**Continuous Delivery**

Getting software released to users is often a painful, risky, and time-consuming process.

Continuous Delivery can help large organizations become as lean, agile and innovative as startups. Through reliable, low risk releases, Continuous Delivery makes it possible to continuously adapt software in line with user feedback, shifts in the market and changes to business strategy. We speed up the whole process while improving quality. Test, support, development and operations work with each other as one delivery team.

## From Continuous Integration to Continuous Delivery

[Continuous Integration](http://www.thoughtworks.com/continuous-integration) is best described in a [landmark article](http://www.martinfowler.com/articles/continuousIntegration.html) by our Chief Scientist, Martin Fowler. Originally one of the fundamental practices outlined in the Extreme Programming (XP) methodology, Continuous Integration (CI) has become an essential ingredient for teams doing iterative and incremental software delivery.

Continuous Delivery is the natural extension of Continuous Integration: an approach in which teams ensure that every change to the system is releasable, and that we can release any version at the push of a button. Continuous Delivery aims to make releases boring, so we can deliver frequently and get fast feedback on what users care about.

"Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly."   
- Martin Fowler, ThoughtWorks Chief Scientist

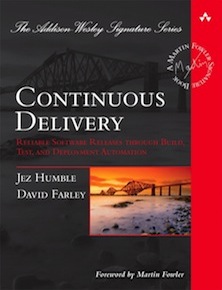
## We wrote the book on it.

Continuous Delivery (CD) is a concept that was first described in the eponymous 2010 book co-authored by ThoughtWorker Jez Humble and ThoughtWorks alumnus David Farley. CD provides a pattern language for the collection of software build, test and deployment activities that happen on the path to production. Further, CD references CI as a starting point, and extends it using lean philosophy and planning techniques that make it relevant to the greater business community. The goal of CD is to put customers in control of an ongoing cycle of software releases.

CI made its way to many delivery teams by way of [open source software](http://www.thoughtworks.com/open-source) called CruiseControl, that was developed by ThoughtWorks in the year 2000. Since then it has spawned a wide market of development tools in the mainstream market. After many years of working with customers to construct deployment pipelines, ThoughtWorks developed Go as the first tool designed specifically for the practice of continuous delivery.

“Business knows all too well that slower and larger releases are buggy and unreliable. With Go we have frequent releases of smaller amounts of functionality. This avoids all the pressure, escalations, and rollbacks, while enabling reliable, predictable releases. They really see a value in this.”

– Alexander Schwartz Sr. Release and Test Specialist,  
Head of Places Delivery Nokia

[](http://www.amazon.com/gp/product/0321601912)

Reliable software releases

## We eat our own dog food.

[](http://www.thoughtworks.com/products/go-continuous-delivery)

The goal of Continuous Delivery is to move your production release frequency from months to weeks or even days. Go is Continuous Delivery software, designed and built by ThoughtWorks to help tighten feedback loops across your build, test and release workflow