EPISODE 885

[INTRODUCTION]

[00:00:00] JM: Knowledge base assembles information from a wide variety of sources into a central platform. The most popular knowledge base is Wikipedia, which covers a wide variety of concepts through a system that attempts to remain authoritative and impartial. Other open knowledge platforms include Stack Overflow, which focuses on programming concepts, and Quora, which adds a social element to the process of information accumulation.

Golden.com is a knowledge base that indexes, categorizes and surfaces information. Golden has information about software, and genetics, and world history, and social media, and sports, and all kinds of other subjects. The company monetizes with a paid knowledge base product for enterprises, which allows for easy querying of the open knowledge base and a private knowledge for internal information.

Jude Gomila is the founder of Golden and he joins the show to discuss the process of building a universal knowledge base and the engineering problems that he's working on to improve Golden.

FindCollabs is the company I'm working on. It's a place to find Collaborators and build projects. We recently launched GitHub integrations. It's easier than ever to find collaborators for your open source projects, and if you're looking for someone to start a company with or start a project, FindCollabs has topic rooms that allow you to find other people who are interested in a particular technology so that you can find people who are curious about React, or cryptocurrencies, or Kubernetes, or whatever you want to build with.

Also, if you're looking to start a podcast, Podsheets is an open source podcast hosting platform that we recently launched. We're building Podsheets with the learnings from Software Engineering Daily, and our goal is to be the best place to host and monetize your podcast. If you've been thinking about starting a podcast, check out podsheets.com.

[SPONSOR MESSAGE]

[00:02:10] JM: Mux is an API for video. Mux makes beautiful video possible for every development team. Post a video, get back a video URL that plays back on any device in just seconds. Video is not easy to manage. There are bit rate ladders and CDN decisions. The decisions about how to serve your video depend on the desired quality and reliability and speed. Mux allows teams of all sizes to get up and running in minutes, and live video is just as easy.

With a single API call, you can get an endpoint where you can push a live video stream and you get an ID to play back stream. Whether you have a specific idea for a video application or you just want to tinker and get creative, Mux is a tool that makes amazing video experiences simple. You can listen back to our previous episodes with the Mux team to find out why video is a complex engineering problem.

Mux was founded by experts in online video, including the creators of the biggest open source video player on the web, which is Video.js. Mux is trusted by leading providers of online video, like CBS, and PBS and Vimeo. You can sign up for a free account at mux.com and get $20 in free credit to get started. That's mux.com. You got to love the three letter domain name. Mux.com, get your free $20 in credit, and make something cool using video.

[INTERVIEW]

[00:04:00] JM: Jude Gomila, welcome to Software Engineering Daily.

[00:04:02] JG: How is it going? How are you today?

[00:04:05] JM: It's going great. How are you?

[00:04:06] JG: Really good actually. We just launched the pay product for Golden and just looking at all the kind of responses on Twitter and looking forward to the weekends a little bit.

[00:04:16] JM: Indeed. Well, to explain what Golden is, I'd like to start by discussing the category of knowledge base with you. There's a type of platform on the internet that is often

called a knowledge base, and the most widely known example is Wikipedia. What do internet uses want out of a knowledge base?

[00:04:37] JG: I think it's a very wide range of use cases. Some users are looking for Ph.D. level explanations of pretty technical subjects, like the Riemann hypothesis or [inaudible 00:04:50] black hole space time. Some users are looking for a quicker explanation of what something is. It may just maybe they want to look up a particular location in Nigeria or they want to quickly find out about a person and understand what this person does.

I think there's a lens of ranges of applications from someone looking at something very quickly to someone doing very deep knowledge research where they're trying to find connections to academic papers and further reading. So that's an interesting dimension.

Then the other dimension is the different types of things that people want to find. Some people want to find information about academic topics, concepts, technologies, concepts and philosophy. Some want to find information about a company, or a person in the business, or a person in science, or location, or historical event, or a current event. So the number of types of things as well is really quite rich.

[00:05:44] JM: What are the shortcomings of Wikipedia?

[00:05:47] JG: Yeah. So Wikipedia is an interesting one. One of the best things that has even been built by humans, it is currently around 18-years-old, and that in the technology space and the software space is actually a very long time. So if we look back at the genesis of Wikipedia and some of the choices that would – Probably the correct choices made at the time. If we look back at the encyclopedia itself, it had limited shelf space on which you could load up your encyclopedia in your house, and then we moved over to like a CD [inaudible 00:06:16] and then multiple CDs and then a DVD and then multiple DVDs. Then we jumped over to the web because that made them parallel. Then we had limited storage. We had limited storage. We had limited processing capabilities. We had a lack of functional plug and play AI systems that were productized. Communities were very early. Real identity was missing. It's pre-Facebook. It's pre-Twitter. It's pre-GitHub. It's pre-YouTube.

So when we look back, we're looking at a system that was designed before many interesting components had been figured on the web and in technology in general. So one of the main ones that that led to is notability. So Wikipedia has famously got a notability requirement for a topic being loaded up to it. In our mind, this has been – It's a fairly arbitrary threshold. So something that is notable to one group of people is different from what is notable to another group of people.

Chris [inaudible 00:07:14] 12E might be notable to a bunch of biology researches, but may not be that notable to sports fans. This leads to inconsistencies in policies of how to enforce what to write about. So we've got rid of that and we've said, "Okay. Well, what can we validate? What can we find out? Whether it's true or not, whether it exists or not, even if it's only appealing to a very niche set of people."

