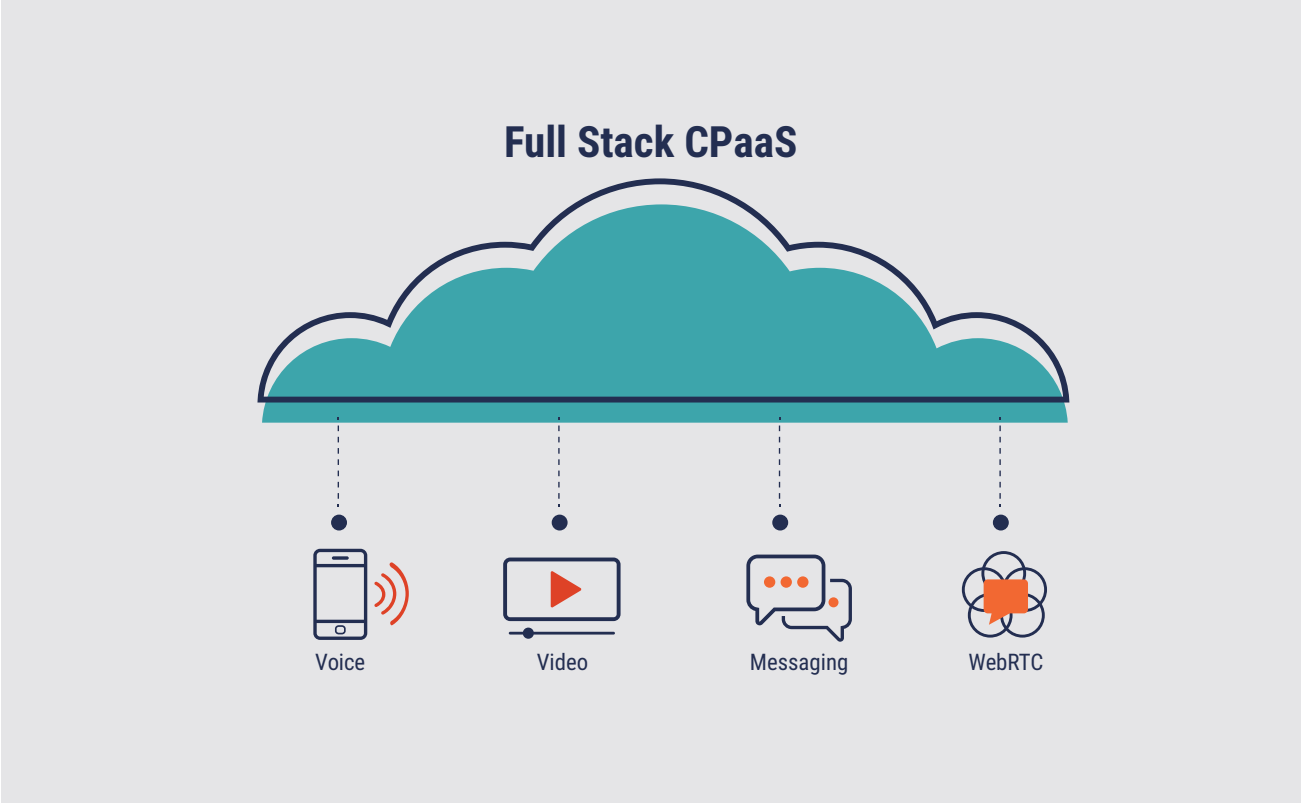
What does the Kindle Fire Stick have in common with CPaaS? They both offer a form of federation that takes the friction out of multiple processes. The Kindle Fire Stick federates subscriptions to Prime, Netflix and Hulu. Before the Fire Stick (or Apple TV, Roku or Google Chromecast), each service was an island that had to be accessed and managed individually. Now with these products, you enter an integrate environment where you can navigate from one streaming service to the next with ease. A CPaaS platform is designed to provide programmable APIs for building voice, video and messaging cloud-based real-time communications applications. Developers enter the CPaaS development environment and gain access to programmable APIs for voice, video and messaging.

You ask what a Kindle Fire TV Stick has to do with Full Stack CPaaS. The two technologies are quite different but the effect when integration goes bad is the same. Actually it is not quite the same. When the Kindle Fire Stick has an issue, the user simply reboots the system. When CPaaS integration goes bad, it is far worse.

