EPISODE 614

[INTRODUCTION]

[0:00:00.4] JM: Searching through all of the videos on the internet is not a simple problem. In order to search through all the videos, you need to build a search index. In order to build a search index, you need to build a web crawler. Video files are large, to crawl all of the actual video files is expensive enough. If you wanted to store all of the actual video files, that would cost way too much money.

In order to build an efficient index in a way that manages your costs, you need to have a way of storing the information about a video without storing the entire video itself. You might be thinking "Why would I care about this?" Hasn't Google already solved video search? Why are we even talking about this?

Google has solved some aspects of video search but there's a different set of challenges that is being tackled by a video search company called Pex. In order to explain what Pex is building, we should first explain the problem set that they are trying to tackle from a business point of view.

Videos across the internet are consumed on a variety of platforms, you've got YouTube, you've got Instagram, Facebook, Vimeo. These videos are sliced up, they're bootlegged, they're repurposed from one platform to another and if you're a content creator who earns their living from hosted video streams, this can be a nightmare.

Imagine you're a musician and you make lots of money from your music videos. You upload your cool new video to YouTube and it instantly gets bootlegged by other users and shared across the internet in hundreds of different places. When people watch these stolen versions of your video, you're not getting compensated. If you could locate all of those stolen videos, you could order them to take it down or claim that the video is your's so that you're going to get paid for it.

Here is the engineering problem. How can you find all of those reposed videos? How can you find sections of your video that have been repurposed and reused in ways that aren't – not okay with you. You can find those videos by crawling the web and building a search index for every video on the web.

Rasty Turek is the CEO of Pex and in this episode, he describes how to build a system that crawls the internet and indexes videos. This is a large scale engineering challenge and there are lots of tradeoffs to be made between financial costs, speed, accuracy, engineering complexity, it was really interesting to talk to Rasty.

Before we get started, we are looking for a videographer. We're also looking for writers for Software Engineering Daily. We are also looking for a couple of other roles. You can find those job postings at softwareengineeringdaily.com/jobs. And if you're looking to get involved with Software Engineering Daily on a lower commitment basis, we've also got the open source projects at gitHub.com/softwareengineeringdaily. We have apps that are purpose built for Software Engineering Daily. And you can also check out those apps in the app store for iOS or for Android.

We just got a new update to the iOS app that creates a feed based off of what episodes you've listened to so if you listen to episodes, you can get recommendations based off of those episodes, recommendations for related content and other episodes.

And, we'd love to have you check it out, we'd love to have you contribute to it so you can check those things out and with that, let's get on with the episode.

[SPONSOR MESSAGE]

[0:03:46.5] JM: Over the last two years, I spent much of my time building a video product. We had issues with latency, unreliable video playback, codecs. I was amazed that it was so difficult to work with videos. As it turns out, video is complex and figuring out how to optimize the delivery of video is not easy. Especially since there is both mobile and desktop, and mobile users might not have as much bandwidth as desktop users.

If you're an engineer working on a product that involves video, you just don't want to think about any of this. I can tell you that from firsthand experience. That's why Mux exists. Check out mux.com and find out how Mux makes it easy to upload and playback video.

Mux makes video hosting and streaming simple. Today, you can get $50 in free credit by mentioning SE Daily in the signup process. Even if you aren't working on video right now, if you think you might work with video in the future, Mux is a really useful tool to be aware of. Check out mux.com and if you're an engineer whose looking for work, you can also apply for a job at mux.com.

On Software Engineering Daily, we've done two shows with Mux and I know that Mux is solving some big, difficult problems involving lots of data and large video files. To find out more, go to mux.com. You can get $50 in free credit by mentioning SE Daily and you can apply for a job if you're interested in working on some of these large challenges.

Thanks to Mux for being a sponsor of Software Engineering Daily.

[INTERVIEW]

[0:05:41.0] JM: Rasty Turek, you are the CEO of Pex. Welcome to Software Engineering Daily.

[0:05:44.5] RT: Thank you very much.

[0:05:46.2] JM: You're building a search engine for video and music and I want to get into the implementation of that because there's lots of great software engineering questions but let's talk about the motivation, the business model.

Why do we need another search engine and why do we need one specifically for video and music?

[0:06:10.6] RT: Well, this one is a little different, or at least the purpose of it is a little bit different. Even very simple is very closely to modern search engines so we go out and we crawl

the web, we extract audio, visual content. The purpose of it is a little bit different and it's used for very direct analytics on the variety of the content and then rights management.

We work closely with rights owners, creators, brands and to help them to understand the spread of their content, who is consuming it and stuff around it. It's not necessarily for the end audience the same way as Google or Bing.

[0:06:46.3] JM: So the business model is that if I'm a creator and I've got videos throughout the internet, I want to find where those videos are, I want to find the bootleg instances of those videos because otherwise I might not be able to monetize the views on those videos?

[0:07:05.8] RT: Monetization is one and it's usually the smaller portion, the larger one is just the general analytics and general understanding of where the content goes and who consumes it. Because the problem is, if you as a creator, upload the content to YouTube, that doesn't mean it's consumed on YouTube. It just means it will spread out everywhere across the internet and then you are missing out on the size of the audience on who is being interested and who is interacting with your content.

As a consequence of that, the creator is then to focus on the direct audience they have data on. You know, was you measure, it's what you become and that doesn't necessarily mean the true audience, it also doesn't offer and show the whole potential of your content so when they are talking to advertisers, it's very different when they say, you know, I reach 14,000 people on daily basis, that when they say, well, actually, it's 1.4 million across the globe and this is how it breaks down.