That to me opens up a huge range, around 1000x the topic space that you can go map out, and Wikipedia today has about 6 million articles in English and Google Knowledge Graph has probably around 10 billion entities. So we're looking at 1000x, 1000 fold the topic space that we could go after. So that's answer 1 or 20 to things that could be considered deficiencies in the current best answer to human canonical knowledge.

[00:08:06] JM: There are general knowledge bases, like Wikipedia. There are also domain-specific knowledge bases. Stack Overflow is a domain specific knowledge base for programming. What have you learned from Stack Overflow?

[00:08:22] JG: So Stack Overflow, what's interesting there is the format, it's slightly different and it's kind of a question and answer format. That serves a very important purpose. If you look at Stack Overflow and Quora, there is a kind of question and answer format there. If someone's looking for something very specific and someone's giving the best quality answer and there's a series of answers around that.

I think that has a very important place on the web, and we would love to point out to these specific answers that might be related to the abstract topic. But sometimes you don't start necessarily with a question. You just start with an area that you want to read about and start to understand questions, start to understand the kind of questions that you want to ask. So, I see it

as something that is a necessary component of the full landscape of the information knowledge and a very useful one.

[00:09:08] JM: I used to be a power user of Quora. Quora merged Wikipedia with social elements. What was innovative about Quora? Do you take any inspiration from it?

[00:09:18] JG: Yeah, there are some learnings as well. So, stack exchange. The model, they got subdomains per area. So they're very focused communities that don't necessarily have – That are not connected super well together. I'm sure multiple people on multiple communities are. For Quora, they haven't subdomains or split up these niche communities, and it's more continuous, so you can be browsing and reading up on physics and then you see a question about how to handle relationships.

So I think that delta there of having split communities versus a continuous community is an important one, but also the mechanics of – In Quora, it seems to be more focused on the real identity component and trying to get – And sometimes that leads to issues where the best answer may not necessarily get surfaced, because it was the most famous person answering something, and that got voted to the top because of their fame. Stack exchange, more so, it seems scared to try and get the best answer for the question.

I think there are different topic areas that they started with. Quora started with a lot of startup VC mechanics that Silicon Valley kind of questions. That was a starting space for them. Stack exchange, programming in more technical areas, and that's led to amalgamations of certain communities around those two web properties.

[00:10:33] JM: How would you describe Golden, which is the company you're working on?

[00:10:37] JG: Golden, for me, is interesting. When you're trying to make something that's quite complex and quite wide-ranging, the descriptors that you end up trying to put together in your head don't become sufficient. They don't sufficient to tell you in a nutshell really necessarily what I wanted to be.

But today, one element would be could we build something that could be bigger than Wikipedia that could cover more topics. So 1000x of topic range, do it, have a deep scheme around each topic. So have timelines around topics. Have video all connecting media around that topic and other aspects that aren't necessarily encapsulated by Wikipedia schema. The other components is then could we make this queriable so we could query over the dataset all integrated into one.

I see with Wikipedia, there are various fragmented pieces, like Wiki species is separate, and there are pros and cons of doing it. So to go verticalized, maybe you can have special features that just work for that dataset. So we're trying to have an integrated single product that is very well-integrated and is very easy to use and solve some of the aspects of, say, the lensing problem where Ph.D. wants to come into the site and read the deep technical information, or someone wants to come in and get a light version of it.

Right now, you've got the simplified English Wikipedia version. There's a lot of fragmentation over the entire product on that side, and then it feels like it's missing the kind of Google Knowledge Graph aspects that could be plugged into it to really run the kind of knowledge graph in a sense. So, I'm looking for a really comprehensive one stop shop where you can look up a topic and find everything around that topic, including all the great core questions and stack exchange answers, but also the best YouTube videos around a topic, the best lectures, the best pet podcast and documentaries, and really dig in to have full pros around it, custom tables that describe useful information around a topic.

So I'm looking for this higher level kind if unicorn not in the sense at a financial unicorn, but the unicorn product that would be amazing to go and use and then have APIs off the back of it. Use AI to get rid of a lot of the boring work that humans have to do to clean up things and spelling and grammar checks and cleaning checks and making sure that the links don't break and link rot is managed and references are backed up and we don't have to necessary rely on will reference be backed or not and just making sure that's very comprehensive, and that's really a long march to achieve that. I think we've done a recent amount of work in the last two years and we've live for – We launched formally a few months ago. So if we continue at that pace and then keep improving, it should go somewhat towards that vision.

[00:13:24] JM: What you're saying there about the willingness to aggregate information from other general knowledge bases, you don't need to be territorial about having things in your personal golden format. You're willing to link to things like Wikipedia and link to things like Stack Overflow and leverage the knowledge bases that have been built before. That's a useful idea, because it becomes more of an aggregator rather than these other platforms where all of the knowledge seems to be built within those platforms, like the Stack Overflows or the Quoras. How do you assemble all the information for all these different topics?

[00:14:05] JG: Sure. There's a combination of human effort and AI effort and AI tools to help humans as well. So, some of the techniques we have, sometimes the AI is running by itself and adding information to the site and we're crawling the web in a targeted way and we're calling news and we're trying to extract facts from pros out in the web and out in news. Also, trying to bring these tools, these suggestions, but we're not completely sure what decision to be whether we should make the decision to include the information or not, refer this out to humans to basically sign off on the AI suggestions.

