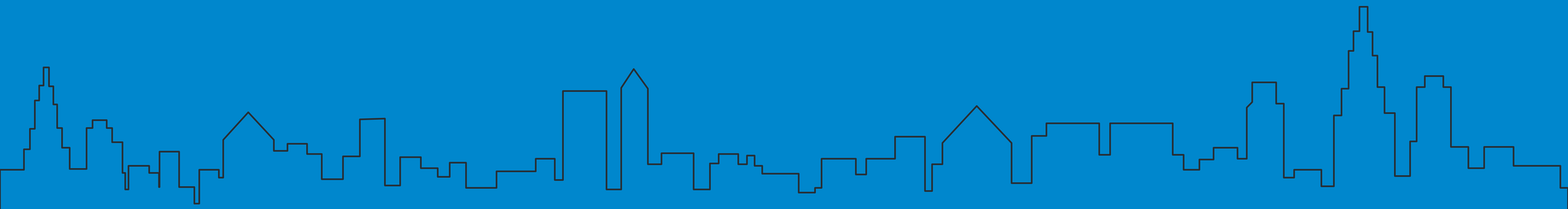
PolygonCast

Data Streaming Workflow

Overview

11-Nov-2015



Our take on (video) streaming

… and how to fix it

Data Streaming

Content-agnostic, modular and customizable method of delivery and playback of multiple types of data on a single interface using multiple, interchangeable transport technologies working simultaneously.

Issues

The video industry is suffering from a single major blocker: a proliferation of standards and a lack of interoperability between them. E.g. preparing and carrying out a live streaming event available on mobiles and desktops means organizing and maintaining multiple video and audio tracks, most of which are designed as separate workflows with their own manifests, CDN requirements, supported DRMs and so on. This requires an army of people and huge resources - and still it's impossible to dynamically adjust to changes.

Features

- Modular design - mix and match your own components.
- Compatible interfaces - even legacy protocols, structures.
- Dynamic workflow - everything works together.
- Content agnostic - can stream ANY content, e.g. newsfeed.

Solution

The PolygonCast Streaming Platform is a workflow focused on data streaming, taking it's inspiration from the way databases handled data synchronization. It's based on the concept of a „bucket", which is data projected on a temporal dimension and later requested based on time by user interfaces. The data can be ANYTHING! The transport layer is fully separated, thus allowing for simultaneous independent content delivery. It's fully modular, so anyone can modify and override components with their own.

Benefits

The PolygonCast Streaming Platform can be deployed on any phase of content delivery, easily combined with your existing stack, used to deliver any type of content to a single interface using multiple transport technologies while being open to modification and easy to adjust according to your needs.

Keep It Simple, Stupid!

Beyond Video

Streaming Workflow Concept

How the Data Streaming concept came to be

0 Concept

We wanted to create a different video (media) workflow starting from scratch. Since we have experienced the industry's issues firsthand, we tried coming up with a solution that would eliminate them altogether.

We set three conditions.

1 Modular Design

Everything should be modular.

If we do some stuff right and some wrong, we could trash the bad parts and keep the good ones. Our protocols, data structures, ideas and code should work just the same even if it were the only part to survive. We just tried solving problems in a more generic way.

2 Compatible Interfaces

It should be compatible with all (meaning the good ones obviously) existing protocols, data structures and tools. So that our toys would work with their toys. The goal is to have all the options without sacrificing anything.

3 Not Just Video

We tried figuring out how the future of online media is going to look like and work. VR is a great example. You can't just take MPEG-DASH, add VR to it and make it work. We think that truly awesome applications require something more.

So we shifted our focus from video to data streaming. Data that just happens to be video from time to time. It worked like a charm.

So this was step 1…
Step 2: demo or it didn't happen!