[0:08:03.1] JM: I am a creator of sorts, I make a podcast. How does the outlook of the podcaster in 2018 compare to the outlook of a YouTuber or a Spotify musician or an Instagram influencer?

[0:08:17.6] RT: I think podcaster a little bit more siloed. I think they're not as spread as any other content. I think there are usually content from two, three, four, five places where the users are kind of used to it. But at the same time, sometimes the podcasts are picked up by other

shows or they're picked up by other podcasters or just in general, maybe even YouTubers and others.

It's fairly interesting to see for the creator, where the content goes and why will they pick up this episode and what they talked about so it's kind of a reference. You can think of it as website links pointing to each other.

We kind of made that connection for the creators and help them to see that in a kind of real time-ish fashion.

[0:09:01.1] JM: As you said, if I'm a YouTube musician and I publish my YouTube video on YouTube then it can potentially be taken and reused in other places on the internet and I would like to know how it's performing in other places on the internet. Why is that though? How would YouTube videos be taken from YouTube and repurposed for other places on the internet?

[0:09:29.3] RT: Well, there are many different reasons why and how, it's very simple. Users just literally download the content that they are watching so the simplest way is just open the browser, wait for the stream to come in and then just take it out of the hard drive for essentially just capture the stream that is coming in.

However, people need to realize that a lot of these platforms are blocked across half of the planet so for instance, YouTube is not available in China, it's blocked in Turkey, it's most popular channel, Vevo is not available in Germany and platform after platform has different rules and they approach things very differently and then the audiences are not spread out evenly and equally.

Even Facebook has two billion or YouTube has over a billion users, it's still not the whole population of the internet. A lot of users will take that content and spread it wherever they are comfortable or wherever they are home so there are platforms like [inaudible] in Russia, [inaudible] in China and even in the US, you have different platforms like Instagram where the content has a little bit different connotation and maybe feeling than it has on YouTube.

Users very often take portions of the videos and re-upload them to their Instagram with maybe some overlay of themselves singing to it. There are whole platforms build it of that like Musical.y or Smule where people who just I'm synced to the tunes together.

I think this is just a form of expression and if you think about how return was always built around spreading, this is just a normal behavior that I will say the older establishments and not necessarily just companies but the society itself wasn't necessarily adjusted to this about until maybe recently, I think as we are progressing, you're saying this is more and more and a good example is unabated gifts.

They're on their own are not expressions of some media content. However, they will be expressed differently even if they're coming from a content that wasn't meant for something, so animated GIFS as memes are very unique form of expression.

Even when they are originating in the content that has maybe a little bit different message and that as animated GIF, with the meme itself, it will become very different too, very different expression. I think users are using their creativity to express themselves this way.

[0:11:55.4] JM: Okay, let's imagine, I'm a YouTuber and I publish a video in the United States and I want to know how that video performs in Turkey. Well, since YouTube is blocked in Turkey, then probably if somebody is a fan of my content, they're going to take it off of YouTube and republish it on to a site in Turkey, maybe they'll also take it and publish it on Instagram without labeling it with who I am. Or maybe they take out some section of it and publish it on Instagram.

As far as I know, you can be on Instagram in Turkey. As a creator, I want to be able to index the different instances where my content exists. Despite the fact that I only initially uploaded it to YouTube and that's valuable because if I know that people are in Turkey are fans of my music, maybe I want to cater to them in Turkey or maybe we want to cater to them on Instagram. Is there any other reason, what else am I missing about the value of knowing where people are watching my videos?

[0:13:02.3] RT: It's not only where but it's also what. As you pointed out, in many cases, the big portions of your content so it's not necessarily that you produce something large and they take it in whole. In many cases, these are the take the portions and those portions are very important because that's a wisdom crowd telling you that these area in your video, audio podcast, whatever that is represents the portion that the people are the most interested in.

It's not only that they watch these in Turkey but this is somehow more resonating with them because they are telling you through their actions. As a consequence of that, let's say you are a musician, now, go and try to get a deal to have a live concert in Turkey because you know that there is audience, it's much more based on the data when you know, if you show up in Turkey and say, well, I'm a musician and believe some people around might enjoy my music.

I think you will have hard time to book avenue and get promoters on board with it. I think the general notion is here with the data, you can actually make more educated decisions and helps you in the overall business and the outcomes of that are very different for different people. As of rights owner, that is representing the rights and wants to monetize the content in the form or shape that you decided to, this allows you to understand where to maybe get a licensing, the other platforms are being the ones popular.

As [inaudible] for instance, this helps you to understand where your customers lie or where actually you are having better ROIs that you expected for movie studios, especially on the trailer side, this allows you to understand what countries may be the first on the box office if you didn't think about them in that way, when you see the trailers are clicking very well in Taiwan, maybe it's a time to have a Taiwan in the first day of box offices the day of hundred where movie is all around the torrents and everywhere else.

And because the world is not treated equal at this point, this information matters significantly to every single rights owner and a little bit different way but that's what we actually allow across the board.

[0:15:16.2] JM: I see, I love your return to the business question and the discussion of creators a little bit later. I'm very curious about the lives of creators and the future opportunities

for creators, how big that market is and so on. But people listen to this show because they want to know about software engineering.

Now that we've stated the problem that you're trying to solve, there's plenty of interesting engineering questions that we could explore. I think the core of the question is I upload a video, I need to be able to detect where that video is playing across the entire internet. And in order to do that, I'm going to need to index the entire internet in a way where I can match my video's fingerprint against all of the other videos in all of the other places across the internet.