So there's human in the loop aspect to this as well where they can sit there and click yes or no or skip on UI to actually incorporate the information. Then there's a fully – Then you can go fully manual if you want and write up a page, and we [inaudible 00:14:59] so that's quite easy to use, whether a user can be in there and use things like keyboard commands and have AI helping them making suggestions as they go along. There's a lot of work to be done on that beside the product. We're not done at all on making that really good experience yet. It's pretty easy to use right now, and we want to get the leverage up so that the AIs making little suggestions to your work. It's giving feedback. When you're making a publish it's telling you, "Hey, you could've written this in the third person." "Hey, you're using some marketing buzzword language here," and trying to give you recommendations on how you can fix it.

Then to your point about pointing to other information, I think it's fine to reuse what's out there on the web. I mean, there's kind of two parts to this. One is there is value in compiling information into one place, one UI, one consistent UI standard, because it helps users flick between different topics and have a regular format. But there is also value to pointing to other formats. So if someone's done questions and answers really well, like Quora and Stack

Exchange, then point out to the questions and answers. If someone's done video really well, it will point out to the best video there.

So, for various parts of the data, like some – There are already existing sites that do this really well. But it's really the accumulation of everything around the topic model. So this is a topic model where we're trying to pull the information around a topic, and whether it'd be a video, question, academic paper. The pros we're pulling into Golden and the structured data, like the info box on the site. On the right hand side, there's an info box that displays kind of quick facts about the topic. That stuff we're pulling in, that's the part that's consistent. For other components where other websites have already mastered the UI around that data type and we're pointing out to it.

[00:16:49] JM: Describe how humans and machines play a role in aggregating this information.

[00:16:58] JG: Yeah. So, AI right now, there are various problems that we can solve that where we can fully automatic. But there are various times it can't get the best answer. So it's getting it right 70% of the time or 60% of the time or 80% of the time, and this is where it needs training data for someone else to signoff who's doing a better job. Say, humans doing 99.9% correct, just sign off on it. You can have multiple sign off from it. So you can have a voting system behind it where multiple people are voting on trying to help the AI label the stuff correctly. This is then going into a feedback look where this specific algorithm can learn from a better answer and then make its own attempt again at predicting the right answer. Then it can go through another cycle of learning again.

So there are interesting – This is kind of piggybacking one intelligence into another intelligence. It's really taking what humans can do very naturally and trying to get it into an algorithm that's repeatable. What we've tried to do is be very modular in solving the problems that underpin this mission, and not all the problems are solved yet, for sure. Open AI had done an amazing research in writing full prose, and that's a brand new arena to try and write full prose for a topic. There are lots of other components, like detecting what something is.

So, detecting something in the first place. So you see a news article and it's talking about some new kind of company, Recursion Pharmaceutical or something, and then so does detecting that

in the news that this is a new thing. So that's an AI problem. Then you can also – A human could have done it. They could have seen it in news and made a topic page about it. Or you can have an AI say, "Hey, we think we've got something here, Recursion Pharmaceuticals. Do you want to create a topic about this?"

Then eventually, it's going to get good enough to do it itself at a certain failure rate, where it's a better use of a human's time to just correct things that are wrong rather than signing off on things are predictions. Different algorithms are further ahead than others in both the academic literature in other kind of companies that are implementing these and aren't internal versions of them. So some will be fully automatic. Some will be semi-automatic, and some maybe kind of manual for a while, and we're going to just try and push all of them forward as far as we can.

[SPONSOR MESSAGE]

[00:19:32] JM: When you start a business, you don't have much revenue. There isn't much accounting to manage, but as your business grows, your number of customers grows. It becomes harder to track your numbers. If you don't know your numbers, you don't know your business.

NetSuite is a cloud business system that saves you time and gets you organized. As your business grows, you need to start doing invoicing, and accounting, and customer relationship management. NetSuite is a complete business management software platform that handles sales, financing, and accounting, and orders, and HR. NetSuite gives you visibility into your business, helping you to control and grow your business.

NetSuite is offering a free guide, 7-key strategies to grow your profits at netsuite.com/sedaily. That's netsuite.com/sedaily. You can get a free guide on the 7-key strategies to grow your profits.

As your business grows, it can feel overwhelming. I know this from this experience. You have too many systems to manage. You've got spreadsheets, and accounting documents, and invoices, and many other things. That's why NetSuite brings these different business systems

[INTERVIEW CONTINUED]

[00:21:24] JM: Do you use any tools for scraping the internet? Like there's a tool called Diffbot that's pretty useful for scraping the internet, but there's also just manual scraping tools you could use. How do you scrape the internet for information?

[00:21:40] JG: Yeah. So we've looked at the various tools out there like Diffbot and we ended up building out some of our own software here. The exact way we do it don't really necessarily go into that much, because it's kind of proprietary information. But there's a few components with this, doing it in a kind of a friendly way respecting the kind of TOS of the sites out there doing it in areas that actively want their information put out there and then giving them attribution when necessary, when they need it and when they're requesting it.

So I think that's one component, and it's kind of interesting that the scraping kind has a bad connotation with it. But if you think about the main tool, we all use Google every single day. That is like crawling the web. Calling is one way to call it. Scraping is another way to call it. We don't do anything where we can't, we shouldn't be taking the information away and it's someone else's proprietary dataset. We're more interested in looking at holistic documents out there and trying to extract facts rather than wholesale pulling datasets and wholesale pulling prose. It's not a unique content when you do that.

