EPISODE 462

[INTRODUCTION]

[0:00:00.3] JM: As the gig economy grows, that growth necessitates innovations in the online infrastructure powering these new labor markets. In our previous episodes about Uber, we explored the systems that balance server load and gather geospatial data. In our coverage of Lyft, we studied envoy; the service proxy that standardizes communications and load balancing among services.

In shows about Airbnb, we talked about the data engineering pipeline that powers economic calculations, user studies and everything else that requires a MapReduce. In today's episode, we explore the business and engineering behind another online labor platform, Fiverr.

Fiverr is a marketplace for digital services. On Fiverr, I have purchased podcast editing, logo creation, music lyrics, videos and sales leads. I've found people who will work for cheap and quickly finish a job to my exact specification.

I've discovered visual artists who worked with me to craft a music video for a song that I wrote. Workers on Fiverr post gigs; jobs that they can perform. Most of the workers on Fiverr specialize in knowledge work, like proofreading or gathering sales leads.

The workers are all over the world. I've worked with people from Germany, the Philippines and Africa through Fiverr. Fiverr has become the leader in digital freelancing. The staggering growth of Fiverr's marketplace has put the company in a position similar to an early Amazon. There's room for strategic expansion, but there's also an urgency to improve the infrastructure and secure the market lead that Fiverr has established.

Gil Sheinfeld is the CTO at Fiverr and he joins the show to explain how the teams at Fiverr are organized to fulfill the two goals of strategic creative growth and continuous improvement to the platform.

One engineering topic that we discussed at length was event sourcing. Event sourcing is a pattern for modeling each change to your application as an event. Each event is placed on a pub-sub messaging queue and made available to the different systems within your company. Event sourcing creates a centralized place to listen to all of the changes that are occurring within your company.

For example, you might be working on a service that allows a customer to make a payment to a worker. The payment becomes an event. Several different systems might want to listen for that event. Fiverr needs to call out to a credit card processing system. Fiverr also needs to send an e-mail to the worker and let them know that they have been paid. Fiverr also needs to update internal accounting records.

Event sourcing is useful, because the creator of the event is decoupled from all of the downstream consumers. As the platform engineering team works to build out event sourcing, communications between different service owners will become more efficient.

Zooming out of the technical depth, we also explored engineering management and the lesson that Gil learned working at Amazon's A9 division and several successful startups that he was at prior to joining Fiverr.

This was a great episode about what it's like to run engineering within a rapidly scaling company. Shout out to today's Software Engineering Daily featured open source contributor Nelly Cheboi. She is working on an open source Software Engineering Daily app for Xamarin, which is a great complement to the Software Engineering Daily apps for iOS and Android that we have already built.

In the other podcast players like Overcast or the iTunes podcast player, you can only access the most recent 100 episodes of Software Engineering Daily. These Software Engineering Daily apps have our entire back catalog, it's categorized, we've got a search engine, and they're open sourced at github.com/softwareengineeringdaily.

If you're a power listener to Software Engineering Daily you might like the iOS and Android apps, or now the Xamarin app thanks to Nelly.

Now let's get on with this episode.

[SPONSOR MESSAGE]

[0:04:18.2] JM: Amazon Redshift powers the analytics of your business. Intermix.io powers the analytics of your Redshift. Your dashboards are loading slowly, your queries are getting stuck, your business intelligence tools are choking on data. The problem could be with how you are managing your redshift cluster.

Intermix.io gives you the tools that you need to analyze your Amazon Redshift performance and improve the tool chain of everyone downstream from your data warehouse. The team at Intermix has seen so many redshift clusters, they are confident that they can solve whatever performance issues you are having.

Go to Intermix.io/sedaily to get a 30-day free trial of Intermix. Intermix.io gives you performance analytics for Amazon Redshift. Intermix collects all your redshift logs and makes it easy to figure out what's wrong, so that you can take action, all in a nice intuitive dashboard.

