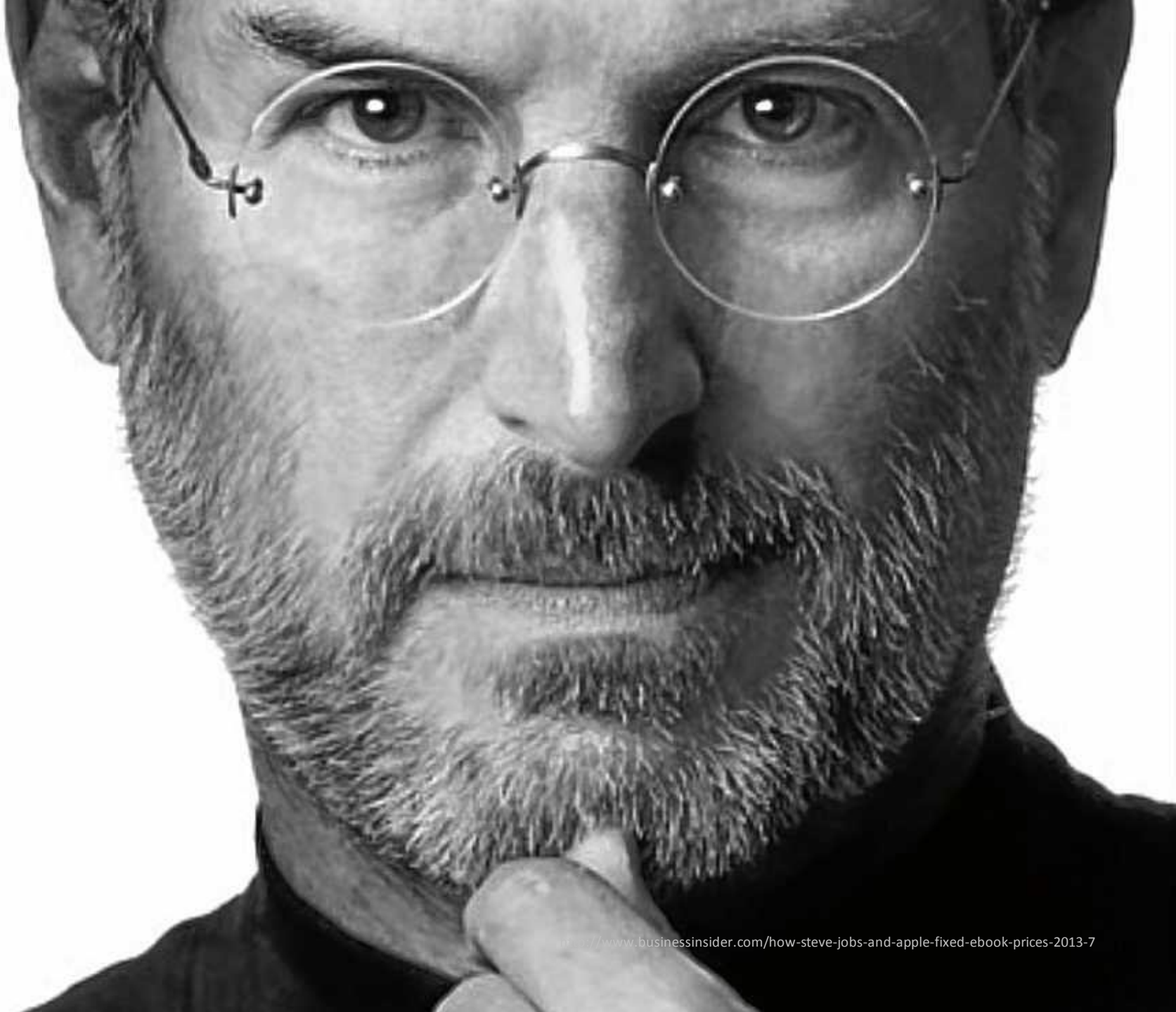
Intro to continuous **delivery**

Vladimir Alekseichenko

Agenda

- Introduction
- BuzzWords
- Benefits
- Deployment pipeline
- Tools
- Patterns
- Anti-patterns



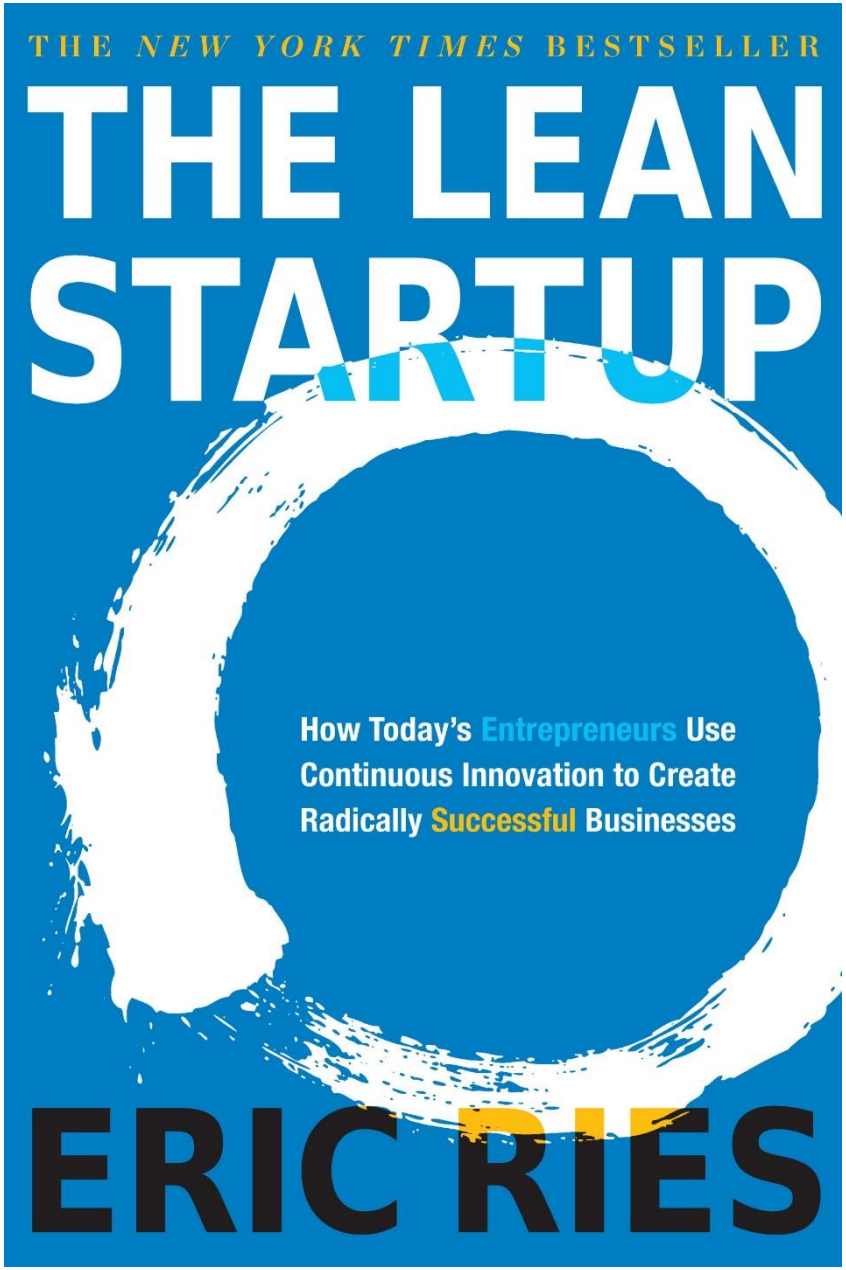
http://www.businessinsider.com/how-steve-jobs-and-apple-fixed-ebook-prices-2013-7

Innovate

You **can't** just **ask** customers **what** they **want** and then try to give that to them.

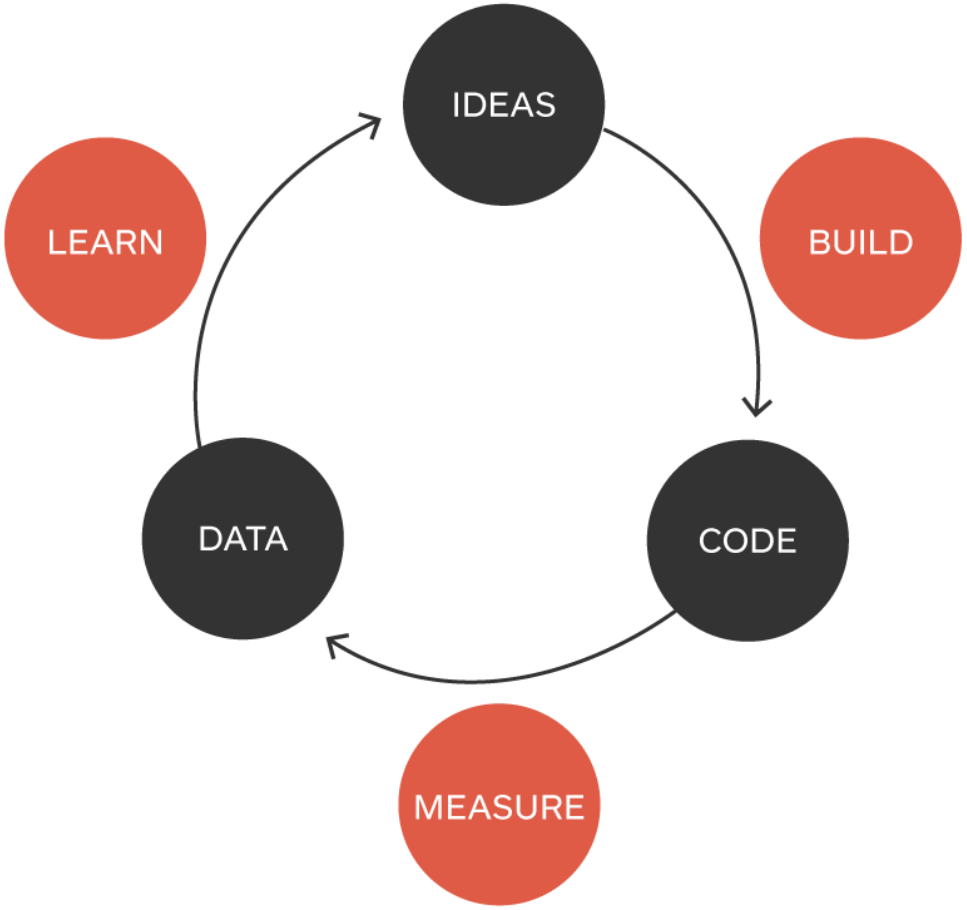
By the **time** you get it built, they'll want something new.