Is that correct? Do I understand the problem statement well enough?

[0:16:15.4] RT: Yes, that's exactly correct.

[0:16:17.2] JM: Okay. Video is inherently hard to index and that's because you have this long sequence of images and sound. What are the steps to building a search engine for video and search index for video?

[0:16:34.0] RT: Yeah, well, there is a lot of moving pieces in place so the very first one is if we start from the surge in point of view. There is a need of for a spider or a crawler both at the same time so that means you have to be able to discover where the video actually is. For that, there is not many, I will say, smart ways to do it, just brute force through those platforms and some of the platforms that are significantly large so for instance, YouTube at this point is a little bit more than 4.5 billion videos and growing by four to five million videos a day.

That means we have to keep up with this load and we have to find each video as soon as it's uploaded, that means we have to reach into this platform very often. We have to follow the lengths and we have to expand on that through each platform. There is just the first step so you would keep up with the information that there is a new video.

The second basically have to be able to download that video so that means in some cases, you have to simulate the browser because a lot of platforms will use techniques to prevent that direct scraping or direct connections so for instance, in some cases, DRM, in other cases you just

build something in house but at the same time, you have to be able to do this across the board so that's a pretty significant thing.

Then just downloading the content. I mean, it's a lot of content and when you already have all the content in, you have to somehow be able to index it so in our case, it's finger printing with our own algorithms and the optimization comes on the speed that you have to keep that content in your hard drive so for us, because we want to make this viable in a business end. we can all saw all the content because it represents exabytes of data so the finger prints represent the very small portion of the original size and usually it's the kilobytes to hundreds of megabytes.

You have to create those fingerprints as fast as possible but as you pointed out, indexing and in general, working with video is very intense on the CPU or whatever processing mechanism we are using so that's where the optimization comes but even when you have all of these done and when you find how they fingerprints that we have to be able to search for them so we have now database larger than 10 billion videos and you have to be able to unearth information about the relationship between video A and the whole catalogue within minutes at worst.

That's where all of these interesting things come together to do this at the massive scale with the high precision and economically it's where the challenges are lying so we spend over four years building these together and I believe we are at the forefront of the market at this point.

[SPONSOR MESSAGE]

[0:19:38.9] JM: You listen to this podcast to raise your skills. You're getting exposure to new technologies and becoming a better engineer because of it. Your job should reward you for being a constant learner and Hired help you find your dream job. Hired makes finding a new job easy. On Hired, companies request interviews from software engineers with upfront offers of salary and equity so that you don't waste your time with a company that is not going to value your time.

Hired makes finding a job efficient and they work with more than 6,000 companies. From startups to large public companies. Go to hired.com/sedaily and get $600 free. If you find a job through Hired. Normally you get $300 for finding a job through Hired but if you see our link,

hired.com/sedaily, you get $600 plus your supporting SE Daily. To get that $600 signing bonus upon finding a job, go to hired.com/sedaily.

Hired saves you time and it helps you find the job of your dreams. It's completely free and also, if you're not looking for a job but you know someone who is, you can refer them to Hired and get a $1,337 bonus. You can go to hired.com/sedaily and click "refer a friend". Thanks to Hired for sponsoring Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:21:21.2] JM: Okay, there's a lot of different points in there that we can dive in to. Let's talk about the process of fingerprinting. If I am a creator and I want to find all the videos that have been created from my video or that may take a sample of my video, I need to create a fingerprint of my video so that when Pex is crawling the internet, Pex can check if the fingerprint corresponds to one of the videos that the Pex engine is crawling.

At what point in the crawling and the indexing of the web, do you compare that fingerprint of my video to all of the different videos that you are crawling, do you crawl the internet and then grab all of the videos and then hold them for a while and then like do the finger printing and then let the videos go or do you like crawl the internet and then one by one, whenever you come across a video, do you check it against all of the finger prints in your database that you need to be looking for?

Tell me a little bit more about that process?

[0:22:36.0] RT: Well, we actually hold all the videos forever, we don't hold the underlying videos because at this point, we are roughly 1.8 exabytes of data in. We process in the lifetime almost two exabytes of data. How we do it is we downloaded each audio visual content with fingerprint it and we throw it immediately away so we only store the fingerprints.

[0:22:58.5] JM: So sorry, you download the content for every single video on YouTube, every video in the instance?

[0:23:04.4] RT: Correct. It's also for live streams or any other source so it's a lot of data. Then the fingerprinting process on its own is optimized to a massive capacity in speed so instead of us going with a VGA or a custom processors, we figured out how to do this as fast as possible on traditional CPUs. and then we utilize the cloud offerings of what some called preemptive either call spot instances of which essentially is short living servers that are cost pennies or cents on the dollars and we are able to run them in masses.

Essentially it's just that massive brute force across hundreds of millions of videos at any given point and then, when we saw these fingerprints and customers comes through us later, we are able to search through them. Essentially, it's not in the any sense of real time, it's more kind of delayed.

Even if when we process videos at the same time, we usually front those fingerprints against everything that we are actively searching for. In some cases the delay can be as short as 30 seconds after the videos, I pull it in the other instances it can be the lifetime of the platforms so if you are still able to match the video that was uploaded, the very first video that was uploaded to YouTube with any video that we have in our database today.

[0:24:28.6] JM: Okay. The process of creating a fingerprint is that some kind of rolling hash function or you don't have to give me the proprietary secret sauce but can you tell me about in broad strokes how you build the content ID system.