Streaming Workflow Execution

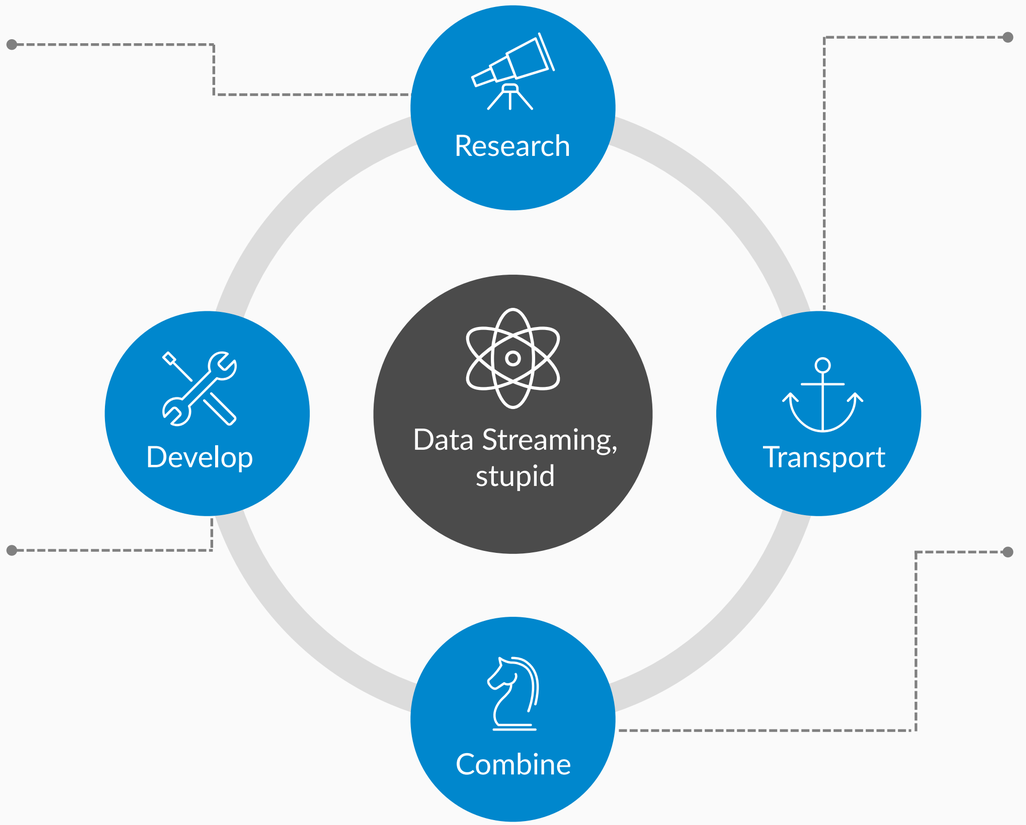
One bucket to rule them all

Common format

We coined the concept of the data bucket. Two video qualities and one audio means 3 buckets. It's data projected on a temporal dimension. And applications (like video players) just want to get it on time and query. Databases have figured that functionality out a long time ago.

Shared tooling

We shouldn't have to solve the same problems again and again. We provide common solutions and allow anyone to override them with better ones.



Expand transport technologies

Separate transport from all the other components so that every application could go with default HTTP but also easily be able to go beyond that. Who knows, maybe stateful UDP streaming servers will be the next bing thing in a year?

Broader scope

Make all the components support even the craziest scenarios. Combining a 3D 360 concert video with 3 periscope-like streams? 5 buckets. 60° viewports for 3D? 15 buckets.

Data Playback

Data Playback

Going a step beyond simple data visualization

Due to separating data synchronization from playback there is no problem with delivering multiple streams to a single player.

The thing being played is data. This means that we can go beyond video and easily display e.g. a live event coupled with tweets regarding said event and even an additional periscope video. All interactive and in a single player.

Modularity comes into play again, meaning that the Data Player can be fully customized not only on a e.g. per event basis, but on a per user basis. Want to give your viewers full control? Now you can.

Because we handle multiple stream of data, the Player can truly adapt per platform. You can combine 3D video stream and normal stream. People with VR headsets willl download an additional 3D content.