-- Steve Jobs



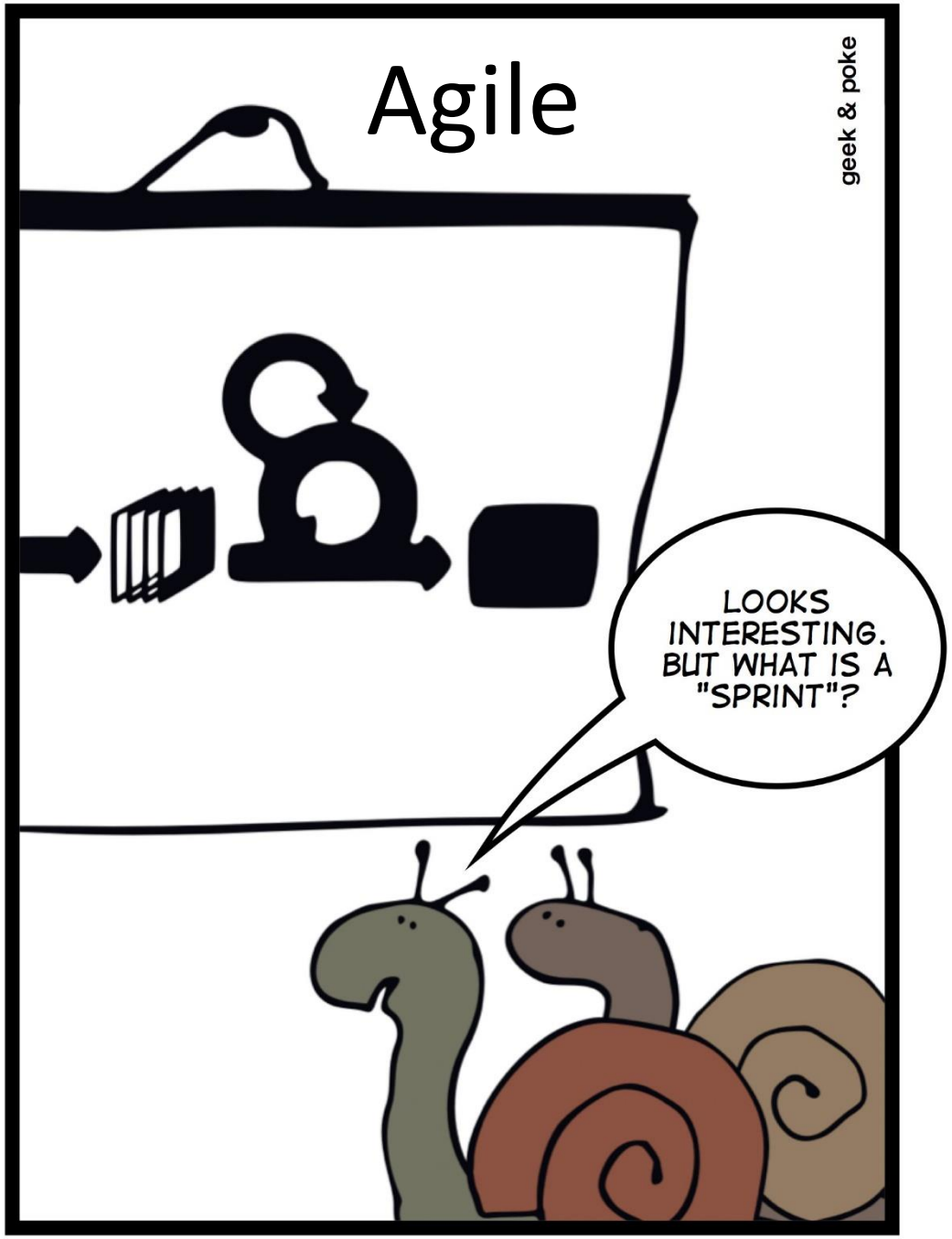
http://drgnmeme.com/business/lean-startup-machine/

Lean startup cycle



http://polisofialabs.com/en/approach.html

Agile



LEARNING AGILE

Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

Principles behind the Agile Manifesto

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence

Principles behind the Agile Manifesto

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software. ← #1

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. ← #3

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress. ← #7

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence

Twelve Principles of Agile Software

#1

Our highest priority is
to satisfy the customer
through early
and **continuous delivery**
of **valuable software**.

http://agilemanifesto.org/principles.html

Twelve Principles of Agile Software

#3

Deliver working software **frequently**, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

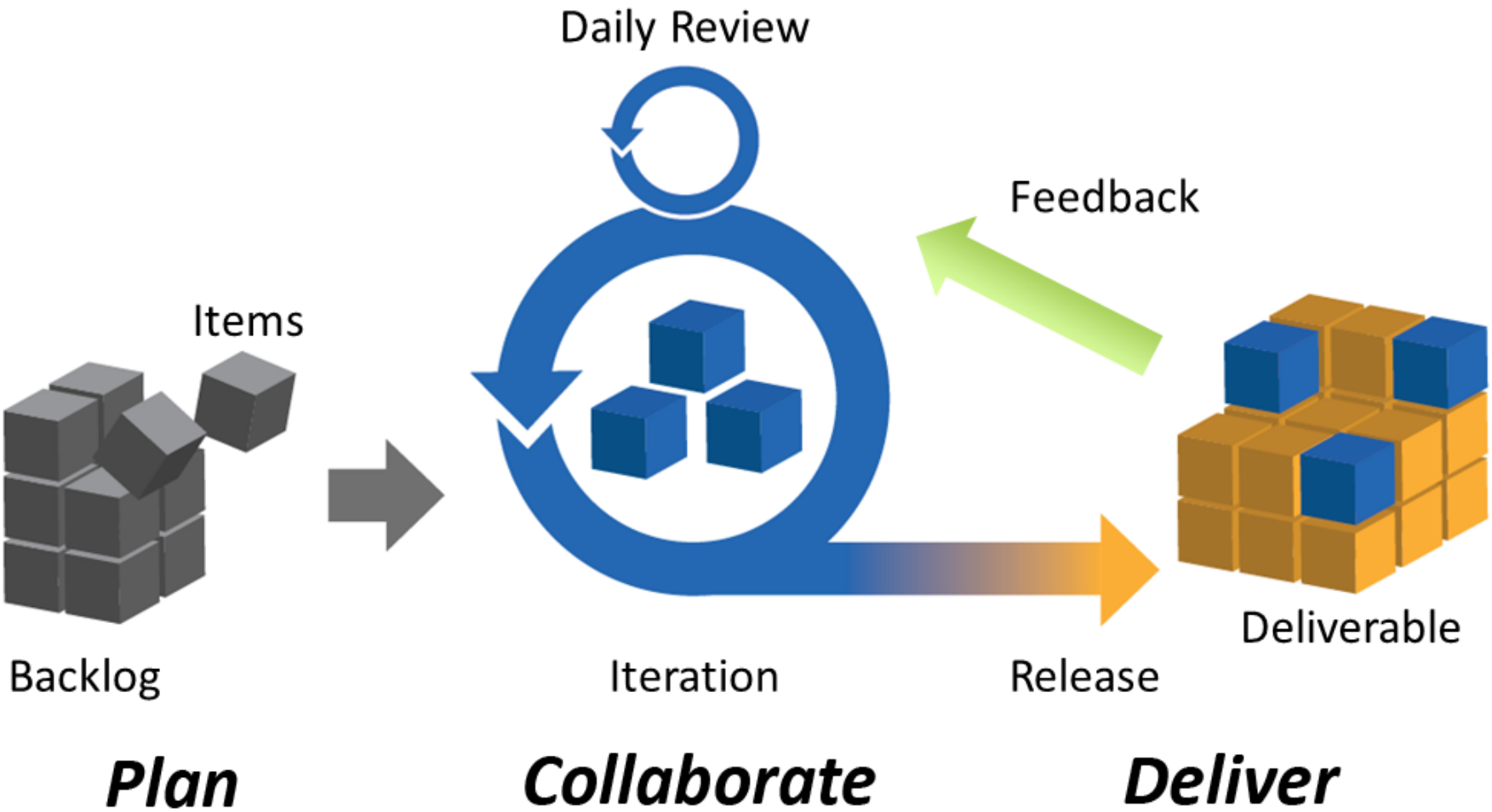
http://agilemanifesto.org/principles.html

