted, making conversations between users secure.

WebRTC enables point-to-point browser communications as well as multipoint communications sessions. In a multipoint session, each browser sends and receives audio, video and data streams to and from every other browser in the session in a fully meshed configuration[4].

## WebRTC Federation with SIP and the PSTN

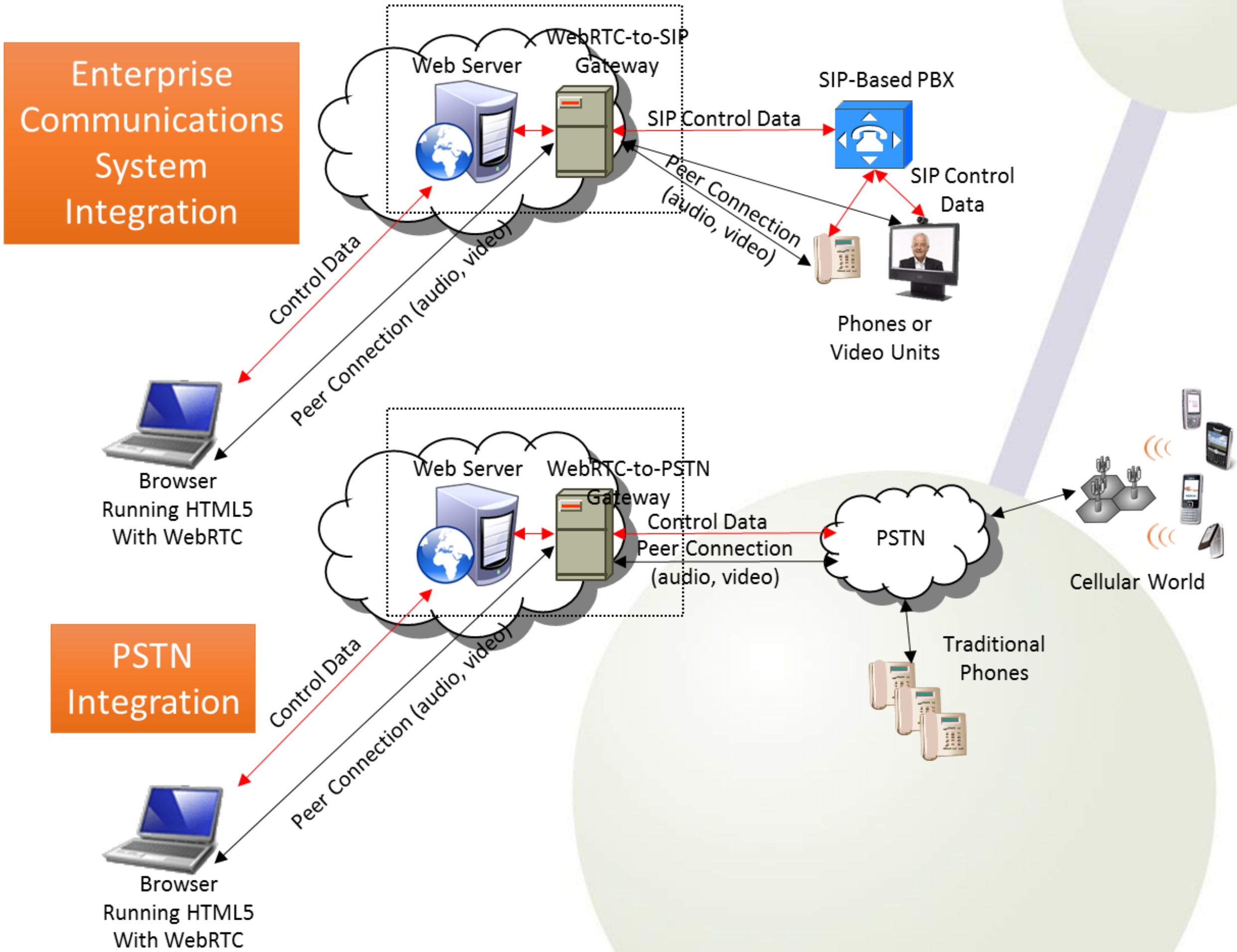
Although WebRTC capabilities are ubiquitous in the browsers many people use, the ability to reach out and connect to others who may not be connected to the same web server is an essential capability. It is also important in many applications that WebRTC-enabled voice and video calls interoperate with SIP-based communications systems and with the public switched telephone network (PSTN). Consequently, web servers supporting WebRTC may ultimately need to be able to federate with one another and with public and/or private voice infrastructure. Federating between the WebRTC domain and SIP or PSTN communications infrastructure results in an architecture as shown in figure 2.

---

[3] Latency is the time delay between when a network packet is sent by one endpoint and when it is received by another. Real-time voice and video typically require less than 150 milliseconds latency. Jitter is variability in the order in which packets arrive. If there is too much jitter, the real-time voice and video application may not have enough time to put the packets in order once they arrive, thus causing distortion to an audio/video conversation.

[4] The ability of WebRTC to support multipoint communications is extremely helpful; however, it does not scale well beyond just a few connections due to the media encoding and decoding demands put on the processor. This is particularly true for tablets and smartphones.

**Figure 2. Integrating WebRTC with the Enterprise Communications Systems and the PSTN.**




## The Need for Media Gateways

The figure above shows a voice/video gateway between the WebRTC domain and the SIP or PSTN worlds. A WebRTC gateways may provide a number of capabilities including:

1. Signaling gateway –WebRTC does specify the signaling that must occur when establishing a communication session; consequently there must be some type of translation between how the WebRTC application establishes and controls a communications session and how the SIP or PSTN systems control communications. Even if the WebRTC application uses SIP for signaling, there are so many different “SIP-compliant” implementations that a signaling gateway may still be required.

2. Media gateway – WebRTC may not use the same voice and video codecs[5] as are found in an enterprise or public communications system. If this is the case, then media transcoding will be required to enable communications between these disparate systems.

[5] Codec is short for “coding and decoding”. When voice or video is transmitted over an IP network, voice or video media must be compressed by an endpoint. It is then packetized into IP packets, sent over the network. The receiving endpoint decompresses the voice and video stream and plays or displays it as

3. Security and compliance – WebRTC uses cryptography to provide security by encrypting both the control data and the media. A gateway may be required to decrypt WebRTC control and media flows before they can be integrated with another WebRTC domain or the SIP/PSTN world.