Full Stack CPaaS Defined

What constitutes a Full Stack CPaaS? A CPaaS is a cloud communications platform that provides APIs to build real-time voice, video and messaging applications. A few CPaaS products also provide WebRTC mobile and web SDKs as well. Anything less than this does not qualify for Full Stack CPaaS. I will explain the issues next. I will also explain how CSPs and Enterprises unwittingly box themselves into what I call the duct tape and bailing wire or Frankenstein approach to CPaaS. Further, I offer a whitepaper called 8 Questions You Need to Ask Before Committing To A CPaaS Provider. While full stack is covered in this whitepaper (in less detail), there are more areas you need to be aware of. I encourage you to access this whitepaper and give it a read.



Omnichannel Cloud Communications Platforms are Vital

Why is it so important for a CPaaS to offer voice, video and messaging? Each plays an important role in real-time communications. Although it is quite possible that your needs might fall initially in one of the three categories, once you see the benefits real-time communications brings, you will quickly come up with use cases that require the other two. Example use cases:

Customer call center

- Messaging to enable customers to text instead of call for service or support
- Voice to connect customer to a call center representative after receiving context via text
- Video to view the actual problem the customer is experiencing

Banking

- Messaging to the 800 number request service
- Voice to offer customers after receiving context via text
- Video chat to have a face-to-face discussion directly from a banking website

The Duct Tape and Bailing Wire CPaaS

In the above two use cases, the first communication API the developer would use is most likely messaging (text). It would not be unreasonable for the developer to focus on messaging only platforms at this point. However, the minute the developer wishes to extend the application to offer an additional mode of real-time communications, he finds himself needing to look for a new platform that supports the desired new functionality.

It is possible that the organization could wind up supporting three disparate communications platforms that were never designed to work together. Now the integration between the three communications platforms becomes the responsibility of the organization. To make matters worse, if a new version of one platform is introduced, the integration work starts all over again.

The Frankenstein CPaaS

Food for thought – not all CPaaS providers design and develop the platform they sell. I am seriously not kidding. To get to market quickly some vendors negotiate contracts with technology suppliers to resell disparate voice, video and messaging components. Again this is less than ideal because these disparate technology components were never designed to work together. The CPaaS provider claims they offer a full Omnichannel communications platform when it is really not much more than the Duct Tape and Bailing Wire CPaaS. The provider takes on the role of integrating and managing the components but it never seems to go as planned. These platforms tend to perform unacceptably, experience unexpected glitches and, in some cases, fall down unexpectedly. And do not get me started about the finger pointing when things do not go as planned. The reality is the same crazy integration and management has to take place, it has just been moved from the buying organization to the CPaaS provider.

The same is true for CPaaS providers who have developed one or two real-time communication component in house and purchased others. How many times have you heard the story of solution vendors who purchase really awesome technology but never get the integration with their existing product working right? More than I can personally count. Never underestimate the complexities associated with integration. Like Frankenstein himself, this implementation promises to be a nightmare.

The Brass Ring CPaaS

This is the true definition of full stack CPaaS. It is designed from the ground up by a single highly qualified development organization. It sits on top of a carrier-grade telecommunications platform that handles all of the network complexities. Each Omnichannel component is intimately familiar with the other. It is assumed and trusted that these voice, video, messaging components will work together seamlessly. It offers a full set of web and mobile SDKs. And it provides sample templates and a visual designer to allow subject matter experts, in addition to developers, to design applications. It is RestcommONE

RestcommONE is a full stack CPaaS, which means an organization can begin with one communications method like messaging and easily add voice and/or video later. With RestcommONE, the organization is billed based solely on what it consumes (the traffic the platform moves). So if an organization is using RestcommONE for a messaging application, they are billed on the number of SMS texts that are sent through the platform – that is it. This allows developers to consume and pay for only what is needed today while having the trust and confidence they will not have to select a second or third platform vendor as their needs expand. In other words – voice, video and messaging integrated within a single platform – all there at the ready.

RestcommONE is built on top of Open Source Restcomm. Some interesting facts about Open Source Restcomm you may not know:

As reported by Black Duck Open Hub

Current number of lines of code	Estimated man years of effort
5,346,799	1,587
Number of commits	**Cost to reproduce (COCOMO method)**
35,094 by 320 contributors	$87,266,838

With statistics like these, it is easy to understand why the other CPaaS providers make the decision to purchase and/or partner. And remember with the RestcommONE platform, you pay for what you use. Need only messaging today – no problem. Want to add voice or video – no problem. Omnichannel real-time communications are at the ready – when you need them. But you pay only for what you use today.

Click here to view 8 Things You Need To Ask Before Committing To A CPaaS Provider whitepaper.

If you are interested please contact us. We'll be expecting you.

Contact Us