HMP-X1

Whitepaper

EMPOWERING TELECOM INNOVATORS

In a matter of a few short years, telecom applications technology has transformed drastically and quickly. Traditional host based APIs are no longer sufficient. We are suddenly at the dawn of a new era where telecom has merged with the internet and social media.

What is needed today is a far more sophisticated platform and a modernized API to fully engage the full potential of the new telecom world.

Since our daily lives depend so heavily on instant connectivity and peer-to-peer interaction with others, the platform hosting telecom applications must be always available and able to provide reliable connectivity with acceptable voice quality - regardless of location and media channel used to communicate. This is the current reality where expectations are very high, where everything needs to work reliably and is expected to be very user friendly.

These current expectations require new architecture and new tools. Unlike limits imposed by traditional host based APIs, network based APIs allow better scalability, more redundancy and un-restricted architecture. From a developer's point of view, time to market has become even more critical while access to highly skilled programming talent has become more limited. Modern day developers expect to have a choice when it comes to programming languages and application platforms.

To effectively deal with this challenge, we have developed and released modern tools that take full advantage of our highly reliable and field proven core HMP telecom technology platform and we call it HMP-X1. With the release of X1, Pika has extended and modernized the existing set of host-based APIs with gRPC/Protobuf (Protocol Buffers) based network API - directly mapped to it's existing HMP APIs.

Through this modern network API, HMP-X1 allows for the expansion of the existing telecom concepts with the addition of complete redundancy while also encouraging software development for more sophisticated distributed network architectures. The use of language and platform agnostic Protbufs and gRPC will allow the freedom to create applications in most modern programming languages (those supported by protocol buffers). In practical terms this means that an application written in Java, Ruby, C#, Go, Python or even PHP can reside on any platform (including mobile devices) while still communicating with and controlling Pika's HMP-X1 engine running in a customer's data center or on a public cloud.

Consider the simple example of a PBX that has been developed using Pika's HMP-X1 API. In such a case, the media processing server(s) can reside in a high availability data center and the logic for such an application (written in C# using Microsoft Visual Studio) would run locally. IVR messages, Voicemail recordings, dial plans, workflows and user data can all be stored local to the application that makes gRPC calls to a media processor for the heavy lifting.

HMP-X1

Whitepaper

EMPOWERING TELECOM INNOVATORS

HMP-X1 will enhance two key factors: versatility and reliability for both, users as well as developers. This can add tremendous value to the product, allow quicker time to market while lowering the overall cost of ownership.

Furthermore, HMP-X1 is the foundation of our more futuristic HMP-X2 (coming soon) which will offer REST/gRPC high level APIs with Multi-user and Multi-tenant features.

Questions and Answers:

What are protocol buffers?

Protocol buffers are a flexible, efficient, automated mechanism for serializing structured data – think XML, but smaller, faster, and simpler. You define how you want your data to be structured once, then you can use special generated source code to easily write and read your structured data to and from a variety of data streams and using a variety of languages. You can even update your data structure without breaking deployed programs that are compiled against the "old" format. (1)

What is gRPC?

gRPC is a modern, open source remote procedure call (RPC) framework that can run anywhere. It enables client and server applications to communicate transparently, and makes it easier to build connected systems (2)

1) Google Protocol Buffers Documentation

2) Google gRPC Documentation