Twelve Principles of Agile Software

#7

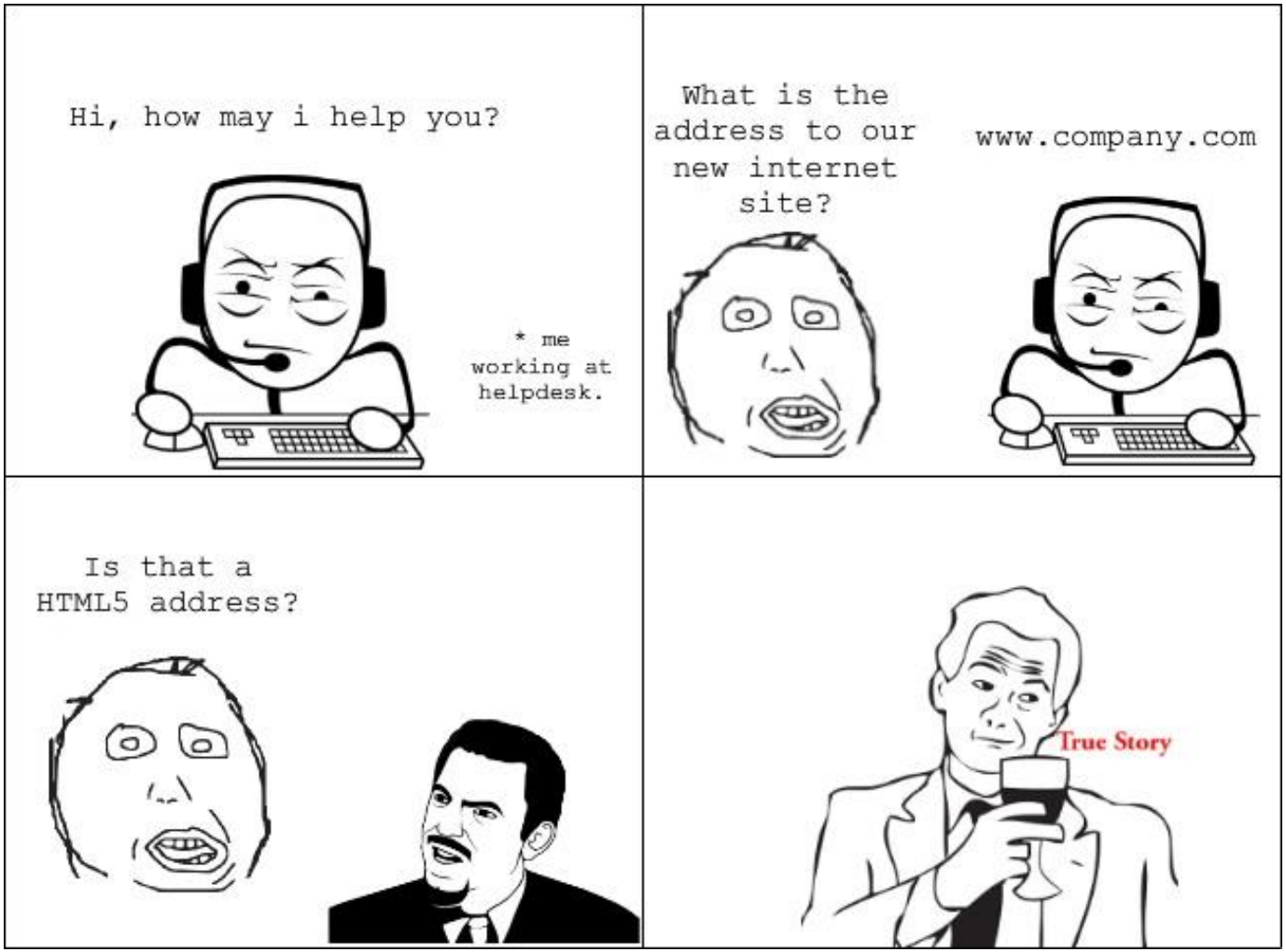
Working software is the primary measure of **progress**.

http://agilemanifesto.org/principles.html



Agile Project Management: Iteration

BuzzWords



http://faildesk.net/2012/09/26/buzz-words/

Continuous

DevOps

testing

deployment

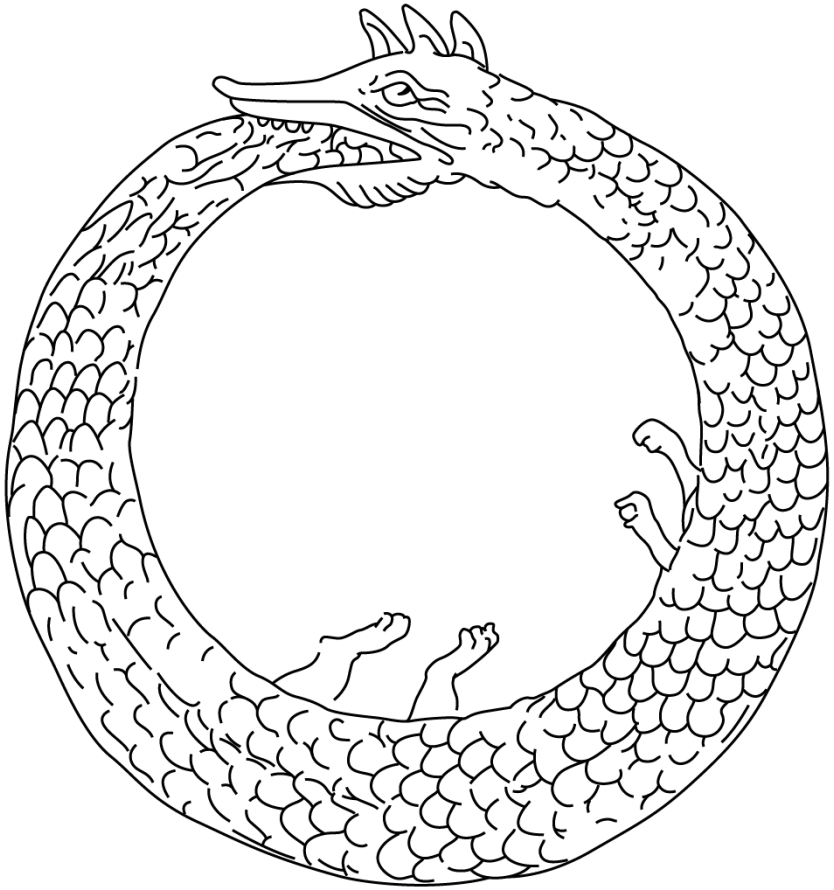
compilation

delivery

integration

improvement

Continuous - big picture



http://commons.wikimedia.org/wiki/File:Ouroboros1.png

Related concepts

Continuous **testing**

↑

Continuous **compilation**

Continuous testing **tools** for the .Net

- NCrunch
- Continuous Tests
- Giles
- SpecWatchr
- AutoTest.Net
- …

Continuous (**compilation** | **testing**) **is** not …

- *continuous* **deployment**
- *continuous* **delivery**
- *continuous* **integration**

continuous **deployment**

↑

continuous **delivery**

↑

continuous **integration**

continuous **deployment**

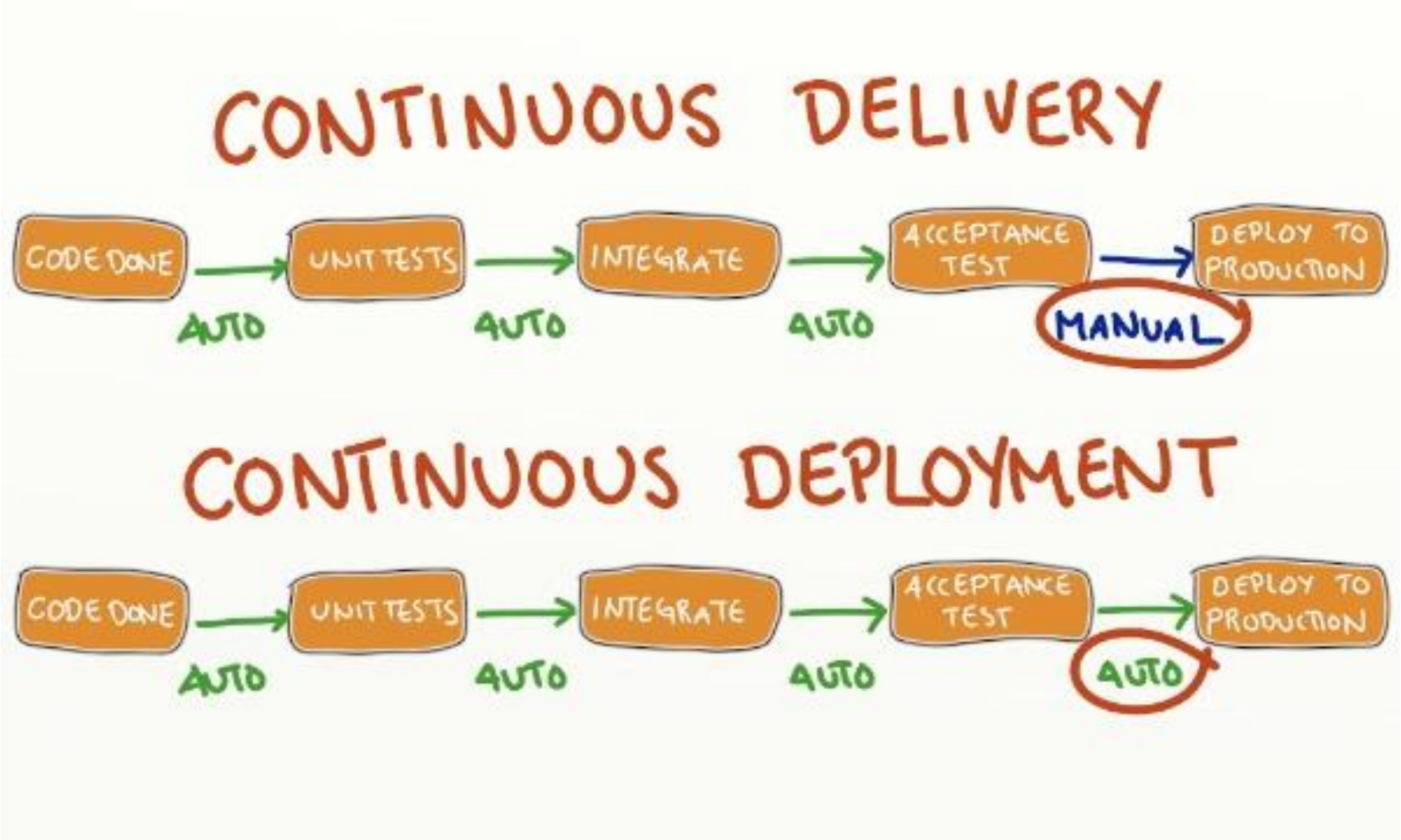
↑

We're here

continuous **delivery**

↑

continuous **integration**



http://blog.crisp.se/2013/02/05/yassalsundman/continuous-delivery-vs-continuous-deployment

