

ASTRON

Netherlands Institute for Radio Astronomy



university of
 groningen

kapteyn astronomical
 institute

Agile and DevOps from the trenches at ASTRON

ADASS XXVIII, 11-15 November 2018

How it all started

- In 2011, LOFAR software development was in crisis
 - Focus had been on getting the instrument to work
 - Little time was spent to make it ready for operations
 - Pressure on the software team to deliver new features

How it all started

- In 2011, LOFAR software development was in crisis
 - Focus had been on getting the instrument to work
 - Little time was spent to make it ready for operations
 - Pressure on the software team to deliver new features
- But
 - Features lacked requirements
 - Huge Technical Debt

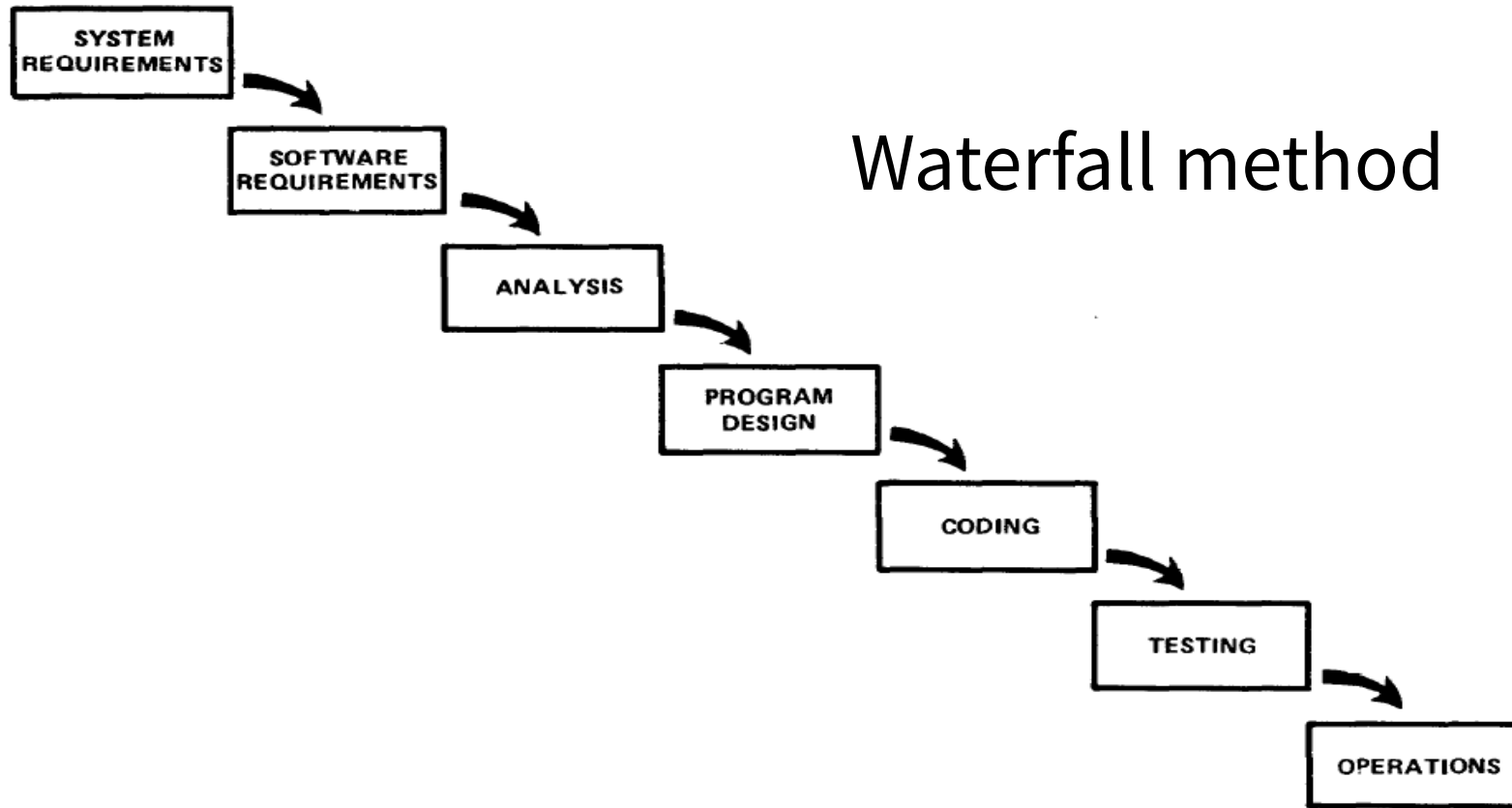


How it all started

- In 2011, LOFAR software development was in crisis
 - Focus had been on getting the instrument to work
 - Little time was spent to make it ready for operations
 - Pressure on the software team to deliver new features
- But
 - Features lacked requirements
 - Huge Technical Debt
- Something needed to change ...



