Glossary

**Blue/green deployment.** Blue-green deployment is a technique that reduces downtime and risk by running two identical production environments called Blue and Green. At any given time, one of the environments serves all production traffic, while the other is idle or accessible only by internal users for testing purposes. At deployment time, the new version can be deployed to the “idle” environment, tested, and then the router is switched to send all production traffic to the new production environment.

**Canary release.** A technique to reduce the risk of introducing a new software version in production by slowly rolling out the change to a small subset of users before rolling it out to the entire infrastructure and making it available to everybody. (Danilo Sato)

**Commit.** A commit to a source code control repository adds the latest changes to the head of the currently checked out trunk/master or branch. Commits remain in the repository indefnitely, and can be reverted or rolled back to a previous version.

**Continuous delivery (CD).** The ability to get changes of all types—including new features, configuration changes, bug fixes and experiments—into production, or into the hands of users, *safely* and *quickly* in a *sustainable* way. (Jez Humble)

**Continuous deployment.** Every change goes through the pipeline and automatically gets put into production, resulting in many production deployments every day. (Martin Fowler)

**Continuous integration (CI).** A development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early. (Thoughtworks) Production deployments are frequent, but not every change results in automatic production deployment.

**Dark launch.** The practice of deploying the very first version of a service into its production environment, well before release, so that you can soak test it and find any bugs before you make its functionality available to users. (Jez Humble)

**Deploy.** Deployment is the process of putting updated components or infrastructure of a software product into a particular environment, such as a test or production environment. By using techniques such as feature flags/toggles, it’s possible to deploy updates without actually releasing them for people to use.

**DevOps.** The term was coined in 2009 from “Development” and “Operations”, but the concept goes much further back. Developers, testers, operations staff and others collaborate to build and maintain build, test and production infrastructure that enables them to improve their customers’ lives.

**Environments.** A delivery team commonly has various environments where they can deploy and run their software product. For example, a development environment where coders make and test additions and changes to the software; a test environment where the team can run automated tests and do various types of testing such as functional, exploratory, and security; a staging environment where the team and business stakeholders can test in a production-like environment, and production itself.

**Feature flag (or feature toggle or switch).** A configuration option that defines whether or not a feature within your software should be executed. You might also hear this concept called feature flags, flippers, switches, feature bits, or latent code. (Katrina Clokie)

**Latency** is the amount of time to get through a pipeline line including any introduced delays.

**Logging.** Log files record transactional and status information, along with errors and warnings that are generated by unexpected activity. (Katrina Clokie) They provide detailed, low-level information to diagnose the problems.

**Merge.** Merging or integration in version control reconciles multiple changes made by different people to a version-controlled collection of files. For example, you create a branch to make changes for a particular user story. When the changes have been completed and tested, you can merge those changes back into the master or trunk. Merging is often done by way of a pull request.

**Monitoring.** The process of maintaining surveillance over the existence and magnitude of state change and data flow in a system. Monitoring aims to identify faults and assist in their subsequent elimination. (Katrina Clokie)

**Observability.** The instrumentation you need to understand what’s happening in your software is available. Observability focuses on the development of the application, and the rich instrumentation you need, not to poll and monitor it for thresholds or defined health checks, but to ask any arbitrary question about how the software works. (Charity Majors)

**Pipeline.** A repeatable, recorded communication of automated feedback. It includes test and deployment scripts, from development and operations respectively, along with a pipeline to illustrate the process by which the scripts run. Also known as automated deployment pipeline. (Katrina Clokie)

**Pull request.** A pull request is a part of a code flow that is created when new or updated code on a branch is thought to be ready to merge into the base branch in the source code control repository. Others can review the changes, make comments on them, and commit more changes under the same pull request, if required. Once the changes are judged to be acceptable, they can be merged into the base branch that was specified in the pull request.

**Release.** Release occurs when a new or updated feature with business value is made available to end users. Business stakeholders decide when to release features to end users.

**Staged rollout.** A canary release with a different focus. Instead of creating a canary by limiting changes to infrastructure, the rollout intentionally limits the number of users with access to the new code. (Katrina Clokie)

**Throughput** is the number and size of items that can be sent at any one time through a pipeline.