So the algorithms that are interesting to us is like, "Okay, when looking at a webpage, what are the important things, what are the facts embedded in this document of this URL?" and extracting that kind of stuff, or, "Hey, there's an event that occurred in this new story. Can we summarize the event in a single paragraph and write our own prose around that to summarize it?"

So, to us, it's really crawling the web and extracting information and trying to compile it and trying to put it in a format that's useful for people and machines. You can't really copyright facts. So there are kind of interesting things here with their usage and you can't really copyright facts

that's embedded inside text. A lot of the kind of modern summarization and extraction is actually quite in line with our current society's views on IP. So I think we're quite good there.

[00:23:38] JM: As you crawling the web and assembling this knowledge base programmatically, you have materialized view of that information that you have crawled. You have summaries and topic aggregation of things that you've crawled. Then you have some degree of human in the loop for reviewing that content for making sure it – I'm assuming, making sure it looks cogent. Making sure it's accurate. Can you describe how human in the loop fits into your knowledge base development?

[00:24:15] JG: Sure. There's a feature called suggestions on the interface, and everyone should go and have a play with it. From there, we're making small modular recommendations which might be, "Hey, we found an image for this topic. Should we use this image for, say, a topic that has no image yet? Has no thumbnail?" and a user can click yes, no, or skip.

It's a very simple model where each time we're learning off the back of that and also the users making progress, because they sit there, making edits. Also, some people are very specialized and that they may be really good at spelling and grammar checking or they may really enjoy selecting images for topics. So it allows people to customize what types of suggestions they're going to be answering, and it's very mobile friends as well. So, more than 50% of our users are using mobile web to actually get to the site, and that's a very natural use case. But editing on mobile, it's classically being almost impossible not just due to technical limitations of what you can do in these editors, but just the UI space and the constraints of finger sizes, to texts, to complexity of what you can display and the resolution that you can see with your eye.

When you have all these constraints, it's really hard to edit things in a kind of natural language way, like a kind of Word document way or a spreadsheet inside on a mobile phone. But what you can do is make a simple yes, no skip system where the user can take bite size pieces to edit and do this on the move, and it also plays into the attention economy and that there's so many things that we could be using right now. So many websites we could be on. So maybe you don't have that much time anymore to do like 15 hours of editing and you've only got a couple of – You got a couple of minutes between the next meeting, and yeah, you can just jump in, make a couple of edits.

So we're trying to make it fun to edit. We're trying to open up editing to our wider set of people. Also, still allow for the most deep, complex writing in a web editor, and that part is an innovation I think in terms of collecting the information. We've seen in Mechanical Turk. Mechanical Turk is used to clean up information. You kind of have to build your own interface in a sense, but you've got access to people out there.

So, this is really channeling these recommendations that the AI is making, in different module AIs. This is called an algorithm's making into allowing some kind of intelligence on the other side to sign off on it and for the algorithm to get better every time.

[00:26:40] JM: Does crowdsourcing still work? Because I kind of feel like there was this phase in the development of the internet where people were very, very willing to crowd source information. We're willing to contribute information because these gamified experience was kind of fun. I contributed a lot of stuff to Quora, and then overtime I kind of realized, "You know, I'd rather just contribute. If I'm going to write, I'm just going to contribute to my own personal brand, my own platform, or to Twitter." But Izguess even if I'm contributing to Twitter, that's basically crowd-sourcing Twitter. I guess I'm wondering, does crowd-sourcing still work as well as it always has?

[00:27:20] JG: Yeah, it's a good question. I'm not fairly sure if it is. So there're a few things on that. One, the users – Platforms have evolved to give more back to the user for the amount of time you're spending. For Twitter, it's associating it with building up your own brand. So things have become more user-centric in accumulation of the value a user puts into stuff. So there's that general trend.

Also, people, there are so many things that you could be doing with your time now, right? There're so many different apps out there. Users have definitely possibly been exhausted by various game mechanics. They've been overexposed to in mobile gaming and all the apps, and that's not really fresh anymore.

But there are some persistent incentives that people still want to get their projects covered. They want to get their companies talked about. Maybe they want to get their profile covered as

well. So there're still incentives there. Then the other component here is that we have a business model where companies can deeply query the information and connect to it by APIs, and this means that we have got finance to go and pay for also paid helpers that can go and collect the data, do research. That is one of the ways to accelerate the collection of the information, is not necessarily to wait. Also, yeah, it's a good point that crowd-sourcing is possibly quite difficult nowadays when there are so many places that a user can be.

But one of the other parts to that is like bring leverage. This is about a ratio of incentives to leverage to friction, and if the friction is really low and the leverage is very high and the user gets something good out of it, then it's going to be a good equation for them. So I think AI editing brings high-leverage to the user and they cannot – Then lowering the friction by having [inaudible 00:29:11] editor. That is also a way to like rebalance the equation.

Also, we even call very real identity into Golden, and that means that you can actually attach a social [inaudible 00:29:21] to your work. So if someone can look at the profile and say, "Okay. Well, this person knows a lot about these particular biotech topics." You can see they're kind of writing. They've done on the topics and this writing has stood the test of time.