Traditional Software Development



Royce 1970, © IEEE

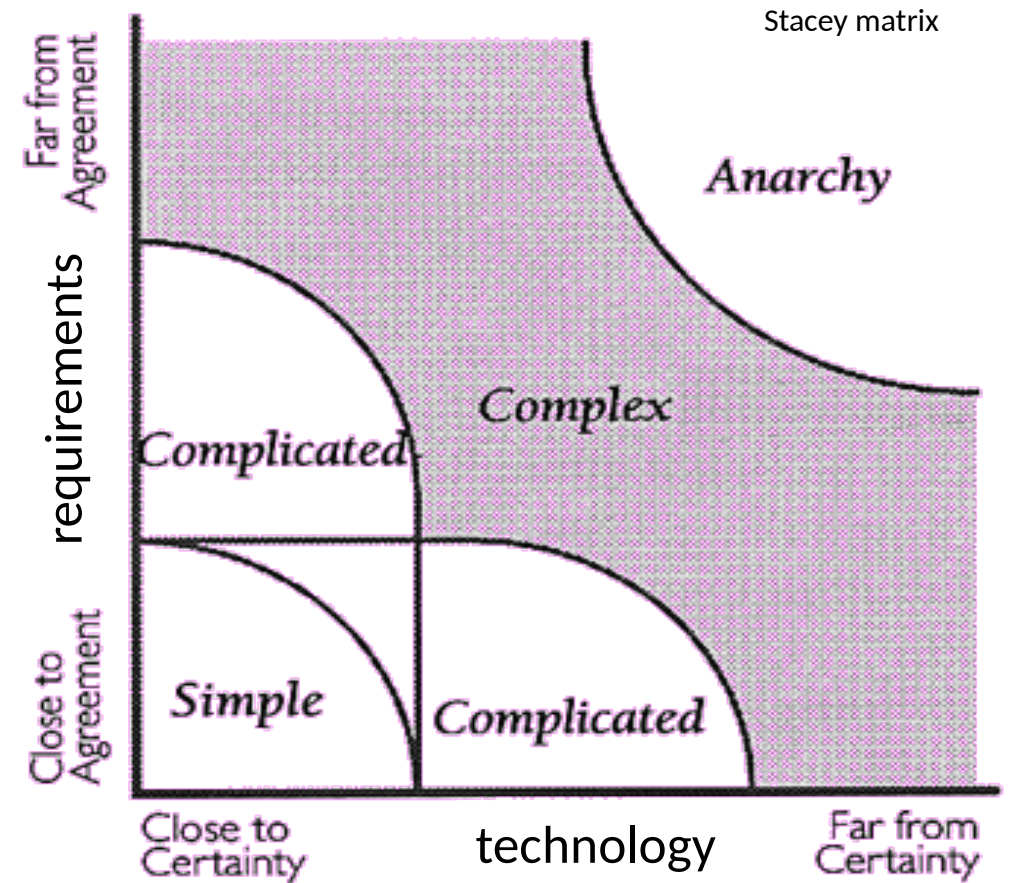
Why does Waterfall not work?

Why does Waterfall not work?

Complexity!

Why does Waterfall not work?

Complexity!