[0:24:44.9] RT: Yes, it's a fairly complicated because you have to deal with distortions and the distortions come in many forms of shapes so for instance, some clever users remove, let's say every 12 frames from the video which doesn't impact the quality of the video but it will throw off a lot of different systems in place.

Some people speed up, slow down, others will put frames there on the videos, put the blurring filters around them, horizontally swap the images, put lots of logos in them so our algorithms have to deal with all of that and have to deal with that in the indexing side so what we focused on is we spend as much as a research as possible on the indexing side where the search is then the cheapest function possible.

In our case, it's kind of, you call it rolling function, it takes in a lot of different inputs and it tries to figure out what is important with the in period of side and the reason this is essentially a very simplistic censor that holds the data as a representation of a portion, not necessarily time based or also not frame based or also not frame based. It's kind of in between and it allows us to identify content short as how second and so that's why we are able to take animated GIFS that are created as a memes, off of two hour log movie and we are able to connect them back to each other.

As I pointed out, we have to deal with a lot of distortions and in some cases, as a good example is Shia Labeouf going bonkers in front of a green screen where people replaced the green screen with lots of different stuff and matching this back to the original, it's a fun challenge.

[0:26:27.2] JM: I don't think I completely understood your explanation of the crawling and the storage and that's my fault. You get a fingerprint for every original video on YouTube so that anybody, if they wanted to look up the videos they have been derived from a video for which you have a fingerprint, you can serve that query because you've made a finger print for everything. I didn't quite catch what you are doing in terms – because like you said, you can't index all of snapchat and all of YouTube and all of Instagram and all these things because if you saved all of the videos, all of the potential derivative sources that you wouldn't have the money, you don't have the budget to store all that data.

I'm sorry, I didn't catch it but for each of those videos, across Instagram and Snapchat and so on, are you making some derivative of that video that captures the essence of that video and then doing a mathematical comparison from those to the finger prints because you've got all the finger prints stored durably all the time, what are you doing with all of that content across snapchat and YouTube and Facebook and so on as you're crawling it?

[0:27:35.9] RT: The fingerprints that are literally the essence of the contents. The fingerprints are essentially able to tell us what the content holds. We do not hold as you pointed out, we do not hold any of the videos ever so once the video is processed, that means fingerprint is created, it's immediately deleted.

At any given point we don't hold more than a couple of hundreds of terabytes of data in the kind of meat cache because we will end up bankrupt very soon. What the finger prints allows us to do is essentially use mathematical formula to compare them between each other and detail us the story about what's portion of the content is being or is not being used between themselves.

And then we are able to reach on them as a visual information back to the user and the user then can go and if the content is still available, that means it wasn't removed in between that query, time and indexing time, they can go and watch them in the precision of like have second, to see where the content is overlapping and what that means and they can verify these information and then we also capture a lot of the metadata surrounding those videos to how many views it has.

Who commented it on it and a lot of other stuff and you're able to expose this back to the users if you're able to give them deep analytics on where the content goes and what it means and who interacts with it and stuff around that.

[0:29:03.0] JM: As you're crawling the internet, for all the videos across snapchat, for all the videos across snapchat and YouTube and Facebook and so on. Are you merely running each of those videos through the same fingerprinting tinder generation process so that you have a mathematical artifact that you can just compare to the fingerprints?

[0:29:20.8] RT: Correct.

[0:29:21.9] JM: Okay, that's pretty cool. And then, because then you can actually store, because it's not expensive to store fingerprints of everything across snapchat?

[0:29:31.6] RT: Actually, for all 10 million videos that we have right now which I think average length is around 17 minutes. The database is only roughly hundred terabytes big so the fingerprints are very small and they allows us to run very fast queries cross themselves for that exact reason because the formula is essentially equal for every single video and that allows us to expose this information across every single video we ever seen including videos that were removed.

We are able to kind of go back towards in the time and tell the story of a video that doesn't even exist anymore.

[0:30:10.2] JM: Right, you can tell the story in terms of maybe why that video got removed but I mean, the fingerprint isn't going to tell you that much information, right? Or I guess you could store the meta data also. You could store the comments and other things but what do you mean you can retail the story?

[0:30:28.8] RT: For instance, let's say, somebody leaks a video, let's say it's about a senator doing something crappy and the senator is able to take that video down or any other situation where this happens. And that video spreads and now you see a [inaudible] of that video going around and then maybe see it and then runs it.

And now your question is where this came from. How this even happened, right? We are able to, if we saw the original video in every single content of that we are able to backtrack back to the first very occasion where it happened, who uploaded it, what people commented on it, how many views it attracted over the time and then when it was removed, why the reason, if that reason was given, why it was removed and then we are able to track this across the internet so if you are able to do kind of the Barbra Streisand movement where you know, somebody goes after the content.

You can tell where it came from and what's going on with it and we are able to measure that virality. And then on the content itself, even we are not able to reproduce it so if you are not able to take the fingerprint and replay that video to you, we are able to tell you if any other occurrence of that video will ever show up. And in many cases, this is very interesting for the authorities where you are dealing with very sensitive content like those videos where you want to make sure that this just doesn't spread.

And you want to catch every occasion of that video and on every single platform before it catches on. You are able to assist with that and many other angles.

[0:32:05.2] JM: Okay, interesting. Governments could potentially work with you if they wanted to prevent the spread of terrorist videos. Wow, fascinating. The crawler, you know, you don't

want to have to crawl over videos that have not changed so like if a YouTube video has not been changed, do you know that?