Wikipedia is classically been anonymous, pseudo-anonymous, sometimes real identity. But that is missing out on an opportunity to add to your personal brand. So by bringing in the real identity that is useful for people to build up a profile as an expert in a field.

[00:29:53] JM: Let's talk more about the data engineering and how you're building the schema that is Golden. I respect that you want to keep the priority bits priority, but can you just describe the data engineering pipeline in a little bit more detail?

[00:30:10] JG: Yeah. I'm going to [inaudible 00:30:11] the kind of detail that you're looking. So we'll see what we can try. I mean, there's all sorts of data coming into Golden, right? So we're looking at specific URLs. If you made a topic page on your website and tried to build a canonical page around that, as soon as you stop putting URLs into the info boxes and into the topic page, we're starting to hit these URLs and fairly try and get extra information, the Twitter account of your particular website. Maybe some people that may be involved in it that maybe on your team page. Maybe we're trying to summarize and produce a description from your about page.

So there're various aspects of targeted crawling there that occurs to try and get relevant information over to building up a picture of that topic. So there are things that can occur instantaneously, right? There are algorithms that are fast enough to go make a suggestion immediately. Some things need to go into a cron job overnight that will sit there and crunch through.

Then there's stuff that we've already precompiled. Then there are other interesting ones. So as our algorithm sit there nighttime to look for duplicates. There might be a duplicate that's gone into the website, and that stuff can run on a daily cycle. Then there are things to look for inference. You may pulled in one fact and you pulled in another fact that allows you to get to something else and infer something else and make a suggestion.

So there's kind of like a little chain reaction that occurs with every time we get any URLs associated with these topics. We get a more visibility involved. Instead of crawling every single website that exists, say, in a classic sense of it on all such engine, we're trying to build out more organically from these URLs and pull them then. Then they have to have particular queues. There's a queue system for it, and then there are various different technologies around that to make sure that this stuff can be handled well.

We've actually got thousands of suggestions waiting to go that need human signoff, but there's also the automatic AI as well, which is making exchanges. Then we've also got import projects. So we're actually specifically trying to – So we're specifically trying to fill out various data areas and we've built a whole templating system, for example, around an entity.

So if you set an entity to be a company, it comes with a bunch of info box properties, the location, the CEO, founder date, inc. name, tagline, business model, and each one of those has a type and we try to be careful about the types and the schema around each and one of these entities and we're extending that all the time.

Eventually we'll build some algorithms to also do schema suggestions to say, "Hey, you're missing this. We've detected this property of this entity and we want to make a schema change." That's a pretty high-level change. For now, that's being architected. We're architecting

that carefully, and we looked at the Wikipedia schemas. We looked at schema.org and various other schemas that are being set there.

So there's a schema design. There are types of things. Even when you're setting a date, should it be the exact day? Should it be a fuzzy date? Should it just be a year and should it have a time range? If a CEO was a CEO of a company, it needs a time range. So there's very subtle ways to store this information and then making it communicate to each other is, or making it consistent holistic is interesting like problems of kind of data architecture. So I hope that answers a little bit about it. I'm not necessarily giving you some exact commercial detail.

[00:33:50] JM: No problem. Can you give me some products that you use perhaps, like your software architecture, like cloud providers and streaming frameworks and machine learning systems? Any kind of lower level infrastructure that you're using?

[00:34:04] JG: So, to be very vague, we're using AWS right now and we're using a bunch of the AWS products, and that's mainly for infrastructure. I've also been weary about whether giving out full details where there's much value for us like as a company to do that. But we've been experimenting with different AWS products there. That's probably mostly because the team is fairly used to the AWS products and it makes sense to try and use different components there. But yeah, we haven't given out details on necessarily any of the specific tooling that we're using to make this happen. It is a competitive market, right? There are other companies that want to also own the space. But, yeah, that's my vague answer for you.

[00:34:46] JM: No problem. Fair enough. How do you evaluate build versus buy? Because in these days, there are so many good cloud tools that you could buy off-the-shelf and build something really, really cool with. But if you're building something data-intensive, like a knowledge base, I can imagine there would be some applications that you might need to build specifically. So I don't know if you can talk about that philosophically perhaps.

[00:35:13] JG: Yeah. We recently had one of those decisions, and it sounds like a very big company type thing, like build versus buy. Really, actually, startups face it all the time. They just don't face the decision probably very correctly, and either they go and buy what they should have built or they build what they should have bought.

So we had a very specific at this the other day, and even though it was a very small cost in dull terms, it was only like a 5k piece of software one-off. We really had like a reasonable discussion about whether we would do it or not. It kind of affected the philosophy of the company, and there're all these like generic little things that people say of like, "Is it a core competency?" "No, shouldn't build your payroll software [inaudible 00:35:54]." "No. We shouldn't built the Cap Table Management software. Let's use Carta." "Hey, we shouldn't build our discord or Google Hangouts, even though we've had a couple of technical issues with them."

But when it comes to some of the NLP stuff, there are interesting packages out there when it comes to some of the startups. I mean, there are various kind of frameworks there where if you're a startup and you partner with a startup, that's usually a pretty shaky ground. But if you have high-velocity, you may change direction anytime. So if you're buying something off a startup which is going to be moving around a lot and you're a startup, then maybe you should have not necessary bought that.