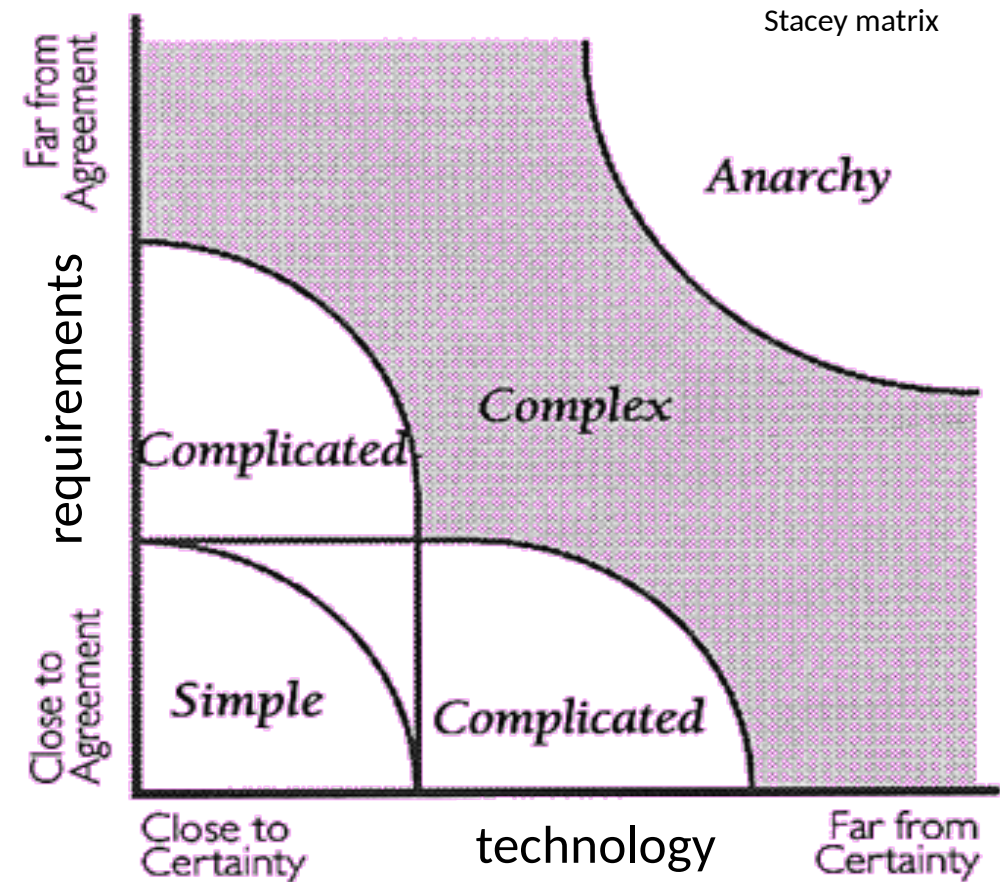


Why does Waterfall not work?

Complexity!

- Waterfall is good for Simple* projects
- Agile works better for Complicated and Complex projects

* Simple does not mean Easy

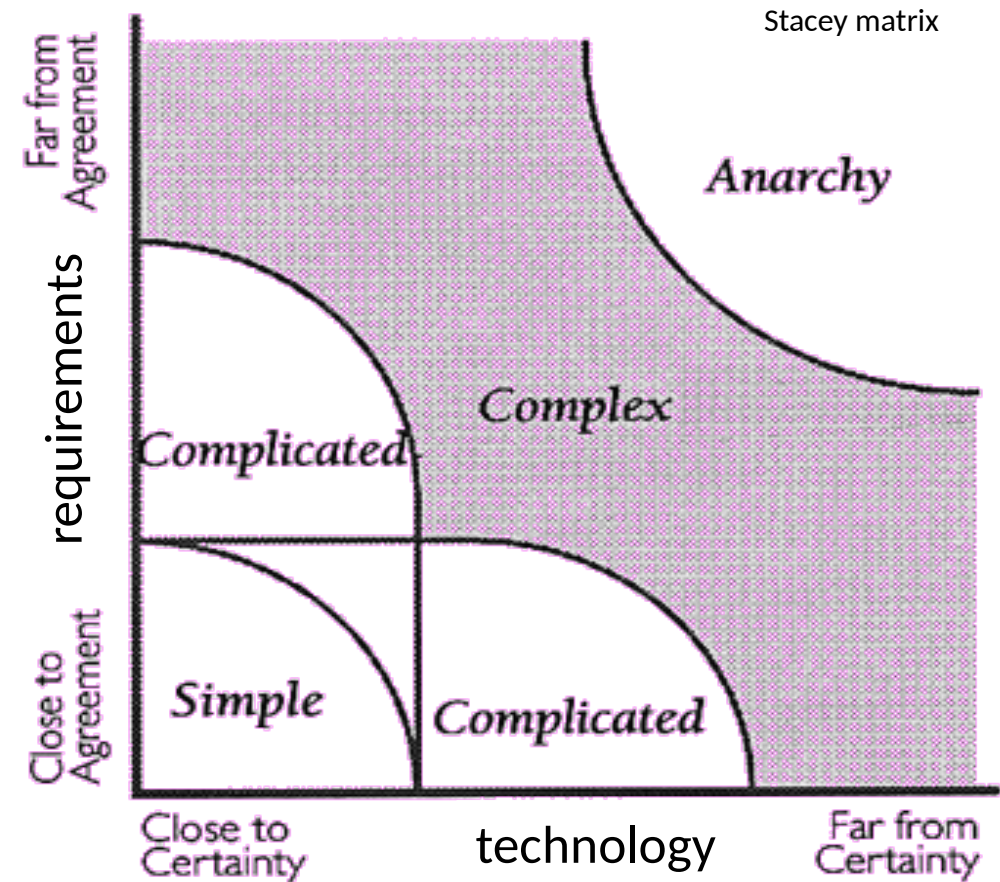


Why does Waterfall not work?

Complexity!