The alternative is doing that yourself; running a bunch of scripts to get your diagnostic data and then figuring out how to visualize and manage it. What a nightmare and a waste of time. Intermix is used by Postmates, Typeform, Udemy and other data teams who need insights to their redshift cluster.

Go to Intermix.io/sedaily to try out your free 30-day trial of Intermix and get your redshift cluster under better analytics. Thanks to Intermix for being a new sponsor of Software Engineering Daily.

[INTERVIEW]

[0:06:03.4] JM: Gil Sheinfeld is the CTO at Fiverr. Gil, welcome to Software Engineering Daily.

[0:06:08.4] GS: Hi, Jeff. It's great to be here.

[0:06:10.7] JM: It's great to have you, because I am a power user of Fiverr. I love it. Fiverr is a marketplace where people get work done. I've gotten podcast episodes edited, I've had explainer videos made for me. I even found someone who sang lyrics on a song that I wrote, and all this was pretty cheap and it was really fun. Describe the vision for Fiverr.

[0:06:37.3] GS: Yes. When you look on the work marketplace, by 2020 43% of Americans are going to be freelancers. Today, only 3% of those freelancers getting the work online, most of them the other 97% getting the work from word of mouth. What Fiverr's mission is to see how we change that and allow those freelancers to get work where they'll spend their time on actually performing the skill that they're good at and would not spend the time on chasing customers, marketing themselves, negotiating a contract, chasing unpaid feeds and so on.

What we've done in Fiverr in order to change the legacy model of freelancing, which is mainly a labor marketplace where people pay by units of time, whether it's by the hour or by the day, or pay projects. We encapsulated a service, a creative service, digital service in what we called a gig.

That gig represent a product, which were called a service as a product, SAP. Our freelancers create those gigs, and say what are they going to do, what's the skill they're going to perform, how much they're going to charge, how long it's going to take, what other extras or other features are available to order as part of their gig.

Those gigs are presented at the Fiverr marketplace as product. At the base of it, Fiverr is a catalog company, so buyers of digital services come to the Fiverr marketplace and they order a product. That product is actually a digital service. They select the options that they want on that gig and they get upfront how much it's going to cost them, when it's going to get deliver, and they can do click-to-buy and have that service performed and delivered to them.

That's quite different than the other freelancing landscape outside of Fiverr. That model and approach allows Fiverrs to be a low-touch, high-volume marketplace for digital services.

[0:09:25.2] JM: My interaction with Fiverr versus the other freelancing marketplaces that I've interacted with has been that the UI and the software design choices have led to a simplified interface. It's much more pleasant to interact within some of the other interfaces at the marketplace interfaces.

I would say that's incredibly important, because you have so many different people who are on the platform looking for work, and then you have so many people on the other side of the marketplace. You've got a very busy and complicated two-sided marketplace simplifying the interface as much as possible to resolve the confusion between those two sides in the marketplace. That's no easy task.

Maybe we can start there. What are some of the design choices? Because we've had these freelancing marketplaces for a while. What are the design choices that has led to Fiverr being such a popular marketplace?

[0:10:27.8] GS: Yes. When we started the Fiverr, it was all about what are you willing to do for $5. A really simple question. Simplicity was always a key tenet for us and still is. Today, it's how at minimum selections and without complicated choices, one can do click-to-buy to order a digital service.

In cases where a service is more complex, we actually put upfront the key questions, whether a buyer needs to send a brief before they get the service as part of their order or a gig, creating a set of toolbars and list that are really simplified and clear so they'll do the selection and they understand what they're getting.

It's really important for us that it will be clear to the seller what the obligations that he'll be obliged to as part of delivering the service, and for the buyer what they're actually going to get. For example, how many revisions?

It's really important for a seller to know that he is now obliged to provide two or three revisions or four revisions. But if you don't have it upfront and clear, you might have a disconnect where if I expect to have as many revision until she believes that the service is perfect. The service I will – I actually only going to do so really, revisions.

Putting things upfront, we believe in minimalistic QI, not over complicating things and have it that unit of gig that presents the service as a product.

