

Agile in a nutshell



Jonathan Rasmusson

<http://agilewarrior.wordpress.com/>



ADAPTIVE PLANS

ITERATIVE DEVELOPMENT

CHANGING REQUIREMENTS



THE WATERFALL HORROR FEATURE SHOW



Beware
Agile Software Delivery

- Coming soon to a project near you -



What we're going to cover

- You already know agile planning
- What to expect
- De-bunk some myths
- Overview some agile methods
- 3 steps towards agility today



Agile planning in a nutshell



Too much to do, not enough time



Credit: <http://www.flickr.com/photos/geneoh/>



You make a list



Alright, what do I need to do, to get ready for this date?



Doing this alone makes you feel good

You size things up



Looks like it shouldn't take more than a couple of hours!



You set some priorities

Most important



Dang, mom says she's going be in in 11 hrs.



Least important

Out of scope



Start executing



Roses are red, violets are blue,
I hope you like this apple juice.
Cheese and crackers too.



That in a nutshell is what most people do

A little secret

Pssst

We do the same thing in agile...



Only instead of



ToDo list

Tasks

Guesses

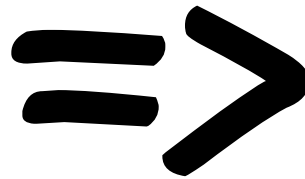
We use fancy names



Master story list

User stories

Estimates

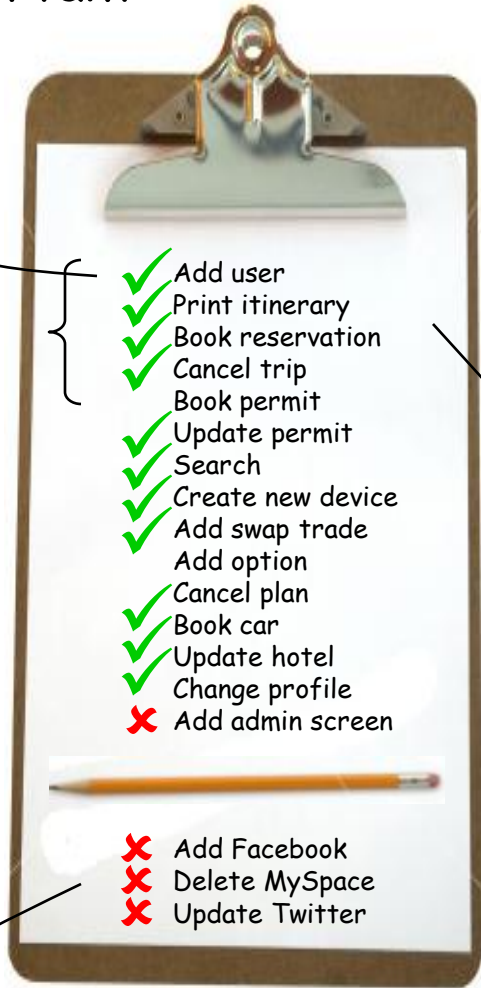


How does this make a plan?

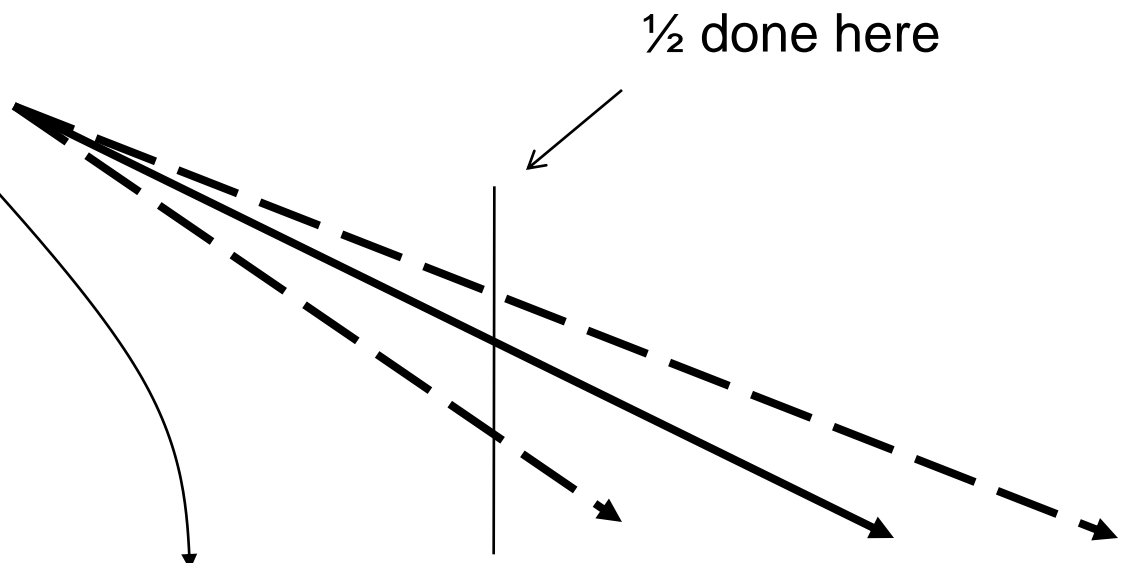
a date
a budget
a plan

Most important

1wk



Least important



✓✓✓✓
1wk

✓✓✓✓
1wk

✓✓✓✗
1wk

Out of money
Out of time
Nothing left



What we're going to cover

- You already know agile planning
- What to expect
- De-bunk some myths
- Overview some agile methods
- 3 steps towards agility today



We flex on scope



Time



Budget



Quality



Scope



Flex here

We have the same definition of done



Are we done raking
the leaves when:

A plan?

A design?

A report?

Not at my house!

