Evolving Software Testing at the Pokémon Company International

Paul Grimes

p.grimes@pokemon.com

Abstract

How can you know that your services will handle the requests of millions of users a day? Or that making a fundamental change to one of your technologies won't break your user experience? By creating a phased approach to testing the right pieces at the right time that your entire team can use and build on. In the Pokémon Company International (TPCI) quality department, we are responsible for the Pokemon.com website, the software for tournaments played by 1000s of players, and the services used for logging into applications like Pokémon Go and Pokémon TV. Since the launch of Pokémon Go in 2016, our quality-focused department has worked to develop strategies that reduce the number and duration of customer incidents in the face of millions of daily active users. We will present the phases that our test engineers follow and the tools we use to ensure that the services we are delivering are capable of meeting our users' demands. From exploratory testing of our API's, to developing unit and integration tests, to deploying scalable performance suites, we will discuss the tactical choices we've made and how they affect our outcomes. This integrates with our web and mobile testing safeguarding the end-to-end user experience. By integrating testing early and doing the right work at the right time, TPCI is ensuring our customers have exceptional encounters as they interact with our brand.

Biography

Paul Grimes has worked in the software industry over twenty years. He loves technology and has worked in many vanguard fields including web services, game network engineering, automation planning and automation implementation. He spent the early part of his career at Microsoft working on Office, then shifted to work on PC and Xbox games. He also has varied experience from working at Barnes and Noble, Disney and various startups in the games and data knowledge industries. He currently works for the Pokémon Company International as a software development engineer in test.

Introduction

The Pokémon Company International (TPCI) is a wholly owned subsidiary of the Pokémon Company of Japan. It is responsible for the development, production, distribution marketing, and licensing of Pokémon products outside of Asia. This includes the Pokémon Trading Card Game, Pokémon video games, Pokémon animated series and live action movies, and Pokémon merchandise such as apparel, toy, and plush.

To support the brand and the many diverse efforts, TPCI has a dedicated technology organization. The tech org is responsible for both internal tools and external offerings. The different offerings are vast.

Pokemon.com is the official website for everything Pokémon which is available worldwide and translated in 12 languages. There are smaller “mini-sites” which are used to announce new video games or large offering such a new set for the training card game. Pokémon TV is where fans can watch their favorite episodes of the animated TV show or one of the animated movies. The Pokédex can help fans find information about their favorite Pokémon.

The Pokémon Trainers Club (PTC) provides user an online identity, which can be used to store and manage both adult and child accounts. PTC allows players to track their progress in tournaments for both the video game and the card game, sign-in to Pokémon Go - a game licensed to developer Niantic, play the Pokémon Card Game Online, or track which episode of the Pokémon Animated Series they are watching on Pokémon TV.

Our quality team inside the technology organization focuses on ensuring quality as we develop these and other upcoming offerings. We work with internal teams on the development of projects, work to improve the development process, validate releases, and try to make sure we are applying the right validation at the right times.

Quality Team Values

Our team is organized around six core values developed by our quality team and our leadership group. They help us inform our decisions. They have been amended and added to by the team as we have changed and grown. We recognize team members who demonstrate these values at monthly team meetings. Our team values are:

> *Great Triumph - We believe every member of our team is working to deliver amazing, memorable experiences. Valuing everyone's opinions and viewpoints will help us reach our goal.*
>
> *Customer First - Using customer-centric practices in our work, we are committed to providing a quality product and positive experience to our customers and fans.*
>
> *Continued Learning - We strive to better ourselves every day. We will do this by continuing to learn new things, take on new tasks, and by asking questions. We're ok with trying something different and failing - I'll know what to do better the next time.*
>
> *People Matter - Our team is a family. It is super important to us to care for each other, even outside of work. We believe the people make the company great and treat them accordingly.*
>
> *Transparent Communication - We are honest and transparent. When discussing the quality of a product, we will use data to communicate the quality. Obfuscating the true quality of a product isn't productive.*
>
> *Courageous Action - We act without fear of impediments or dissenting opinions. We confront risks with prudence and tenacity even if they leave us vulnerable. We support each other because we trust that our actions improve quality for everyone.*

Team Organization

Our team organization has changed organically. Our director has mangers reporting to him. The managers handle people management tasks as well managing teams of four to eight people. Our director works with our managers to produce a vision for where the team is headed. This vision is then translated into the action by our steering committee, which is made up of the managers and selected individual contributors. They focus on issues such as inclusion and working direction to drive changes to make us more effective. The discrete day-to-day work of the quality organization is divided across the team into pods and Scrum teams.