Projectors, buckets, fillers and managers

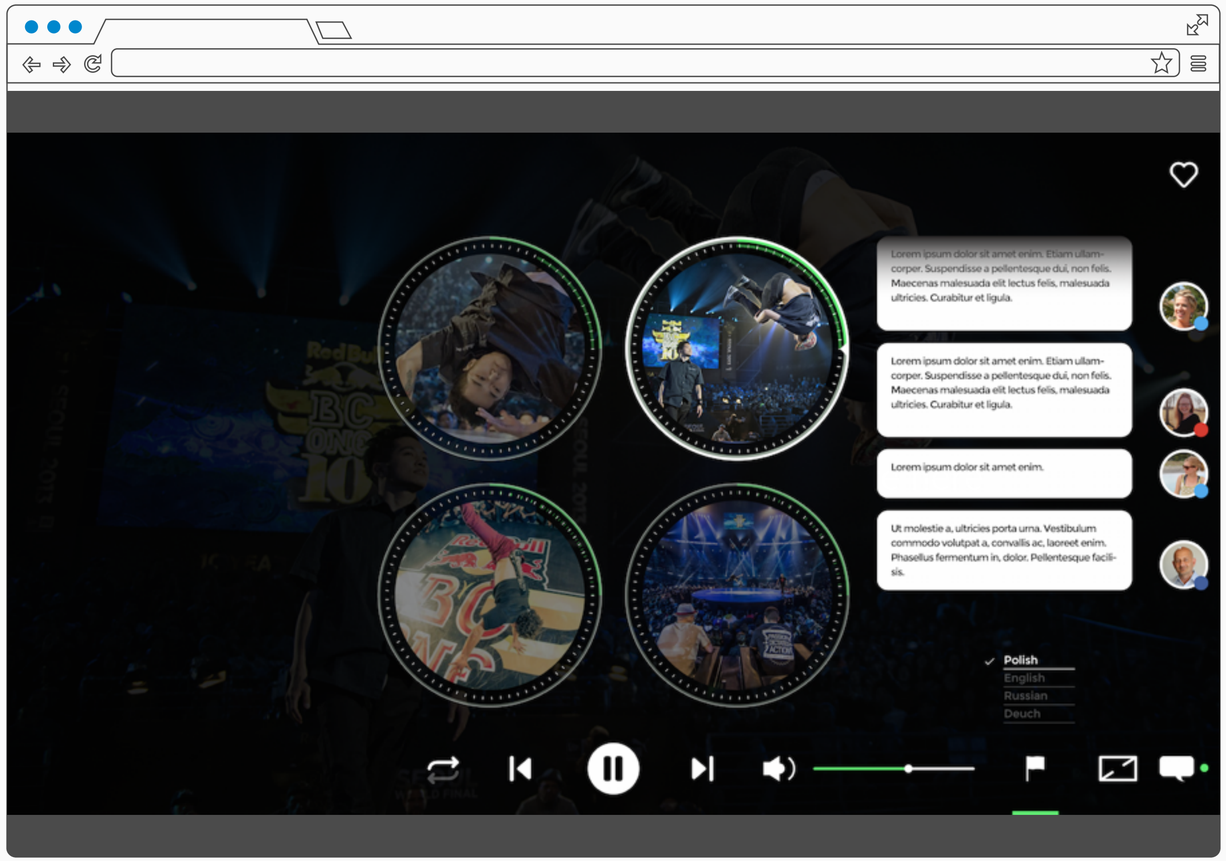
The data player is an interface through which users can interact with the data being provided. Is consists of several sub-entities called Projectors, which can be dynamically created to display a given bucket of data / data type. This means that in one data player window several of these Projectors can display content hand-in-hand, not being limited by it’s type (e.g. text on video or video in video).

The manifest like information is provided by a process called a Filler. It synchronizes the fronted with the backend in a fashion similar to databases.

When multiple qualities are detected, the Resource Manager requests said data to be provided in the most adequate quality possible. The request is handled by the Download Manager, which handles selecting the most efficient transport technology available (or a mix of said technologies) and the best edge cache server. And projects are informed that the type of data they’ve subscribed to is ready for consumption

Interacting with the Data Player

Multiple data streams seamlessly displayed on a single interface



Synchronized stream of information

The video you see on the left is an effect of playing the main stream (top right) and switching into stream selection mode. The player works with 1HQ primary, 4 LQ video streams, multiple audio streams and social data.

Video

One bucket per video quality and one bucket per audio track, ad infinitum.