4. NAT and firewall traversal – Most browsers reside behind both firewalls and network address translation (NAT) devices. These devices tend to block real-time voice and video. WebRTC uses a protocol called ICE (Interactive Connectivity Establishment) for traversing NATs and firewalls. ICE relies on STUN (Session Traversal Utilities for NAT) and TURN (Traversal Using Relay NAT)[6] protocols and servers. A WebRTC deployment may need to establish STUN and TURN servers to provide higher likelihood that voice and video packets will traverse the network boundary.

## WebRTC Requires Directory Services

One of the elements native WebRTC does not supply is a directory service. A directory service is necessary so that WebRTC users can find one another or so that they may interact with someone who is on a standard telephone or a mobile phone.

In some cases, directory services will be provided by interfacing with a website's authentication mechanism[7] or with an existing enterprise directory. When a browser connects to a website, the application can ask the user for login credentials. As the user is authenticated, the web server creates a directory that maps authenticated users to active browsing sessions. Directory information can then be pushed down to the browser interface, allowing people to find and communicate with one another.

An alternative scenario would be a customer service website that interfaces to a contact center. In this scenario, the user browsing the website does not authenticate, only the contact center agent requires authentication. The web server can automatically create the linkage between the customer and a contact center agent. This can be a direct WebRTC-to-WebRTC communications session, or it can be a WebRTC-to-PSTN session, depending upon how the WebRTC capabilities are integrated with the call center (we discuss WebRTC to contact center integration scenarios later on).

# *Examples of Customer Engagement Using WebRTC*

Customers show extreme loyalty to those companies offering excellent customer engagement options, and forward thinking organizations are investing in application development as a way to better connect with their customers. The more innovative companies are already crafting engagement-enabled websites and applications equipped with live assistance and self-service capabilities that deliver a transformational experience and drive very high levels of customer satisfaction.

---

appropriate. In the video world, the use of the term codec is so widespread that video endpoints are often called "codecs". WebRTC uses the wideband Opus codec and the narrow band G.711 codec for audio encoding.

[6] A STUN server simply provides the IP address of the outermost NAT. This becomes one possible IP address the WebRTC server can send real-time voice and video traffic to. If this does not work, a TURN server may be used. A TURN server is like a trusted intermediary server through which all voice and video traffic is routed. In the WebRTC case, rather than having peer-to-peer connections between browsers, each browser would establish a peer-to-peer connection with the TURN server, which would relay voice and media between the browsers.

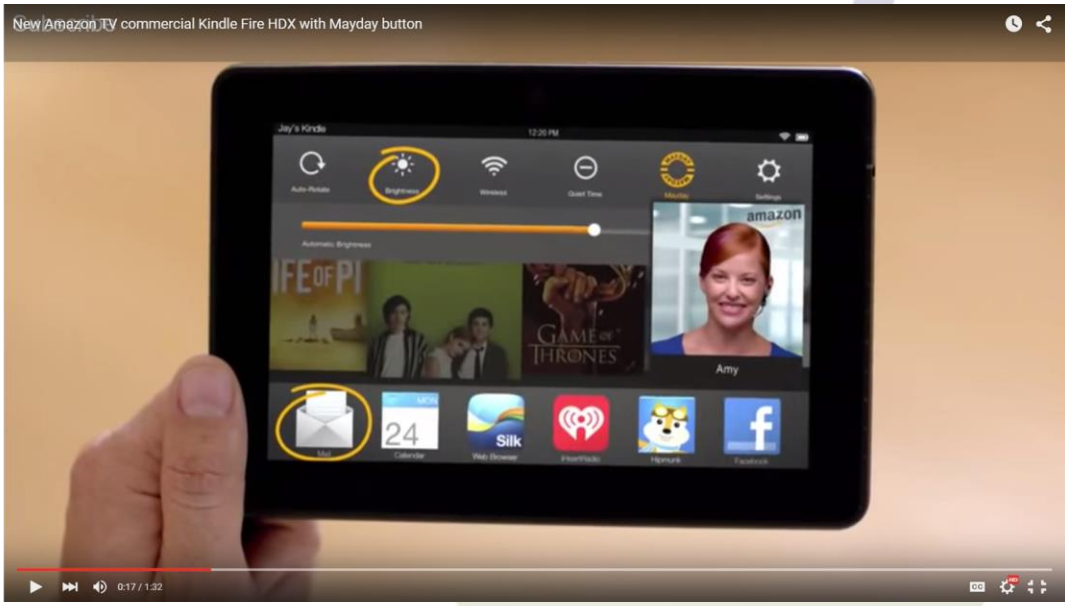
[7] Think of sites like LinkedIn or Facebook. In order to communicate with others, users would need to log in using their LinkedIn or Facebook credentials so that the system would know who they are.

## The Kindle "Mayday" Button

An example of this type of customer engagement is found in the Amazon Kindle HDX. A simple "swipe down" from the top of the Kindle screen exposes a Mayday button, which when touched enables Amazon live customer assistance.

By clicking on the "Connect" button, a Kindle user can be in live audio and video contact with an Amazon tech support person within 15 seconds. The support agent can then provide helpful information including "how to" instructions as well as on-screen annotation and co-browsing.

**Figure 3. The Amazon Kindle HDX Mayday button provides instantaneous live assistance (source Amazon Mayday Commercial on YouTube)[8].**


The Mayday functionality illustrates how a company can take an existing application and enhance it to provide better and timelier user engagement, all while continuing to use existing back-end infrastructure investments. In the Amazon Mayday case, the existing contact center infrastructure is integrated with WebRTC technology.

