# DevOps Samurai interview

As a DevOps evangelist in Japan, I'd like to introduce some cool DevOps guys in Japan.

This is not only Microsoft technologies, but also non-Microsoft technologies.

## Rakuten DevOps practitioners on transitioning to DevOps

I asked about the transition into DevOps on Rakuten. Rakuten is one of the biggest electronic commerce and Internet company based in Tokyo, Japan. Its B2B2C e-commerce platform

Rakuten Ichiba is the largest e-commerce site in Japan and among the world's largest by sales.

They are also known for "Englishnization" which means they use English as the company's official language from 2012.

Today's guests



**Kotaro Ogino** : Testing and Automation professional. He helps to adopt testing automation for over 20 projects in Rakuten.



**Hirobumi Takahashi** : Testing and Automation professional. He helps Kotaro and works with him.



**Yasunobu Kawaguchi :** Agile Coach in Rakuten. He is one of the most influential people in the agile community in Japan.



**Eiji Ienaga** : Agile coach and TDD and Testing professional in Eiwa system management.

  
**Tsyoshi Ushio** : Senior Techinical Evagelist – DevOps Microsoft Japan

## Part 1. Kotaro Ogino on Continuous System Testing

**Tsuyoshi** : Thank you for coming to this interview. Could you introduce yourself?

**Kotaro** : I'm part of the Development support division in Rakuten. I help to adopt automated testing in projects. It is over 20 projects now. In addition, I develop a platform for testing, deployment and automation of operations.

**Hirobumi:** I lived in Spain for 3 months. Rakuten has a lot of subsidiaries in the world. This allowed me do the job in English. Now I help Kotaro.

**Tsuyoshi :** How about the overseas job? Did they do a good job?

Notice : In Japan, not many people work internationally so they are afraid to use English

Now they are about to change to use English, so they don't know much about

the working environment in foreign company and are really interested in it.

We just admire the overseas job and we want to know the actual status.

**Hirobumi :** Yes. The engineering culture is different in Japan and Spain.

**Tsuyoshi** : Could you explain more detail about this?

**Hirobumi** : This can be my personal experience, but I felt that the responsibility of individual engineers is clear in Japan, so they focus on their responsibility area. In Spain, the responsibility boundary is not clear and the team tries to achieve the goal.

**Tsuyoshi** : I think DevOps has three elements. People, Process and Product. People refers to mindset and Product refers to technologies and tools. First of all, could you tell me about "Product", please?

**Kotaro**  : When I automate something, I always consider about the Value stream. For example, testing is often considered as a dedicated process from Dev or Ops but it should be considered as a part in a flow from requirement analysis to release. Test automation can provide the values to both developers and operators such as speedy and reliable release..

**Tsuyoshi :** Any challenges for automation?

**Kotaro**  : We have 2 groups. One is Devs division and the other is Ops division. When a development team needed a server, they needed to ask Ops division to set up and run it which took some times.

**Tsuyoshi** : Could you tell me about the servers that you look after?

**Kotaro :** Web / API servers. We use Java, PHP and Ruby.

**Tsuyoshi** : Could you tell me about the toolset of the automation.

**Kotaro** : Chef, Docker and OpenStack for provisioning servers, Capistrano and Fabric for deployment and JUnit for unit testing and so on.

**Tsuyoshi** : It is very common that developers don't write test code properly. Do they write these well?

**Kotaro** : Well, it's difficult... However, I think the development team should decide the policy. I just tell them how to do it.

**Tsuyoshi :** Could you tell me about non-unit testing?

**Kotaro** : Ok. I love this term called "Continuous System Testing" which I coined.

**Tsuyoshi** : Tell me more about it?

**Kotaro :** Continuous integration is about automating integration and testing continuously. Continuous System Testing is about automating system tests and then doing it continuously.

**Tsuyoshi :** I think it is one of the most difficult practices in DevOps.

**Kotaro :** Yes.

**Tsuyoshi :** What is the outcome?

**Kotaro :**  Before the adoption of the continuous system testing, it took 5 days for a system test which includes manual testing and deployment. Now it takes 2 days even if it has some problems.

Jenkins fires the automated system testing every night, so we can see the results the next day. Which means it takes 2 days maximum even if we had some problems.

**Tsuyoshi :** How long does it take?

**Kotaro :** 2 hours. It will take 50 hours unless it is executed concurrently. We use Jenkins with

20-30 concurrent tasks.

**Tsuyoshi :** Cool. In DevOps world, it is said that we have a system which enables "10 deployments per day", however it does not include deep automated system testing to make it easy to notice their own problem and roll it back. I'm impressed because yours includes deep automated system testing.