- Waterfall is good for Simple* projects
- Agile works better for Complicated and Complex projects
- Anarchy should be avoided wherever possible

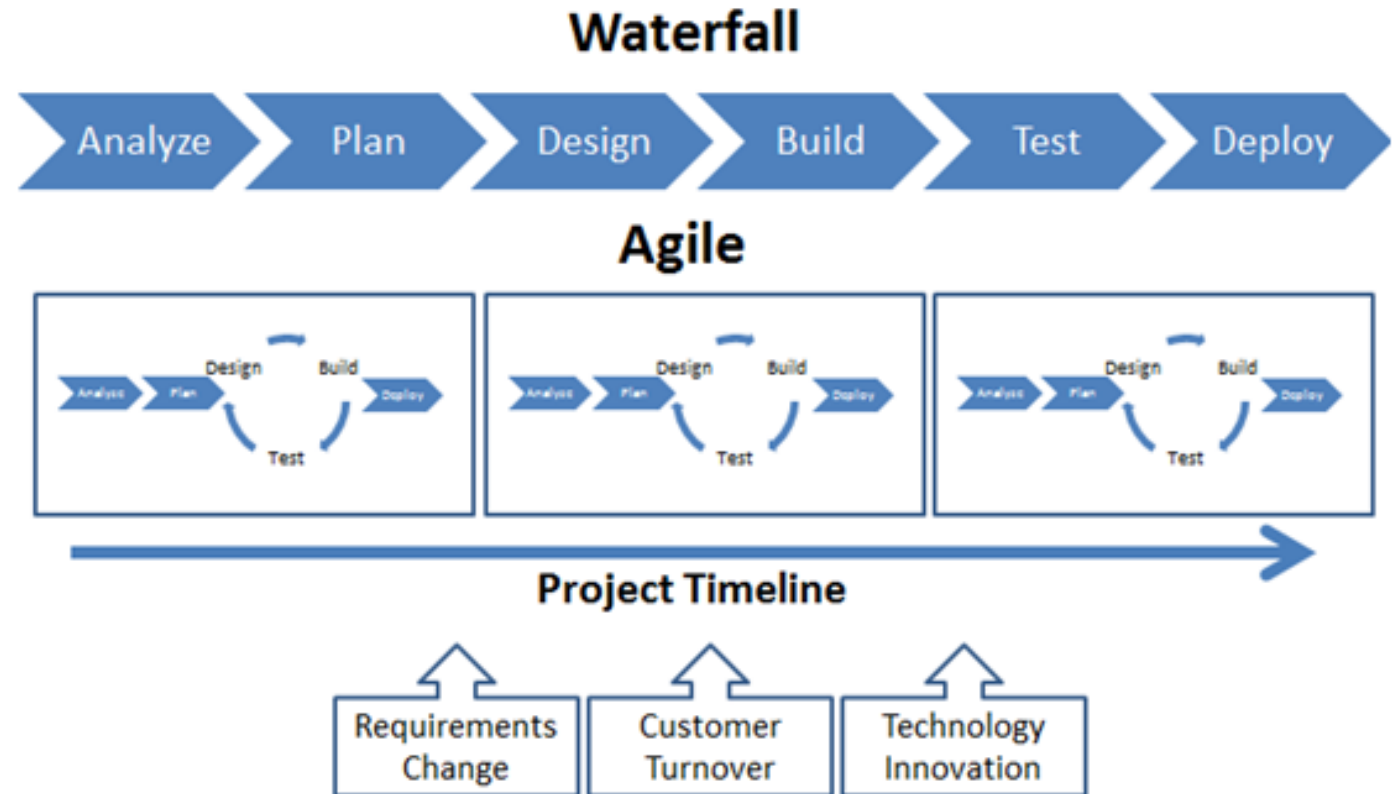
* Simple does not mean Easy



Waterfall vs Agile

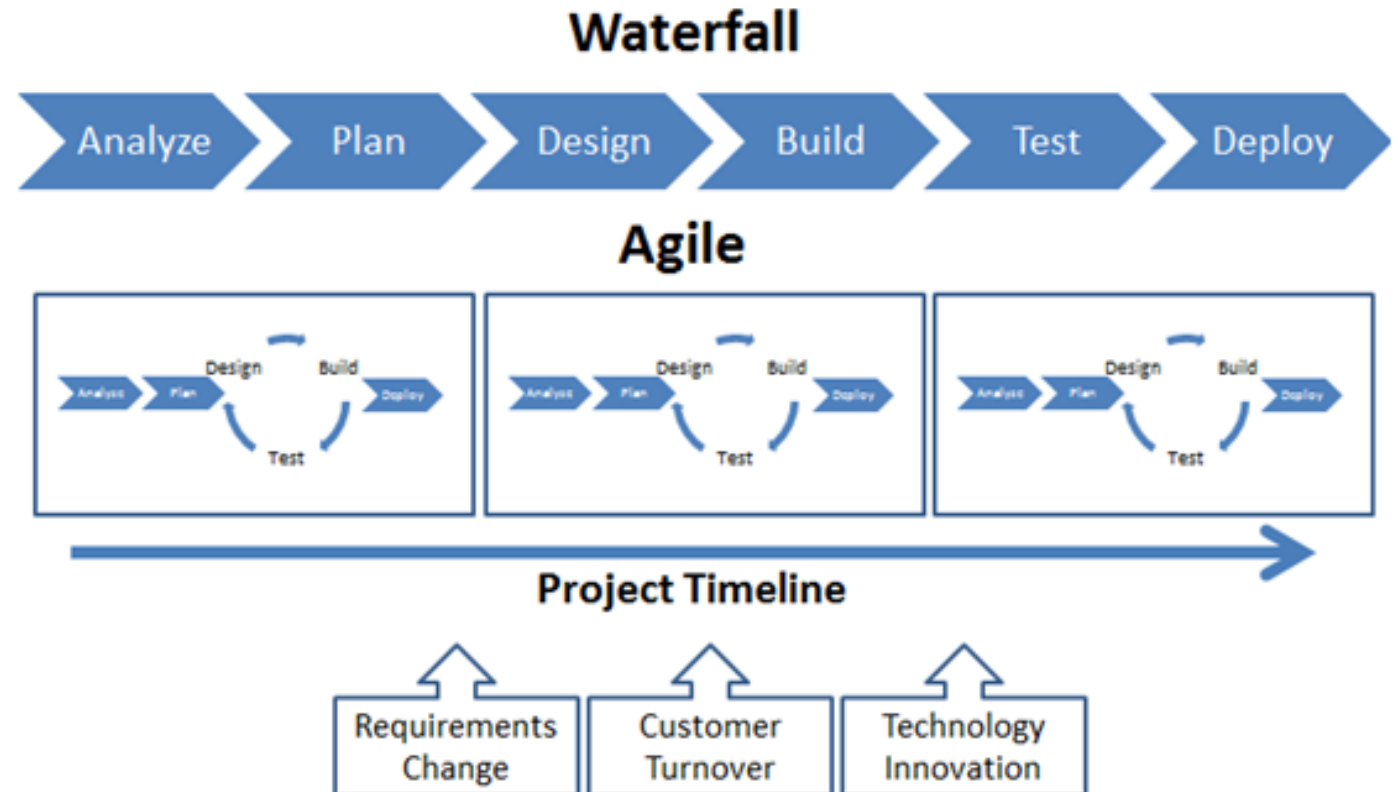
Waterfall vs Agile

- Cyclic approach
- You still use Waterfall but with (very) short iterations
- This makes you *Agile*, because you can easily adapt to change.



Waterfall vs Agile

- Cyclic approach
- You still use Waterfall but with (very) short iterations
- This makes you *Agile*, because you can easily adapt to change.
- But Agile is more ...



Agile Software Development & Scrum

Manifesto for Agile Software Development

Agile Software Development & Scrum

Manifesto for Agile Software Development

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

Agile Software Development & Scrum

Manifesto for Agile Software Development

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

Scrum is a *framework* that aims to implement these Agile principles

Agile Software Development & Scrum

Does Agile/Scrum work in a scientific environment?

Agile Software Development & Scrum

Does Agile/Scrum work in a scientific environment?

- Reasons why it could work:
 - Projects are generally complex
 - Requirements constantly change (both user and system)