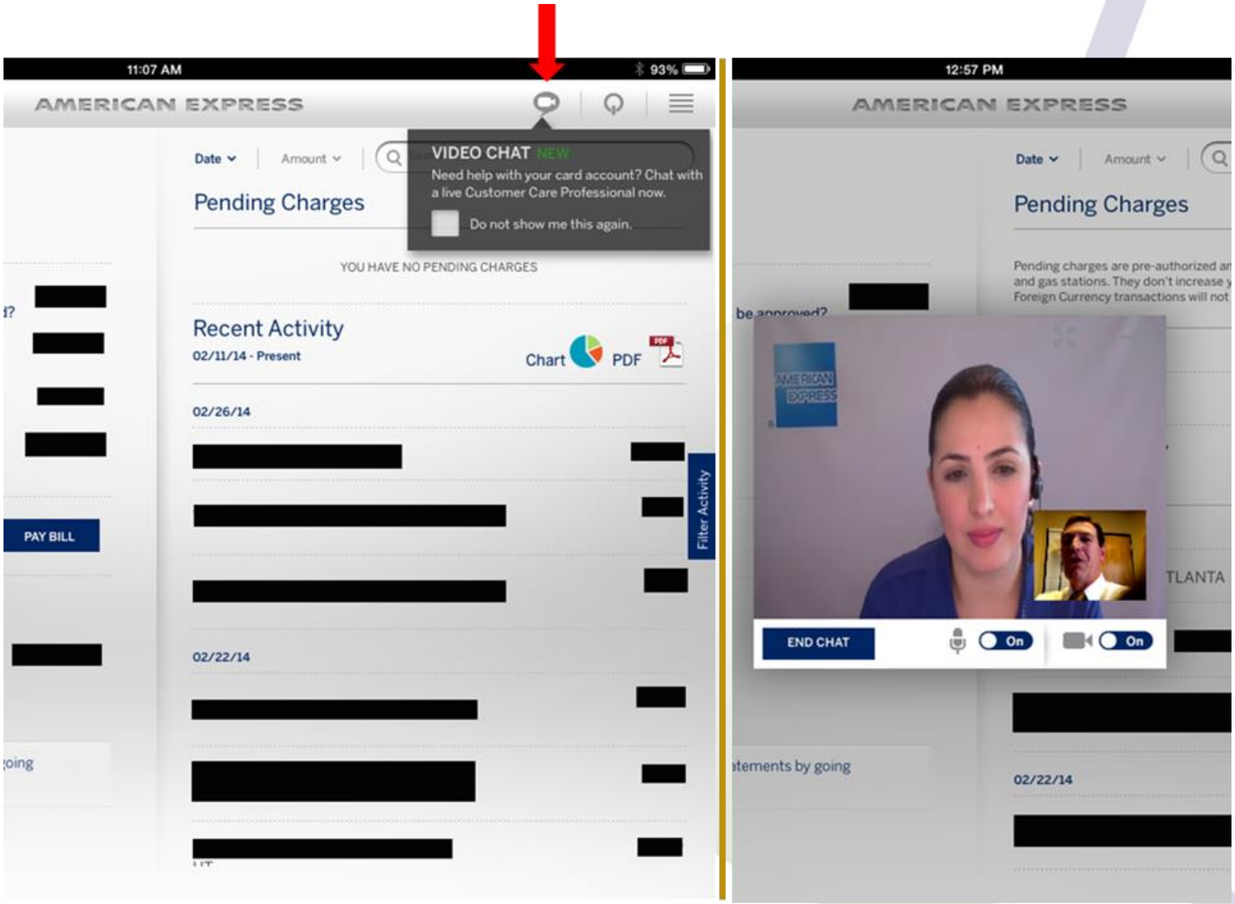
## The American Express Customer Live Assist Example

American Express has also adopted an improved customer engagement strategy for its card holders that involves in-context communication. Any card holder that authenticates on the American Express mobile app is provided with the ability to instantly launch an audio/video conversation with an American Express agent. The functionality is enabled by a small icon at the top of the screen. When a person clicks on this button, a voice and video call is launched to an agent. At the same time, the person's context is transmitted to the agent so that the agent knows which screen on the app a person was viewing when the call was launched. Besides receiving application context, the agent gets a screen pop showing data from the CRM system for that person because this app requires authentication.

[8] The Amazon Mayday Commercial can be found at https://www.youtube.com/watch?v=X40j57v5g6I.

The American Express enhanced engagement strategy is also based on WebRTC technology. On the app side WebRTC is used while on the agent side, American Express' existing contact center agent and telephony infrastructure is used. In between the app and the contact center is a WebRTC-to-SIP gateway that translates the media protocols.

**Figure 4. An example of the American Express mobile app interface when WebRTC-enabled voice and video are used.**




American Express recently released the some interesting statistics about this mobile app[9]:

- Calls have come in from 44 different countries even though American Express is primarily a US-based card, implying that customers use this service from all over the globe.
- Approximately 92 percent of calls connect; 8 percent do not connect; this is because American Express has no control over the user's network environment.
- 1% of American Express card holders used the service.
- 67% of those using the service chose to use the two-way video option.

[9] This data was part of a presentation by Brian Barnes, Vice President, World Service and Global Credit Administration Technology, American Express delivered as part of the Enterprise Connect 2015 trade show held in Orlando, Florida.

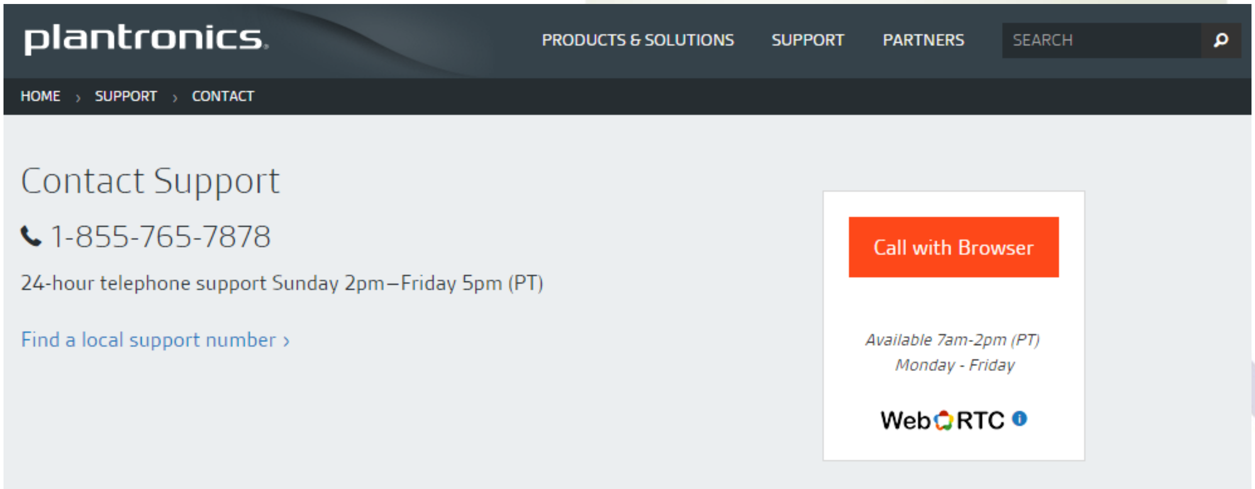
- 88% of these callers had first call resolution.
- Interaction time averaged five minutes.

## The Plantronics Customer Support Example

Plantronics is a global manufacturer of industry-leading audio headsets and wearable technology. The company implemented an enhanced customer engagement strategy by voice-enabling the support portion of its website. On this portion of the site, a button appears that allows the customer to launch a WebRTC call from within the browser that connects into the Plantronics contact center. Plantronics has chosen to use AT&T's Enhanced WebRTC API to implement this functionality. This API allows Plantronics to embed WebRTC communications capabilities in its web pages, and AT&T's back end provides the functionality to convert the WebRTC voice stream to a PSTN call.