[0:12:30.2] JM: I go on Fiverr, I say in the search box I'm looking for podcast editing and I see somebody that offers that service, I click on it, I arrange with that seller how many revisions I am willing to go, or how many revisions they're willing to go through. I set along the price, I set along any sort of extras. Maybe I am going to get a transcript along with my podcast editing, and I get it all done.
It's very simple. I think at this point, we've described the high-level product offering. For people who are unfamiliar with it, you could also go on and get Photoshop done, or get your photos edited, or get sales leads generated, all kinds of interesting tasks. It's really fun creative environment.

This is Software Engineering Daily, we should talk about the software engineering. Give a high-level perspective for the technology stack of Fiverr, then we'll dive in to some of the individual components.

[0:13:31.3] GS: Cool. When Fiverr started, it was built on top of Ruby on Rails monolith. Today, we are at the version four of our architecture. The key principles that we have as part of our architecture and platform, one is microservices. We broke the monolith to a set of microservices, allowing each one of the Fiverr development teams to work independently and that high cadence feature delivery.

Second, CQRS; Command Query Responsibility Segregation, allowing to get high responsive website differentiating the reads from the writes. We also have separation between the frontend and the backend, allowing logic to be written quickly and frontend to be modified and developed independently from the backend.

The last elements that we're adding now as part of the V4 of the architecture is event sourcing, moving from database to event sourcing architecture allowing us, handling larger number of events and moving away from a predefined set of databases.

We have still some leftovers from the legacy monolith's Ruby on Rails, we're leveraging Ruby, Go and Node.js for the backend services, React and the JavaScript for the frontend stack.

[SPONSOR MESSAGE]

[0:15:23.9] JM: Are you a Java developer, a full stack engineer, a product manager or a data analyst? If so, maybe you'd be a good fit at TransferWise. TransferWise makes it cheaper and easier to send money to other countries. It's a simple mission, but since it's about saving people their hard-earned money, it's important.

TransferWise is looking for engineers to join their team. Check out transferwise.com/jobs to see their openings. We've reported on TransferWise in past episodes and I love the company, because they make international payments more efficient.

Last year, TransferWise's VP of Engineering Harsh Sinha came on Software Engineering Daily to discuss how TransferWise works. It was a fascinating discussion. Every month, customers send about 1 billion dollars in 45 currencies to 64 countries on TransferWise. Along the way, there are many engineering challenges. So there's plenty of opportunities for engineers to make their mark.

TransferWise is built by self-sufficient autonomous teams. Each team picks the problems that they want to solve. There's no micromanagement, no one telling you what to do. You can find an autonomous, challenging, rewarding job by going to transferwise.com/jobs.

TransferWise has several open roles in engineering and has offices in London, New York, Tampa, Tallinn, Cherkasy, Budapest and Singapore among other places. Find out more at transferwise.com/jobs.

Thanks to TransferWise for being a new sponsor of Software Engineering Daily. You can check it out by going to transferwise.com/jobs.

[INTERVIEW CONTINUED]

[0:17:15.0] JM: We've done a bunch of shows about event-driven architecture in recent memory. I would love to talk about that for a bit with you. My understanding of the motivation for an event-driven architecture is you often have events that occur across your software that multiple data models want to update in response to.

For example, if a buyer buys something and they purchase it for $15, you have a transaction for $15, and you want to be able to easily update multiple different databases. Maybe you have a search index over past purchases of a user. Maybe you have an analytics database of all of the transaction values over time so that you can quickly and easily aggregate revenue. Maybe you have another database.

The event sourcing model, one thing that makes it really useful is you log that event, you know there has been a purchase for $15. You log it on this immutable queue of events and that queue is a pub-sub system, where that $15 transaction has been published to the queue and you can have event handlers that read from that queue and update each individual database.

This is useful, because it decouples the event creation process from the responses to that event and whoever wants to subscribe to particular events throughout the infrastructure could do that. Then the contract is they just publish their events back to the queue. Am I describing event sourcing accurately?