Working software is the primary measure of success

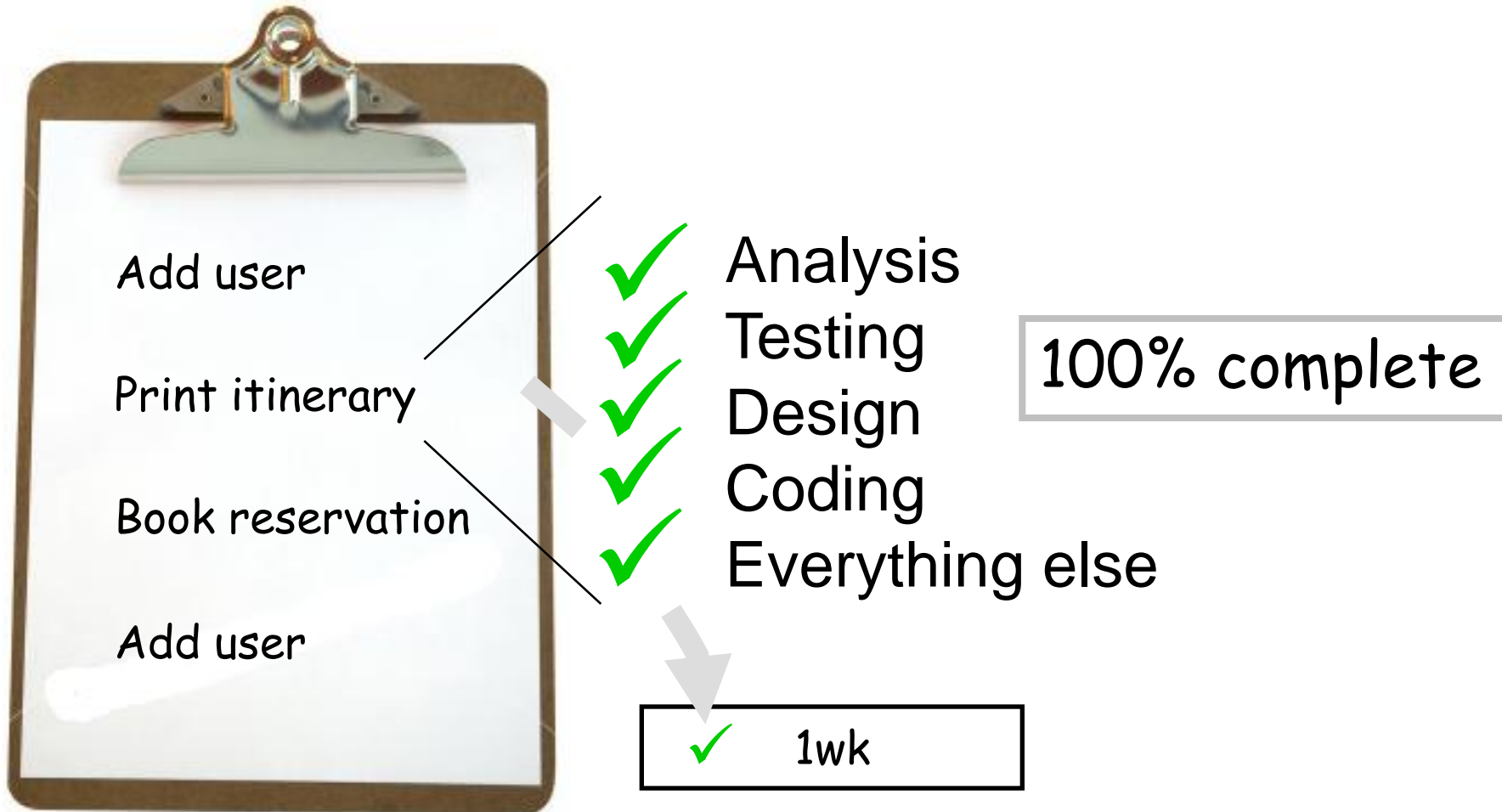


Are all fine
and dandy ...

Project plans
Test plans
Requirements docs
Architectural diagrams
Analysis models
Security reports
Deployment plans

... but they are of no
value to the customer.

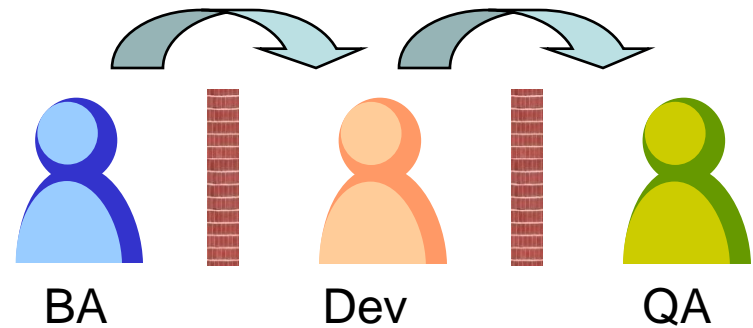
That means ...



