

Oredev

Malmo - November 2016

# Xanpan

*Pronounced “Zanpan”*

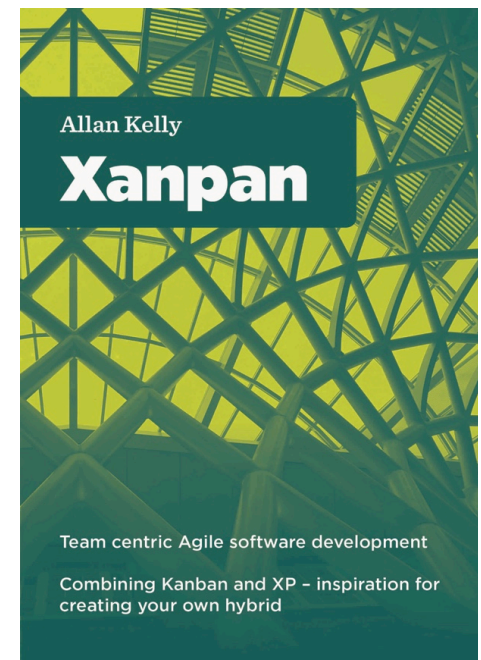
*What do you get if  
you cross Kanban  
with Extreme  
Programming?*

*Team Centric Agile*

allan kelly

[allan@allankelly.net](mailto:allan@allankelly.net)

@allankellynet - #Xanpan



Who is doing...

Scrum?

Kanban?

Extreme Programming?

Waterfall?

We don't need another methodology





Ken & Jeff's  
Scrum-Cola



The Real  
Thing

# Choose your Cola

8 out of 10  
teams prefer.

Will you take  
the Kanban  
challenge?



David Anderson  
Kanban-Cola



Kent Beck  
XP-Cola



High in Caffeine  
For Real Programmers



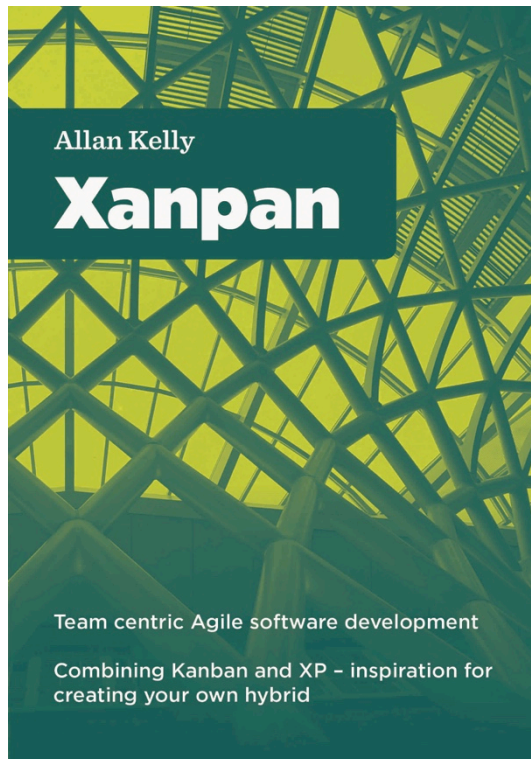
Allan Kelly  
Xanpan-Cola





# Xanpan

Take it, use it  
Inspiration  
Roll your own



<http://leanpub.com/xanpan>

Discount code: **Malmo2016**

Half price for 24 hours

Printed: Lulu.com [tinyurl.com/zf5hke4](http://tinyurl.com/zf5hke4)

Image from Ildar Sagdejev under Creative Commons license  
[http://commons.wikimedia.org/wiki/File:2009-02-15\\_Rolling\\_a\\_cigarette.jpg](http://commons.wikimedia.org/wiki/File:2009-02-15_Rolling_a_cigarette.jpg)

# Principles

## 1. Iteration routine

- Humans are good at deadlines

## 2. Invest in Quality

- “Quality is Free”

## 3. Visualize

- *See to learn*



# Principles

## 4. Dis-economies of Scale

- *Small batch size*

## 5. Emphasize Flow

- Level, Span, Constrain

## 6. Team Centric – stable

- *#NoProjects*
- Planned & Unplanned work



# Principles

7. *Constructivism learning*

8. Goodhart's Law





# Goodhart's Law

Any observed statistical regularity will tend to collapse once pressure is placed upon it for control purposes.