Can you tell that from the meta data on the site? If I'm a creator and I upload a video and then you crawl it, and you do the fingerprint comparison and then 25 minutes later, I replace my video with something with Kyle Minogue in it, you would like to be able to get that signal and you know, re-crawl it, you would re-crawl the video itself.

Does YouTube give you meta data for that or do you actually have to do some manual crawling over material that might not have changed.

[0:32:57.4] RT: We have two systems in place and that one is checking the meta data and that one is just dealing with the audio visual content. The meta data itself doesn't tell any story outside of itself. You know, how many views, what the description is and these kind of things but it doesn't point out that the video was changed.

For that, we came up with very simple solution to it where we check the information regarding the video and audio stream. If that video changes and you don't keep it precise or for instance, the original video at three minutes and 27 second but a new video is three minutes and 25 seconds, we know something changed. Also, we extract very small portions of the content.

Usually just first or second, 512 bytes of that video and we check if that hash simple shot 256 or something similar, we take that hash and we check if any change was happening in that video and these information is sufficient enough for us to know if we have to repurpose that video or not.

Actually works quite well across the board.

[0:34:05.2] JM: Tell me a little bit more about the infrastructure that it takes to run this crawler?

[0:34:11.1] RT: The crawler itself is not as big as it looks, although it sends roughly 70 billion requests a day. It is not that complicated, it's written in Go and it has a very few moving parts, it only looks for signals that we are interleaved or that the system can interpret in its own way.

What is maybe more interesting and more heavily deployed as caching so one of the very important things for us is to cache every single request that we have and hold out information as long as possible so we want to know that for instance this page that holds this video is changing very sporadically so we don't have to access it 20 times a day but instead, it's enough when we show up once a week or maybe even less.

So they crawl in each surface very straight forward, very simple. It doesn't have many complexities in it and I think that is why it is also able to scale. Where the complexities comes from is we don't want to take down platforms when we set a lot of traffic to them. So if you are trying to be very mindful of the way that we are doing this and we also hit a couple of bugs in the real lifetime sort of we took the massive platforms down in for hours.

So over the years, the developed systems that prevent us from doing so. For instance, the idea is we shouldn't sending more than a couple of request the second to any given platform no matter the size. So we try to bundle the request together. We also try to be very mindful of what we have to touch and when we have to touch it and we have algorithms that are making this decisions and preferring the requests along the way.

So the crawler themselves then just interact on these tasks I would say and grab the data that they see and then send them back to the pipeline and everything they build in the infrastructure is built on a synchronous pipeline. So you can think of it as a factory where it goes from one conveyer belt to the other and they work independently from each other. So one component doesn't block the other from functioning and that allows us to kind of smooth the sailing for the crawlers.

To grab the data on its own phase and then let the other components deal with the data on their own pace, scale at their own size whatever they want to be without manually needing to tweak any of this. It took us a couple of years to get here but we are now able to scale using infinitely if there is enough researchers open to us.

[0:36:47.4] JM: What are the different databases and storage systems, you know a bit of caching layer and the persistence layer?

[0:36:55.9] RT: Also the persistence layer for us is a very heavily sharded Postgres. We love Postgres because it is a battle tested database and we know it is fairly well. So we utilize a lot of its functions like triggers that allows us to run story procedures on every single role of data that we insert and that as a consequence we have all the algorithms that make the decisions every time directly in the Postgres itself. So we have kind of ETL in a sense to be able to weaken the Postgres.

We also build a lot of extensions around it and it allows us to take actions directly from the Postgres instead of having third party components that we all have to write and that we'll have to interact with Postgres. So it's kind of our heart and brain and it also makes over our SRE job much easier where we know that we have to keep other Postgres instances up and running and then 24/7 and we have to protect them from anything to be happening.

But it allows us to run everything else as a form of paddle essentially where we can shoot servers when they start limping essentially. So we have very few servers that have to be protected and have to be evolved but at any given point and allows the servers to daunt. The caching later was implemented in a mix of different databases but we are now moving towards foundation DB and we will be utilizing that solely for the whole caching purpose at it scales very nicely.

And it allows to be able to hold pentabytes of data and still have a millisecond responses for all of them so we are quite happy with the performance that we see and we will be expanding on it but as a history, we had an implementation with Redis and then it depends on what kind of data that the videos usually stored in something like S3 or Google storage for the period of time that it needs to be processed because the price for that storage is very low.

And then the webpage is are usually stored within something like Redis or meme cached where they are held for as long as possible to be used between the components and the interesting part about that is that the components because they are first of all independent from each other

but also because they don't have to interact and they don't have to communicate. It is a different component. It's needing the same data or at least data from the same page.

It can just hit the cache instead of going back to the page so we save I will say many billions of requests a day towards the platforms which allows us to kind of operate a little bit of sneakishly and don't hurt the platforms with extra traffic that we don't need to get.

[0:39:45.5] JM: Yeah, the thing that struck me about what you just said was the stored procedures in Postgres having a lot of the functionality of your core platform in Postgres stored procedures.

Could you tell me more about that decision and what those stored procedures are?

[0:40:02.6] RT: Absolutely. If you build even out of Postgres, we build it a little bit differently than you think of SQL databases. We think of it still as a pipeline. So we have multiple tables that get data at the forefront. So we have writers that essentially grab data from different components and then write them to the correct table but they don't make any calculations there is nothing. There is no external component that touches those tables.