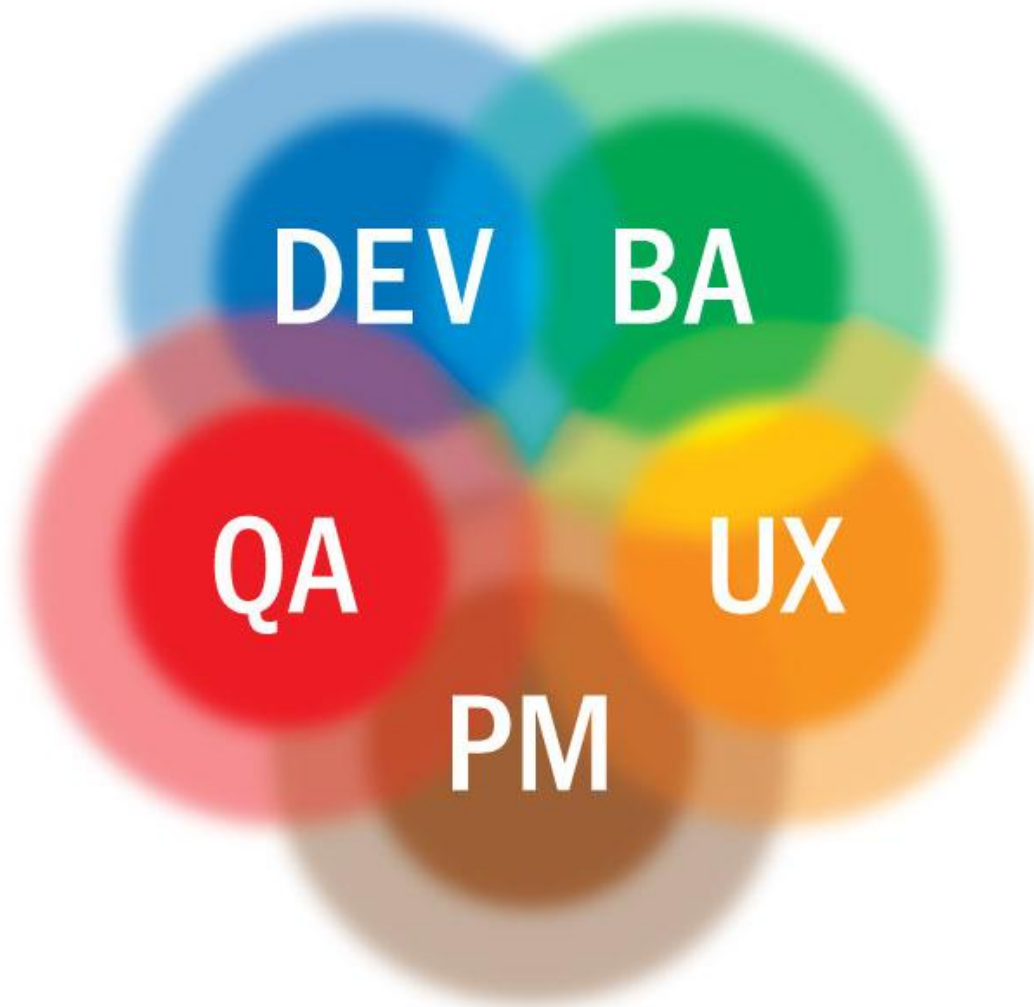
And we have to work as one team



VS



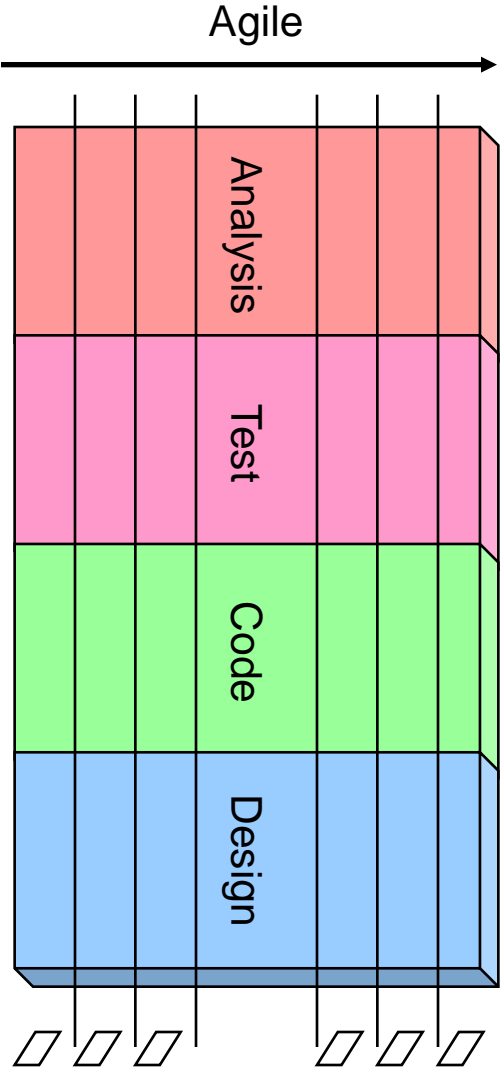
More overlap between roles



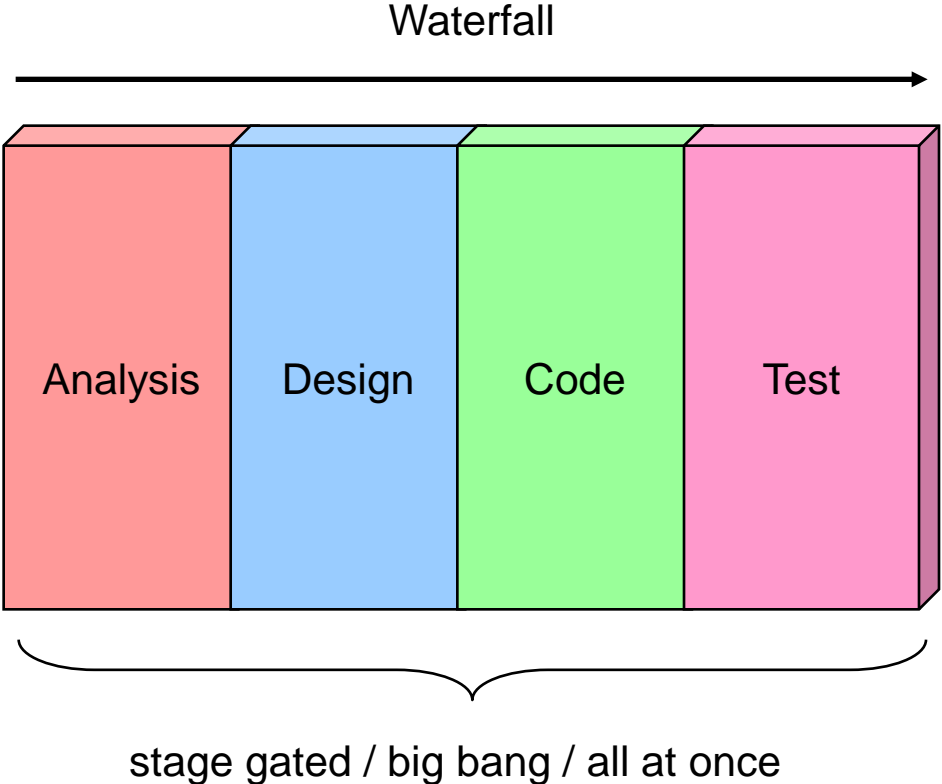
The team is accountable