Professor Charles Goodhart, CBE, FBA

Some details

# Team Centric



# Stream Teams

With business: product, service, business line





# One Team

## Delivers product/service



# Teams

- Keep teams together
  - Why break up successful teams?
- Stable teams
  - Improve performance
  - Velocity & estimation can become predictable



# Sausage Machine

Requirements go in

Working Software  
Comes Out



# Teams



Flow the work to the team



# One Team

Many projects  
Many pieces  
of work

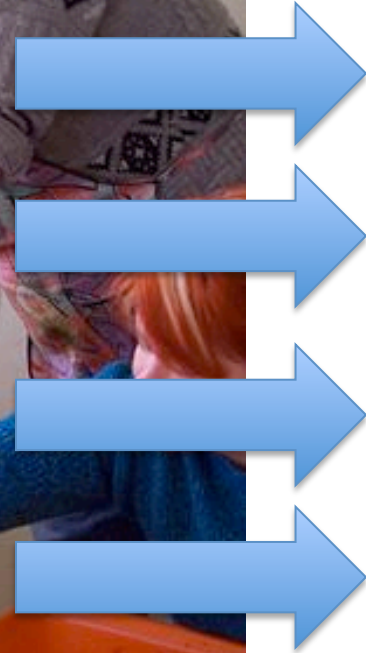
## Focus on Flow

Bring the work to the team

Bring the work to the team

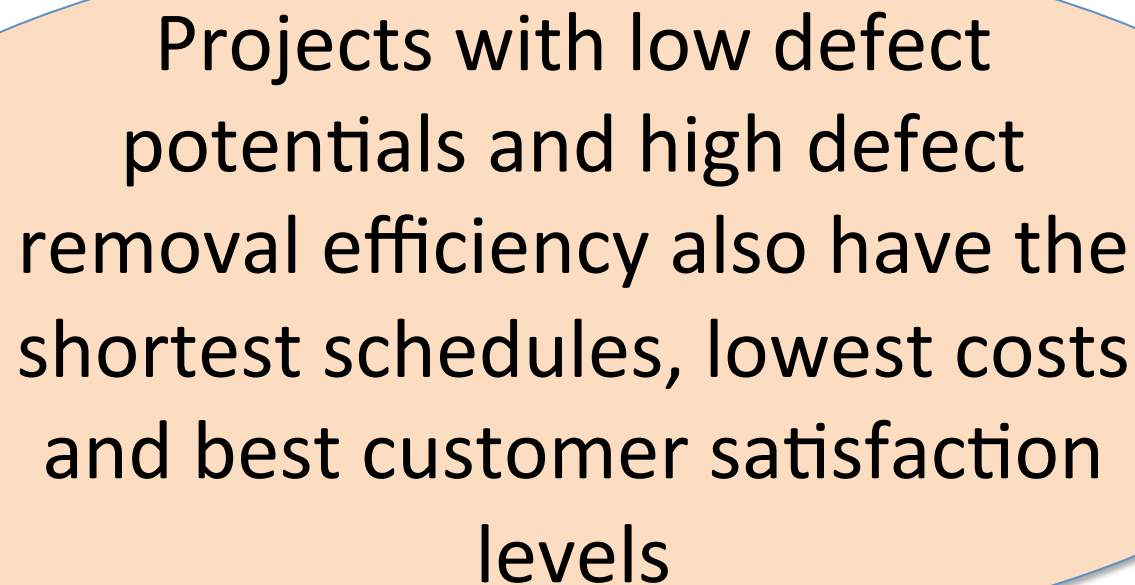
Bring the work to the team

Bring the work to the team



Quality

# Quality -> Quicker



Projects with low defect potentials and high defect removal efficiency also have the shortest schedules, lowest costs and best customer satisfaction levels

Capers Jones, 2008

*Applied Software Measurement*

Investing in quality up front

Is

Cheaper than fixing it later

**No license to gold plate**



There is no such thing  
as “Quick and Dirty”

Only

“Dirty and Slow”

**You always pay  
for dirty**



Quick &  
Dirty



Dirty &  
Slow

How do you improve quality?