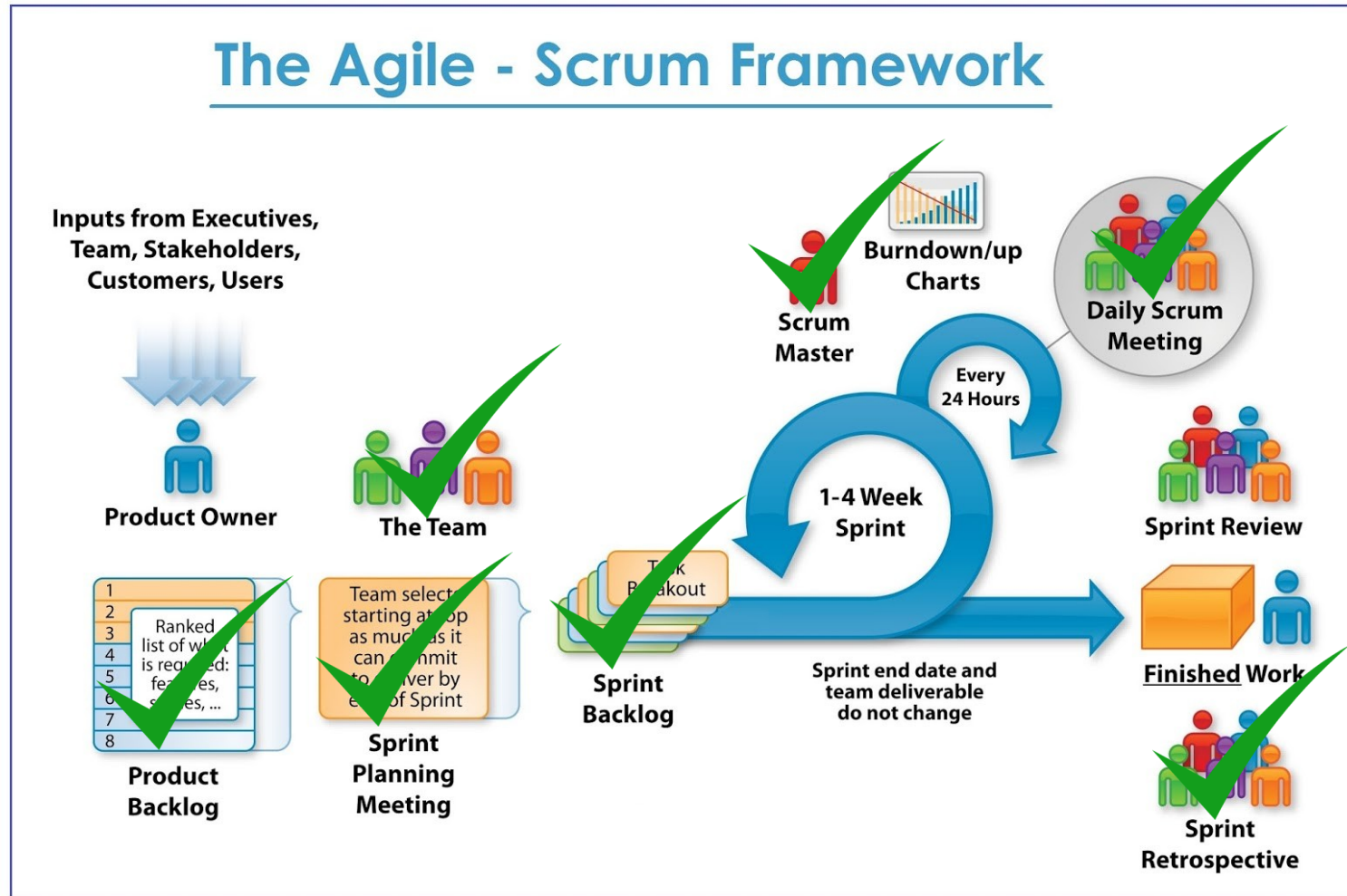
Agile Software Development & Scrum

Does Agile/Scrum work in a scientific environment?

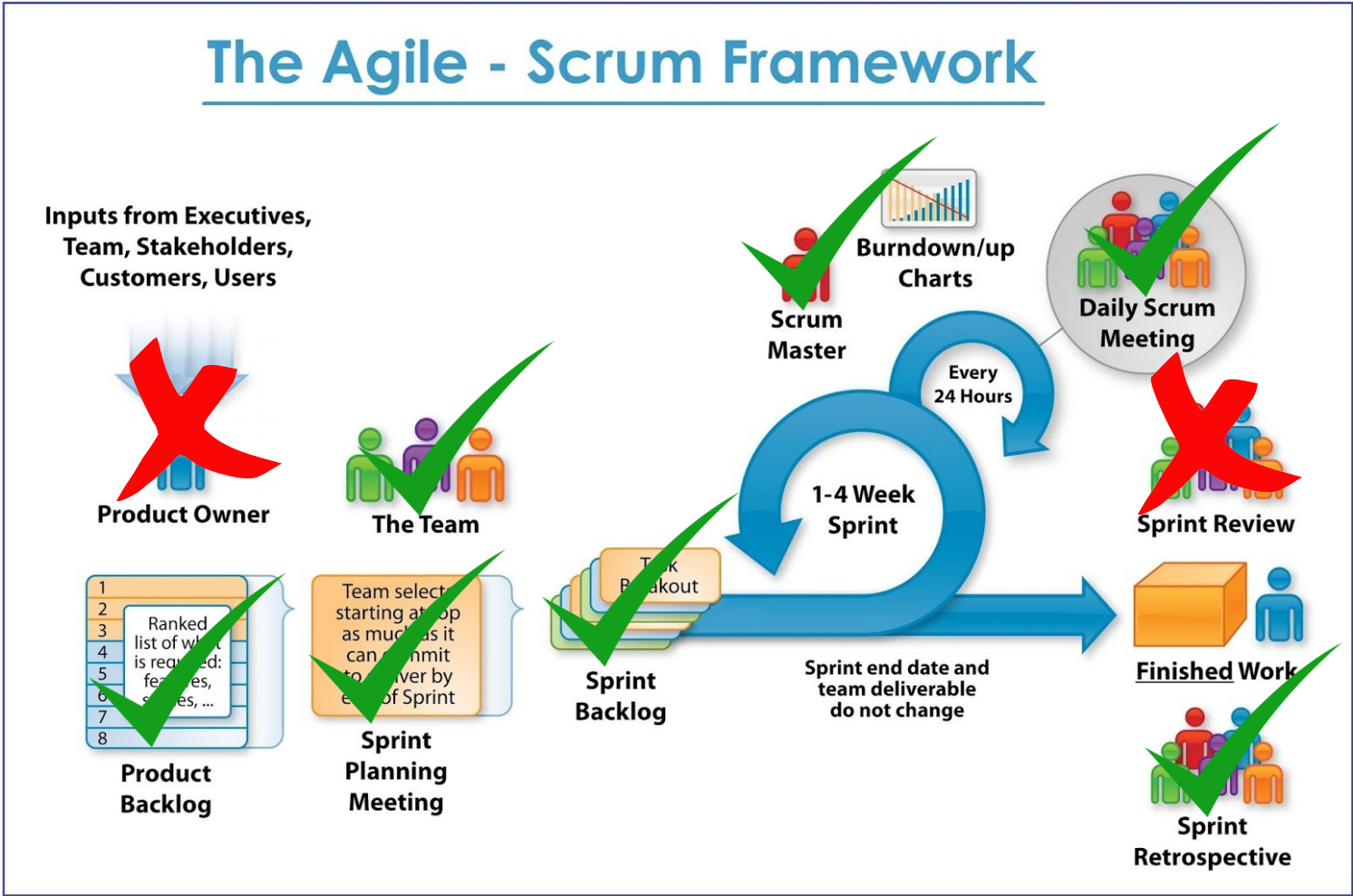
- Reasons why it could work:
 - Projects are generally complex
 - Requirements constantly change (both user and system)
- So, we gave it a try ...