Pods and Pod Leads. Pods are small groups of two to six individuals working on a specific product. For example, there are pods for organized play tools, pokemon.com, and Pokémon TV, and Pokémon Trainers Club to name four. The individuals on the team may be from different managers. They are a mix of Software Development Engineers in Test (SDET) who work on tools, developing automation, and frameworks and Software Test Engineers (STE) who develop test cases and collateral and work on execution. A Pod Lead is chosen by the managers to own and drive the quality responsibilities for the pod. Pods can be scaled for the demands of a team and rearranged to best serve the business.

Scrum Teams. Scrum teams are assembled to work on a specific project that may be longer or shorter than the duration of a project. They use Scrum procedures with a scrum master, Product Owner (PO) and team members doing development. These teams tend to stay together so that estimating remains constant. The Scrum teams are typically 3-7 contributors, including the scrum master and PO. These Scrum teams focus on things like adding automated regression testing to legacy projects, developing continuous integration/deployment pipelines for projects where they are lacking, improving the quality team's data gathering and dashboarding capabilities, and performance testing.

Individuals may be part of a pod and a scrum team simultaneously. This structure allows us to share knowledge about projects across the group giving each individual an opportunity to see projects across the technology organization. Being a pod lead gives individuals a chance to develop the skills of owning a project and planning the work for a project without having to do career growth or people management tasks such as employee reviews. It also provides an individual ownership of a project and allows partners to know who to go to with concerns around the direction the pod might be taking. An individual's manager remains their advocate and helps them plan their growth. Managers also remain aware of projects throughout the organization because employees they manage contribute to projects across the organization instead of in a single focus area. The department can react to changes in priority because of the flexibility of the team composition. This structure allows individuals to remain focused on delivering quality and confidence to our partners while giving people the opportunity to develop the skills they wish to improve.

Here is an example of how this might look in practice. Our director's name is Klein. Klein has three managers, Adams, Baker, and Clark. Each manager has a mix of SDET and STE's reporting to them.

For a release of PTC we needed to replace the database structure. One of Adams SDET's, Davis, was chosen as the pod lead for the project. The demands on the team were high because they needed to guarantee both functionality and performance of the chosen migration. Another SDET from Adams' team was added to the pod, Evans. Frank, an STE from Baker's team, Ghosh and STE from Clarks team, and Hills an STE from Adams' team were also added to the pod. As demand for more automation became apparent, a third SDET, Irwin was added to the pod from Baker's team.

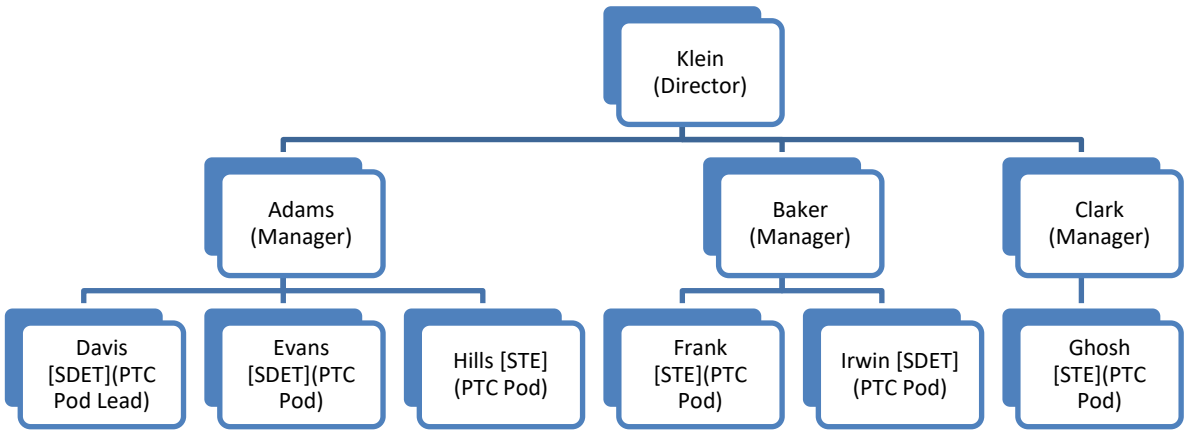

Figure 1. Example Org Chart

After shipping the database migration, PTC work drastically reduced. Evans became the pod lead for PTC pod, and Hills remained on PTC pod for continuity. Davis and Frank became members of a scrum team focused on automation regression (ART). Hills contributed to that scrum team as well. Irwin became a pod lead on PCOM, and Ghosh joined that pod. Frank began working on a separate pod focused on PTV.