Instead of that each single table is running a sort of procedures through a trigger and now in those store procedures we make decisions like for instance, if this video gained a thousand views in less than an hour, I want the spider go back and essentially grab that same page in the next hour and this information is to send out the managing part for the spider or for the crawlers and it's then acting on at the right time. So this is kind of the first wave.

The second wave is where we merge the data to each other to eventually expose them to our customers. So our customers don't interact with the whole database. They instead get their own place where they have information stored and even when we have to store the same information in multiple times which requires a little bit more space on the hard drives, it will allow us to run a ridiculously fast queries where we have now almost 2.2 trillion rows of history call updates.

And we are still able to run queries in under one second for any single customer. Of course all of this information and the reason for that is that if you have broken those tables down to small

pieces and then at each step and each time a table gets an update, these procedures then makes decisions on what you'll be exposed and it's then called be to a table number three for instance and this table hold intermediary data for the customers and then other table gets a new update.

That is now mixed with these table and it goes on up until it reaches it's destination and we have learned a lot of different procedures including actual learning which is exposed directly in Postgres through our extensions where we run interesting algorithms and to above the data that we are seeing. So for instance for every single video we process that means it has to have both video or audio. We also run annotation algorithms that allows us to expose information like this portion of I'll do your track this condemning music.

This other portion contains, I don't know shooting is the other thing is the human speech. This other portion is animal sounds and then if you are able to store this information and exposed and the customers and do something later with it and we have these for both video and audio where we expose or extract lots of additional information from the content itself and now lots of these algorithm is afterwards are running or lots of algorithms afterwards are running on these data directly on the Postgres where they make decisions.

Like for instance, if the fingerprints are saying these two things are matching however the annotation algorithms are not saying the same story we then proclaim them as a false positive and sent to our RND department that they then look at the content and see if there is anything going on with the fingerprints maybe we made a mistake, maybe there is something wrong with annotations but these things need to kind of confirm each other and it allows us to be very, very precise in comparison to our competitors.

Where we use lots of different signals to make those decisions and all of that is completely automatic and why we choose to do that in Postgres is first of all, it loves us to not to fry extra components because if you are going off the database you will now have to have the drivers to connect to it. You will have to communicate with it that means read and write back to it where the procedures are already working on the data that are stored in the memory at any given point.

And second of all because the Postgres runs in transactions that means that procedure needs to finish successfully. So if it doesn't finish successfully, it is then returned at the forefront. So if we have seven tables connected to each other, the first table will return an error to the writer will have to repeat that whole writing sequence.

So even it's a little bit more expensive on the resources and yeah on the resources and gives us a 100% accountability for the data. So we don't have situations where something happened in between of the transaction and then it just died and it is very important for us and our customers that we can verbally give them data at any given point. So we much rather sacrifice performance that means we have to take a little bit more time before the brightness of the data.

[0:45:00.4] JM: Are there any disadvantages to putting so much functionality in a place where it's tightly coupled to the database?

[0:45:08.8] RT: Well the downtimes are then severe. So essentially whenever there is any problem with the database on its own that means everything is impacted. So every single component and every infrastructure gets in the halt that is interacting with the database, we minimize this to very small portions but it is still a significant part. It also influences our customers that have no access to the data whatsoever. So we have to keep those databases up and running no matter what.

The other thing is it is very hard to debug and just measure and monitor these things because the procedures have ran on every single road that is inserted into the database and as a consequence of that, if you have no visibility into what happened when it happened and why it happened in the process. So usually the benefit of that or kind of the disadvantages and benefit at the same time, once you have tried to develop then there is never a problem.

But writing to develop can be a challenging thing especially when you have chained tables through triggers towards each other so far as we have seven tables that are copying the data between each other or kind of like each other to the next table and those are written by different people in our organization. So we have to interact very tightly on all of those decisions because if one person makes a mistake then it's propagated up to the end to the customer.

The good thing as a consequence of the databases, we are able to reply any situation from the kind of the backlog or from the previously start information. So if you made a mistake at some point, you are able to start from that point again and just run it again. I will say that the hardest part is definitely debugging and monitoring and then also the deployment. I mean Postgres was never meant for this. It was never built for this so we have to kind of twist his arm a little bit.

I will say the impressive part on the Postgres side is that the database can really withstand a ridiculous amount of pressure and is able to deal with incredible complexities that we introduced there and I think that is one of our biggest kind of secret air quotes weapons where we don't have to build a complicated procedures around the database which is built in the database.

[SPONSOR BREAK]

[0:47:41.0] JM: Every team has its own software and every team has specific questions about that internal software. Stack Overflow for Teams is a private, secure home for your team's questions and answers. No more digging through stale Wikis and lost emails. Give your team back the time it needs to build better products.

Your engineering team already knows and loves Stack Overflow. They don't need another tool that they won't use. Get everything that 50 million people already love about Stack Overflow in a private, secure environment with Stack Overflow for Teams.

Try it today with your first 14 days free, go to s.tk/daily. Stack Overflow for Teams gives your team the answers they need to be productive. With the same interface that Stack Overflow users are familiar with. Go to s.tk/daily to try it today with your first 14 days free. Thank you Stack Overflow for Teams.

[INTERVIEW CONTINUED]

[0:49:19.9] JM: Well we have done three shows with people from Citus Data which is the company that you've probably heard of. They've built a lot of functionality around the PostgreSQL extension capability and I mean they're sponsors of Software Engineering Daily,

full disclosure but I find it interesting that their approach to enabling people to build a data platform because you see all of these companies that want to build a data platform now.