[0:19:12.8] GS: Yes, you are. Pretty accurate. You have done your homework and have probably sessions with people who are experts at event sourcing. For us, the additional element to what you have described is we do think that dependency on centralized database.

When you look at Fiverr as a marketplace, we have a common set of users, we have a catalog which is built from what they call the gig. We have a large catalog of gigs. We have thousands of new gigs created every day. We have orders where those users actually buy a gig, and each order has a buyer and a seller.

As we grew the Fiverr centralized database became more congested. We have many services that needs access to those areas in the database and also need to augment it with information.

Those services are sometimes independent to one another and we want them to be independent. We don't want to have dependency.

By moving to event-sourcing architecture, each one of those services can create a view of its own on those data streams, can augment them and send them to other services and can leverage the data for that service-specific role.

As example, user provide the rating after in order as being completing. That rating is relevant for the order, it's relevant for the user who rated it, for the user who created a gig who got rated, and we have several services that use that rating either to promote gigs on search or to show that to the user or perform analytics.

Each one looks slightly different on that information. Before we move to event sourcing, all those services actually went to the same database to pull the information, created congestions, which required us to create specific databases or additional databases just to handle those specifics with a lot of dependencies every time we augmented or added information.

With event sourcing, we can do that in a loosely couple and allow each one of the team to move at fast cadence of featured delivery to the marketplace.

[0:22:13.0] JM: When you're breaking up centralized database, or a few centralized databases into more event-driven decoupled architecture, I imagine there are a number of different teams that have to do some refactoring to setup their own domain-specific event sourcing side of things. Do you have a pattern that each team or each database is doing to go through that refactoring, or is it fairly ad hoc from service to service?

[0:22:49.6] GS: We're at the stage where we're still defining and refining our processes. The way that we organized our teams is based on what we called customer journey. We have groups that handled sellers, group that handles buyers, group that handles content, which is mostly algorithmics and search recommendations. We have group that handles mobile and group that handles each one of the specific verticals.

Those groups contains two to three feature developments team, small teams with product manager, development lead, two, three, four engineers and designers and analyst. On parallel to them, we have a platform theme that handles the infrastructures, the API, the communication layers and the foundation of the marketplace.

At the current stage that we are, we're involving the platform team making sure that they are aligned on the change that we want to do, and then they guide the different feature teams as they create a new dataview or create a new event to our event sourcing.

[0:24:21.3] JM: You also mentioned CQRS, which is Command Query Responsibility Segregation. This is the idea of having different patterns that your reads go through to access data, than the writes. Explain why CQRS is relevant to Fiverr's engineering infrastructure?

[0:24:45.1] GS: CQRS allows the separation of reads from the writes. Let's say there is a change in the marketplace, and that change requires several actions to happen. These writes are happening asynchronous to the reads that waiting. There is no pending of frontend or UI that waiting for the backend services.

The way that we implemented CQRS, we created a model that we call the Chimera. Chimera is a two-headed beast from the mythological – from the Greek mythology. We have one head that we call the service, that's actually pulling the data from the database or the dataview and sending it as fast as possible to the frontend. We have what we call the worker, will then perform a task and logic and will write the data back either to the event sourcing queue or to a database.

To answer your question in general, it allows Fiverr to be a site that is very responsive and quick, and that is not pending on backend process and activities.

[0:26:19.7] JM: In recent episodes, we've evaluated some different options for large-scale queueing. There is Kafka, there is Google Cloud pub-sub, there is Amazon Kinesis. What are you using at Fiverr?

[0:26:33.7] GS: We're using Kafka. We evaluated several options at the end. We decided to go with Kafka. We thought that it's the most mature for our needs. We're using Kinesis for some

other data queuing and handling. Or there is where Kinesis and having a manage service is more applicable to us.

The ability to control the space, the ability to replay events and have that on a distributed service that we know that can scale with no performance limitation. The fact that Kafka has been adopted by large portions of the software development community help us to take that decision and go with Kafka.