Analysis, design, testing, and coding are continuous activities



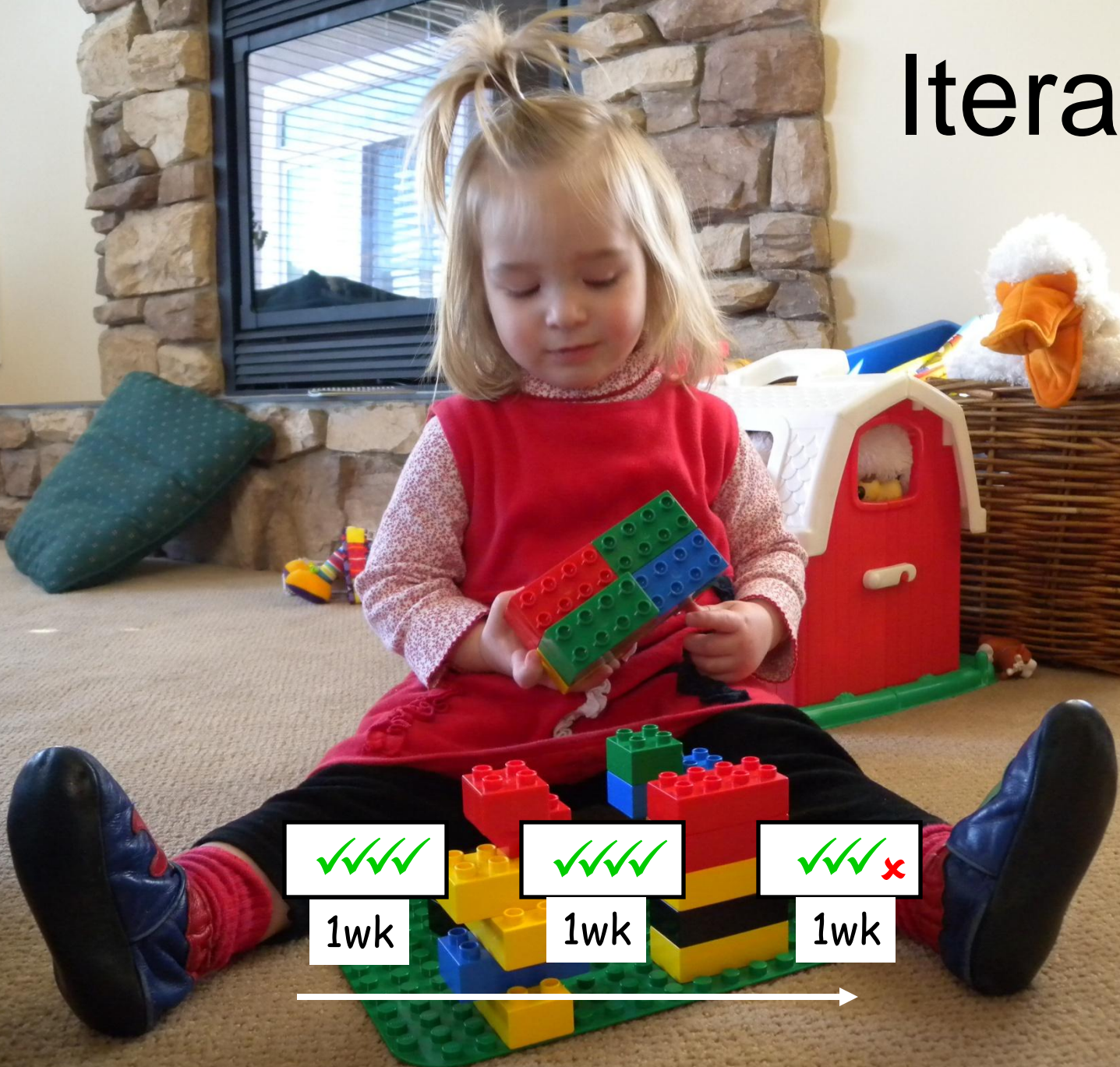
VS



Some terms you'll hear



Iterative



✓✓✓✓

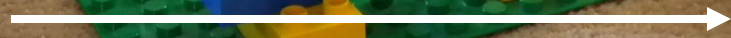
1wk

✓✓✓✓

1wk

✓✓✓✓x

1wk



