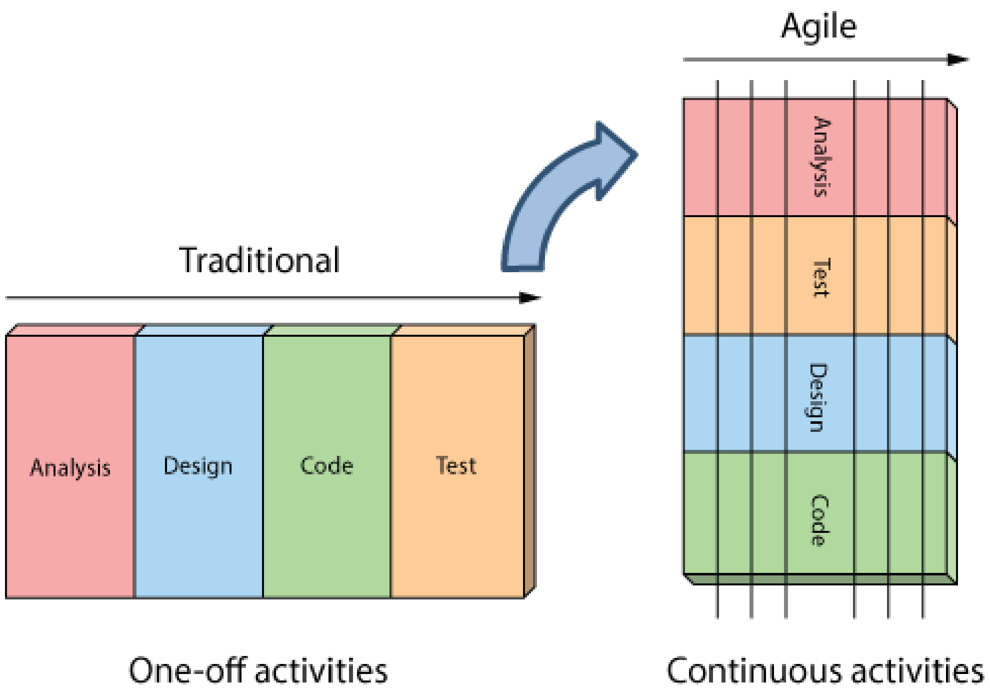
NEW BOOK
Automated Testing

How is Agile different?

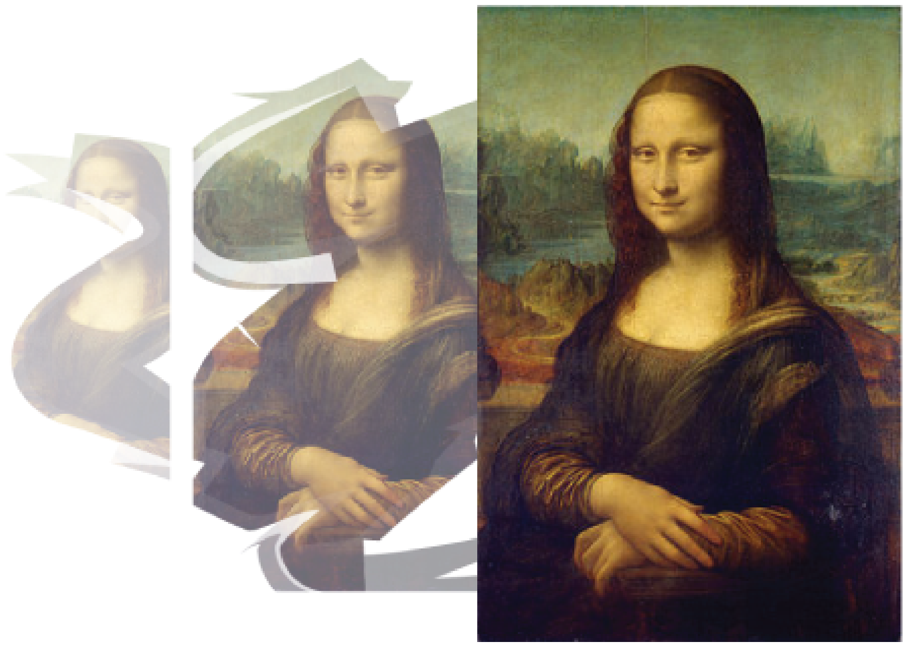
Analysis, design, coding, and testing are continuous activities

You are never done analysis, design, coding and testing on an Agile project. So long as there are features to build, and the means to deliver them, these activities continue for the duration of the project.