**Kotaro :** Thanks. In addition, the Jenkins system has 100 slaves.

**Everyone** : Cool!

**Tsuyoshi :** This is the famous Mary Poppendeck's question. "How long would it take your organization to deploy a change that involves just one single line of code? Do you do this on repeatable, reliable basis?"

**kotaro** : It depends on how much lines of code it has against a production line of code.

**Eiji :** Hmm. The code smells. Something wrong happens in your code.

**Kotaro** : No. For example, we have a search engine. It requires us to change 10 thousand lines of code, if we change one line of code.

**Tsuyoshi** : Ok. It must be a special case. Please tell me the normal case, please?

**Kotaro :** It will take 3 hours for us to deploy an application into production.

**Tsuyoshi** : Cool. Especially if you are enterprise company. By the way, could you tell me the definition of "System testing"?

**kotaro :** End to end functional testing and non-functional testing which includes performance, data-consistency, availability, search-quality, redundancy and index testing for distributed environments. It includes Recovery testing and Operability testing as well.

**Tsuyoshi** : Perfect! Tell me more about the tools for it.

**Kotaro :** We use Jenkins. We use JUnit/Selenium/CucumberJVM for web-based acceptance testing.

**Tsuyoshi :** When does the team write acceptance tests?

**Kotaro :** It depends. A scrum team writes it during a sprint. However, another writes it in the endgame.



**Tsuyoshi :** Do you use Azure for automation?

**Kotaro** : Now we need to write system tests for other regions in the world. Azure has a lot of data centers in the world. We are going to use Azure for performance testing across multiple regions.

**Tsuyoshi :** Could you tell me about another toolset besides acceptance testing.

**Kotaro :** It is totally same as acceptance testing. JUnit / Selenium / CucumberJVM.

**Tsuyoshi :** By the way, where is a bottle of soy sauce?

**Yasunobu** : I will show you. This is the highest-ranking pun. Because it will require to understand both Japanese and English.

Note : Soy sauce is called "Shou yu" in Japan. Which is very similar to "Show you".

They successfully implement "Continuous System Testing". They automate a lot of system testing for keeping quality. It is quite the Japanese style to improve the software lifecycle and

keeping up high quality. Kotaro presented at JaSST (Japan Software testing symposium). You can

refer to his presentation http://jasst.jp/symposium/jasst14tokyo/pdf/D2-2.pdf or his blog http://kokotatata.hatenablog.com/entry/2014/03/14/075842 .

To be continued.

## Part 2. Yasunobu Kawaguchi and Kotaro Ogino on Challenging for silos

DevOps guys always encounter the "silo" problem. How do they conquer it? Stay tuned!

**Tsuyoshi** : Then, could you tell me about the "People" element?

**Kotaro** : In terms of opinion, we had some conflict between Devs and Ops.

**Tsuyoshi** : How did you kaizen it?

**Kotaro :** We showed them some positive outcome. After automating their system testing, They gradually understand the merit. They could notice problems quite quickly. Then they said that it is cool.

**Eiji :** Did you have any trigger for the change?

**Kotaro** : Yes. It's a "Smoke test". We wrote a simple smoke test for the search engine.　We have a problem that non-Japanese developers didn't notice about Japanese processing. It is very difficult for them to understand the Japanese character's problem. However, it is crucial for the Japanese market. We added Japanese-character-testing for the smoke test. Then they understood the importance of testing.



**Tsuyoshi :** It is very hard to notice the problem related to foreign languages. Do you have any challenges related to Dev and Ops?

**Kotaro** : These are separated by division. Ops guys learn about infrastructure as code and all infrastructure configurations are managed and treated in Continuous Integration manner.

**Tsuyoshi :** Splendid!

**Yasunobu :** Now we have a DevOps meeting , on a regular basis to promote DevOps.

**Kotaro** : Ops uses infrastructure as code related technologies. However it was separated from Dev's system. Now they collaborate and integrate their own automation tools.

**Tsuyoshi** : Interesting. Both silos have high technical practice but did not collaborate. A lot of companies struggle with silos between Devs and Ops. Why do we have these silos?

**Kotaro :** I think we need to understand each other. Devs know about TDD and CI but Ops does not. Now infrastructure as code is coming, it is very common to fire chef recipes from Jenkins. Same thing happens on Dev side. System testing was executed only by the Ops person.

**Yasunobu** : I think a silo is just a structure. You may think "Hey, what a silo. It is ridiculous!" but the structure must have been reasonable in the past. According to the progress of technologies, it would have become a silo.

**Tsuyoshi :** What do you mean?

**Yasunobu** : We had someone who is in charge of operating tapes for a computer. It was effective at that time. Now we can easily back up a system, now we don't have that role.