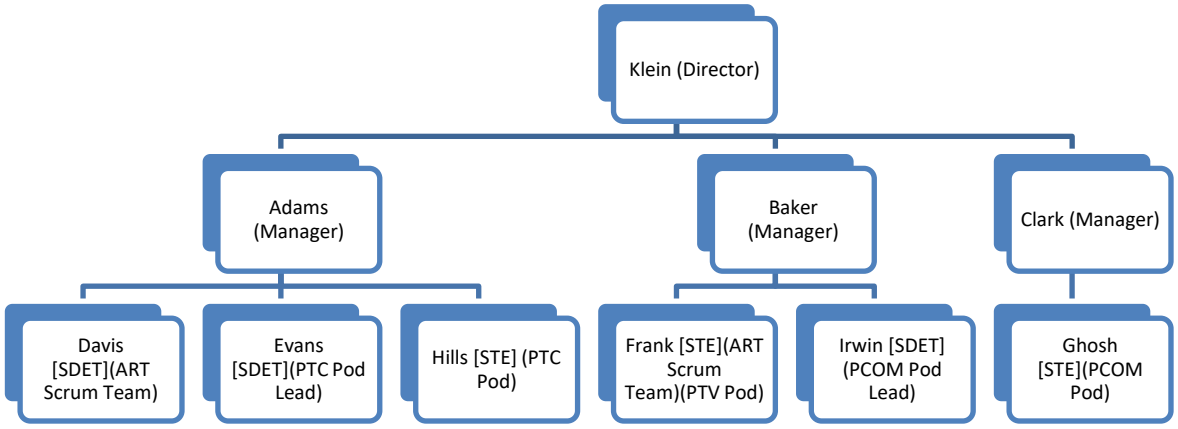

Figure 2. Org After Shipping

Approach to Quality

Inside each pod, there are a number of phases that each team may need to complete depending on the project. The pod lead will help derive the plan with input from the pod and the partners. Each phase when done in order allows the team to surgically address the most important issues at the right time. These phases tend to build on each other, so the work done in a previous phase helps inform the work of the next phase.

First, case development occurs in the exploratory phase. Next, the libraries or test runs are created during a test development phase. Automated tests and system verification occur in an integration phase. The systems are stress tested and their performance determined in an evaluation phase. Finally, confidence and readiness are determined in the release phase.

These phases may happen seemingly concurrently during a sprint or take multiple sprints to complete. They may align to their own cadence or happen as tasks on a Kanban board depending on the team they are being done for but, by acknowledging which phase of testing is appropriate at the time, the pods and individuals can determine where a feature is and how much confidence the team has in its readiness for release.

Exploratory Phase

During our exploratory testing phase, we are discovering what the extent of the testing for a feature set may be. This often includes setting up a local development environment so we can replicate what the development team is doing. It may include setting up a test environment or performing testing against the first environment that the development team is using. Tools like the developer tools in a browser for web testing and Postman for API testing are invaluable at this time. Typically, test engineers are reviewing basic functionality, discovering edge cases, and evaluating implementation strategies.

For a recent PCOM release, an update was being made to the Pokédex. We decided to investigate if we would be able to use mabl, an automation tool, to develop a sustainable automation suite. We used the exploratory phase to develop reusable components in mabl called flows to determine if automating this task was feasible. In this case, the actual test cases had already been developed and we were exploring the ability to automate the cases. With some experimenting and collaborating with the mabl team we were able to develop working flows that would allow the development of an automated pass of Pokédex changes.

Test Development Phase

Once the basic shape of testing is understood, engineers can begin developing tests and organizing test suites. Discussions occur with the development team around where they have unit tests and how that testing is executed (locally, part of CI/CD). Test is involved in code reviews to determine where gaps may be occurring. Common test libraries may be explored or developed at this point to make developing automation easier. Many of our pods and scrum teams will publish libraries to internal repositories for common tasks such as integrating with CI/CD, publishing results of tests to information repositories so that other pods can easily integrate saving time and effort.