[0:27:33.2] JM: What are the use cases where you chose to use Kinesis?

[0:27:37.2] GS: We use Kinesis for our data event pipelines that we send to our data lake, or offline data processing, business analysis, business insights. That's I would say the data, pipelines that take events outside of the platform, put them into a data lake that is used only for research and data reporting and analysis. It's outside of the ongoing function of the platform. While we use Kafka for those events that are every action and activity events within the Fiverr marketplace.

[0:28:30.1] JM: Is the choice there based on you want to use Kafka for the more business critical stuff, because it's more widely adopted and you feel it's more tested?

[0:28:40.7] GS: It was clear to us that going with the event sourcing grout is a journey. It's an area where the industry still adopt and changes and Kafka is being evolved. We felt that with Kinesis, we have a good solution for what we need for our event pipeline, for the data event pipelines that help us to reduce and have a low maintenance and low operations, having a managed service from AWS.

For the core event sourcings and activity, we zoom the marketplace and the platform. We believe that overall Kafka is the right selection and that the different options of replaying events at scale by any service or theme to their needs is more applicable for that use.

[0:29:48.6] JM: Right. There's an entire company that has raised almost like a 100 million dollars I think to build Kafka infrastructure, Kafka connectors, the Kafka streams API. Those things might change over time and they're going to update them aggressively. They're going to

be at the leading edge of the creating the best interfaces for event sourcing or CQRS, because Kafka is at the center of that, or your queuing system at the center of that, whatever it is.

As great as Kinesis is, you want to go with the giant open source community project where you just got so many people who have a vested interest in keeping Kafka up to date for that mission critical event sourcing system. You said as a jumping off point into the discussion of managed services that you do use. How unified is your choice of cloud service providers? Are you mostly on AWS?

[0:31:02.6] GS: Yeah. We're not running on AWS. All of the platform services today, we shifted to AWS late last year from other cloud/infrastructure provider. Not to do promotion here for AWS, we're very happy with our selection and it helped us to have higher stability and better performance across the board for our needs.

We use Google BigQuery outside of the platform as a data lake where we felt it's a good solution and a good platform for our data analysis needs.

[0:31:53.3] JM: How hard is it to have a cross platform where you touch other service? Because I've actually heard that from a couple of other people where they say, "Yeah, the only service that we use that is non-Amazon is BigQuery." Is it tricky to get AWS and BigQuery to play nice together?

[0:32:16.2] GS: We didn't have issues with AWS and BigQuery. We actually export events, which we called row events to externally and important into BigQuery. That works well for us. We're also saving them on S3 and using Spark and using Databricks on top of those for some other data analysis, more on the data science side for our needs.

We have a set of options that we optimized space on I would say ease based on purpose and also based on price. There is different pricing model between BigQuery and Databricks for example. For some activities we'll use Databricks, in some activities BigQuery is actually a better selection.

[0:33:20.4] JM: Yeah, the build versus buy question seems pretty interesting here, because I could see Fiverr is a marketplace business. You probably want to try to keep costs down where you can. It's not like a super high – I mean, I don't know what the margins are like, but it seems like the type of company where you do want to actually evaluate your build versus buy a little bit more closely than some company like Facebook or Google or something where they're just selling advertising, and that's quite a high-margin business. You're in quite of a bit different business. How do you look at those build versus buy decisions?

[0:34:05.3] GS: At the core of the company, we're as you said a marketplace company. As such, we focus our development and product team on what we believe is core. The core are those modeling of freelancing services, creative services as product, and exporting them in a such way that will allow our freelancers and the buyer of services to have a great experience.

For example, communication. The communication there are on messenger is a component that we build and we keep inside. We will not outsource it or we will not use a third party too for it, because the element of pricing and clear communication, sharing those digital services are working together on them are really key part of the Fiverr experience.

Another example, search. Search on digital services marketplace is actually a very complex and very interesting area. Just an example to the complexity, when you order on Amazon a product whether one person order it a thousand people order it, it's still the same product and you're going to get the same product assuming that it's available on the Amazon data warehouse, on the Amazon warehouse.