改善 - „improvement”

- *Continuous*
  - **compilation**
  - **testing**

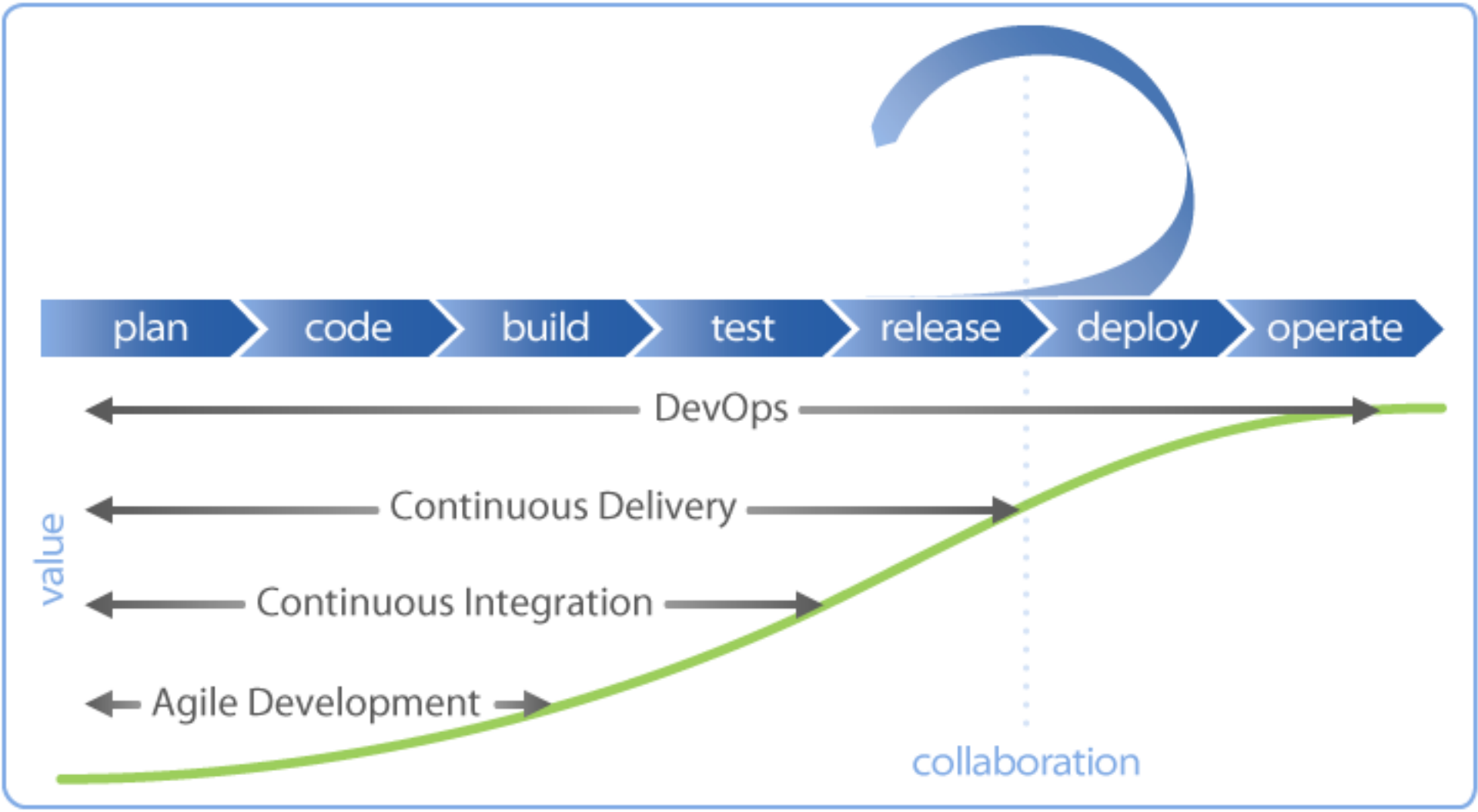
- *Continuous*
  - **deployment**
  - **delivery**
  - **integration**

改善 - „improvement”

- *Continuous*
  - **compilation**
  - **testing**
- *Continuous*
  - **deployment**
  - **delivery**
  - **integration**

Continuous

improvement



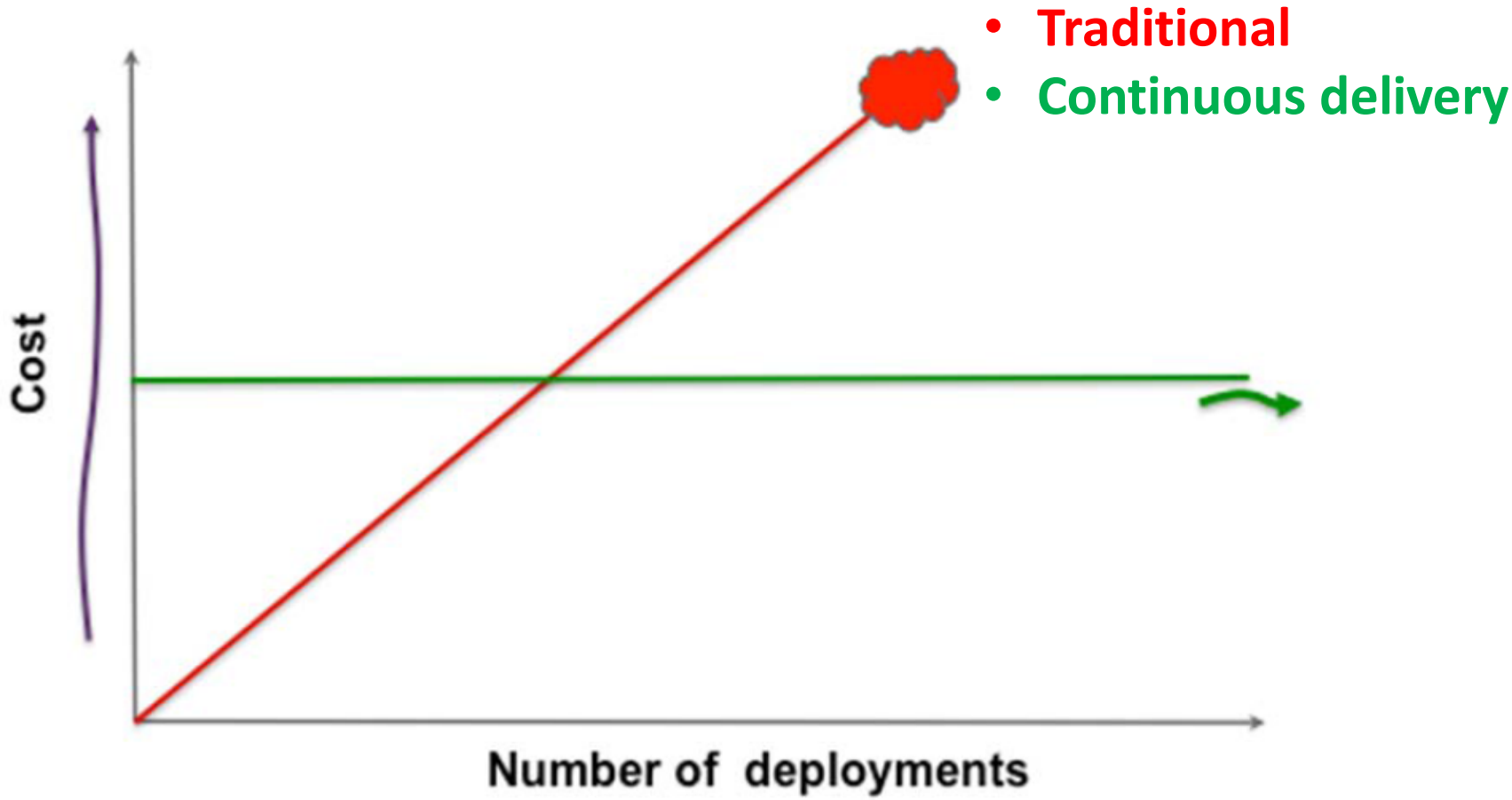
http://www.collab.net/solutions/devops

Benefits continuous delivery

- Empowering teams
- Reducing errors
- Lowering stress
- Deployment flexibility
- Practice makes perfect