Iterative



Time-boxed delivery

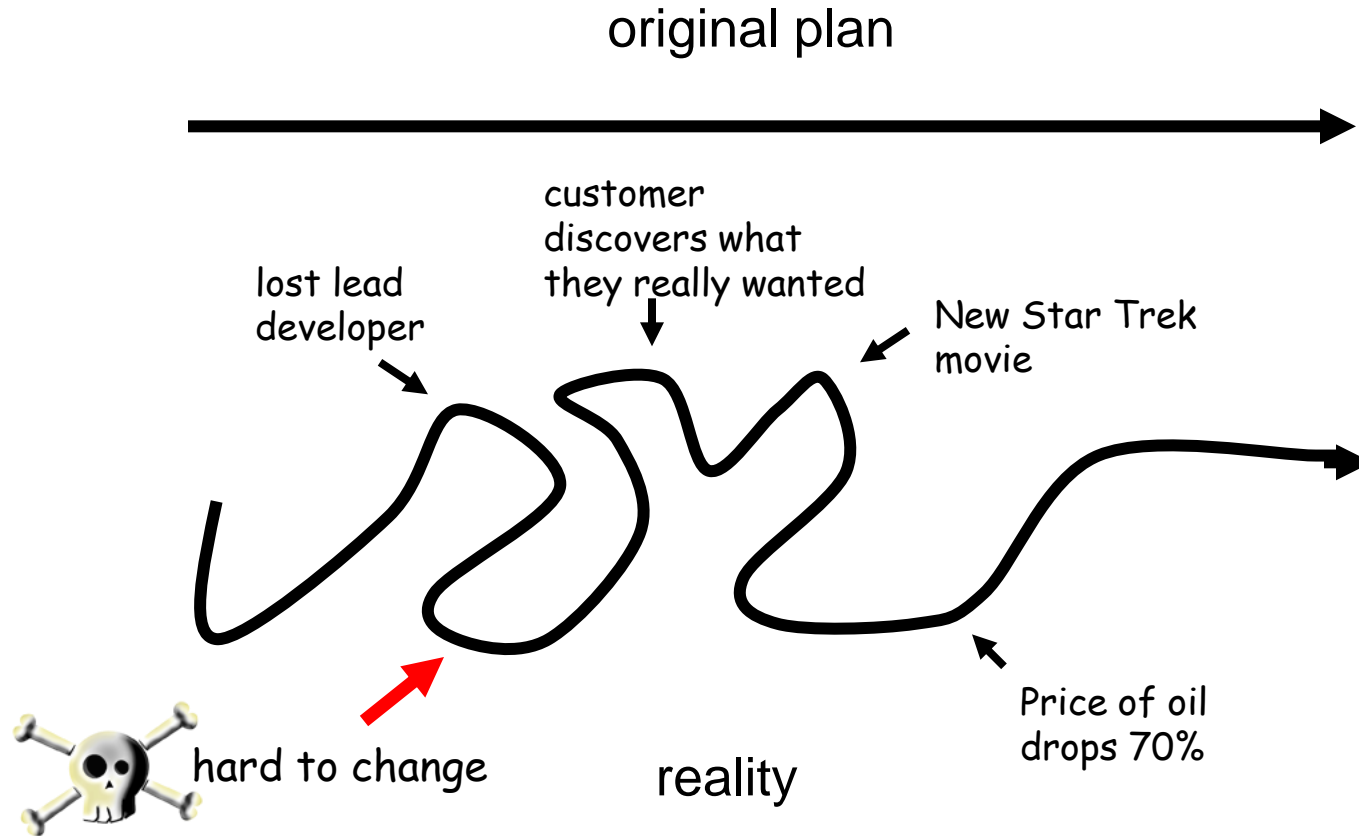


Credit Yogi: <http://www.flickr.com/photos/yogi/1147960/>

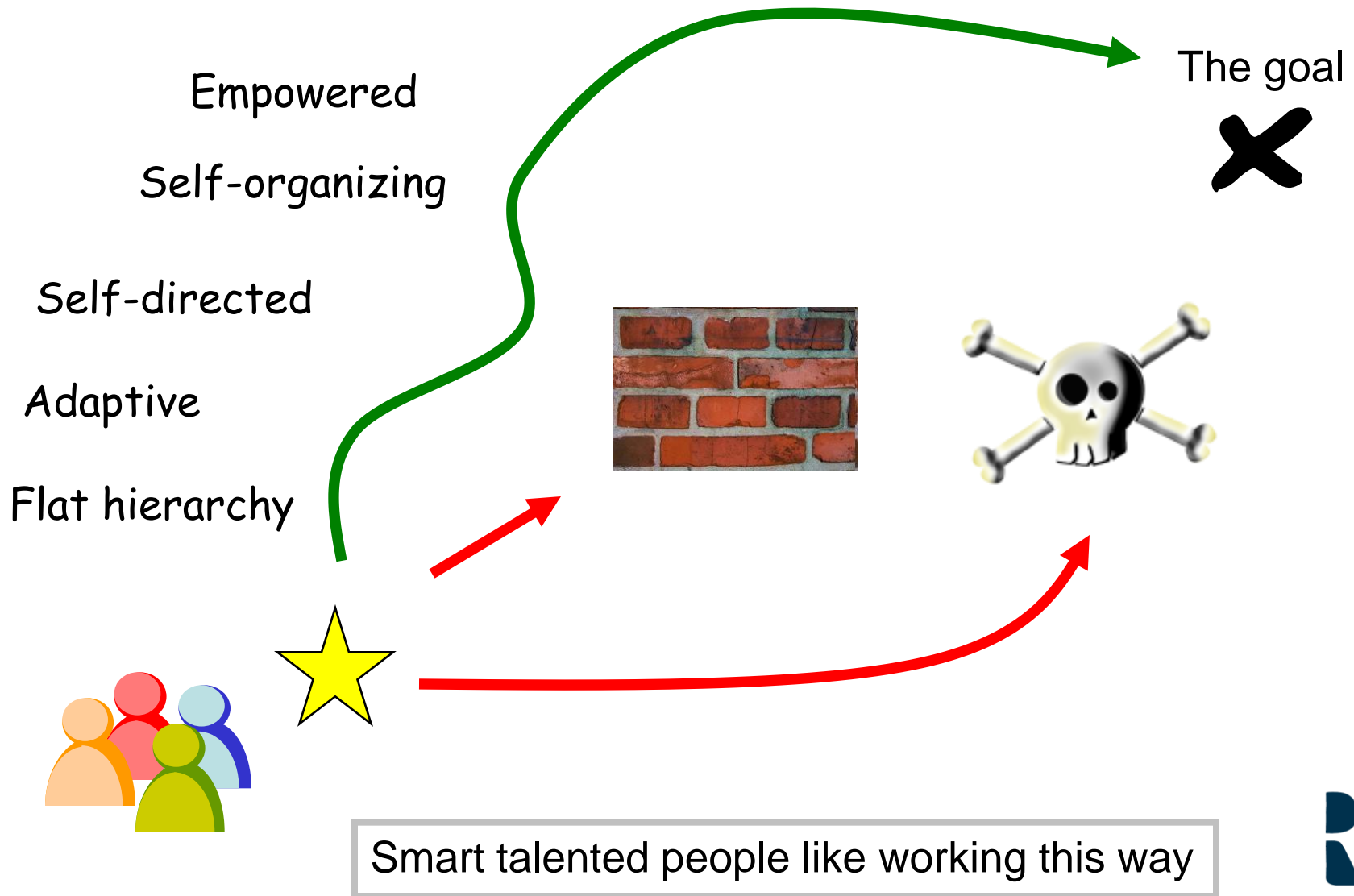
© Copyright 2009, Rasmusson Software Consulting