Agile/Scrum at ASTRON

Agile/Scrum at ASTRON



Agile/Scrum at ASTRON



First Lessons Learned

- Not having a Product Owner is *really* problematic
(Even if you have involved users)

First Lessons Learned

- Not having a Product Owner is *really* problematic
(Even if you have involved users)
- It is one of the main reasons for not having Sprint Reviews
(We sometimes give demos, but not on a regular basis)

First Lessons Learned

- Not having a Product Owner is *really* problematic
(Even if you have involved users)
- It is one of the main reasons for not having Sprint Reviews
(We sometimes give demos, but not on a regular basis)
- It results in bad User Stories
(Can be a big issue)

Agile/Scrum at ASTRON

In practice

Agile/Scrum at ASTRON

In practice

- Three-week Sprints
- Sprint Planning based on Product Backlog
- Development on branches
- Nightly builds
- Build after each commit on the trunk
 - early warning for errors
- Code review before merge to the trunk
- Definition of Done

Agile/Scrum at ASTRON

So, do we do Scrum?

Basically: no

Agile/Scrum at ASTRON

So, do we do Scrum?

Basically: no

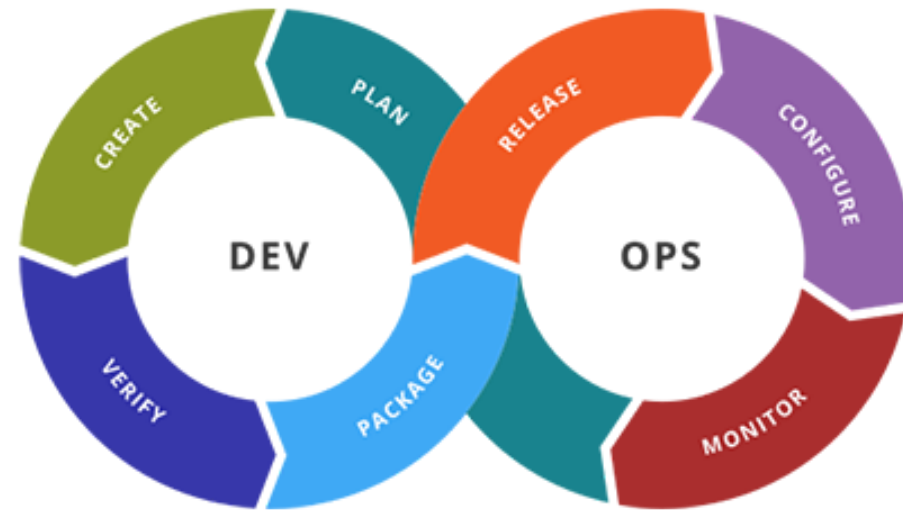
Are we Agile?

I think we are.

What about DevOps?

What about DevOps?

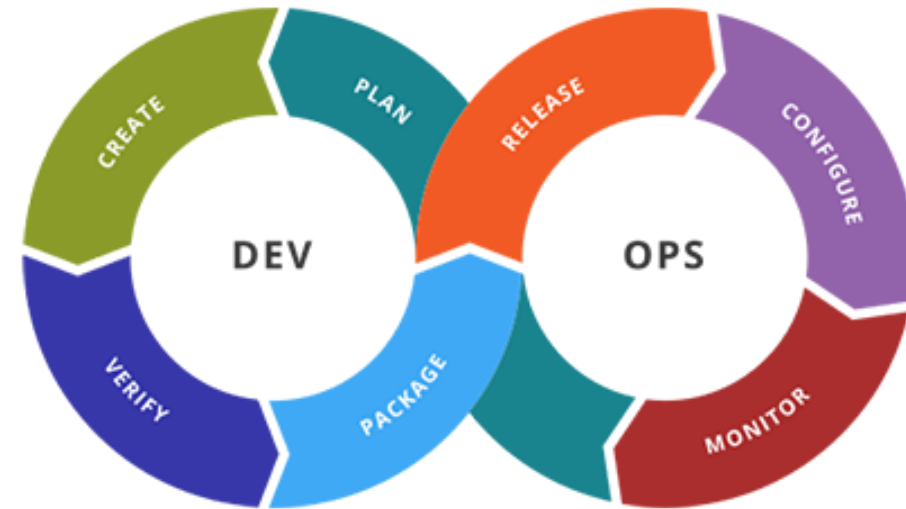
- DevOps is the combination of development (Dev) and operations (Ops)
- Goal is to shorten the development life cycle