As part of this implementation, Plantronics tags each user session with a unique "ID", which happens to be a virtual telephone number obtained from AT&T as part of its enhanced WebRTC API offering. This virtual phone number becomes the caller ID when the call reaches the contact center ACD/IVR. Based on this phone number, Plantronics can identify which browser session launched the call as well as which page the person was on when the call was launched.

**Figure 5. Screen shot of the Plantronics Support Website showing the ability to launch a call from the browser using WebRTC. This was implemented using the AT&T Enhanced WebRTC API framework integrated with Plantronics' existing Avaya contact center.**




Plantronics' ACD/IVR, located in the contact center, has needed no modification whatsoever as this solution was deployed, other than modifying routing rules based on the incoming caller ID phone number. The company also modified its Salesforce.com screen pops to reflect when an incoming call is WebRTC-based, and agents have had a small amount of training so that they can properly respond to calls that come in from the website. Otherwise, WebRTC-based calls appear just like any other call.

Although it is early in the process for Plantronics, the company is pleased with how the solution is working, and it has plans to incorporate additional contextual information and more functionality in the future.

# Introducing the AT&T Enhanced WebRTC API

It has been said that "WebRTC will be to communications what the original Internet was to information"[10]. WebRTC offers a tremendous opportunity for developing new applications and solutions as well as transforming how organizations interact with their customers.

Although WebRTC promoters often state that a WebRTC-enabled communications session can be implemented with just a few lines of JavaScript code, the reality is that to make a useful application, there must be much more than just establishing audio and video connections between two browsers. Real-world applications often need to integrate with SIP-based communications systems or with the PSTN. This requires WebRTC gateways that translate media to SIP or that integrate with the public phone system.

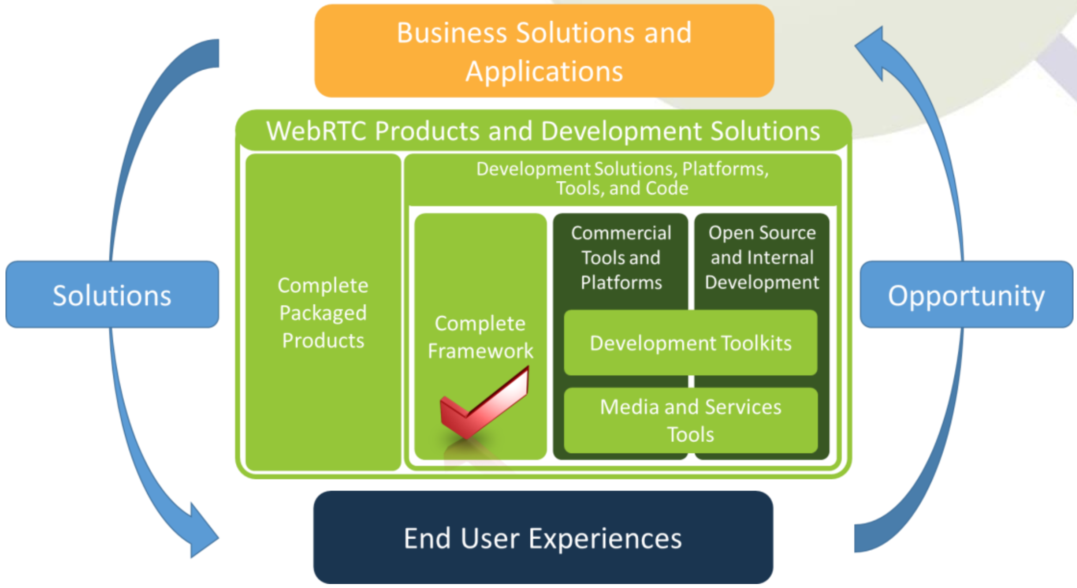
Other issues developers face include network address translation (NAT) device and firewall traversal. Like most audio and video protocols, WebRTC is not NAT and firewall friendly, meaning that many of these security devices will block real-time voice and video traffic. Consequently, most real-world implementations will require some way to securely traverse these NAT and firewall devices.

WebRTC applications also need some sort of directory or rendezvous service so that users can figure out how to call one another or so that organizations can properly route a communications session.

Finally, the WebRTC standard is a work in progress: the standard has not been formally ratified when this paper was printed. Consequently, there are still modifications to the underlying standard occurring.

## WebRTC Development Frameworks

**Figure 6. Between business solutions and end user experiences, one has numerous options for developing WebRTC applications. For developers, the easiest is a complete WebRTC framework, like that provided in that AT&T Enhanced WebRTC API. (Source: © Kelly and Edholm, The WebRTC Ecosystem, 2015.)**




[10] This idea was first articulated by Phil Edholm former CTO at Nortel Networks and president of PKE Consulting at the WebRTC Expo 2013 held in Atlanta, Georgia, USA in 2013.

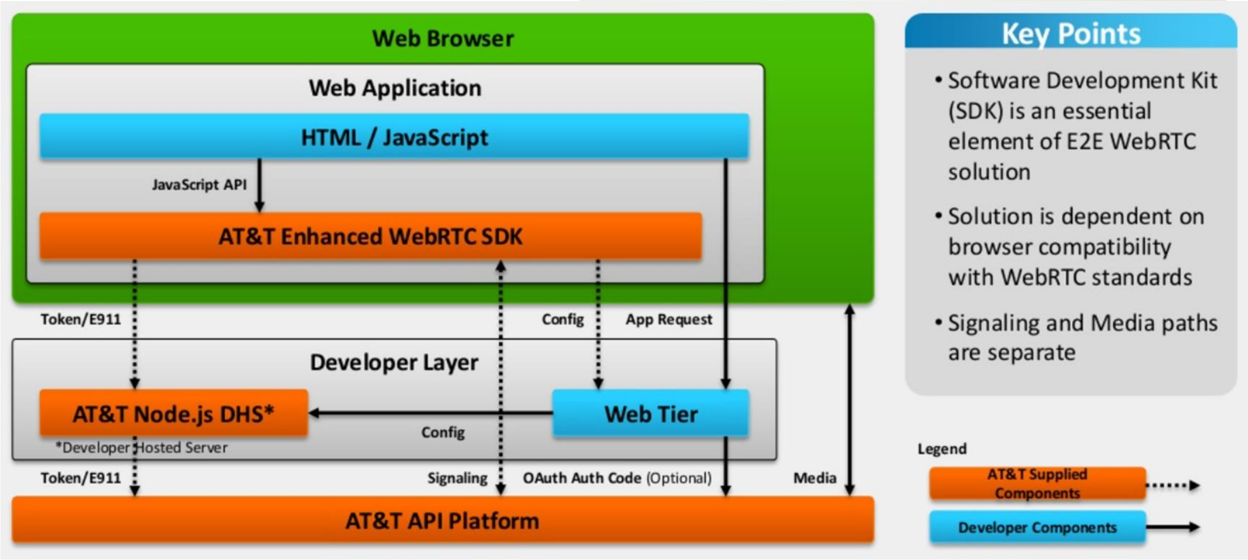
To overcome these real-world developmental obstacles, WebRTC frameworks consisting of API calls and backend infrastructure have been developed. The better frameworks provide a number of capabilities developers need, including gateways for integration with the SIP or PSTN worlds, directory services and advanced call routing capabilities, NAT/firewall traversal capabilities, and a layer of abstraction separating the developer from low level, WebRTC standard function calls, which are still in flux.