Twitter (or any other data feed)

One bucket that streams a live snapshot from twitter's API of the tweets coming in.

This is just the beginning. Feel like providing a live transcript for the hearing impaired? Add a bucket. Want to add a livestream from a drone outside the window in 3D WITHIN the video? Add a bucket.

By approaching playback from a modular standpoint and focusing on data instead of only video, we allow you to focus on the experience you want to provide, simultaneously displaying unlimited types and quantities of content.

Transport Technologies

Multiple Transport Technologies
allow you to deliver data via any delivery technology. Be it HTTP, QUIC, WebRTC, WebSockets, P2P or anything else.

They all work together.

Psst… We are streaming video over UDP in HTLM5 right now

Introducing Multiple Transport Technologies

Streamlined data delivery without a hassle

We allow you to use multiple transport technologies at the same time. Mix multiple HTTP CDNs, WebRTC, P2P, QUIC or anything else.

Our reference client adaptively selects data streams based on their quality. If HTTP is slow, why not try another server? If WebRTC is fast why not switch all streams to it?

Modular design and separation of concerns allows you to experiment with technologies without interfering with other components. Adaptive Bitrate doesn't care if the video was downloaded over HTTP or WebRTC.

The idea of having a more universal **data manifest** and not just a video manifest allows for a broader and more creative technology application. We are just moving bits. Why label them with video or audio?

Transport technologies

When it comes to video streaming MPEG-DASH has introduced native support for multiple CDNs at the protocol level through Base URLs. This is a great idea! It allows broadcasters to diversify risks and provide additional (backup) CDNs that would help keep the perfect user experience no matter what. This is a much more scalable solution then serving your clients different HLS playlists to basically achieve the same results.

We take this idea a step further. We expand video streaming to data streaming. And with that we provide support for multiple delivery technologies at a per stream or per segment definition level. This means that the same clients can download audio via HTTP servers from different CDNs and download video streams via a mix of WebRTC (Server to Client UDP), P2P (client to client UDP) and HTTP or QUIC.

Since most of the data being transferred online right now is video, we think of media streaming as our top priority.

How we use Multiple Transport Technologies

This is an example of what has been tested and proven to work

We have placed both audio and video on CDN1

We have placed only video on CDN2

We have placed only the highest quality video on a WebRTC server that connects with clients over an unreliable UDP. And our application makes it reliable. If we lose e.g. 2% of the data, we can just request those 2% again or get them from a any CDN.

We have tested QUIC for a video delivery. Unfortunately, Chrome can make only one request and then it needs an application restart. Truly experimental technology.

And it just worked. Since the idea is the same (get that data), it is pretty easy to replicate HTTP-like Request Response logic via any delivery technology. And this is what makes them all work together.

PS. In our tests UDP works exceptionally well with LTE networks.

Tools & Tech

Our Stack

Our stack, based on micro services enables us to deliver some unique features

Docker

All our micro services run within Docker containers. This gives us a possibility to easily parallelize and distribute workers

Node.js

Even driven environment for rapid feature implementation

MongoDB

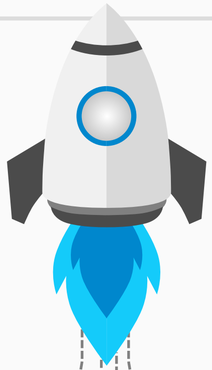
A document oriented, scalable and schemaless database is ideal for storing practically unlimited meta information about media content

NATS

High performance message broker for our distributed system. Used for heavy task parallelization

Our approach to Scaling

In our case, scaling translates into performant video workflows



1

Parallel

Video processing is a CPU consuming process. The only way to speed it up is to process individual fragments in parallel

2

Multiple Workflows

Different video workflows require different setup. We spawn workers to handle customized workflows

3

Containerizing

Containerizing technologies gives as an option to cherry pick the most awesome ones and utilize their full potential

4

Operational Mindset

Technologies we've chosen force us to think about every microservice in an operational way, ready to join the production fleet anytime

We've aimed to support both complicated and simple workflows. With every feature being optional. This leads to a scalable microservice - worker infrastructure.

Every workflow is simple, scalable and unique.

PolygonCast fits right in!