Development is iterative

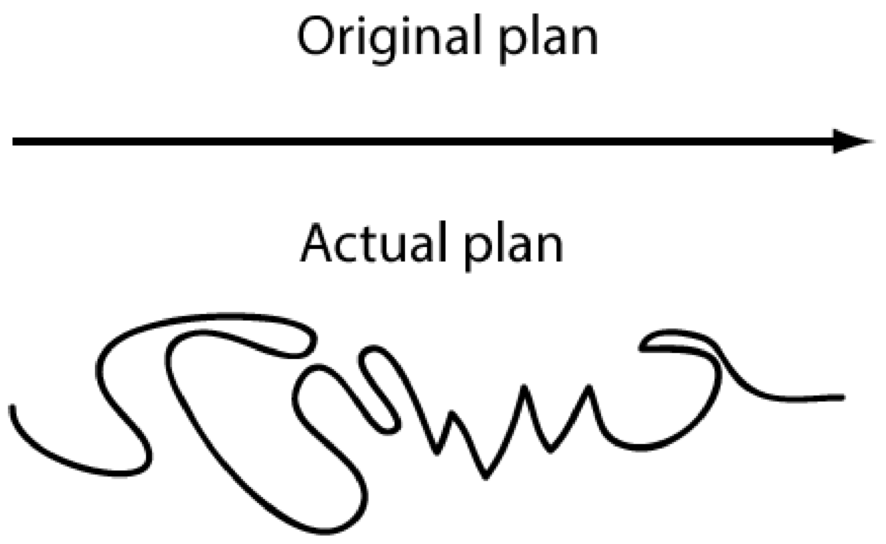
Iterative development means starting with something really simple, and adding to it incrementally over time.



It means evolving the architecture, accepting that your requirements are going to change, and continuously refining and tweaking your product as you go.