Since WebRTC was first introduced in 2011, AT&T has been one of the thought leaders and early innovators of practical WebRTC solutions. One of the early examples AT&T showcased was with mobility partner, Ericsson, in which a browser-based WebRTC call was successfully connected to a mobile phone on the AT&T wireless network. Since then, AT&T has been hard at work broadening its ability to integrate with, enhance, and control WebRTC-based communications. To that end, AT&T has released its own Enhanced WebRTC API, making AT&T the first carrier to offer a WebRTC development framework that eases implementation while shielding developers from the underlying complexity of creating real-world WebRTC-enabled applications.

## The AT&T Enhanced WebRTC API

The AT&T Enhanced WebRTC API has been developed using a standard JavaScript development framework. As such, pre-built JavaScript functions and objects are provided for simple integration into any web application. The AT&T Enhanced WebRTC API provides features such as E911 capability along with telephony gateways that eliminate the need for developers to learn telecommunication protocols and regulations in order to develop telephony-enabled apps.

**Figure 7. An architectural overview of the AT&T Enhanced WebRTC API. (Source: AT&T)**




Apps can be built for the Google Chrome browser; consequently, apps developed using the AT&T Enhanced WebRTC API will work on PCs, Macs, smartphones, and tablets with the Chrome browser installed. When the WebRTC standard is fully adopted and when it is fully supported by other browser manufacturers, these apps will work in any WebRTC-enabled browser[11].

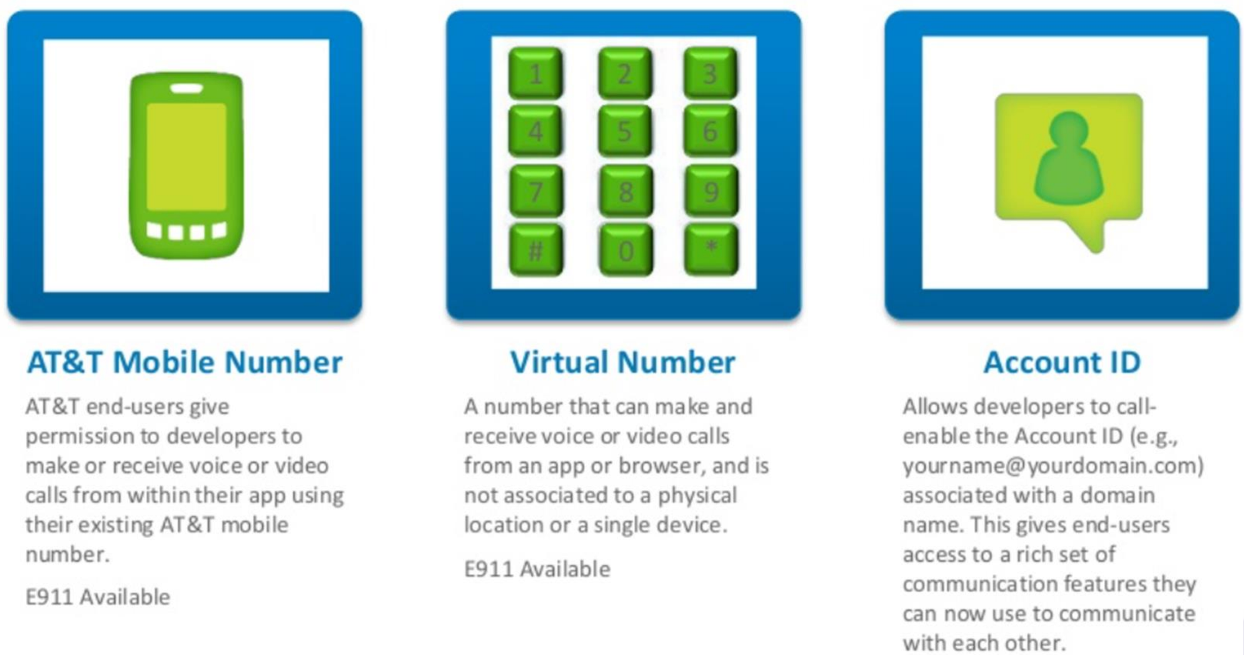
[11] Many of the capabilities of the AT&T WebRTC Enhanced API will work in Firefox as well as Opera; however, AT&T has indicated that it works best with the Chrome browser. This is because there are subtle

## What the AT&T Enhanced WebRTC API Provides

At a high level, the AT&T Enhanced WebRTC provides three key features that shape the user experience:

- AT&T Mobile Number: If the user is an AT&T mobile subscriber, that person's AT&T mobile number can be used as the person's ID. This is useful because outbound WebRTC-based calls show the AT&T mobile number as the caller ID while inbound calls are routed to both the person's browser and to the mobile handset. Another way to use this feature is that a WebRTC call can be transferred from the browser to the AT&T mobile handset. Subscribers also have the ability to call out to any US-based wireless or wireline number, and their AT&T Mobile Number will show up as the Caller ID, just as if they were making a call from their wireless handset.

**Figure 8. Three key functional components of the AT&T Enhanced WebRTC API include making or receiving calls from an AT&T mobile device, the use of virtual phone numbers, and registration of account ideas to create calling directories. (Source: AT&T)**




- Virtual Numbers: A user or a webpage can be assigned a virtual number. Virtual numbers are helpful in two similar but distinct ways:
  1. When a virtual number is assigned to a particular user, this virtual number maps the user to that number effectively creating a directory capability for call routing. Thus, when a user runs the app or points the browser at the designated website, the user can then make calls to the PSTN, and the caller ID will show this virtual number; alternatively, people can call this virtual number from the PSTN, and it is routed to the browser of the user running the app or browser.
  2. If a button on a web page is assigned a virtual number, then that virtual number becomes the caller ID for any calls made by clicking on that button. For example, suppose a customer

differences in how WebRTC has been implemented by the browser manufacturers. Once the WebRTC standard has been ratified, these browser differences will diminish or be eliminated.

support website had a button that said, “Call Us”. If a user clicked on that button, a WebRTC call would be placed from the browser, interconnected to the PSTN, and the call center agent would see that number as the caller ID.