It can be integrated into your streaming setup during any phase - from raw image processing to media delivery and playback

1

Full PolygonCast deployment

What: You supply the raw video, we supply the software.

Approach: We deploy PolygonCast in its entirety: we take care of the muxing and encoding, deliver the content to your viewers and play it. Our video platform will take your video, send it to an encoding servers (e.g. Elemental), add ABR, different audio and/or video tracks, partition the deliverables into chunks, distribute them to viewers using various available transport technologies (WebRTC, UDP, P2P, QUIC or MultiCDN), reassemble and play it back to the user using our player. All that's required from you is providing the video content.

Result: You get a full-scale video platform which can easily handle any of the currently market-available functionalities and is fully future proof, capable of streaming 360 or 3D content. You can also get cutting edge P2P and real-time video analytics.

Media Delivery System deployment

What: You have video content and want to stream it.

Approach: You already have a backend infrastructure which handles video muxing and encoding? Recently implemented MPEG-DASH and don't want to revamp everything again? No problem! We can deploy PolygonCast as a media delivery system and you will still get all the really important functionalities.

With this deployment method, you simply feed our platform your encoded files (e.g. DASH mp4). We will handle the chunking/DRM/ABR aspects via legacy compatibility and deliver your content using MultiCDN, WebRTC, UDP, QUIC, P2P and provide playback with our player.

Result: You get a cutting-edge content delivery system which support EVERYTHING that's on the market right now, coupled with a truly future-proof HTML5 player which can handle everything that's coming in the future (e.g. QUIC).

2

3

Player only deployment

What: You have an infrastructure, want a future-proof HTML5 Player.

Approach: We assume you have a fully deployed CDN solutions and only implement HTML5 playback functionality with possible future support and easy integration with new and emerging standards (e.g. UDP, WebRTC).

Result: You get a fully functional HTML5 player which can be easily expanded in a modular way e.g. via our analytics and P2P solutions, which can function as entities separate from the PolygonCast video platform.

While we strongly recommend doing a full deployment, whatever phase you decide to integrate on, you will still receive major benefits.

Summary of business opportunities

Give the PolygonCast Streaming Platform a shot.

Multiple types of content

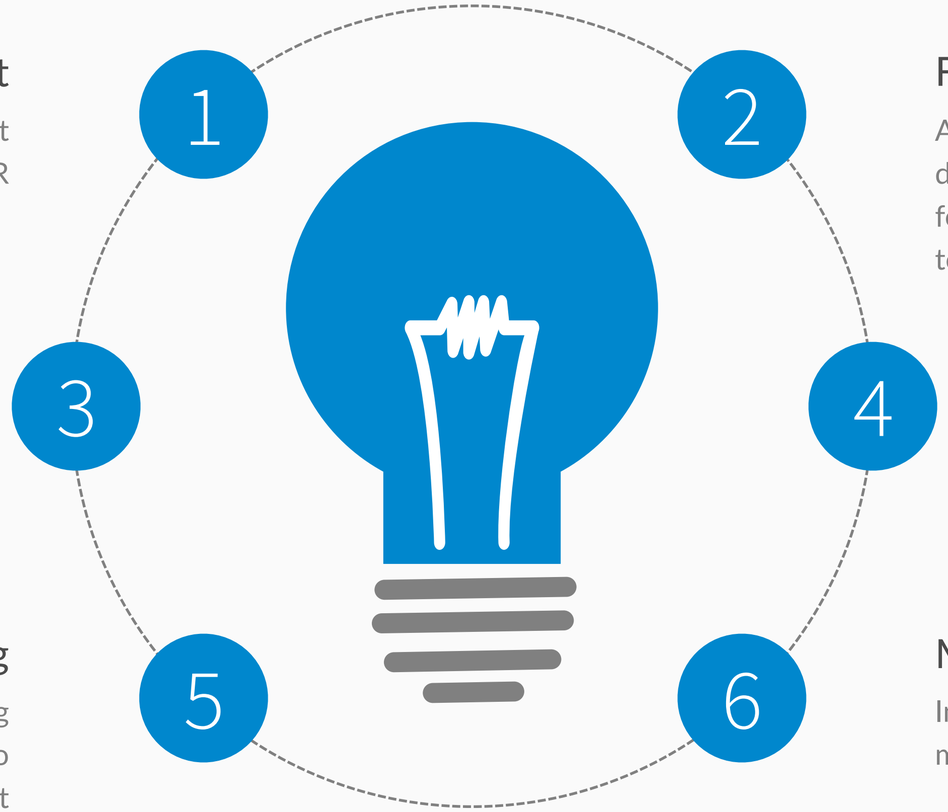
Combine various types of content like different camera views, 360 video, social feeds and VR