While if you have a digital service provider, which is a freelancer, if he's providing a service one time or five times or 20 times, the service might change. Our search algorithm needs to take into account how occupied or how busy each one of our freelancers, and also look into what quality of service each or one of them provides based on how busy they are.

Those are example for services and technologies that we build in-house and we invest on. Infrastructure services that are open sourced or available outside that are not core to the experience we will not spend a time on building this in-house and we'll use the ones that are already available as open source, or even paid assuming that they justify the call.

[SPONSOR MESSAGE]

[0:37:11.0] JM: Do you have a product that is sold to software engineers? Are you looking to hire software engineers? Become a sponsor of Software Engineering Daily and support the show while getting your company into the ears of 24,000 developers around the world.

Developers listen to Software Engineering Daily to find out about the latest strategies and tools for building software. Send me an e-mail to find out more, jeff@softwareengineeringdaily.com. The sponsors of Software Engineering Daily make this show possible, and I have enjoyed advertising for some of the brands that I personally love using in my software projects.

If you're curious about becoming a sponsor, send me an e-mail, or e-mail your marketing director and tell them that they should send me an e-mail, jeff@softwareengineeringdaily.com.

Thanks as always for listening and supporting the show. Let's get on with the show.

[INTERVIEW CONTINUED]

[0:38:13.5] JM: That search engine where you have to update it to list if a freelancer is busy, so maybe you need to mark a freelancer as busy so that they don't show up in search results, which is more dynamic than ordering couches and, "Okay, Amazon has a stock of 500 couches."

I suppose if they ran out of that specific type of couch, or if they ran out of all couches they would have a similar problem. But this is a regular occurrence within Fiverr. When you have that search problem, can you just alter elastic search, or solar or something to be able to do that for you? Or do you have to build a search engine entirely from scratch?

[0:39:03.8] GS: We didn't build a search engine entirely from scratch, but we're leveraging solar and elastic and using them in a way that is slightly different than other companies. Our search leverage large amount of features beyond relevancy in order to have better performing ranking.

As example, we find out that when a seller is available, he's actually online, there is more likelihood to have a great experience of buying and selling. One of, I would call it boosts that our search algorithm gives is floating up with some volume, those sellers who are available online. In a way that the overall experience of both buyer and seller is actually better. That's an example for one feature that our search engine looks at as we do the ranking of available gigs for search that is different than for example on Amazon.

[0:40:33.3] JM: At Fiverr, you have the two-pizza team model where these different teams have some number of backend and frontend and design and project management, and you have a roughly standardized model of the different characters that are in a given team. Those different teams are managing their own services.

I imagine you have a somewhat standardized model for deployment and service maintenance. I'd like to talk about that. Let's start with the deployable unit. I'm assuming you're deploying in containers of some kind. Maybe you could talk about the container orchestration and deployment process that you have for each service.

[0:41:23.6] GS: Sure. We've started working with Docker about six months ago. We're not in the phase of actually content analyzing our architecture and platform, and we're adapting Kubernetes for container orchestration. Prior to that, we have a set of services and the unit of deployment was a service or independent components that we wrapped as components that available to deploy.

Although we have a centralized integration to send, we use CircleCI, the deployment system, the way that we've built our culture is that each team is empowered to push their own services and called to production. We don't stop them. They are responsible. Each one of those team is responsible for the quality and for the performance of their services.

They will test them before they deploy them. They deploy them and monitor closely that they didn't make any negative impact. They will also monitor the business aspects of that's relevant to the service that they deploy. In cases of things go wrong, they'll roll them back up.

In general, we believe in team independence and team empowerment. Each one of our teams coincides from a development lead and a product manager. The way that we work is rather than telling each one of the teams what to do, those teams act each one as a small startup, where they come to us and tell – when I say us, it's the executive team, and they tell us those are the initiatives and the projects that we want to work on for the coming quarter.