It is my assumption that it is called "silo" when some technologies change something.

**Tsuyoshi** : By the way, you guys are Dev guys except for Yasunobu, right? From the Dev's view point, what do you think about the ops guys?

**Kotaro** : Generally speaking, I want more communication with Ops guis. It was really great to see the presentation from Ops guys in Dev's event.

**Tsuyoshi :** What do you learn in DevOps meetings?

**Kotaro :** Well, We don't know much about each other. Ops don't understand what Devs need and what they want. They can solve real problem of software development by Infrastructure as Code.

**Eiji :** I think some of them don't understand the vision like "who will be happy from this project".

**Kotaro :** They are now changing. They start accepting the passion of Devs.

**Eiji :** Now we are about starting to have mutual understandings.

**Yasunobu :** An Ops person has craftsmanship. Like a master carpenter. Their job is about to creating stable environments and operating it safe and sound. It is really important. That is why

they keep a high-level of self-control. So they try to make it perfect. They design and test it. Then, they hand you an environment, saying "Hey, now you can use this!" with full confidence.

However, no matter how hard they try, it may include some small problem. Then the Dev guys say,

"This is rubbish. They are not professional and perfect at all."

**Tsuyoshi :** Everyone makes mistakes. I think.

**Yasunobu :** Yes. So Ops guys try to polish perfectly. This hands-off-model leads to a boring job to an Ops person where they need just follow the Dev's opinion. It is not cool.

But from Devs point of view, Ops don't need to solve a small problem from the start. That is why, Devs need to understand the catharsis of Ops. We need to understand each other's passion.

**Kotaro :** I have an experience related to this topic. At first, we called it non-functional testing, then we changed the name into operability testing, then Ops guys started to give us requirements related to operations.

**Tsuyoshi :** Tell me more about it.

**Kotaro** : When we called it non-functional testing, they thought the project was led by Devs and they didn't have a right to give a requirement. After we called it Operability testing, they noticed that they can provide some requirements.

**Tsuyoshi :** It must be a breakthrough. Cool. By the way, if you were Ops guys, what is an Devs problem?

**Everyone :** ...

**Tsuyoshi :** I think it is very difficult to cover all learning areas, so we need to collaborate with others.

They told me about the Real-World-Example of DevOps. This problem occurs everywhere and it is still the most difficult problem to adopt DevOps. The next blog post will be the last part of the interview of Rakuten guys!

Stay tuned!

## Part 3. Yasunobu Kawaguchi on The impact of Small Team, Microservices and DevOps

Why a big company like Rakuten needs DevOps? Yasunobu told me about the three most important ICT strategies for an enterprise company.

**Tsuyoshi :** Why did you start the DevOps meeting?

**Yasunobu :** I met Ed Kraay who is an agile coach of Yahoo! in 2009. He talked about DevOps and release in Yahoo on the Agile Conference.

**Tsuyoshi :** Is this the guy who is presenting at Rakuten technology conference this year?

**Yasunobu** : Yes. The Rakuten technology conference has a lot of DevOps sessions. Ed and Microsoft DevOps guy, Ryuzee and Dana Pylayeva who will present about DevOps with chocolate, lego and scrum game.

**Tsuyoshi :** Super. Why were you inspired by him?

**Yasunobu** : I was not surprised about Netflix and Facebook's DevOps. However, Yahoo! is an enterprise company which has a lot of legacy systems. It was established about 20 years ago. They successfully stood up a DevOps environment in just a year. It is amazing.

Facebook already has a culture of startup from the beginning so I'm not surprised by their DevOps adoption.

But Yahoo! didn't have the culture. They have established a DevOps environment from scratch.

**Tsuyoshi** : What outcome did they achieve?

**Yasunobu** : They used to take 2 months for development and 4 months for QA for its release. Now they can enable several deployments per day.

**Tsuyoshi** : It's amazing.

**Yasunobu** : Why did they try DevOps? I guess they must have had a reason.

A big company used to have an advantage for building a service. They have a bunch of money, so they can invest money into QA activities. After the bankruptcy of Lehman Brothers, Startups got some power.

Because the cutting edge technologies enable us to build a profitable service by a small team (3-5 people).Startups beat big companies. At that time, big companies couldn't beat startups because 4-month-QA now becomes disadvantage for the market.

**Tsuyoshi :** I agree with you.

**Yasunobu** : I think now is the time for the break-through of DevOps.

**Tsuyoshi :** Please expand.

**Yasunobu** : When the world is about to change, sometimes we have a timing that management can't judge something. Because it requires company-level-judgement.

For example, when we first met Virtualization, we had a service division and an operation division. At that time, the OS was set up by the service division, and a server was set up by the operation division.