Whether it's yourself or it's Uber or Facebook of Thumbtack, companies that are – they have all of these transactional data, they have all of these OLAP data, the analytical data and getting between those two phases I mean it's always been a challenge but the scale of data continues to grow and also the access patterns of different people who want to get at that data whether you are a data scientist or you are writing machine learning algorithms or you want to write algorithms for what you are doing.

The fingerprinting kind of thing, you know you have all of these different things that you want to do on top of your data and so you have a data platform problem. It seems to be a problem that people are attacking from different angels. One of which is this all Postgres angle and you know as you mentioned, this deployment problem that sounds like a big deal because everybody wants to be continuous delivery and I guess you could do continuous delivery with the database, right?

You could just have some sort of staging thing and some sort of testing thing where you have a subset of the database or you have some sample transactions that you are going to run. Do you do anything like that? Do you have a continuous delivery pipeline for your database?

[0:51:03.4] RT: Yes, sort of. It is not as sophisticated as it sounds but they do have a staging environment where we are exposing a portion of the data to the system and then if you are able to rest out live what will the consequences of any of those deployments be. They do hold versions of the triggers and the storage procedures in our internal Wiki. We don't have any automated system behind them because we've written those procedures very sporadically.

I will say one a month, maybe even less. As I pointed out, once it's rigged in it is actually very stable and it is very nicely performing. So we don't have anything very complex behind it but we do have a manual mid-level where we deal with these complexities. Actually interestingly enough when you pointed on Citus Data, I have here a very happy customer of theirs, very happy paying customers so –

[0:52:01.7] JM: This was not planned.

[0:52:01.8] RT: Yes, we've used Citus Data very extensively for the last two years seeing that we are Postgres. So we do actually horizontally shard through Citus and it's our only database that we use and it's true that I have nothing but praises to tell about these guys and their work and in all truth, it allows us to build the complexities into the Postgres because the horizontal sharding allows us to break these tasks into small instances where they are run on a single shard which is fairly small in comparison.

If you have to run a massive deployment of each Postgres on every server. So now this is a consequence of that and we have hundreds of thousands of transactions occurring at any given second but as a benefit of that, if one transaction dies it doesn't influence the whole database and influence just that one single transaction and that if you are able to also achieve a ridiculous scale in terms of performance for instance.

At any given point, we've write roughly 60,000 rows of data into the database on the forefront. So that means in the interesting tables and then we take hundreds of thousands of decisions within the next minute on those data and all of that is ran almost in real time. So for instance when we measured how quickly will the occurrence of the data be from the forefront table to the back table that means to the table that is visible to the user and there are five tables in between, it takes roughly 17 milliseconds.

And that is with all applications of all the machine learning models and all of the procedures that needs to be acting on and the reason why this is, is because the data is already memory. Postgres is already working on the data actively at the point of where we are evaluating it. So the overhead is very small and then it's just going down to the engine how quickly it can perform the information that we need. And then we've rolled in lots of upper extensions and [inaudible]. So we can take advantage off the speed and the performance on the language itself.

[0:54:18.9] JM: I heard you say at some point part of the persistence storage is in S3, part of it is in Google Cloud storage, are you running lots of infrastructure on both cloud providers?

[0:54:34.1] RT: Actually we have seven.

[0:54:36.2] JM: Seven cloud providers?

[0:54:38.0] RT: Yes.

[0:54:38.9] JM: Okay.

[0:54:40.1] RT: There are many reasons for it, one being that we compete with YouTube in some sense and is having everything Google is quite challenging maybe more on the business and legal side than it is on the technical side although I must say we are very or quite fond of Google Cloud platform. I think it is on our most preferred platform to deploy. It could deploy anything but we utilize different platforms for different things. Although other crawlers are in general deployed in every single platform.

It is also for altitudes reasons and different geographical reasons. So for instance, some content that is available only in China is not available anywhere else. So we have to utilize Alibaba's cloud for that content and then some other content is maybe available only in Germany. So we have to have that at centers that actually are able to wrap the content in Germany. So we don't use proxies for this purpose. We build our own infrastructure that my colleague named Hydra.

Which essentially is overall routing that goes off. It ends up in any of those data centers of the cloud providers but then it gathers the data back to the origin. So if it is running in GCP or it's AWS then if that's where the data will be eventually processed. So we do live with a lot of complexities in the world that we live in which is not necessarily technical. It is more around that content is now doubled in portion of the world or it is not exposed because the creator doesn't want it to.

And if you want to be able to expose that content to everybody on our customer side. We have to be able to process the contents that you have to deal with those things and then we also like to spread the traffic evenly. We had some occurrences of situations where we took down peering centers for our customers or for our customers for the cloud providers when we ran a massive deployment in a single data center. So we've started kind of spreading this evenly.

So if you don't impact anyone along the way and also it's easier for the platforms that have been touched with different CD and notes instead of the one. We don't hammer the one single instance with all the traffic that we sent.

[0:56:57.6] JM: All right, I could keep asking about this cloud providers a lot. Actually I had one weird question I like to ask you. So let's say you had a million dollars in cloud provider credits but you can only spend them on some of the newer higher level managed services, what would you spend that million dollars in credits on?

[0:57:21.0] RT: Higher level, you mean by?

[0:57:23.2] JM: Let's say things like BigQuery or Kinesis, you couldn't just throw this million dollars at infrastructure.

