RAYGUN

Error Monitoring Drives Innovation

Automating software quality liberates teams. Development is proactive rather than reactive.

Summary

Customers want applications that solve real world problems, and their hunger is increasing at a phenomenal rate. The level of demand is shocking, and as their desire increases, so does the demand for software developers. The US Bureau of Labor Statistics projects 30% growth in the need for software developers over the next few years. CareerCast recently upgraded software engineer to the highest rated job in 2012, beating out actuary, HR manager and financial planner.

Naturally, the pressure of keeping up with software releases is also on the rise, as is competition between not only applications, but also software that provides the same functionality. This hurried pace and competition means development teams must innovate quickly. However, while innovation is an easy goal to set, execution can be at odds with rapid software releases, quality and maintenance. The key to focusing development teams on both innovation and quality rapid release is to optimize and synthesize the methods and processes for catching, preventing and addressing errors.

> "Error monitoring is a tooling approach that allows companies to spot issues prior to release without additional effort, and to more quickly address issues that do make it to production."

The result is software teams that are more proactive and less reactive.

Push vs Pull

There are two major trends that impact IT operations and developers. They are:

1. Quality Assurance (QA) responsibilities being passed to front-end and back-end developers; and
2. Autonomous delivery pipelines and analytics.

Previously, development teams staffed large QA groups to run functional tests prior to releases. In today's development shops, quality is everyone's responsibility. QA teams are called Quality Engineering (QE), and they build test case strategies and automation for end-to-end testing.

While front and back-end developers are the first line of defence for bugs, they are also expected to live test driven development, run unit and initial functional tests, and, if given proper lab environments, conduct end-to-end testing on every commit across the entire multi-tier application. In many development shops, this organizational structure is already there, but overall it should be the aspiration of all modern development shops.

From orchestration to deployment to test runs, automation backed by a strong reporting and analytics engine is the only way to make this new organizational structure possible, extensible and sustainable. As the transition from the push vs. pull method is completed, IT and development need an advanced tool to provide information on problems, identify areas of improvement and report on trends with very little outside input required.

Culture, Process, Tools

Together, the trends of shifting QA responsibility and autonomous delivery pipelines are the foundation of DevOps with one critical missing element: culture.

Culture does not define what is popular in mainstream development or the handful of startups in Silicon Valley who use a method unique from enterprise development. Culture is a deliberate implementation of a results, metrics and quality-driven development team. If this culture is not created consciously, it will develop organically—usually contrary to the overall objectives of the company.

Considering the trends in processes and tools, the speed and quality of the development team should increase. But in many cases, an increase does not occur because the development team is focused on either upcoming releases or what is currently in production. Developers cultivate an intentional tunnel vision so they are able to complete their tickets and address bugs efficiently. IT operations use their tunnel vision to keep production running and identify potential issues. So while they intend to approach the delivery pipeline holistically, the “right now” demands often get in the way.

Let Error Monitoring Take a Load Off

The solution is something developers do every day: exception handling. Try/Catch blocks are in every application. (If they are not, there is a more serious problem!) In most current systems, exception blocks belong to a specific developer on a specific branch of the application with results leveraged only in the Integrated Development Environment (IDE) and in production to prevent catastrophic failure. But exception blocks can be used far more proactively during production and releases.

Adding one more line of code in error blocks to process a large amount of data and push it into a modern analytics and reporting engine allows developers to expand the benefit they receive both individually and across the whole team. This analytics engine synthesizes errors, including complete stack traces, across the entire application and pushes critical events and trends to relevant team members. This lets developers address issues during continuous integration (CI) prior to hitting production and respond before greater problems are caused.

By using this line of code, tasks developers used to look at manually are now completed automatically. Since the tool tracks current and previous releases, critical errors are more easily emphasized. For example, if an error has repeated across multiple releases without a major impact to application quality, the system will list this error as a low-level priority.

Moving Faster, Freeing Brain Cycles and Increasing Innovation

The net result of a tool that can take on the burden of exception handling is that:

1. Developers have less to think about;
2. Unit tests become simpler; and
3. Quality Assurance requires less effort and focus.

> "Implementing error monitoring directly results in more developer time focused on adding new features, improving user interfaces and addressing customer concerns."

Removing the pressure of the release-to-release mentality lets the team exist in a more relaxed environment that promotes creativity and drives innovation—the exact type of environment that will result in groundbreaking applications.

Conclusion

No tool can produce innovation, but automation tools can give developers the time to be creative and produce new and original technology. As you explore this opportunity, identifying automation that easily integrates into your current processes without additional effort is critical.

Error monitoring is the perfect solution. Error monitoring provides a historic, holistic view of application quality by pushing critical events directly to the team and prioritizing issues by understanding how previous errors have impacted the application quality.

Since developers are trained to create exception handling in their code, it's an appropriate place to implement this simple line of new code that takes exception blocks and expands their results beyond individual developers and individual releases.

Error monitoring is the easiest way to support innovation without additional effort, direct more of your developer time to proactively coding and spend less time reacting to software issues.

About Raygun

Raygun Inc is the market leader in error tracking and crash reporting tools for software developers. Raygun's powerful yet easy-to-use error tracking solutions transform how software teams of all sizes work – and work together – to drive dramatically lower software error resolution times and associated costs. The company's proven technology, comprehensive services and expert tools are helping more than 1,000 companies around the world to turn software errors from a costly process to a revenue driver.

For more information, visit www.raygun.com, or subscribe to Raygun's blog at www.raygun.com/blog

U.S.A Office

Raygun Inc
1670 South Amphlett Blvd
Suite 214
Palo Alto, CA 94402

NZ Office

Raygun
175A Cuba Street
Te Aro
Wellington 6011