- Account ID: An account ID can act as a directory listing in the AT&T Enhanced WebRTC API framework. When users authenticate, an account ID can be created that represents that user. Users connected and authenticated to the same domain can then be provided with a rich set of calling features. Alternatively, users in different domains registered to the AT&T Enhanced WebRTC API framework can connect as long as at least one of these users has the account ID and domain name of the other.

  A sample usage of such a directory capability might be to create some sort of “buddy list” or “authenticated user list” which people could see. By clicking on that person’s name from the list, an audio and/or video call could be launched.

## But There Is More!

The features above are high level capabilities, but the AT&T Enhanced WebRTC API provides a number of additional capabilities, including

- E911 capability: because the API allows users to dial out to the PSTN, E911 must be part of the offering. For example, suppose a browser user dialed 911. Where should the call be routed? The AT&T Enhanced WebRTC API handles this by requiring all outbound calls to include 911 information such as name, address, etc. There capabilities within the AT&T Enhanced WebRTC API that facilitate collecting and transmitting 911 information to the proper agency.

**Figure 9. The AT&T Enhanced WebRTC API provides numerous capabilities for creating a complete calling solution. (Source: AT&T)**


- Integration with the PSTN for dial in and dial out capability: users can dial out to and receive calls from the PSTN.

- Integration with SIP trunking: inbound and outbound calls may be routed through a SIP trunk depending upon how the organization has set up its calling infrastructure. Where this might be particularly convenient is when a person on a sales or support web page clicks a button to connect to a contact center agent. The internal communications infrastructure for the organization may be a SIP-based PBX. The AT&T Enhanced WebRTC API can integrate directly to this infrastructure via a SIP trunk using AT&T's built-in WebRTC-to-SIP gateway.
- Conferencing rooms: AT&T has provided conferencing capabilities within the WebRTC API. Developers can expose functionality allowing users to create conferences with up to 30 participants (participating calls can be WebRTC-based or PSTN-based). The conference moderator can add others to the conference either via WebRTC or by dialing out via the PSTN.
- Directory services: a key capability of the AT&T Enhanced WebRTC API is directory services that allow people to connect with one another. A person can be represented by virtual number, an account ID, or by that person's AT&T mobile device number.
- Mid-call controls: the AT&T Enhanced WebRTC API allows a variety of mid-call controls including mute, call hold, call transfer, call resume, call transfer, and moving a call between endpoints (i.e. moving a call from the browser to the mobile phone). Thus, developers can create a fully-featured calling application using any or all of these mid-call controls as applicable.
- Call states: Developers can use the API to know a variety of call states so that they can know when calls are in-process, connected, terminated, etc.

## Getting Started with the AT&T Enhanced WebRTC API

For developers interested in building apps using the AT&T Enhanced WebRTC API, there are a few simple steps required. These are outlined below.

1. Create an account at the AT&T Developer Program website.
2. In the "manage my account" area of this site
    a. Enter your organization's domain.
    b. Order any virtual numbers needed.
    c. Add one or more Cross-Origin Resource Sharing domains. These are the domains where your app will be hosted. It could be the same as your organization's domain, or it may be different domains. Entering these domains allows the API calls in your app to be serviced by AT&T.
3. Create an app name and get the app's "credentials". This step involves registering your application's name with AT&T and provides your application with two unique authentication parameters: the App Key and App Secret. The App Key and Secret are used to authenticate the app with AT&T and your account for billing purposes.
4. Install Node.js and then download the AT&T Enhanced WebRTC SDK from AT&T's GitHub repository.
5. Build your app.

### Pricing

Developers can get started with the AT&T Enhanced WebRTC API at no charge. However, once an app is ready for production, the following charges will apply:

- $99 annual access charge plus usage fees.
- Usage fees are as follows: the first one million calls (audio or video) per month are free. After that, there is a fee of $0.01 per call. Note that there is no "per minute" charge for these calls[12].
- Virtual numbers may be rented for $3/number/month.
- High volume customers can arrange custom terms with AT&T.

# *Integrating WebRTC with an Existing Contact Center*

With forethought as to business process, people, and back-end supporting infrastructure requirements, organizations can create strategies that facilitate much more effective real-time engagement with customers and prospects. Using a WebRTC API framework, applications can be created that readily integrate with existing enterprise communications systems and customer contact center investments.

As we discuss some ideas how to do this, we will focus on the following concepts:

1. Planning and testing, including the human aspect of these.
2. Media and context integration.
3. Lessons learned from existing WebRTC-Contact Center Integrations.

## Planning and Testing

When considering integrating WebRTC with the contact center, it is important from the outset to view this as a long-term strategic initiative. While the development may be completed in a relatively short period of time, the impacts of WebRTC are likely to permanently transform some parts of the contact center along with the supporting stakeholder organizations. Consequently, **it is important to make sure all of the stakeholders are involved from the beginning to make sure there are no disconnects between these groups**. For example, in the Plantronics use case discussed earlier, stakeholders from the following organizations were part of the overall team:

- Customer Service
- Telecommunications Group
- Digital Marketing Group
- Enterprise Product Marketing Group
- Engineering (Developers)
- Server Group

It is also important to understand the target audience and to initially be selective on who can call the WebRTC-enabled engagement facility. At issue is the challenge contact center management will face trading off the value of the real-time WebRTC-enabled interaction versus the cost of the agent. Many organizations have been trying to minimize the number of calls that come into a live agent because these agents are expensive. With WebRTC-enabled web pages and apps, you are inviting customers to engage

[12] This is incredible pricing. Most other WebRTC service providers do charge on a per-minute basis. What AT&T is doing here with its pricing will likely shake up the market because this pricing is so compelling.

more often, or at least more easily and with less friction, with the contact center. Consequently, **you will want to make sure the interactions you are enabling, which may utilize more contact center resources, are of sufficient value to offset any increased costs**. Thus, you may need to enable WebRTC engagement on a selective basis, perhaps to high value clients or on a sales site, but maybe not on the general support site. In fairness, we do need to state that some organizations with WebRTC-enabled engagement applications are reporting very high levels of customer satisfaction and first call resolution. So, while there may be more agent interaction time, if customer satisfaction scores, sales, and first call resolution increases, WebRTC-enabled apps may actually become revenue generation activities. It may take some experimenting and time to get the levels of engagement right.