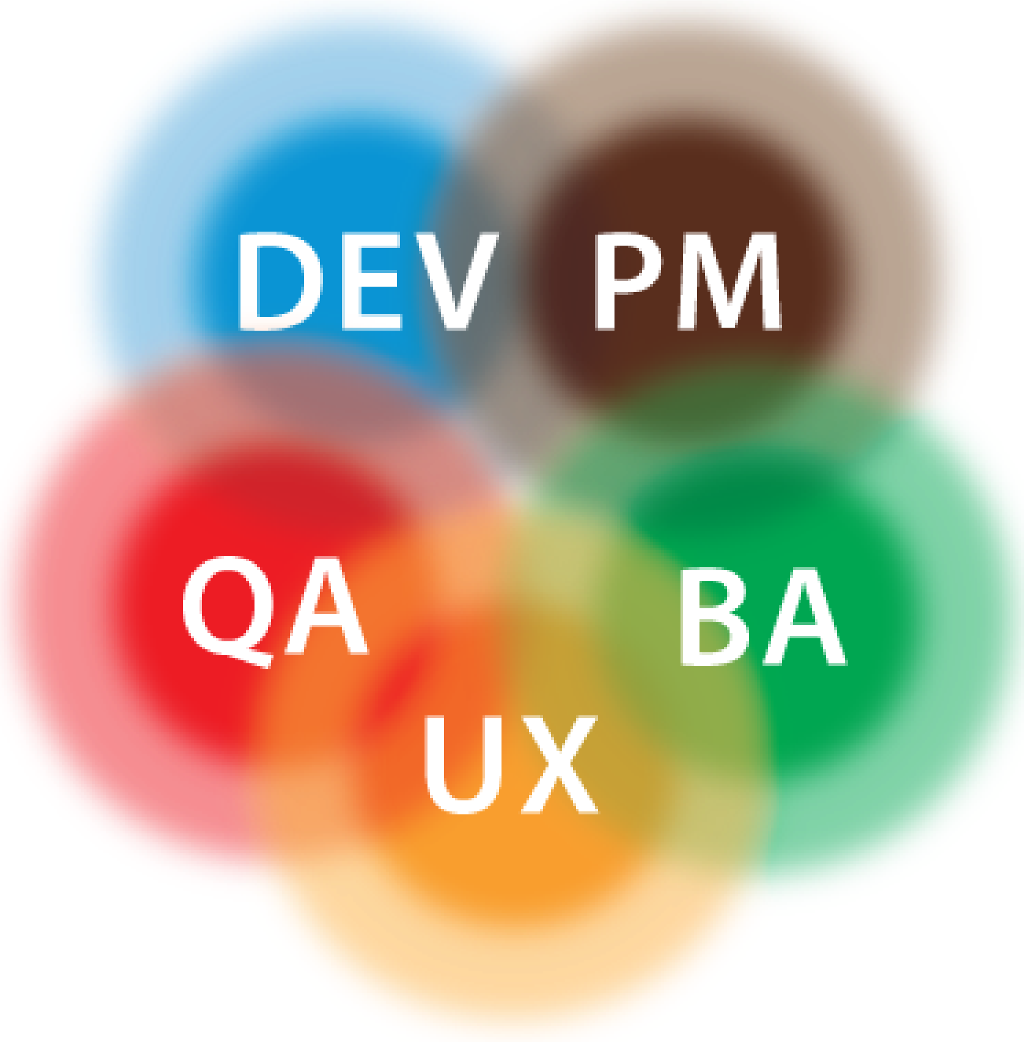
Planning is adaptive

When reality disagrees with their plans, Agilists find it easier to change their plans than reality. They call this adaptive planning.



And while there are many ways to changes plans, the preferred way is to flex on scope.

Roles blur



Roles really blur on Agile projects. When it's done right, joining an Agile team is a lot like working in a mini-startup. People pitch in and do whatever it takes to make the project successful—regardless of title or role.

Yes, people still have core competencies, and, yes, they generally stick to what they are good at. But on an agile project, narrowly defined roles like analyst, programmer, and tester don't really exist - at least not in the traditional sense.