Benefits CD

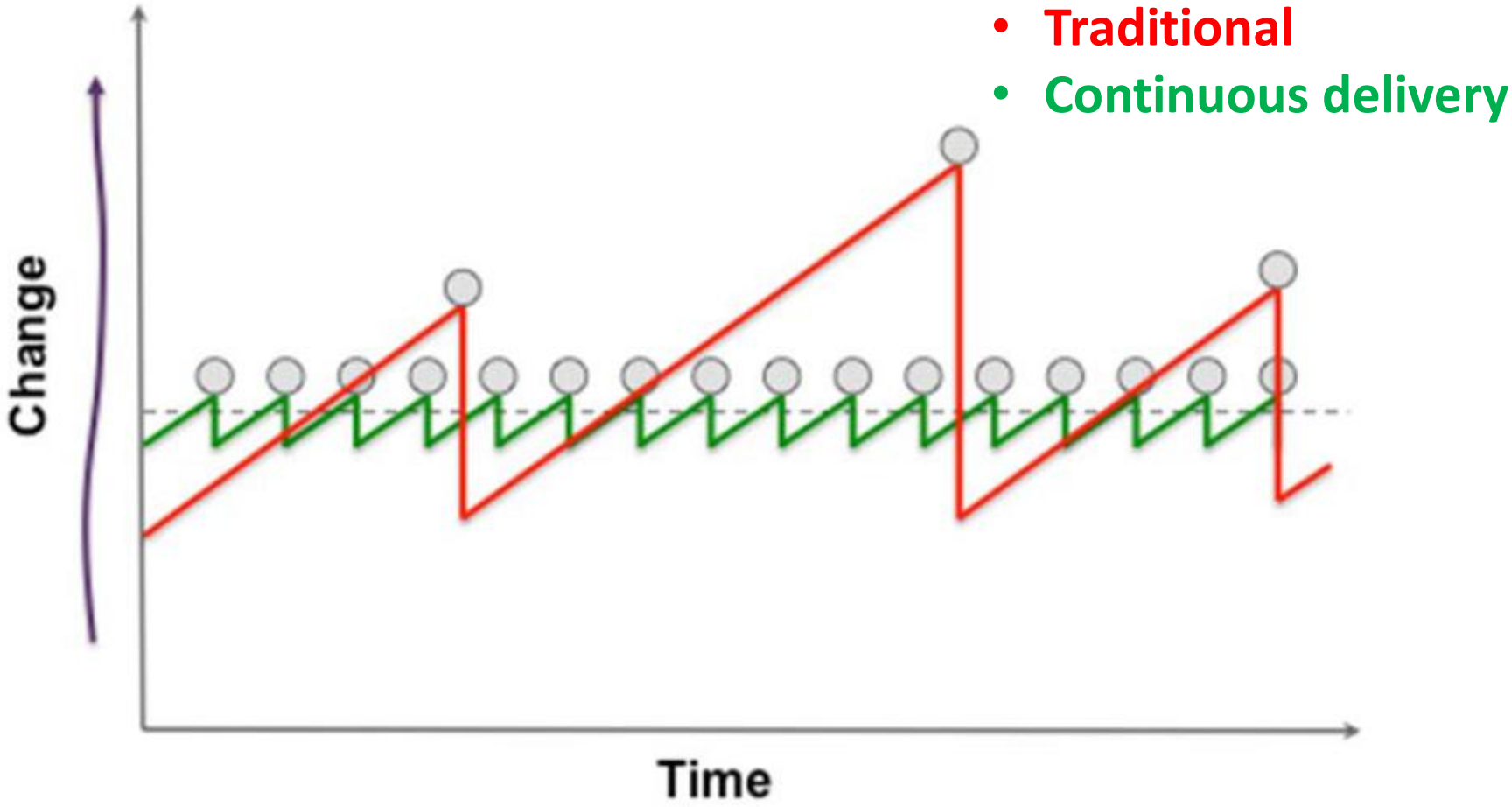
It **lowers** your **cost**



http://xebia.com/sites/default/bestanden/ContinuousDeliveryPaperXEBIA27112012.pdf

Benefits CD

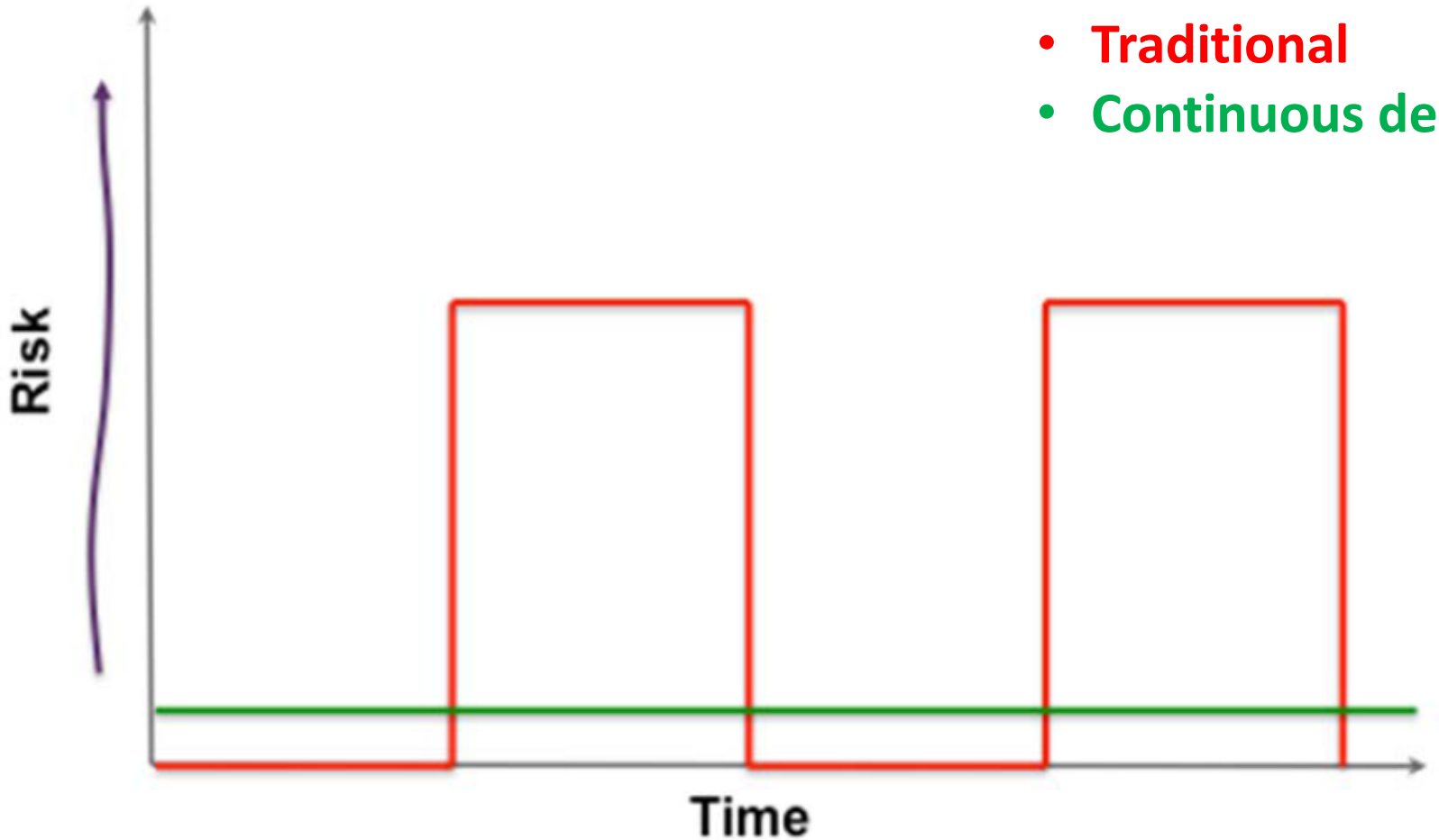
It **shortens** your **time** to **market**



http://xebia.com/sites/default/bestanden/ContinuousDeliveryPaperXEBIA27112012.pdf