So, one of the conditions is like is what you're buying going to radically change or give up or like shut down. Are you going to waste a bunch of time? The other part is that how much you need to customize it. So, if it's part of your core offering, or the core innovation, defining the boundary of the core competency is quite difficult, and defining what your core competency is when you're in a product market fit searching mode is also difficult as well. So I think it's okay to buy things – Say you're doing do some crawling. It makes sense to go test out Diffbot. I think part of this, you go test it out and say, "Okay. Well, do I need to customize way beyond this? Okay. I'm going to have to build it in-house."

If not, can I get away with doing prototype code for this to try and get away with it and maybe we'll switch over and buy it later. You may not be able to buy some software until you get your business model up and running. It may be too expensive, or it may be something that you're going to buy now but build later or build now and then buy later.

So it's really about looking at the build buy decision as well. It's like a roadmap, long-term roadmap decisions, because it could be build-by-build or build prototype code, then we're going to buy it, then actually we're going to end up building it again. So it's not necessarily a binary

switch as well. It's about what is the philosophy of the company. What areas do you want to own? What areas do you want to get really good at? Do you want to be really good at the AI component? Do you really want to be good at the UI component? You can't be good at necessarily everything. Then there's the question of how many plates do you want to spin on your technical stack if you're going to go and build all the pieces yourself?

So I think there's an art form to that, and I don't think there's any easy rules to say, "Oh, just focus on your core competencies." It's like, "Well what are my core competencies?" So I would love to like hear more thinking on this and if anyone can – When this goes out on Twitter, people can put it in the discussion, like links to defining what your core competency are. I think that's the hard boundary here to define.

[00:38:22] JM: All right. What's the business model for Golden?

[00:38:25] JG: Sure. Currently, we are offering a query tool for companies in organizations, and that's on seated SaaS model where currently $99 per month. So all the topic pages open it for you, and that's the really important thing. That's the main reason that I'm working on this, is to get an open website with all the topics on there.

But one of our pictures is that this is a company, and we believe a company can make a better product and a company needs to have a business model. The more money we make, the more we can invest in a software. So the actual query tool is a very powerful query tool where you can query any kind of entity inside Golden. Set up various filters. Fine grain alerts. There's also an API as well so you can go into enterprise modes and you can pull the data at which might be enriching your Salesforce or Affinity. There's integrations as well coming where you can actually say, "Okay. Well, I want to directly integrate that into my CRM."

You might be an insurance company and want to pull data on people, management team that you're about to back the company off, ensure the company off, or the company itself, or you need some location information about where this particular company is based and to feed your insurance model.

So we're using the use case of – I mean, the values is the data and how to access it for corporates and for organizations via like deep querying, advanced querying, APIs, integrations. So that's product level one, and I think there probably will be other paid products coming up from Golden, because we think that it's going to be very valuable to also apply these algorithms that we've developed to private knowledge. That's probably going to be the second phase for us where we start using the algorithms we developed to help shape your internal private knowledge. Then it's this connection between public and private knowledge. So if you've collected all the private knowledge in your organization and organized it really well and it's self-healing and you also want to keep it in-sync with what's happening in the public as well to make sure you're on point and your private data is also correct.

Hopefully, then there's this next fore phase where – And we saw this kind of with GitHub, where you had open source repos. You had closed repos, and people [inaudible 00:40:35] open repos into closed and they also took private repos open source. So I'm hoping as well that, overtime, in mapping also the private knowledge companies, we will come back full circle to the vision and provide a channel and affordances and incentives for companies to open up their information and not just open up code, but open up their knowledge. That happens for academic papers. It happens through open source code. It happens for blog posts, but I'd love to see it happen canonical information as well get opened up to the public.

So, I think having them next to each other, there are some people that say, "Oh! This is really bad there's a business model attached to it." Your incentives are all going to be wrong. Actually, the incentives to be correct on the data level are very, very strong, because a company will not buy the information if it's wrong. If a company trades on the information that's wrong, then that's not good for Golden if we get it wrong. So we got to get the information right. I'd argue, we have even more of an incentive to get the information correct on the public side, and by putting the two things next to each other, we are opening up a door for companies in the future to start open sourcing their information and making the open side richer than we could have every possibly imagined previously.

[SPONSOR MESSAGE]

[00:42:01] JM: My favorite way to hire developers is contract to hire. In the contract to hire model, you pay a contractor to work with you on your company, and if you enjoy working with them, you hire them fulltime. Compare this to the traditional hiring model of bringing and engineering candidate in for several rounds of interviews. Instead of spending lots of time crafting questions to be done in front of a whiteboard, contract to hire lets you actually work with the engineer on a real project and it compensates that engineer for their time. It's always surprised me that contract to hire is not more widely used, but part of the reason for that is that there hasn't been a great platform that encourages contract to hire.

Today, that has changed. Moonlight is a platform for hiring high-quality software developers from all over the world to work on your project or your company. You can hire part-time or fulltime developers. If you enjoy working with them, you are free to bring them on to your team.

There are no lock-in effects. Moonlight does not charge you a finder's fee. They will work with you to make sure you and your developers that you hire are all happy. Go to moonlightwork.com/sedaily and get 50% off your company's first month of hiring access. That's moonlightwork.com/sedaily. If you're a developer who's looking for a remote work and community, check out moonlightwork.com to join for free.