On each one of those projects when they're about to go into execution, they'll come and lay down the plan and the details on what they're going to build and how they're going to build. We act as a board giving them a clear direction and strategy and providing feedback and guidance. But rather than have a model where a small team of executives tell the details to a larger organizations, we have each one of the teams coming to us and tell us what they want to do and how they want to do.

We believe that that allows us to scale faster and have distribution of the brain power and ideation power, rather than small team, a large number of teams actually driving the company. In a way, it's an upside model of executive team telling what the company to do is executive team guiding a large set of portfolio companies. Those teams by the way, we call them task forces. Each one of the Fiverr task forces tell us what they think is the best thing to do.

We provide them guidance. That works really well for us. Each one of those task forces are tasked with making impact. The language in the company, the conversation is about how a team and then how each one of the team members, how a group of teams can actually make larger impact.

[0:45:29.7] JM: Do you have a platform engineering team, like they do at Netflix has a platform engineering team where they build services and APIs and tools that are useful across the different teams within the company?

[0:45:49.4] GS: Yes, we do. The feature teams, or those task forces are busy on making impact, which is mainly business impact product and features impact. The platform team is being freed from that. Their goal is to enable those features team to have higher cadence by providing to them better platform, better tools, clear APIs.

We have a portion of our platform team that focus on improving the performance overall, so having some more holistic view. We also have a dev ops team that's in charge on building the tools for the development team and for running the software production.

[0:46:48.8] JM: With the freedom and responsibility model, you have these different teams and they're looking at their own business metrics, they're looking at what they can improve, they have certain KPIs or some other kind of metric or deliverable that they're going for. That gets you a system of constant decentralized improvement.

Is there also a top-down goal setting or vision setting or mission setting that helps the company grow in significant leaps and bounds, rather than the incremental improvement? I should say that just getting constant incremental improvement on a platform as useful as Fiverr is really high upside, but I know that the company releases new features and new platforms, like Fiverr Pro, which is a professional marketplace, the high-end of Fiverr in contrast to the starting with the $5, these are the things I'm willing to do for $5. This is what I can buy for $5.

You've now expanded into the high-end. There is a culture of being able to experiment and have totally fresh ideas. Talk about how you have those – both of those cultures in place, the consistent constant improvement, paying attention to KPIs and metrics, gradual improvement, but also experimentation in doing completely new things.

[0:48:19.3] GS: Yes. First, in order to have the entire company and the entire production organization focus towards the same goal, we have a topline metric for the company. A topline metric is the North Star. It's that goal that everyone are looking at and measuring what impact is about.

Impact is about growing our topline metric. That topline metric is one for the entire company. Why one? Because if there are multiples, you might not be able to take decisions. If there are conflicts or differences you always go with the options that increases the topline metric. For Fiverr, that metric is happy buyers. Those buyers of digital services and came out from that experience satisfied and that will know they'll come back again and buy again. It's a growth metric that we put as the North Star of the company.

Within that, we have went a look on the portfolio of initiatives and projects that are being run within the company. We have two key flavors. One flavor is the flavor that we call zero to one. The second flavor is growth.

As example to a growth feature, growth activity is tweaking something in our search algorithm running AB test on that and think that that change actually created more happy buyers than the other option, or what was that data control until that. Or changing a place of a line or selection, or changing the flow of a selection on specific page or a window and think that actually increases conversation, which will that means that increases the number of buyers on the platform. We have those type of growth projects running.

The other family is those zero to one. Example of zero to one, and Fiverr Pro is a zero to one project that we have is where it's clear to us that opening Fiverr to a new market segment will create more happy buyers. Those projects of zero to one start small with our product team focus on taking a feature, or a product, perfecting it to a way where the people, the customers who use it are happy with it, and come back and use it again.

Once we reach that point, this is why we call it zero to one. Once we reach that point of product market fit of zero to one project where customers love it and they come back again to them and we actually have a validation for the value for our customers of a feature, then we go from the one to end and apply growth enhancements and high cadence releases to have growth on that feature.