Benefits CD

It mitigates your risk



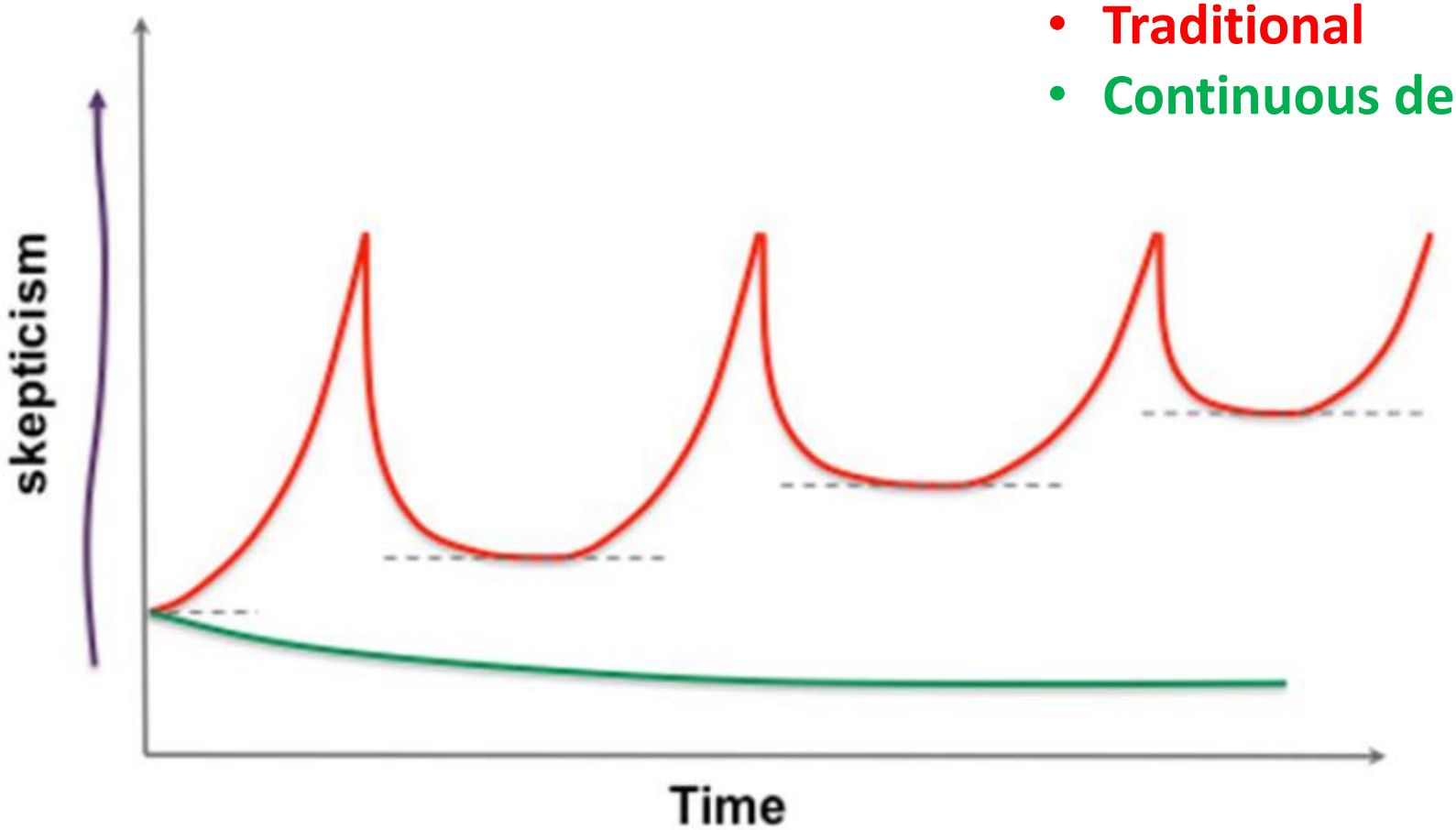
- **Traditional**
- **Continuous delivery**

http://xebia.com/sites/default/bestanden/ContinuousDeliveryPaperXEBIA27112012.pdf

Benefits CD

It (re-)builds trust within your IT organization

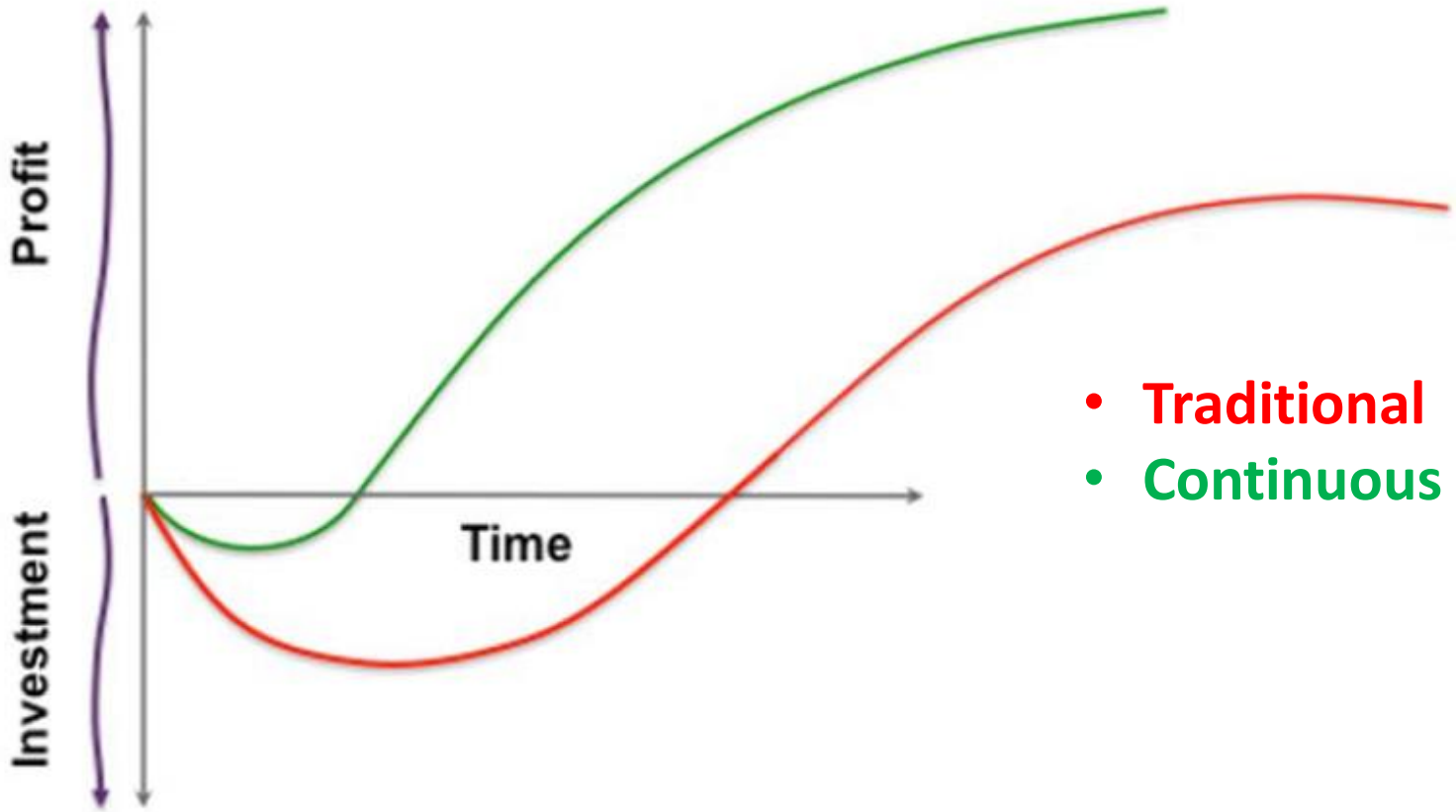
- **Traditional**
- **Continuous delivery**



http://xebia.com/sites/default/bestanden/ContinuousDeliveryPaperXEBIA27112012.pdf

Benefits CD

It helps you to understand your customer

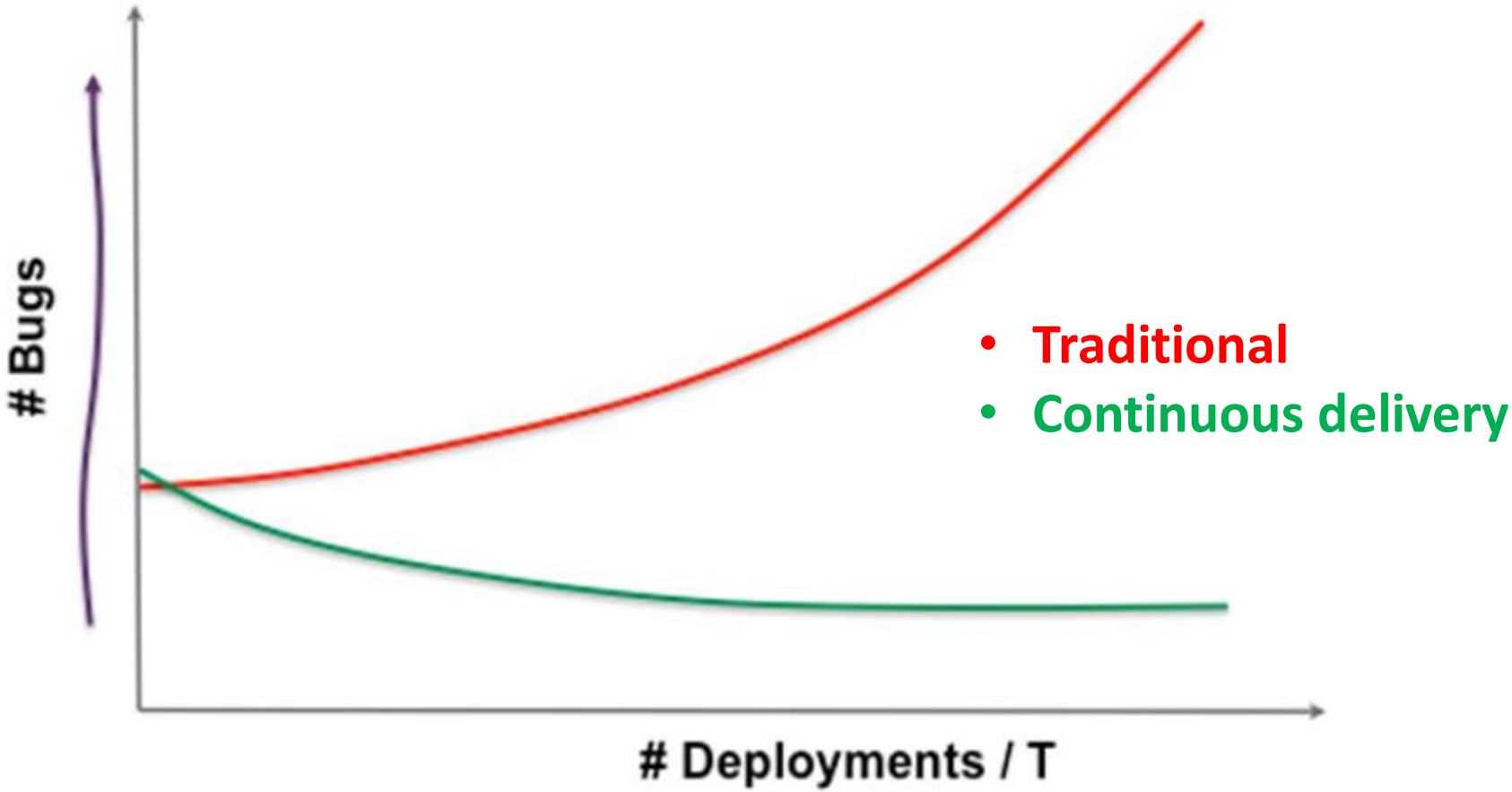


- **Traditional**
- **Continuous delivery**

http://xebia.com/sites/default/bestanden/ContinuousDeliveryPaperXEBIA27112012.pdf