Understanding the limitations of WebRTC is also important. As this paper went to press, the AT&T Enhanced WebRTC API and WebRTC in general is not supported by all browsers, particularly Internet Explorer and Safari. Microsoft's new Windows 10 Edge browser does have support for WebRTC, although it is too new to know what tweaks may be required to make it fully operational. So, a decision must be made, which relates to target audience, as to whether rolling out a WebRTC-enabled website which does not work will all browser type is ok. For some companies, this will be fine while for others it will not be. Consider the American Express example: AMEX decided that since not all browsers support WebRTC, it would not roll out WebRTC-enabled web pages; instead, it did choose to roll out WebRTC-enabled mobile apps because it would have control complete control over that app environment.

Another consideration is the network. **Because you may have no control over the quality of the end user's network, you will need to understand that some WebRTC calls just will not work well.** AMEX notes that about 8 percent of the calls placed through the app do not connect well – not because of any problems on the AMEX side, but due to network issues on the end user side.

Next, be agile! Test and prototype. Plantronics reported that it took just over a week to get its website WebRTC-enabled using the AT&T Enhanced WebRTC API framework. In fairness, Plantronics did have some previous experience with this API when it built a semi-functional app for the AT&T developer conference. Nevertheless, a good JavaScript programmer can do an implementation relatively quickly. **The longest part of the development process is not implementing the WebRTC technology; rather it is making sure the other stakeholders are on-board and ready to perform when the technology goes live**.

With respect to human factors, **it is critical that contact center agents are trained in how a WebRTC-based interaction may differ from a regular telephone interaction**. For example, in the Plantronics case,

## Planning for WebRTC

- ✓ It is important to make sure all of the stakeholders are involved from the beginning to make sure there are no disconnects between these groups.
- ✓ You will want to make sure the interactions you are enabling, which may utilize more contact center resources, are of sufficient value to offset any increased costs.
- ✓ WebRTC is yet not supported by all browsers, particularly Internet Explorer and Safari. Therefore, a decision must be made, which relates to target audience, as to whether rolling out a WebRTC-enabled website which does not work will all browser type is ok.
- ✓ Because you may have no control over the quality of the end user's network, you will need to understand that some WebRTC calls just will not work well.
- ✓ The longest part of the development process is not implementing the WebRTC technology; rather it is making sure the other stakeholders are on-board and all parts of the organization are ready to perform when the technology goes live.
- ✓ It is critical that contact center agents are trained in how a WebRTC-based interaction may differ from a regular telephone interaction
- ✓ If the agent is going to be viewed on video, then the company is going to need to pay attention to how agents look and how they are dressed.
- ✓ Never shirk on testing time.

they modified their Salesforce.com interface so that the screen would pop differently if the call were a WebRTC call. Agents may also need to have somewhat different scripts as WebRTC calls may contain more or less information than a standard PSTN call[13]. Furthermore, it may be important to help agents understand what is going on when a user makes a WebRTC call and how it differs from a PSTN call.

**If the agent is going to be viewed on video, then the company is going to need to pay attention to how agents look and how they are dressed**. In addition, cubicle lighting and camera placement also become critical issues so as to avoid some of the negative artifacts poorly lighted video sessions create (i.e. looking up someone's nostrils or the "Frankenstein" views shadows can cause).

**Never shirk testing time**. AT&T's main contact center consultant advises customers to make sure that they plan time for thorough testing as they deploy their WebRTC-enabled applications. Plantronics noted that testing "from all angles" is a key to their success. Both AMEX and Plantronics started slowly, with limited availability, as the live system was being tested. They intentionally did this to prove system stability and to make sure agents were ready.

## Integrating WebRTC Voice and Video

One of the important decisions the development team and stakeholders will make is how they want the WebRTC-enabled application's voice and/or video to integrate with the contact center. This can be done in either of two ways, with implications for each method:

1. Integration via the PSTN or a SIP trunk, and
2. Direct WebRTC-to-WebRTC integration.

### Integration via the PSTN or a SIP Trunk

The AT&T Enhanced WebRTC API provides both a PSTN gateway as well as a SIP gateway. The user on the WebRTC side uses the WebRTC audio and video protocols while the agent in the contact center uses the same phones or video terminals they are accustomed to using. Behind the scenes, the AT&T Enhanced WebRTC infrastructure does the necessary conversions between them.

If the decision is made to convert WebRTC to the PSTN or SIP, then it is likely that nothing in the contact center infrastructure will need modification. A voice or video call will be tagged either using the AT&T virtual number, or it can be tagged using some other mechanism. The important thing is that when the call reaches the contact center, the call has some type of caller ID associated with it so that the call can be associated with the user's contextual data (see the next section).

In this scenario, all of the existing contact center tools such as recording, voice analytics, etc., can work on the audio stream.

This is the way both American Express and Plantronics have integrated WebRTC in their apps and contact centers. In the American Express case, the WebRTC-enabled application uses the G.711 audio

[13] This point may require additional explanation. In its initial implementation, Plantronics agents only knew that a WebRTC call was from the support website, but no other information was passed to the agent. However, in the typical 'call back' scenario used previously, a text box would appear in which the user could enter text regarding their support issue, which would accompany the call back request. Thus, unless user context is passed with the WebRTC call, agents may have less information available to them. We will discuss passing context to the agent later in this section.

protocol and the VP8 video protocol on the mobile device side. The gateway between the AMEX mobile app and the contact center decrypts the WebRTC stream, and first transcodes the VP8 video to the H.264 video protocol, it then puts the G.711 audio and the H.264 video into a SIP stream which goes to the normal call manager, and from there into the standard ACD/IVR. Agents answer calls either on their regular phone, or if they are video-enabled, on a SIP standards-based video personal video unit.

In the Plantronics case, the WebRTC audio passes through a gateway to the PSTN. The PSTN routes the call to the Plantronics contact center, where the caller ID is one of the virtual numbers Plantronics has procured from AT&T. The PSTN call hits the Plantronics ACD/IVR and is routed according to routing rules associated with a WebRTC call. The system knows it is a WebRTC call because the caller ID identifies it as such.

### Direct WebRTC-to-WebRTC Integration

WebRTC calls are not required to pass through a gateway if the contact center agent can respond to the call using a WebRTC-enabled device or browser. In this scenario, when the end user clicks on the WebRTC button to reach support, the web server links that call to an agent who is logged in to his/her browser and ready to take the call. In this instance, a direct WebRTC-to-WebRTC call is established between the user application and the agent in the contact center.