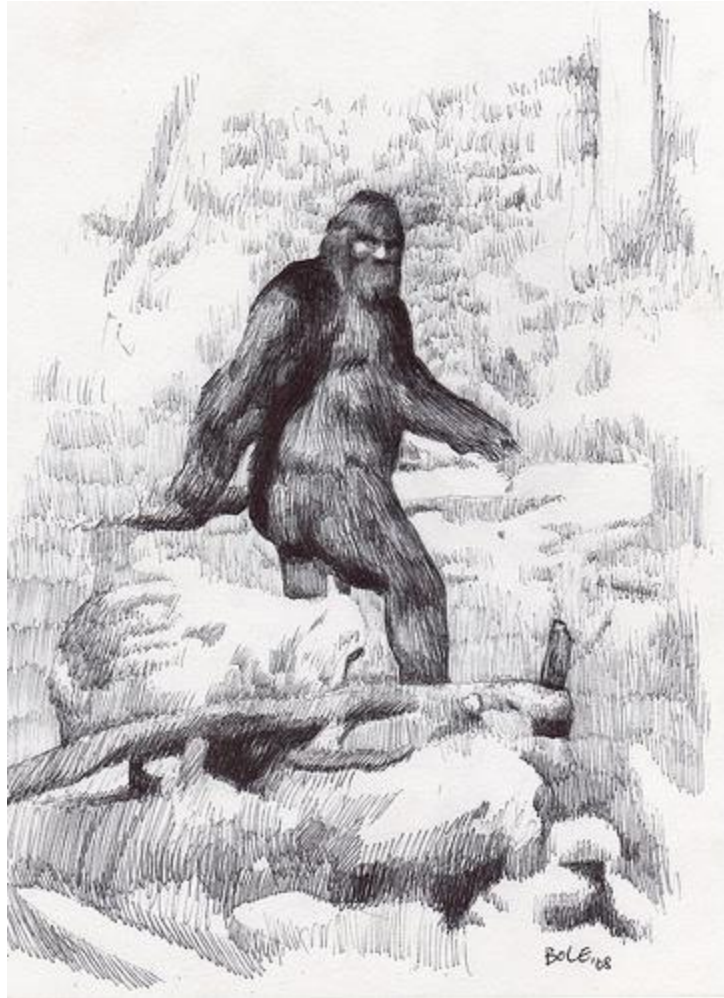
Adaptive planning



Characteristics of agile teams



Agile Myths



What agile is not



Agile is a silver bullet

- You can fail just as dramatically
- All your problems will still be there
 - Agile will just highlight them sooner



Agile teams don't write documentation

- More accurate to say agile teams don't write unnecessary documentation
- Agile teams prefer a face-to-face over documentation
- Documentation gets treated like anything else
 - Prioritized, estimated, delivered



Agile is anti-planning

- Agile teams plan extensively
 - Every quarter (release)
 - Every couple weeks (iterations)
 - Every day (daily stand-ups)
- Uses different tools
 - Project burn downs vs Gantt and PERT
- Planning is very visible
 - Stakeholders know early if there is a problem



Agile is anti-architecture



Agile is anti-over-architecture

Don't spend time designing this ...



... if this is all that is needed.

Agile development doesn't scale

- Agile scales just like any other method
 - Not that great
- Instead of looking at how you can scale up
 - See if there are ways you can scale down



Agile teams don't model

- Agile teams model
 - They just don't believe them
- The sooner they can turn a model into working software the better



Agile is undisciplined

- Agile has been bruised by people taking the easy stuff
 - and leaving out the hard
- Truth is agile is very disciplined
 - You have to write tests
 - You have to design
 - You must regularly integrate your code
 - You have to regularly release working software
 - You must deliver something of value each iteration
- The more agile a team – the more disciplined



Agile development is unpredictable

- Agile acknowledges the inherent complexity and uncertainty of software
- Agile quickly incorporates actual data into it's forecasts
- It doesn't dwell on reconciling actual and original plans
- Accepts that things are going to change, and incrementally re-plans when better data is available

If you crave predictability,
you picked the wrong profession

Credit: <http://www.flickr.com/photos/shadphotos/>



Agile requires talented people

Skill-set of people



Project success



Creating valuable, innovative software requires skilled people

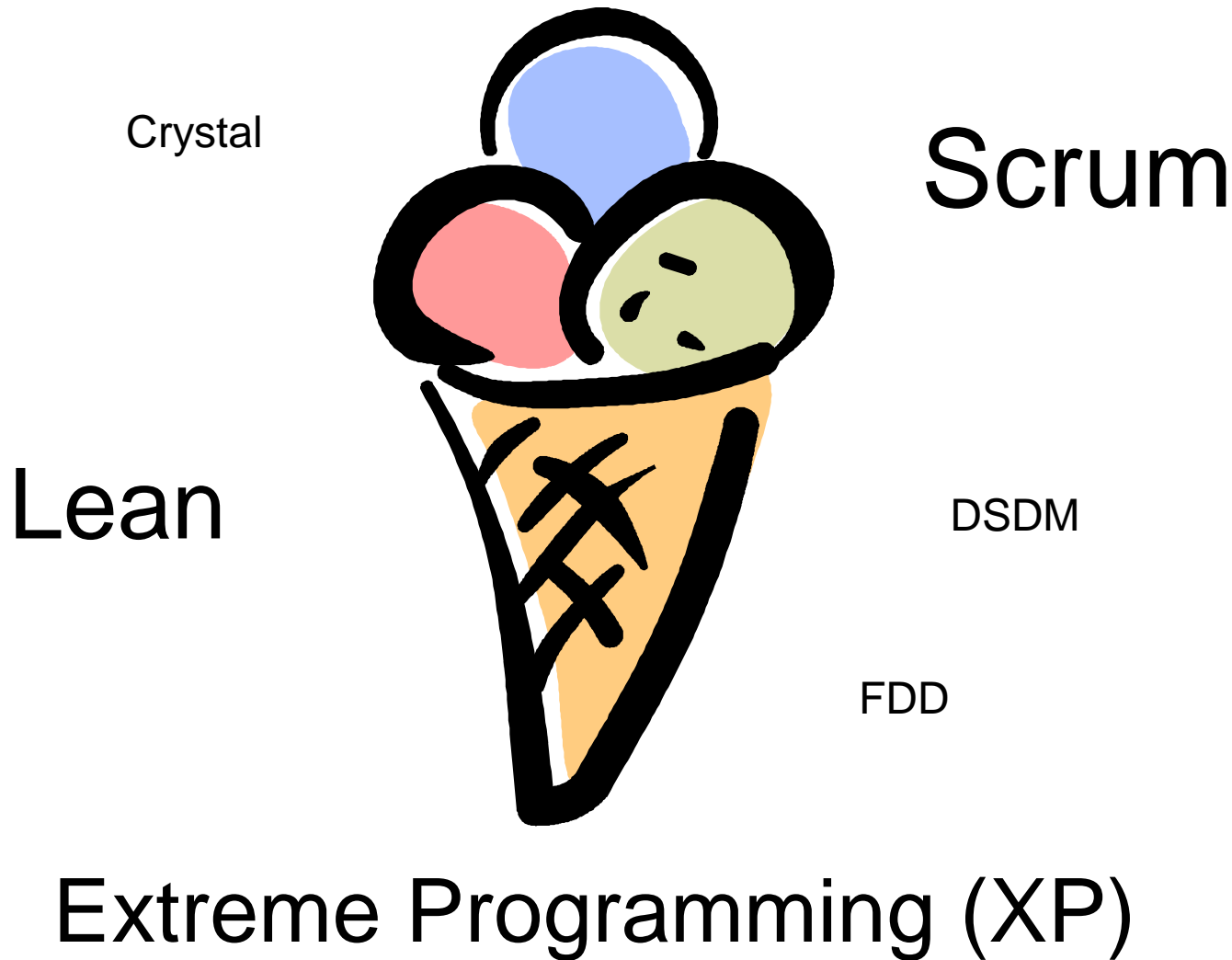


What we're going to cover

- You already know agile planning
- What to expect
- De-bunk some myths
- Overview some agile methods
- 3 steps towards agility today