Benefits CD

It raises the overall quality of your application

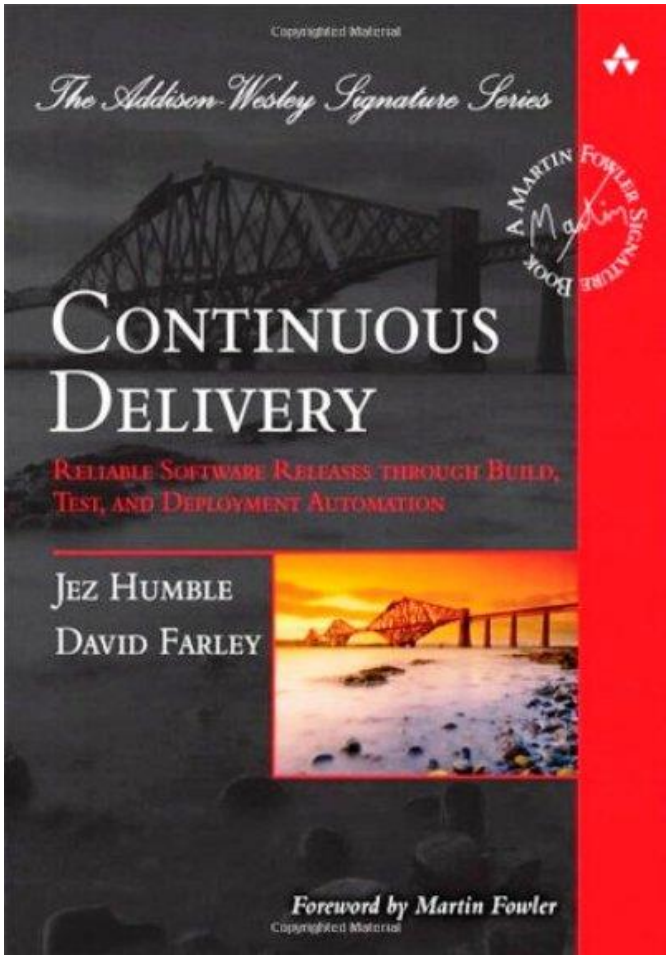


http://xebia.com/sites/default/bestanden/ContinuousDeliveryPaperXEBIA27112012.pdf

Jez Humble

http://blogs.agilefaqs.com/2012/12/05/continuous-delivery-workshop-by-jez-humble-agile-india-2013/

Continuous Delivery



http://blogs.agilefaqs.com/2012/12/05/continuous-delivery-workshop-by-jez-humble-agile-india-2013/

Continuous delivery

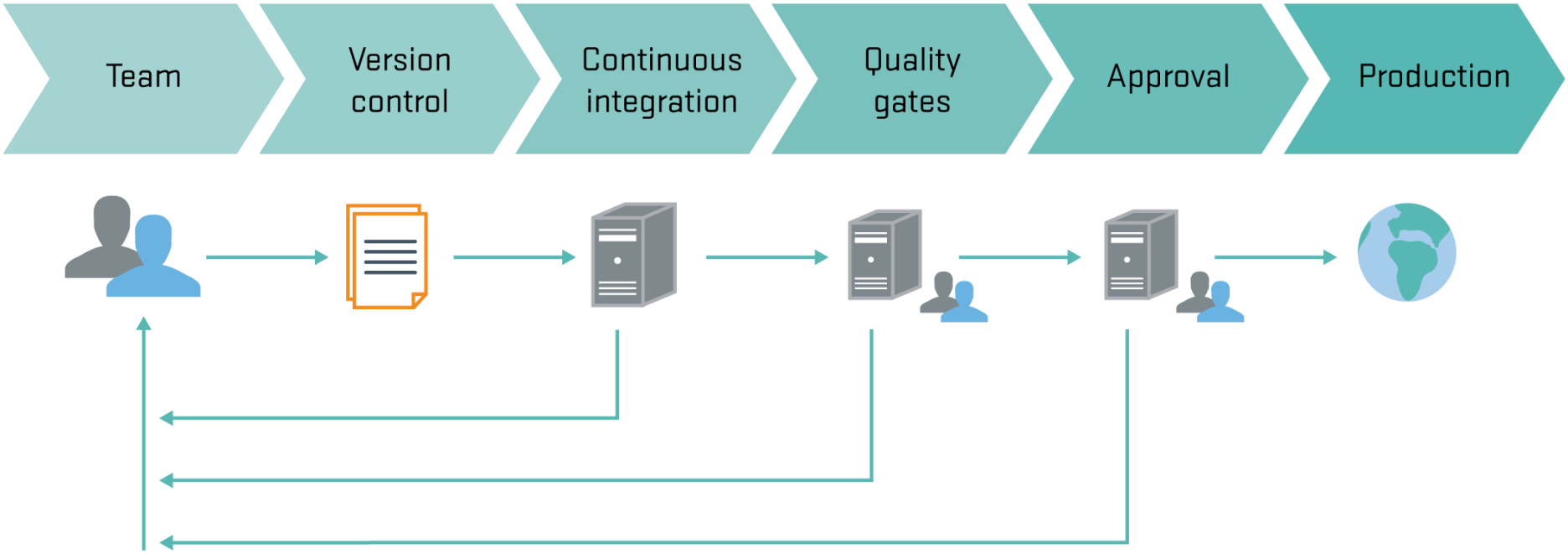
It is a **design practice** used in software development to **automate** and **improve** the **process** of software delivery.

http://en.wikipedia.org/wiki/Continuous_delivery

A pipeline

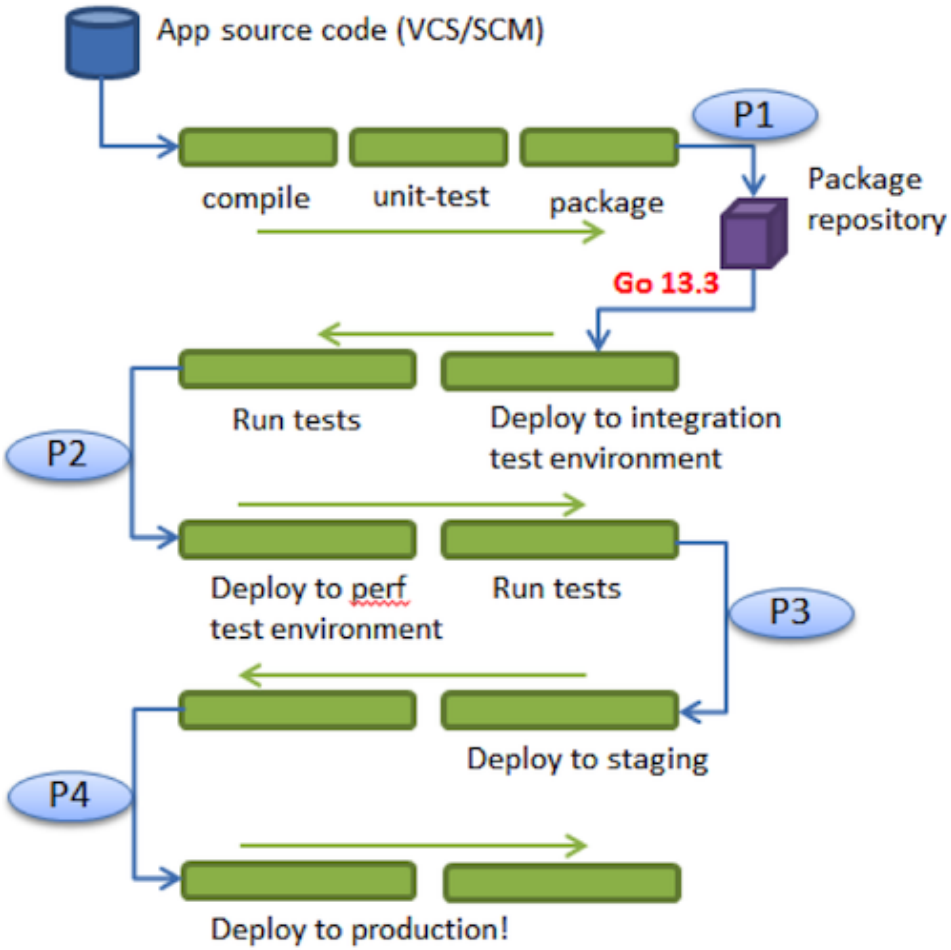
http://www.zinccharare.co.zw/2012/08/23/new-oil-pipeline-on-the-cards/