[0:52:25.7] JM: Fiverr is growing really fast, both as a business and as a company. I believe the company raised a lot of money recently, and I'm sure you're hiring like crazy. What's your mental framework for making good engineering decisions when there is a lot of change?

[0:52:46.2] GS: It starts with talent. We didn't event it, but we're key advocates and followers of hiring very high talent into the company, knowing that when you bring really smart people, highly motivated people and you allow them to make an impact, they just love it and they all go behind it and perform really well.

The other I think that we're doing at Fiverr that actually help us with having good engineering decision and good product decisions is we have people that are very senior people, that I call them executive individual contributor. For example, we have two architects in our engineering team. One is focused on overall software architecture, the other one focus more on the way we do frontend.

They are at the top of the level of the industry. They focus not on managing people and not on execution, but on making sure that the decisions that we take are the best decisions. They work with a team, they help and mentor the teams, they go outside to conferences and interact with other companies in order to make sure that we always have the best knowledge on how to do things within the company.

The combination of topline metric and impact on one hand, having a lot of empowerment to the teams to run at high cadence, bringing really strong talent within the company and having people who are at the top of the professional expertise focusing on those expertise across the company help us to have a highly efficient and a super impactful engineering and overall production organization.

[0:55:12.6] JM: You and I talked offline about the other engineering roles you had in the past and you've had quite a wide variety of different roles, leadership, different types of engineering leadership. How does Fiverr compare to the other engineering roles that you've had in the past?

[0:55:32.9] GS: Fiverr is a marketplace. I was with Amazon.com as part of A9.com subsidiary, and we spoke about that. I was the CTO of a mobile advertising technology company that was sold to Singtel.

Fiverr has those trades of Amazon, I would say 15 years ago, even before I joined Amazon where the company is still small enough that each person within our engineering and production organization makes an impact. That create a common stance of accomplishment and mission for every member of the team, and especially for me as the person who helps the Fiverr production organization.

At the Amazon, I was in charge on the advertising platform and I was exposed to a certain elements of the business. With Emobi, we sold software advertising platform to large companies. It was a B2B business.

Consumer business that's growing fast and has the element of changing something in the world, and what we change with Fiverr is allowing people to work differently in a manner that helps them or more suitable to them, means being a freelancer, but freelancers that doesn't need to spend their time on things that are less exciting for them, like chasing customers or chasing getting paid, but focus on their service.

Having that sense of doing good to their ability of delivering great feature, great products with complex engineering gets scaled, and seeing the impact fairly immediately, it is a great sense of accomplishment.

[0:57:52.2] JM: Gil, it's been great talking to you. I love Fiverr. I'll be following the company closely.

[0:57:58.5] GS: Thanks, Jeff. It has been a pleasure. I enjoyed it.

[END OF INTERVIEW]

[0:58:04.0] JM: Spring Framework gives developers an environment for building Cloud-native projects. On December 4th through 7th, SpringOne Platform is coming to San Francisco. SpringOne Platform is a conference where developers congregate to explore the latest technologies in the spring ecosystem and beyond. Speakers at SpringOne Platform include Eric Brewer, who created the CAP theorem, Vaugn Vernon who writes extensively about Domain Driven Design, and many thought leaders in the Spring ecosystem.

SpringOne Platform is the premier conference for those who build, deploy and run Cloud-native software. Software Engineering Daily listeners can sign up with the discount code 'SE Daily 100' and receive a $100 off of a SpringOne Platform conference pass, while also supporting Software Engineering Daily. I will also be at SpringOne reporting on developments in the Cloud-native ecosystem. I would love to see you there and have a discussion with you.

Join me December 4th through 7th at the SpringOne Platform conference and use discount code ‘SE Daily 100’ for a $100 off of your conference pass. That’s S-E Daily 100, all one word for the promo code. Thanks to Pivotal for organizing SpringOne Platform and for sponsoring Software Engineering Daily.

[END]