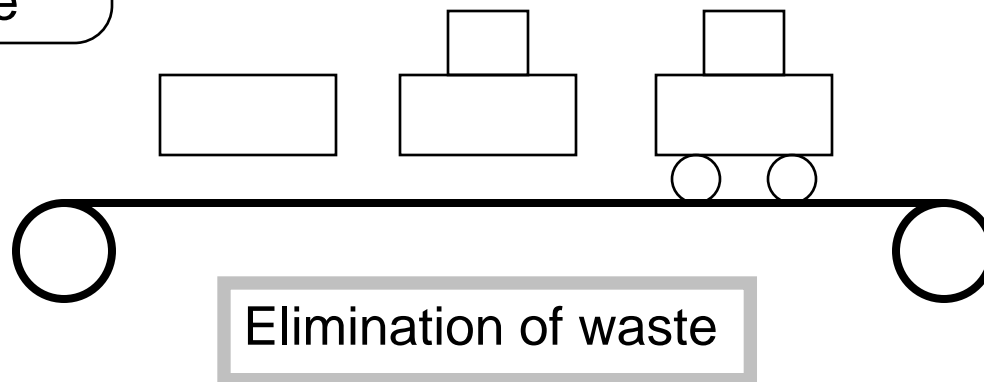
Agile comes in many flavours



Lean

I would like to buy a Toyota Prius please

Toyota's ultra-lean manufacturing process.



Lean

+ pluses

- minuses

- Very good high-level principles and practices
- Addresses systems and organizational improvements across the board
- Advice / practices are not IT specific
- Harder to implement

Lean has much to offer - very worth of study



Scrum

- A project management wrapper for incremental delivery of projects, independent of technology or business vertical.
- Can be used in non-IT projects.



Scrum

+ pluses

- Easy to understand
- Low barrier of entry
- Easy to pick up
- Speaks well to project managers
- Non-threatening
- Most popular

- minuses

- Silent on engineering
- Easy to do the easy stuff while skipping the hard stuff

Easy to adapt - most non-threatening



Extreme Programming (XP)

- Popularized software engineering practices necessary for agile development
- Emphasizes
 - upfront testing
 - automation
 - evolutionary design
 - continuous integration



Extreme Programming

+ pluses

- Sound engineering practices
- Strong in development community
- First real popular agile method

- minuses

- Very developer focused
- Hard for other disciplines relate
- Sometimes characterized by zealots

Essential engineering practices



Options for adopting

XP
full on

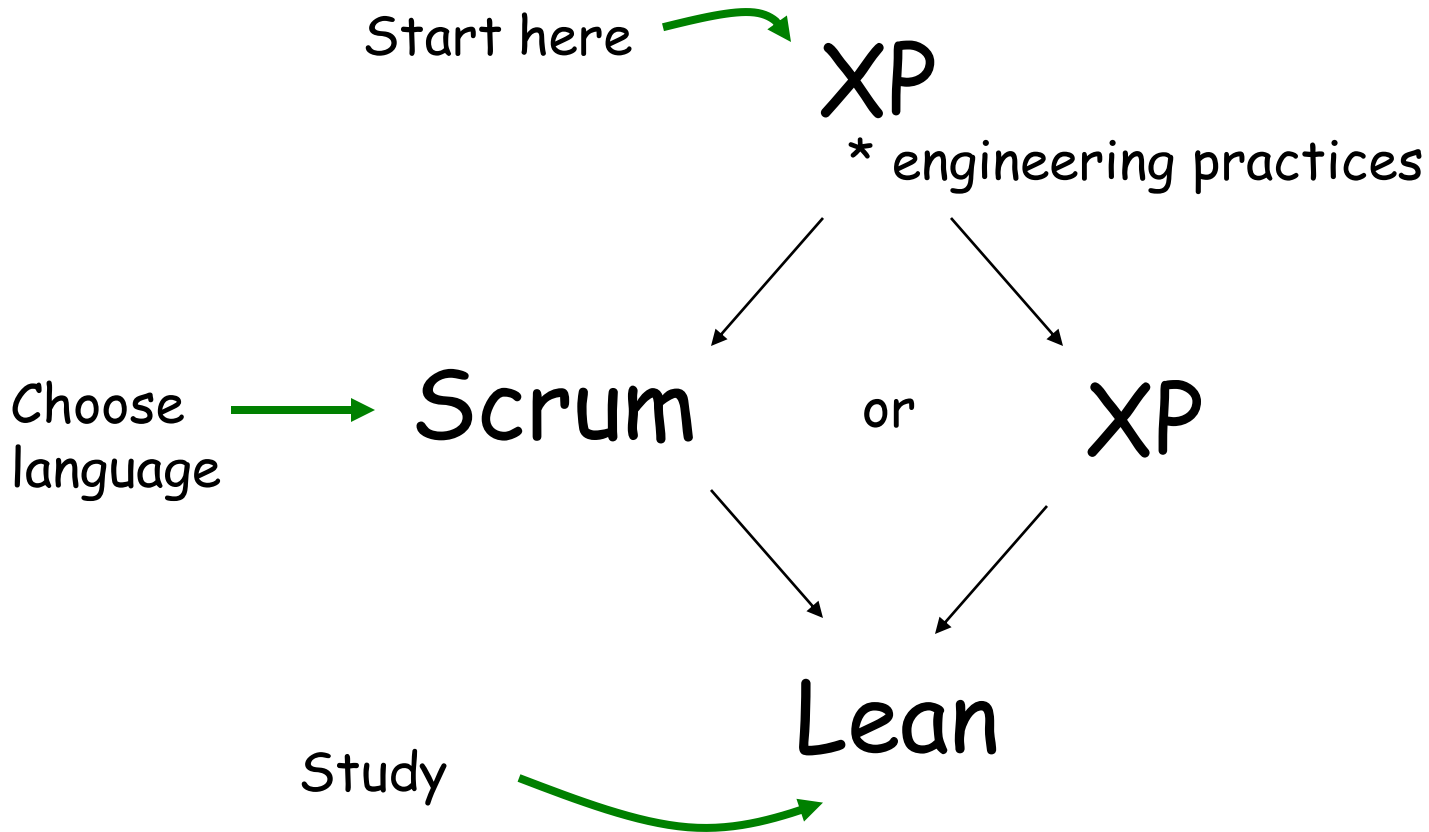
Scrum + XP
engineering practices

Scrum + XP + Lean
engineering practices spirit

- * engineering practices
- unit testing, refactoring
 - continuous integration, Test-Driven Design (TDD)