Scope can vary

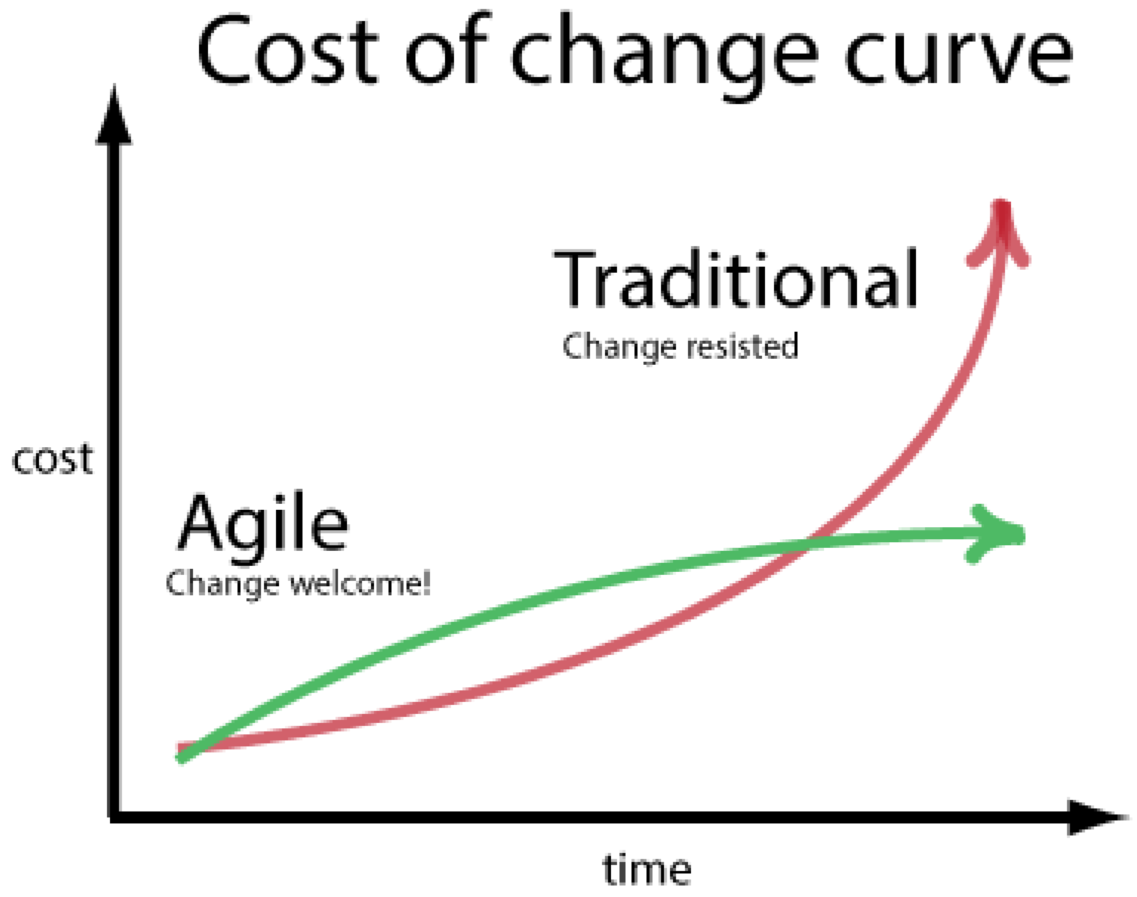
Time
Budget
Quality
Scope

Agile deals with the age old problem of having too much to do and not enough time by doing less.

By fixing time, budget, and quality, and being flexing around scope, Agile teams maintain the integrity of their plans, work within their means, and avoid the burn out, drama, and dysfunction traditionally associated with our industry.

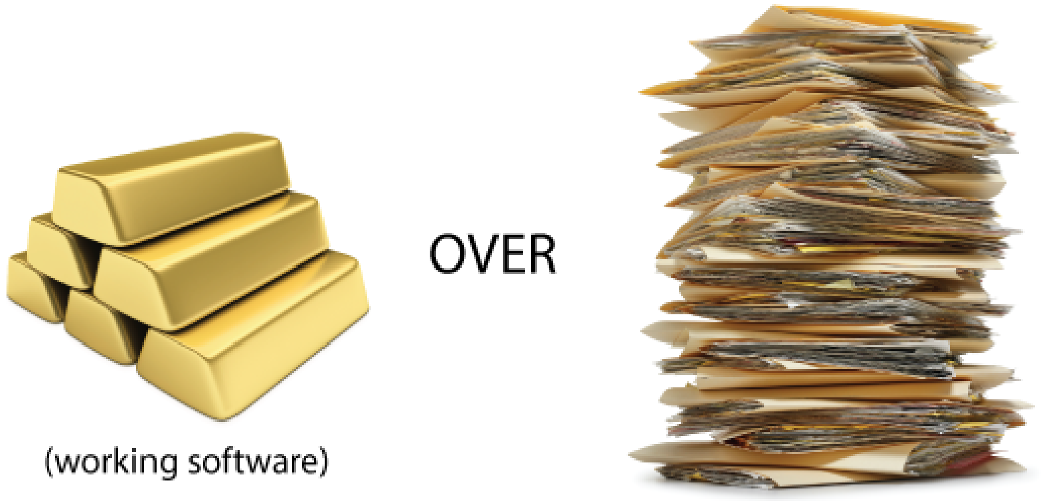
Requirements can change

Traditionally change has been shunned on software projects because of it's high perceived cost late in the game. Agile challenges this notion and believes the cost of change can be relatively flat.



Through a combination of modern software engineering practices, and open and honest planning, Agilsts accept and embrace change even late in delivery process.

Working software is the primary measure of success



The rate at which teams can turn their customer's wishes into working software is how Agilists measure productivity. Project plans, test plans, and analysis artifacts are all well and good but Agilists understand they in themselves are of no value to the end customer.

← Previous (/how_does_it_work)

Next → (/agile_myths)

Continuous Integration (/continuous_integration)

Test Driven Development (/test_driven_development)

CONCEPTS

Burndown Charts (/burndown)

Cone of Uncertainty (/cone_of_uncertainty)

Management by Miracle (/management_by_miracle)

Three Simple Truths (/three_simple_truths)

XPISMS

Bill of Rights (/bill_of_rights)

YAGNI (/yagni)

Yesterday's Weather (/yesterdays_weather)

The Simplest Thing (/simplest_thing)

Testing (/could_possibly_break)

Production Readiness (/production_readiness)