What about DevOps?

- DevOps is the combination of development (Dev) and operations (Ops)
- Goal is to shorten the development life cycle

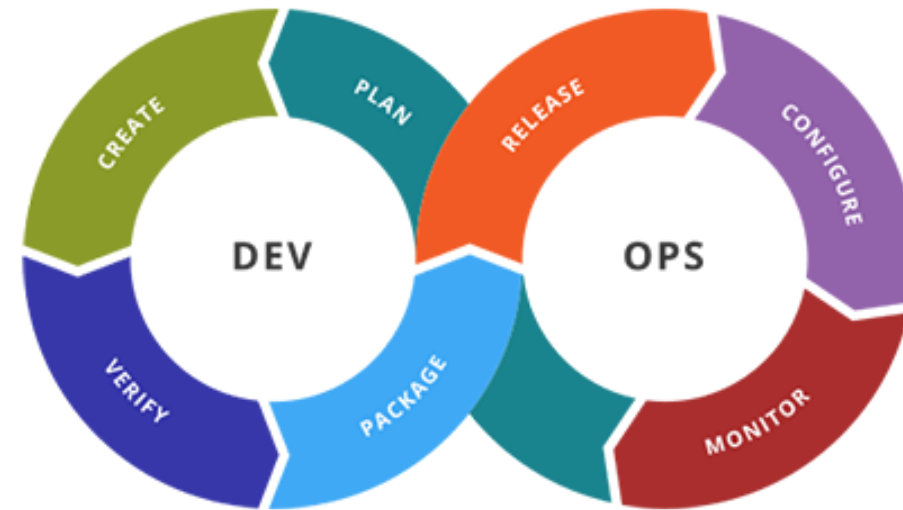
- Is DevOps Agile?
No.



What about DevOps?

- DevOps is the combination of development (Dev) and operations (Ops)
- Goal is to shorten the development life cycle

- Is DevOps Agile?
No.



- But, Agile is an *essential* part of *successful* DevOps.

DevOps Tools used at ASTRON

DevOps Tools used at ASTRON



Jenkins



ANSIBLE



JFrog Artifactory



docker



logstash



HashiCorp

Vagrant



ASTRON

DevOps at ASTRON

- Daily builds, and commit-triggered builds
- Frequent trunk releases
- Automatic deployment
- Continuous system monitoring
- Collecting log-files for system debugging

Agile/Scrum: Lessons Learned

Agile/Scrum: Lessons Learned

What worked for us

- Better planning
 - More grip on progress
 - Accurate planning for the next milestone
 - Good ball-park estimates for future milestones
- Improved software quality
 - Stable trunk, thanks to the use of feature branches
 - More focus of the team, thanks to short cycles
- More involvement of users and commissioners

Agile/Scrum: Lessons Learned

What did *not* work for us

- Really work as a Scrum *team*
 - Too much specialism makes it hard to take over someone else's work
- Sometimes too many unknowns and unexpected setbacks

Agile/Scrum: Lessons Learned

What did *not* work for us

- Really work as a Scrum *team*
 - Too much specialism makes it hard to take over someone else's work
- Sometimes too many unknowns and unexpected setbacks

What we found hard

- Plan for the unknown
- How to handle software architecture?

Agile/Scrum: Lessons Learned

Improved understanding means improved planning

Agile/Scrum: Lessons Learned

Improved understanding means improved planning

- Do *not* start to work on stories that are unclear
- Break-down a story into smaller tasks
- If stories are too big, chop them up
- Involve *all* stakeholders
 - Operators and Science Support are often forgotten

Agile/Scrum: Lessons Learned

Improved understanding means improved planning

- Do *not* start to work on stories that are unclear
- Break-down a story into smaller tasks
- If stories are too big, chop them up
- Involve *all* stakeholders
 - Operators and Science Support are often forgotten
- ... And make sure you have a Product Owner

Conclusion

Agile/Scrum works!

But it requires:

- *organizational* change
- *social* change, and
- a team that is willing to *continuously improve* itself.

This is *not* a technical challenge, but a *social* challenge!

Questions?