What I recommend



There is no one way

Extreme Programming (XP)

Crystal

Scrum

Lean



DSDM

FDD

Do what works for you

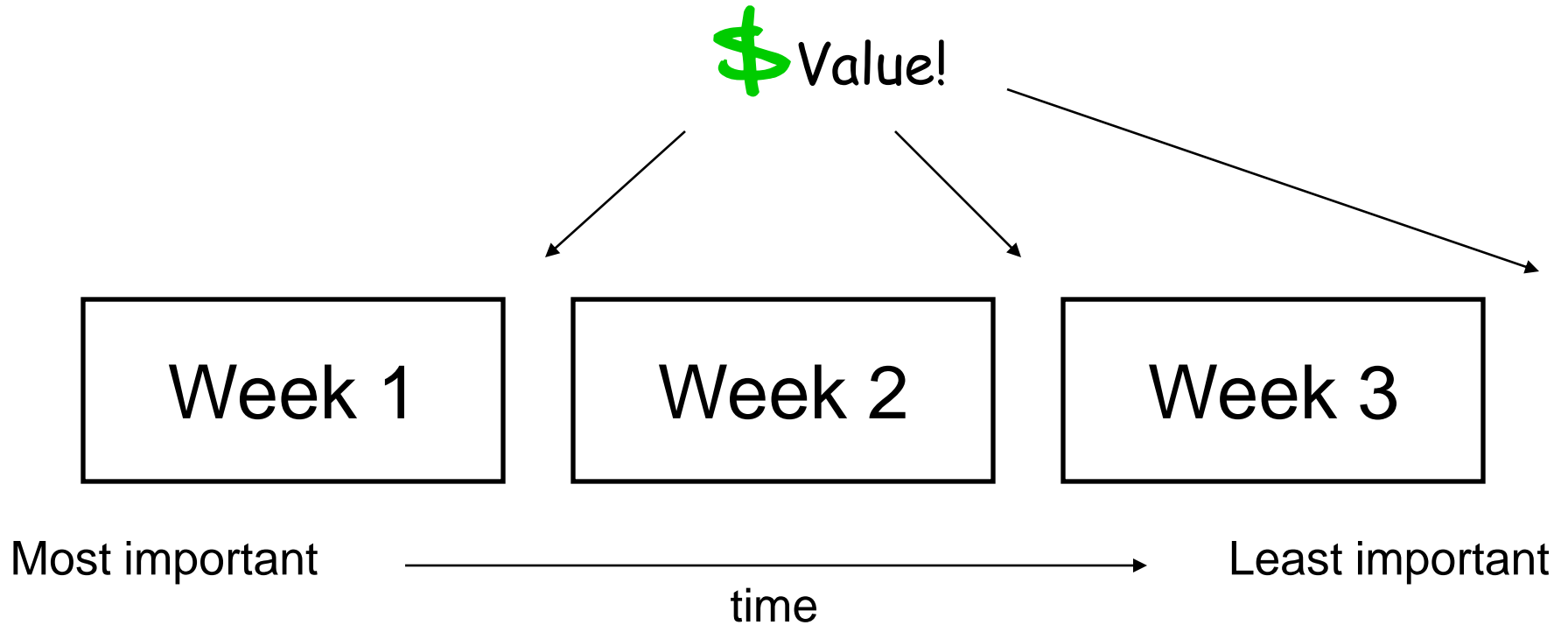
Make it your own



3 steps you can take towards agility today



Deliver something of value every week



Start doing these 4 practices

today!

- Agile is underpinned by technical excellence
- Practices like these are essential:

Unit testing

Test-Driven Design

Refactoring

Continuous integration

- If you get these right
 - everything else becomes a lot easier



Accept 3 simple truths

1. It is impossible to gather all the requirements at the beginning of a project.
2. The requirements are guaranteed to change.
3. There will always be more requirements, than time and money allow.



Final words



Final words

- No magic – you already think agile
- Expect change
- There is no one way



Do I think everyone will one day be doing agile?



No

For the same reason most people
still don't eat right or exercise.

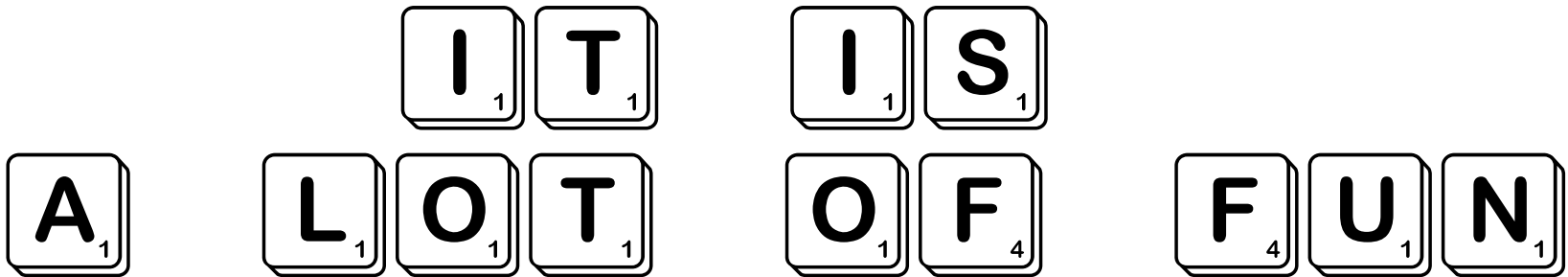


Agile is tough
Agile is hard work
Agile requires great discipline

Not everyone is into this kind of stuff!



But it is a very natural way to work



And it works – very well



For more information

<http://agilewarrior.wordpress.com/>



jonathan@agilewarrior.com