Continuous delivery pipeline



http://www.opscode.com/solutions/continuous-delivery/

Continuous delivery pipeline

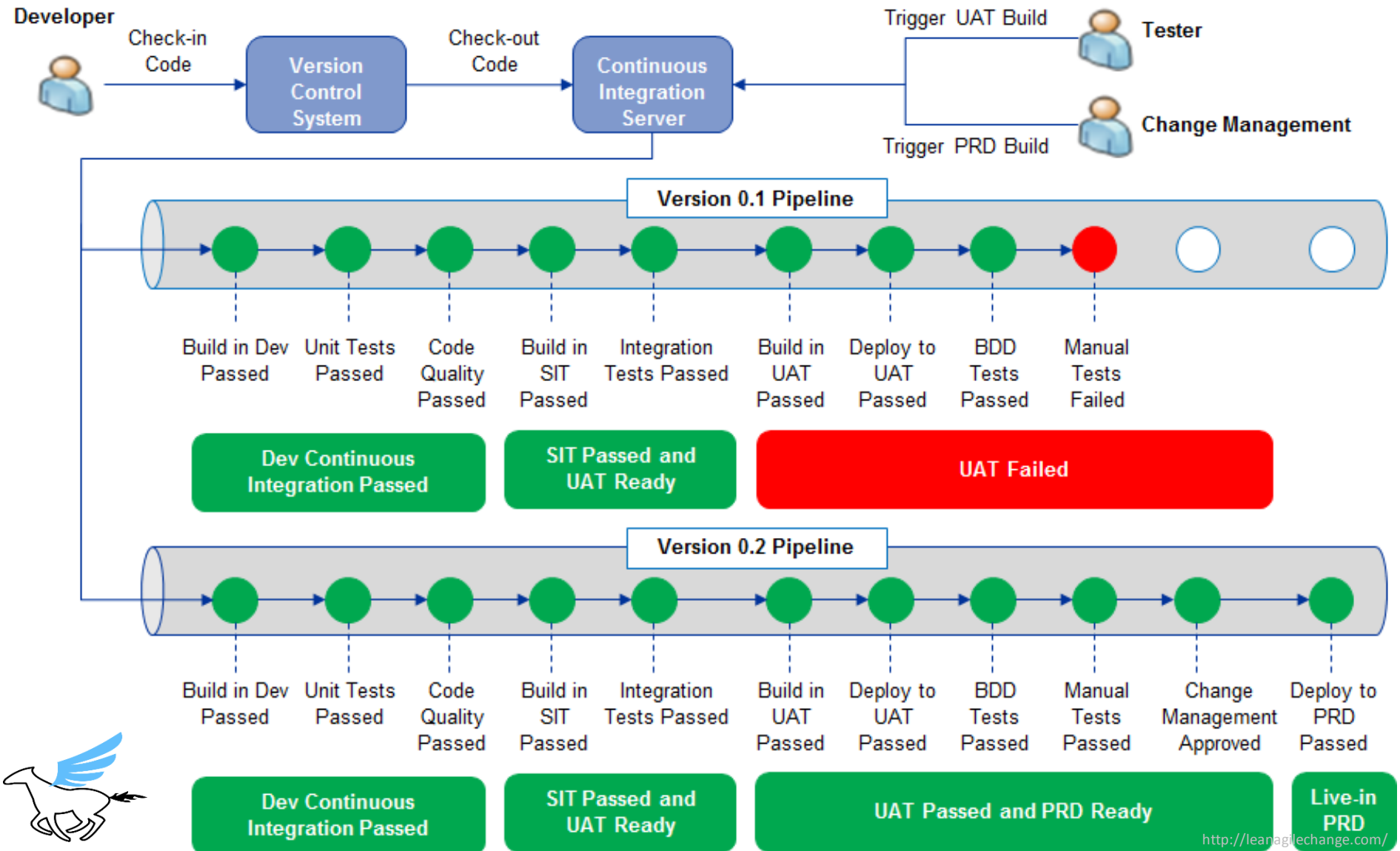


http://www.thoughtworks.com/insights/blog/build-promotion-go

We need to go deeper

http://imgur.com/gallery/peXGg

Continuous delivery pipeline



Enterprise challenges

- Large, monolithic applications
- Low levels of automation
- Contended environments
- Release management requirements
- Scaling up jobs
- Job ownership and security

http://www.infoq.com/articles/preparing-for-cd-in-the-enterprise

Tools

http://photo.elsoar.com/construction-tools-collection-pictures.html

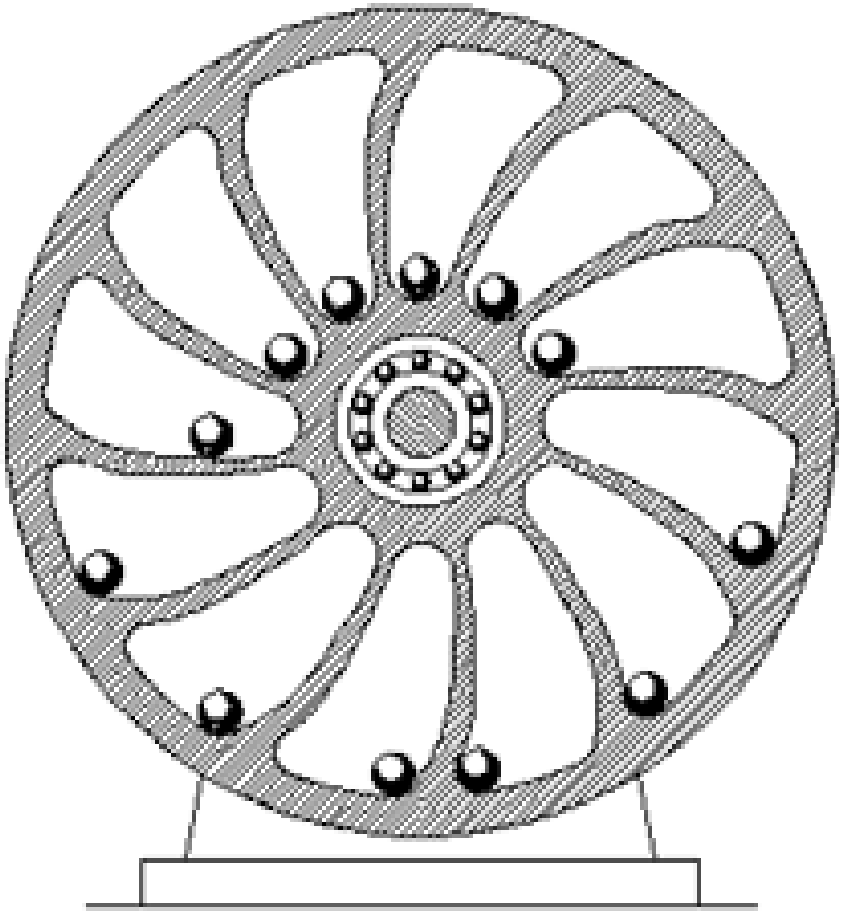
Tools

Puppet
Octalforty-wizardby
NDepend
Kalistick
PowerShell
Vagrant
Ansible
TFS
VirtualBox
git
xUnit
NuGet
MbUnit
MSTest
HyperV
MSpec
jMetter
RoundhouseE
.NUnit
Chef
Selenium
SubSonic
Jenkins
AppDynamics
TeamCity

Tools

- **Virtualization:** HyperV, VirtualBox, Vagrant, …
- **Source control**: TFS, Git, SVN, Mercurial, ...
- **Build**: MS build, NAnt, ...
- **Dependency managment**: NuGet, Artifactory, ...
- **Testing**: Selenium, NUnit, xUnit, MsTest, Twist, ...
- **Infrastructure**: Chef, …

Patterns



http://tainy.net/11354-vechnyj-dvigatel.html

Deployment is not a **Release**.

*Release is a decision**.

* This is the decision of a person who is responsible for this.

Deploy early

Deployment is **never easy**, so try to deploy as soon as possible to **remove** all **roadblocks**.

Create a **repeatable**, **reliable** process for releasing software.

Automate almost **everything**.

Have a **clone** of your **production** environment.

Have **everything**
under **source control**.

Also deployment artefacts.

If It **hurts**,

do it **more frequently**,

and bring the pain forward.

Build **quality in**

(catch defects as early in the delivery process as possible).

Log **failed** and **successful** builds.

Done means r**eleased.**

Anti-patterns

http://photos.imageevent.com/motorbiker

Deploying software **manually.**

http://www.keithmcdonnell.net/continuous_delivery_notes.html

Deployment rarely

(avoid late contact with reality).

Deploying to a *production-like* environment **only after** development is **complete**.

http://www.keithmcdonnell.net/continuous_delivery_notes.html

Manual environment **configuration**.

A **code freeze** ceremony.

Not repeatable proces.

Slight differences.

In summary

- **CD** offers great benefits to both business and technical people within an organization.
- **CD** goes hand in hand with Agile, the method of developing software in small increments.
- Without a culture that emphasizes active collaboration between **dev** and **ops**, **CD** becomes **very difficult** - if not impossible.

http://puppetlabs.com/blog/continuous-delivery-devops-jez-humble-its-all-about-the-customer

Books

- **Continuous Delivery:** Reliable Software Releases through Build, Test, and Deployment Automation by *Jez Humble, David Farley.*
- **Lean Enterprise**: Adopting Continuous Delivery, DevOps, and Lean Startup at Scale by *Jez Humble, Barry O'Reilly, Joanne Molesky.*
- **Release It!**: Design and Deploy Production-Ready Software (Pragmatic Programmers) by *Michael T. Nygard.*
- **Continuous Delivery and DevOps**: A Quickstart guide by *Paul Swartout.*

Links

- http://www.thoughtworks.com/products/webinars/continuous-delivery-iis-nuget-chef-and-tfs
- http://www.thoughtworks.com/insights/blog/challenges-implementing-enterprise-continuous-delivery
- http://msdn.microsoft.com/en-us/library/dd647551.aspx