This type of integration, while conceptually simple, may require some additional developmental considerations. For example, the application developer will need to create the contact center agent queuing logic for these WebRTC calls. Once an available agent is identified, then the standard function calls in the AT&T Enhanced WebRTC API can connect the user with the available agent.

Direct WebRTC-to-WebRTC has both upsides and downsides. Organizations deploying direct WebRTC-to-WebRTC calling will need to understand that some standard contact center functionality, such as call recording or speech analytics, may not be possible because WebRTC calls are encrypted. Alternatively, the contact center WebRTC client could decrypt the audio stream, clone it, and send the audio to a recording device or speech analytics engine. This may require some additional development for the contact center, however.

### A Variation on WebRTC Integration

In the scenarios above, it has always been the end user who has been using WebRTC in the browser or app. Another scenarios that is becoming more common is for the contact center agent to be using a WebRTC-enabled contact center client in an environment like a Google Chromebook. In this scenario, calls from the PSTN come into the contact center, and a gateway converts them to WebRTC and routes them to agents who have these WebRTC-enabled contact center clients. Several contact centers have clients supporting this type of “reverse” WebRTC integration with the contact center.

## Integrating the User Context

One of the key opportunities a WebRTC-enabled engagement application provides is the ability to send contextual information about the user to the contact center agent. For example, it is possible to know which screen on a web page or mobile app the user was looking at when she/he pressed the button launching the WebRTC call. More sophisticated solutions can also send the agent a customer’s browsing history for a particular session. If the website places cookies in the user’s browser or the user has authenticated in some fashion, the website logic may even know the user’s identity, so that when the agent comes on the line, the agent can greet the person by name. Context can help agents serve users better and faster because they may know the user’s intent at the start of the call. For example, if a user is on an ecommerce web page, the agent may be able to tell exactly which item the user was looking at when the call was initiated.

### Contextual Integration with Existing Contact Centers

Major contact center manufacturers have the ability to associate user context with a WebRTC call. Each WebRTC call will need to be tagged with some type of a unique identifier so that the call can be linked with contextual data. This tag can be one of the virtual phone numbers AT&T provides, or developers and contact center managers can create their own tagging scheme. Once the call comes into the contact center, this unique tag can be used to pop up the context information to the agent.

Integration with Cisco's contact center is through the Cisco Context Service, a cloud-based service that provides storage, tagging, and management of the data from interactions between organizations and their customers. Customer data is stored within a POD (Piece of Data) in the Cisco Context Service. PODs are objects used to exchange data between different systems. To put customer contextual data in a POD, Cisco provides a Remote Expert Mobile SDK. Once in a POD, Cisco's contact center can tie the call to the contextual data.

Avaya allows developers to put contextual data into an Avaya contact center system using the Avaya Engagement Development Platform (EDP). Once in the EDP, the information is passed through a "conversion" service that matches the call ID or tag with the contextual data so that these data can be pulled out and displayed to an agent. Alternatively, rather than putting data into the EDP, a developer can pass a UUI (user to user information) link through a SIP header to the Avaya contact center, and the contact center application can pull contextual data directly from the website.

Although few details have been provided, Interactive Intelligence offers an API in which user contextual data can be passed to the Interactive Intelligence Interaction Center.

Organizations who WebRTC-enable their websites and mobile apps, should be heartened knowing that with a little additional development, they can place highly valuable user contextual information into their existing contact center application, thereby creating highly robust, useful, and potentially profitable customer engagement solutions.

## *Conclusions and Recommendations*

WebRTC is becoming a pervasive technology. Any device with a Chrome, Firefox, Opera, and now Windows 10 browser are already WebRTC-enabled. Many mobile applications developers are also using the WebRTC technology to voice- and video-enable these apps.

It is important to note that while the WebRTC standard is still being tweaked and that not everyone agrees on the video codecs WebRTC should use, the audio portion of the WebRTC standard is solid. Thus, organizations can proceed creating WebRTC-enabled applications with confidence, particularly if they initially focus on voice in the near term. In environments where the organization has some control over what type of browser the end user has, video applications can also work very well.

Some very compelling customer engagement applications have been created using WebRTC as a fundamental enabler. Three of these were discussed in this document, but many more exist.

When developing a rich and fully-featured WebRTC-enabled application, it is going to be much easier to develop the application using a WebRTC framework, like the AT&T WebRTC API, as opposed to trying to use raw WebRTC calls and building all of the connecting functionality around it. Furthermore, this type of framework keeps developer code abstracted from any changes to the emerging WebRTC standard.

The AT&T WebRTC API has a compelling price point as opposed to competing frameworks, and it is now generally available for developers to try out. For more details go to https://developer.att.com/enhanced-webrtc.

## *Disclosure on Editorial Control*

KelCor has been compensated by AT&T to write this case study. However, KelCor has maintained full editorial control throughout.

## *About the Author*

Brent Kelly is Principal Analyst at KelCor, Inc. where he focuses on the intersection of business technologies comprising unified communications, social networking, video, cloud services and mobility. Dr. Kelly provides strategy and counsel to key client types including Chief Information Officers, Chief Technology Officers, investment analysts, VCs, technology policy executives, sell side firms and technology buyers. Dr. Kelly has been an independent consultant and analyst since 2001, writing hundreds of articles and presenting at numerous public and private events. He has served as the Vice President of Marketing for Sorenson, an early innovator in the IP communications space, and as the CEO of a privately held manufacturing company. Prior to this, Dr. Kelly was part of the team at Schlumberger that built the devices Intel used to test Pentium microprocessors. He also led teams developing real-time data acquisition and control systems, and adaptive intelligent design systems in several Schlumberger Oil Field services companies including four years of R&D experience in Europe. He has a Ph.D. in engineering from Texas A&M University and a B.S. in engineering from Brigham Young University. Dr. Kelly is serving in his second term as an elected official in his community.

## *About KelCor*

KelCor (www.kelcor.com) is a specialized consulting and analyst firm with a passion for providing client satisfaction through product and service excellence. We have laser focus on the business communications market, emphasizing those products and services that are proven to accelerate an organization's business processes.

We provide value to our end-user and vendor clients by offering an unbiased, 360° view of the unified communications and collaboration marketplace. We prepare research reports, vendor profiles, market forecasts, white papers, case studies, and presentations designed to inform, educate, and assist vendors with strategy, tactics, market approaches, and end user attitudes to help them identify and capitalize on opportunity. We help our end-user clients understand options, strategies, competitive vendor offerings, and best practices engaging our collaborative process engineering expertise, all designed to improve organization efficiency while increasing top line revenues or bottom line profits.

The depth of our reports and our ability to discern key market trends significantly differentiates us from any other consulting and analyst firm you've ever worked with.