[0:57:32.1] RT: BigQuery is a pretty good one, like that one a lot. As deployment, we tend to build the laws of our own but BigQuery is a very, very good piece of technology and we actually store lots of data there for quick adhoc analysis. I don't know if a billion is that, but you know what? Actually there is a thing, it is essentially now called Apache Beam.

[0:57:56.8] JM: Oh yeah.

[0:57:57.6] RT: It's essentially dataflow so if you utilize that very heavily. Yes we love that piece of technology. It is incredibly powerful because it allows us to spin tens of thousands of servers at any given point to run very quick computations at a mass scale and it's costs us hundreds of dollars, thousands of dollars and is able to attach as much data as we want across all the providers that we have and because of the integrations done within GCP ecosystem it allows us to – there is not much configuration.

And the other one that we quite of grew fond of is Airflow from Airbnb which is now fully hosted on GCP and it's also one of the things that we really like for any TL pipelines of our data analyst that don't necessarily know how to code very deeply. They are able to build pretty significant pipelines on the data after the fact when they want to run things like questions or want to answer

questions like how much content was added to YouTube to a particular category within the last 24 hours and how that compares to Twitch for instance.

[0:59:08.7] JM: Beam is something that slots in like the Apache Beam dataflow slots in for something like Spark, right? Like Apache Spark?

[0:59:19.4] RT: It's called - yeah, we use it on our side it is still going through our own systems but at the end of the day it's called end up in Spark. It's called even end up in BigQuery. So the idea of Apache should be essentially that you can apply any rules on your ATL and that ATL is essentially ran by a trigger.

So if you take a data let's say you have pentabyte of data stored within a Google storage and you want to be able to add plus one to every single dive stem, you can do something like that.

Or you want to aggregate the data across the globe or across the whole data set and you want to be able to expose the data back to maybe your analyst or for instance you want to groom a massive loads of data based on some complicated calculations or in other case, we want to touch binary data which the query came out so we want to touch over fingerprints which I just pointed out is a tensor. We want to be able to run calculations on top of those finger prints and then expose them directly to us so we actually run something what we call a backlog search.

Which allows us to run backwards queries on the whole contents, we are able to take a video that was produced nine years ago and measure it against everything that was produced ever since, against all of the platforms and all of the databases that, or all of the index that we created since.

[1:00:46.8] JM: Okay, interesting answer. I want to wrap up with just one more far flung business question. Do you have any ideas for what people misunderstand about this new world of creators and influencers? Where anybody is making video content or music content, anybody can start from their bedroom and make something creative.

What's your vision for how this world changes the next five or 10 years?

[1:01:15.8] RT: I think that what was kind of addressed within last decade or so, with YouTube and other platforms in place is that the distribution is easy than ever. A lot of creators and influencers that they become influencers over time are now able to do these because the platforms allow them to.

What is missing from their kind of tools - it's everything after that. How they manage their rights, how they enforce their rights, how they even understand the content and their audiences where lots of large creator who write holders mostly on the corporation side like Hollywood studios and others. Are utilizing a lot of deep analysis on the data so they're trying to make a very complex predictions on the content and on the environment and they think what the next five, 10 years will look like of the individuals or at least wmaller creators and influencers, they will get access to this kind of tools and hopefully we'll be able to challenge the incumbents on the scale.

Maybe not necessarily all of them but at least it takes significant portion back to them so it was because the intermediaries take a lot of money out of the table and I think –

[1:02:29.2] JM: The YouTubes.

[1:02:30.2] RT: YouTubes, but also the large rights holders. If you are under MCM. MCM takes portions of your revenue that you generated in exchange for some services that they give you which is not necessarily worth as much or not necessarily for nothing but I believe eventually those tools will be available to all the creators and then once the playing field is leveled.

I think what will happen after that is that we will hold more direct relationship the same way as now the brand are becoming more direct to consumer. I think the creators will be much more connected to bigger audiences while today, it's kind of finicky when they are connected to portion of their audience, it's not actually the whole audience.

I was maybe not even their audience as to people who spread their content is essentially not allowing them to interact with this portion of audience. From other measurements, it's for every single viral video, the audience that they don't see, at least the size that the audience they see.

If a video has a million views on YouTube, there is a chance that it has maybe five somewhere else.

That's the audience that they just don't have access to and I do believe that they will eventually have visibility into that audience and they will be able to take care advantage and expose that relationship which will lead to better content created for the right audience.

[1:03:57.2] JM: Rasty Turek, thank you for coming on Software Engineering Daily, it's been great talking to you.

[1:04:00.5] RT: Thank you for having me.

[END OF INTERVIEW]

[1:04:06.0] JM: If you are building a product for software engineers or you are hiring software engineers, Software Engineering Daily is accepting sponsorships for 2018.

Send me an email, jeff@softwareengineeringdaily.com if you're interested. With 23,000 people listening Monday through Friday and the content being fairly selective for a technical listener, Software Engineering Daily is a great way to reach top engineers. I know that the listeners of Software Engineering Daily are great engineers because I talk to them all the time.

I hear from CTOs, CEOs, Directors of Engineering who listen to the show regularly. I also hear about many newer, hungry software engineers who are looking to level up quickly and prove themselves and to find out more about sponsoring the show, you can send me an email or tell your marketing director to send me an email, jeff@softwareengineeringdaily.com and if you're a listener to the show, thank you so much for supporting it through your audience-ship, that is quite enough but if you're interested in taking your support of the show to the next level then look at sponsoring the show through your company.

Send me an email at jeff@softwareengineeringdaily.com. Thank you.

[END]