I'm a customer of Moonlight and I'm also a small investor. I love the platform and I'm actually amazed that something like it did not come out sooner. Check out moonlightwork.com/sedaily and get 50% off your first month of hiring access today. That's moonlightwork.com/sedaily. Again, full disclosure, I'm a small investor in the company, but I'm also a customer and I love what they're doing.

[INTERVIEW CONTINUED]

[00:44:10] JM: When I think about a problem with so much data that you could be aggregating, so much scraping you could be doing. Cost controls seem relevant to me. Do you have any strategies around how you control costs for data engineering and data aggregation?

[00:44:32] JG: Yeah. This is a good one. So we have some ideas here. So there are data points out there that are clearly economically viable to collect. So if you collected the post-money

evaluation of the company, that would be useful information for some other company to pay you for that data point. As you add more of the data points to your network of data, then your entire network becomes stronger, because you've got relative data, like the post-money evaluation relative to something else, some other attribute.

The more data you collect together, the more the per unit value goes up of these data points. Then there are topic areas where they're uneconomic currently to go collect information. At least companies necessarily wouldn't care about them, but you users care about them. The users – I just saw a particular type of black hole and I was no Wikipedia reading about the free body mathematical problem and I saw some section in there and I saw a link there and I clicked on it and there was no page on this particular type of black hole.

So I started writing a version on Golden about this black hole. I'm going to try and find someone over the weekends to fill out who knows a lot more about it than me. So that information, that esoteric information may not be very commercially viable. But it's still really interesting to the community, and there're indirect effects on the economics of collecting their information. So I argue we should go and get that information. We should cover all of physics, all of math, all of philosophy. Maybe Wikipedia will be the best place for historical information, and we'll point to that.

Then there's information kind of outside of this band, like there is a [inaudible 00:46:13] on my desk right now in front of the screen or this podcast and it's not that interesting to go and make a page about that pen. It doesn't really have any commercial value. It doesn't really have any economic value to anyone. No one cares about it, and it becomes difficult to validate as well when no one cares about the information.

So, there is definitely a boundary where it probably will never be valuable information for me to get a page on that and keep it – There's a cost associated to keeping it valid. Storage cost is going to zero. The cost of keeping it valid is also going to zero, but it's not zero. Both of them are not zero. So, there is a cost to pages, but we want to go quite far on this and try and cover everything, where people are interested in it. I think there are other types of graphs out there. There are people building location graphs, where the economics makes sense to be focused around location graph. There are people building information on statistical datasets and the

economics – When you package it together, the economics get better. So the economics can work for them.

So we do have – It's a soft boundary of the 10 billion entities and hopefully the economics get better overtime, because the cost goes down, the validation cost goes down. The number of customers you have go up. The number of inferences you can make from a data and the number of other data points go up. So the economics get better and the boundary starts moving outwards. It doesn't move outwards at a greater and greater speed. Well, it doesn't like necessarily go to infinity, right? Otherwise you'd be simulating the universe inside the universe by mapping every single atom and subatomic particle and labeling everything that possibly exists. So that's not quite what we're trying to do. We're trying to get the topics that you really expect, and very niche scientific topics.

A lot of these startups had it in cycles. There is a cycle, there is a kind of VC cycle, there's an IPO cycle, there's a business cycle, a general business cycle 7 to 10 years. There are founders, lives, cycles as well. If you look at a lot of the projects out there, they tend to fit in these market cycles. So you really have to think about – I mean, Joshua Schechter, who founded Delicious, told me this that really – I'm not probably correctly paraphrasing, but he talked about you really got to think about this company, your company as like a 5 to 10-year kind of experience at least like in its product form, because all the dynamics is going to completely be thrown out the window in 5 to 7 years anyway. You got to make a push. Then from there, you can start to do new things.

I see this as like what is the right model for the next 5 to 10 years? Then when you're at the 10-year point, I think it becomes really hard to predict what really are the products that come off the back of that or what could displace you on the future, etc.

[00:49:03] JM: To draw to a close, I was looking at some of the writing that you've done and the past inventions that you've made. I think you're a capable hardware engineer. Why are you working in software?

[00:49:18] JG: That's a funny one. I really enjoy hardware actually. So, I like cacking on hardware I guess the scale. So if I was going to do a hardware project, it would have to be

something pretty hardcore and – I've invested in Boom. They're building a supersonic jet. Invested in Astranis and relatively space, they were doing 3D printed rockets and software-defined radios. Well, that's a good little answer there.

Astranis, it's a software-defined radio and it cuts down the weight of the satellite from ta hardware-heavy satellite to very a light satellite that's software powered. So I think the scale – The other part I love tinkering with is actually UI and UX, and that can have a hardware component or it can purely in the browser. Actually, you really enjoy personally working on trying to make snappy products. Trying to improve the UI, removing steps from things, improving affordances and constraints.

Abstract design is interesting to me no matter what the substrate is, and that actually generally means that I'd like improving software. The scale is very interesting of how many people you can reach. So we could build something that could be reached. Maybe get to a billion people. That's very interesting to me. The lack of capex required to go build it – I mean, don't get me wrong. I love hardware. But mostly the things I want to build require $500 million worth of capital to build on the hardware.

I think scale on a day-to-day basis, I really enjoy improving the UI. Removing ambiguities in the UI, and that's actually my wheelhouse to play interfaces and improve products. So software is for now something that is my focus, and then also algorithms. I'm interested in these specific algorithms that are occurring in AI. If you're to bind yourself to a wave that's coming through and take a good decision to not operate in a market that is shrinking or boring or not going anywhere, where would you want to be? You want to be in a market where things are rapidly changing where no one knows the new rules. There's a giant wave coming through, which is this wave or algorithms becoming easier to implement and doing things that people thought were not possible 20 years ago.