Compatible

Every component was designed with compatibility in mind, so you can opt in for only client or server or a particular server feature

Video, Media and Data Streaming

Today, the same teams that have been working with video are faced with a task of making video more interactive. But their toolchains do not support that. And ours do



Fast streaming

A multi - transport technology approach opens a door to higher download speeds that are crucial for high quality media streaming we will need tomorrow

Modern and interactive

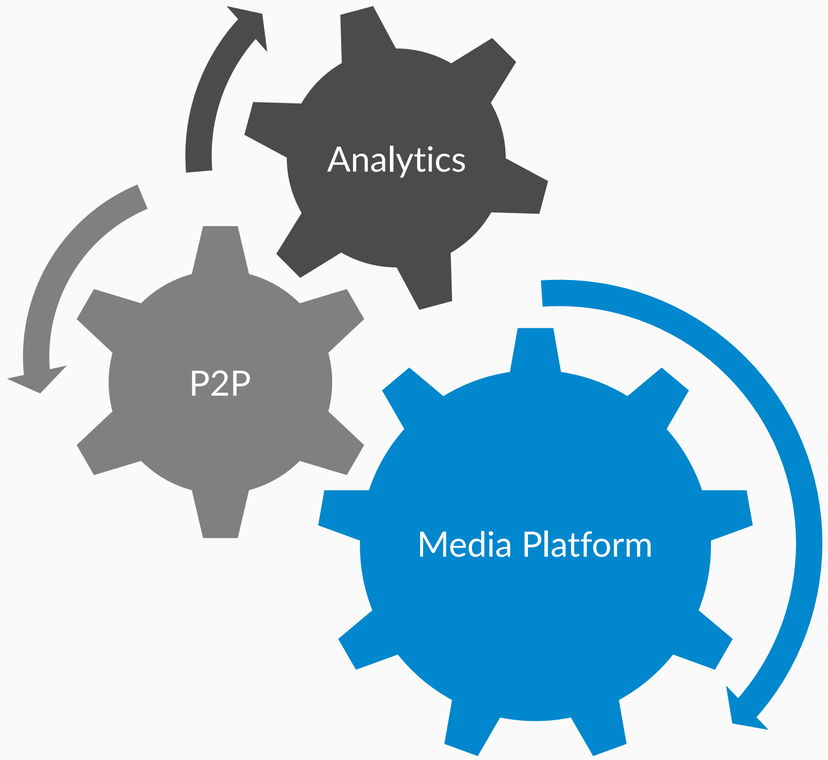
Content and data remix is built into every corner of the stack. This will help you to add interactive features faster

Monetizing quality

Interactive content we promote has a greater monetization potential then plain videos

Works best together

PolygonCast is a whole family of streaming products including analytics and P2P delivery



Media Platform

Can be integrated on any level of content delivery, supports WebRTC, UDP, QUIC, P2P, MultiCDN, MPEG-DASH, ABR, any DRM and so on. Allows for easy VR/360 3D video delivery including segmented ABR delivery of 3D content depending on viewing angle.

Real-time QoE analytics

HTML5 player agnostic video analytics (work on ANYTHING) with intuitive dashboard visualization system. Hosted, fully scalable with unlimited data retention, graphical/text data export functionality and FULL customization including data gathered and/or visualizations.

P2P delivery

WebRTC/UDP based P2P delivery system, DRM agnostic with user geo-grouping for internal corporate streaming optimization.

Thank you for your interest

We are always happy to answer any questions you may have.

Contact Us:

www.polygoncast.com

contact@polygoncast.com

+48 605 26 26 25

+48 880 883 147