In a prior release, we had developed integration tests for the services that PTV uses. These services allow users to continue watching episodes when switching devices. As use of these services expanded, we determined that we should publish a test library for those services for reuse. In the test development phase the pod working on those services factored out the common functionality into a Python library, and then refactored the existing integration tests to use the library. That library was then published internally for other products to use as part of their integration testing.

Integration Phase

With cases in hand, the team will begin work on cases that address the entire system. Often the focuses of these tests are to ensure multiple components work together as designed. These help validate architecture choices made by the development teams as well as our devops team. These tests are often automated and run as part of CI/CD pipelines using Jenkins or Team City. Often the results of these and the unit tests are used to evaluate code coverage, highlighted by tools such as Sealights. Where it makes sense, teams will publish tests in Docker containers so that they can be run by simply executing

the image and collecting the results. This allows these tests to be repeatable in multiple environments. The results of the tests may be written to a test case management system such as qTest.

The PTC team has integrated Python tests that cover 69% of their API. They have also incorporated mabl tests to cover the UI through the web client. By integrating both of these strategies into a single CI/CD workflow in Jenkins, each build has 86% or more total coverage per build.

Evaluation Phase

In the evaluation phase, the focus is no longer just functionality, but performance and stress as well. Using the libraries developed in the previous quality phases, stress and load testing can be performed to simulate the behavior of the system when thousands or millions of users are accessing it. Tools like Locust are used to create load on the system to simulate scaling events or determine starting parameters. Resources may be manipulated in stress tests to simulate running out of memory or database access issues. Estimates for the predicted number of users are sourced from business partners as well as historical data.

Using Locust, we were able to determine that we were seeing errors we didn't expect in the services used by PTV when the system was under large loads. We were able to determine that there were issues with replication of data across regions, and that best effort routing was not appropriate for our design because the data replication may not be as fast as user requests for data. To address the problem we changed routing from best effort to location based.

Release Phase

In the release phase we are looking to provide confidence to our stakeholders. The focus is on providing data that helps make business decisions around a products readiness. Often the team is answering questions such as "How many users can access the system at a time?" or "What monitoring will exist in production?". Often run books have been reviewed and checklists developed so support engineers know how to address incoming issues.

Conclusions

By taking this phased approach, many positives have occurred on the team. The quality team tends to possess very deep understanding of our products and services. We are often able to help make improvements to the development process such as continuous integration testing rapidly due to our experience implementing it at TPCI. We have seen dramatic reductions in the number of incidents in our legacy products because the goals and metrics are known and evaluated for prior to releasing to production. For the Pokémon Card Game Online, for example, we have reduced the number of incidents from around four to six per release (with 4-6 releases per year) to less than three incidents per year. The quality team is able to iterate rapidly, finding solutions to problems that may exist across pods and products and addressing them at their root. Most importantly, we have been able to measurably increase our customer satisfaction, both with internal customers and partners and consumers.

The Future

Part of our culture is to always be growing and looking for opportunities to make improvements. We are looking to create common methods for producing dashboards to make communicating project status with our partners even easier. This will focus on developing a common set of tools for collecting and accessing the data produced in our testing. We hope to continue developing our common libraries so teams that rely on internal services can simply choose. A tool instead of needing to develop one. We also hope to move teams towards CI/CD pipelines that automate much of the evaluation of new services and automatically publish all the way to production when applicable. We are also developing deeper ties

with our customer service teams to try and understand where consumers are encountering issues, how we can address them and provide customer service representatives better tools to mitigate issues.

By constantly refining our processes and evolving how we approach challenges, our quality team is ready to tackle new opportunities.

References

Docker Inc. “Get Started With Docker.” https://www.docker.com/ (accessed August 1, 2020)
Jenkins. “Build Great things At Scale.” https://www.jenkins.io/ (accessed August 1, 2020)
Locust. “An open source load testing tool.” https://locust.io/ (accessed August 1, 2020)
mabl. “Intelligent Test Automation for Everyone on Your Team” https://www.mabl.com/ (accessed August 1, 2020)
Tricentis qTest. “Agile Test Management for the Enterprise.” https://www.tricentis.com/products/agile-dev-testing-qtest/ (accessed August 1, 2020)

Sealights. “Smarter testing means delivering high-quality software faster.” https://www.sealights.io/ (accessed August 1, 2020)
Jet Brains Team City. “Powerful Continuous Integration out of the box.” https://www.jetbrains.com/teamcity/ (accessed August 1, 2020)