Then there's the mission. Sometimes you go to a mission and you want to get the mission done and it really doesn't matter what the substrate is that you're going to work with. So, some people I know that are not biologists, but they want to go and solve a particular type of Alzheimer problem and they have to reskill. I've seen [inaudible 00:51:46] get into transvection of cells and

get into biology. So I think – Maybe you bring something different to the table when you go and attack the problem with the different kind of backgrounds and you see it in a different way.

[00:52:01] JM: If you had to start a company in the physical world, like a manufacturing company or an agriculture company or an energy company, do you have any ideas for what company you would start?

[00:52:13] JG: Yeah. So part of it, while working on Goldman, I actually looked at different companies to go build. One of them was, "Hey, what about building a supersonic jet company?" So I came across – I started researching that, and I came across Boom team and I thought, "Wow! These guys have two years ahead and they've compiled a great team and they're really, really going to do this. It looks like they are on their way."

One of the other ones I looked at was trying to take what Boston Dynamics did, and instead of having million dollar machines that were one-off, that would stuck in the office, their office, like build very cheap versions with replaceable parts that maybe you could 3D print not fully open source. Enough to make it commercial and get the price point down to 10k where you could have rapid iteration of these [inaudible 00:52:58] and build like a Tesla of robotics. So that one is still interesting to me. I haven't seen anyone do it. There's been like robots in factories [inaudible 00:53:06] stuff and there's been toy robots, but I've never seen any Boston Dynamics done a 10k price point and take all the hardware stuff into software if you can. Maybe we'll see that happen after the last mile delivery services, delivering thing son wheels, on these kind of little mini-drones.

Aerospace is very interesting still. Engines are interesting to me. Rapid construction is interesting. I think the carbon sequestering is also a great engineering problem. We got massive problem with the environment [inaudible 00:53:35] methane. So looking at solutions for that which might end up biological solutions or cleaning air. Electronics has always fascinated me as well.

So, I'm always looking out for kind of interesting ideas there. But for now, I'm focused on Golden fully and hardware-wise. Maybe I'll do a hardware company in the future. Who knows? Part of Golden as well is like you interesting by looking at new things all the time. The variety, it's very

high when you get to see all these new pages being made, and new companies going into there.

[00:54:09] JM: There is a company in the low-cost robotic space that we had on the show recently, Slate Robotics in case you're interested in checking that out.

[00:54:16] JG: I would check that out, yeah.

[00:54:18] JM: Last question. Tell me an unusual belief that you recently developed about technology.

[00:54:23] JG: That's a hard question. An unusual belief. One of them maybe that I genuinely think that most things are possible. Not like most configurations that matter in the universe are possible. That's the most technological problems. If you really want to solve them, they are doable, and you just have to fully dedicate yourself to solving them and be absolutely relentless in solving them.

So all of these things of, "Oh, that's not possible. No one's done that before. No one's done that before." That to me just doesn't matter. It's like I think that's probably unusual and that most people would look at some technical problems and say, "This is too difficult. You can't do that." For me, nature can be very non-linear, and there are all these solutions around the corner.

So my unusual take is that pretty much anything that you challenge us to do, we can do it. But you can't do them all at once. You can maybe only do one really hard problem if you went all in on it. I'm talking about engineering problems. I'm not talking about, say, a math problem or a specific physics problem. That may be actually not possible for a team to say, "Yeah, we could do that. We could solve the Riemann hypothesis in the next five years," but maybe you could. Maybe you could say, "Hey, we're going to set up a massive website to brute force workout how to solve the problem and get everybody excited and get loads of teenagers excited and also people working on the problem and set up incentives and set up capturing of a research and linking it altogether and everyone attacking the problem. Maybe it's possible to do it in 5 years. If you said, "Let's do the Riemann hypothesis in five years." So anything is possible.

[00:55:56] JM: Okay. Jude Gomila, thanks for coming on the show. It's been great talking.

[00:55:59] JG: Yes, thank you for the questions. It was really fun.

[END OF INTERVIEW]

[00:56:04] JM: Podsheets is open source podcast hosting platform. We are building Podsheets with the learnings from Software Engineering Daily, and our goal is to be the best place to host and monetize your podcast.

If you've been thinking about starting a podcast, check out podsheets.com. We believe the best solution to podcasting will be open source, and we had a previous episode of Software Engineering Daily where we discussed the open source vision for Podsheets.

We're in the early days of podcasting, and there's never been a better time to start a podcast. We will help you through the hurdles of starting a podcast on Podsheets. We're already working on tools to help you with the complex process of finding advertisers for your podcast and working with the ads in your podcast. These are problems that we have encountered in Software Engineering Daily. We know them intimately, and we would love to help you get started with your podcast.

You can check out podsheets.com to get started as a podcaster today. Podcasting is as easy as blogging. If you've written a blog post, you can start a podcast. We'll help you through the process, and you can reach us at any time by emailing help at podsheets.com. We also have multiple other ways of getting in touch on Podsheets.

Podsheets is an open source podcast hosting platform, and I hope you start a podcast, because I am still running out of content to listen to. Start a podcast on podsheets.com.

[END]