However, a Virtual machine includes both OS and virtual machine. Which division should look after the virtualization? Who is in charge? It is a turning point for the ICT industry.

**Kotaro :** Yeah. Now is the time. Who's in charge of chef recipe? Is it code or infrastrucute?

**Yasunobu** : The way of thinking of software development has been changing. Virtualization, Scrum and so on. The same thing is about to happen for the DevOps, small teams and microservices.

**Tsuyoshi :** You mean now is the time for the big bang that is coming again.

**Yasunobu :** Yes. As I said, for a long time, enterprise companies didn't know how to beat startups. But now, we've got the answer.



**Tsuyoshi :** Tell me more!

**Yasunobu** : Startups is really flexible and easy to adopt for change. Startups can create software quicker than an enterprise company. An enterprise company couldn't do it.

Now we can use a microservices architecture. Which means if an enterprise company can create some small teams, microservices are loosely coupled and a small team can handle it. Then they use DevOps practices for the development. So they can successfully deploy several times per day.

**Tsuyoshi :** Divide and conquer.

**Yasunobu** : Yes. We can decompose a big service into microservices. Small teams can handle microservices. Now enterprise companies can use a DevOps approach like startups which means enterprise companies are about to have huge advantages because they have huge resources against startups.

**Tsuyoshi :** It must be a big turning point. It is disruptive innovation.

**Yasunobu** : That is why it is very important for enterprise companies to try small team , microservices and DevOps. In Japan, when I noticed change like this, about three years later, it becomes popular.

**Tsuyoshi** : It is like a Yasunobu's law. haha.

**Yasunobu :** Netflix 2011. Fowler wrote a blog about small teams and microservices and DevOps. It took about a year for me before I notice it.

Cookpad is cool as they already tried it in 2014. Once I notice a change, it will take about 3 years to be popular. In Japan, Jeff Sutherland came to Japan in 2011 and Scrum became popular in 2014.

Note: The agile approach was not so popular until 2014 in Japan. Now they are trying.

**Tsuyoshi :** Sometimes, someone asks me what is suitable domain for DevOps. For example, is front-end system good for DevOps, but not for backend or something. However, I think it is non-sense. DevOps practices will be a standard of software development.

It is an intrinsic change. Backend system should use these, because no one will disrupt the service stability but still has an ability to deploy quickly and safely.

**Kotaro :** Now new grads know about Scrum. They also know about TDD and CI. They said that they learned in University.

**Tsuyoshi :** It was surprising. I think this change includes Lean startup and Jeff Patton's agile product ownership. One of the value of short release cycle is a quick feedback loop. Yasunobu translated the User Story Mapping by Jeff Patton. What do you think about it?

**Yasunobu** : Think about a big hospital and a small clinic. If some same problem like fire, happened these two facilities, what would happen?

In the small clinic, it causes a panic and quick response for it, on the other hand, the big hospital won't. Because people in the big hospital might not think they need to evacuate from it. This is the feedback loop differences between two.

But now, an enterprise company can use small teams. It used to need multi layered management structure, but now they can take an Amoeba Management style (Note : <https://en.wikipedia.org/wiki/Amoeba_Management>) that includes a lot of self-organized

teams which have 5 to 50 people are in charge of making profit by itself.

**Kotaro** : It is said that small team is great but it requires 3-5 unicorns and it is very difficult. What do you think about it?

**Yasunobu** : The thing is Passion / Skill / Small team. Especially passion is important. Skill will follow.

**Tsuyoshi** : I agree with that. Passion is crucial. When I was in charge of the 2nd agile project in my life, only myself and one developer can handle TDD. The team includes a new grad.

**Kotaro :** The small team includes 5 Devs or 5 people that includes Dev / Ops / QA?

**Yasunobu** : 5 people for Dev / Ops / QA. I mean cross functional. No such guy can handle everything.

**Kotaro** : I think small teams are originated by QC activities and small group activities in Japan.

Japanese bottom up kaizen activity is reimported from Western countries.

**Yasunobu** : The bottom up team and small team with passion. Jeff Sutherland focused on it. In the western culture, individual happiness comes first then the team. However, in Asia, it is the opposite. But a small team is not only in Japan, but also Asian culture I think.

Toyota is the same. Before Tayler's management style, then GM production system beat the Toyota system. Toyota couldn't beat them for a long time and they didn't have much budget and resources. They came up with an all-round worker style. The testing phase come last at first then they kaizen it finally, the testing was decomposed into every phases...



Thank you for coming guys! I was really impressed about your story.

An enterprise company can beat startups by using small teams, microservices and DevOps.

It was an impressive idea.