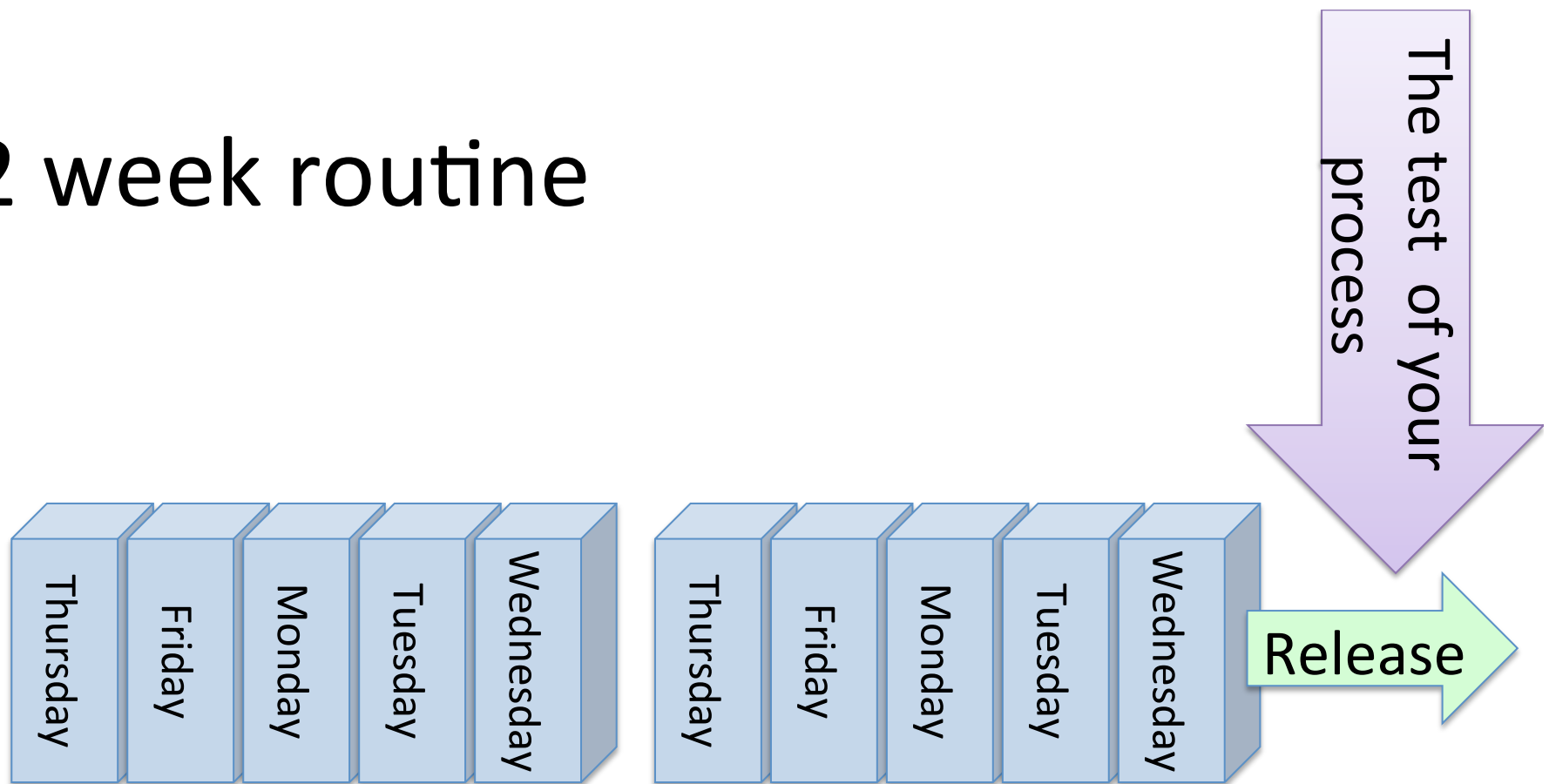
T D D

A T D D

B D D

# Iterations

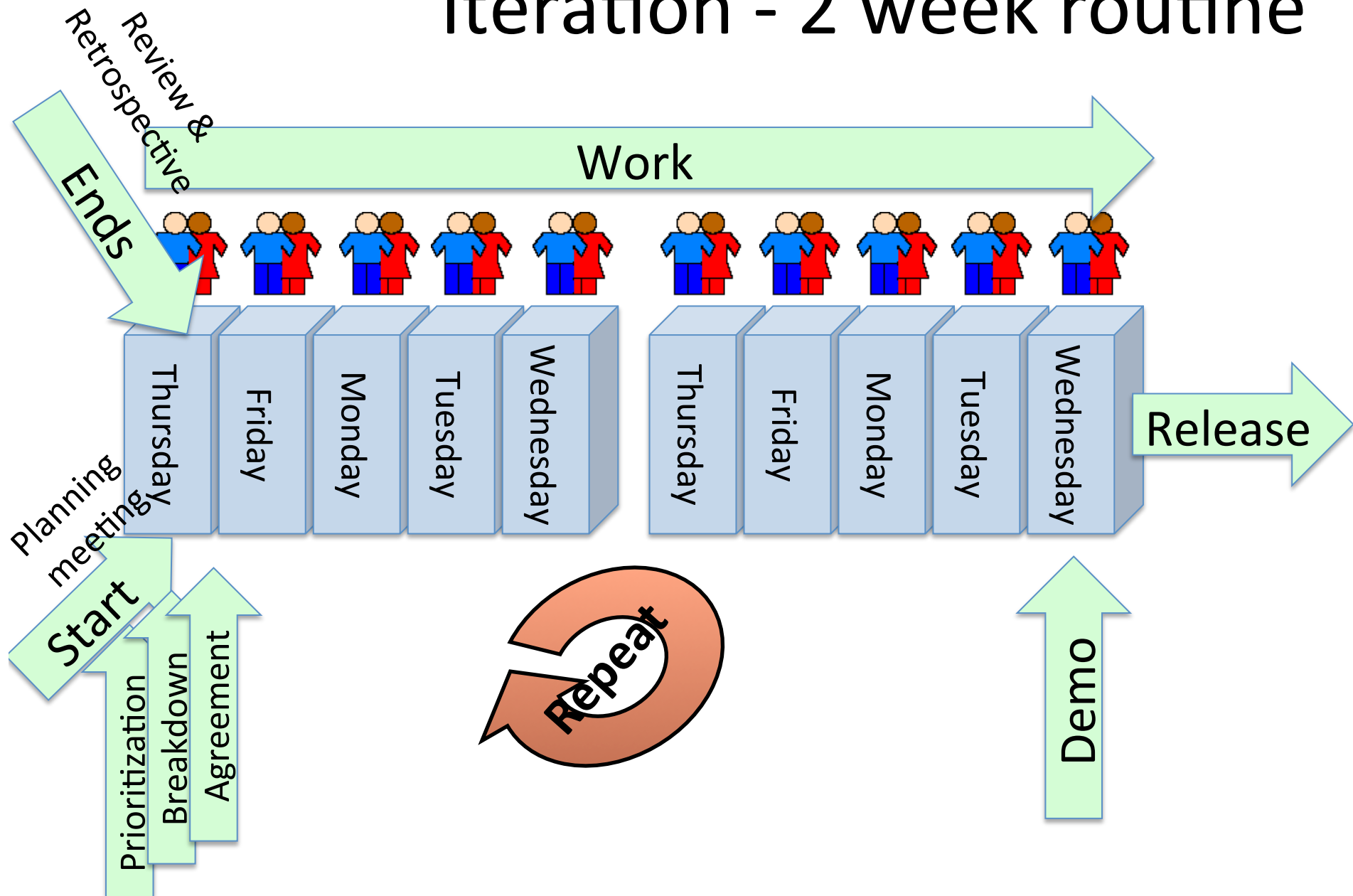
# 2 week routine



Every 2 weeks you have a shippable product

Whether you ship or not is a marketing decision

# Iteration - 2 week routine

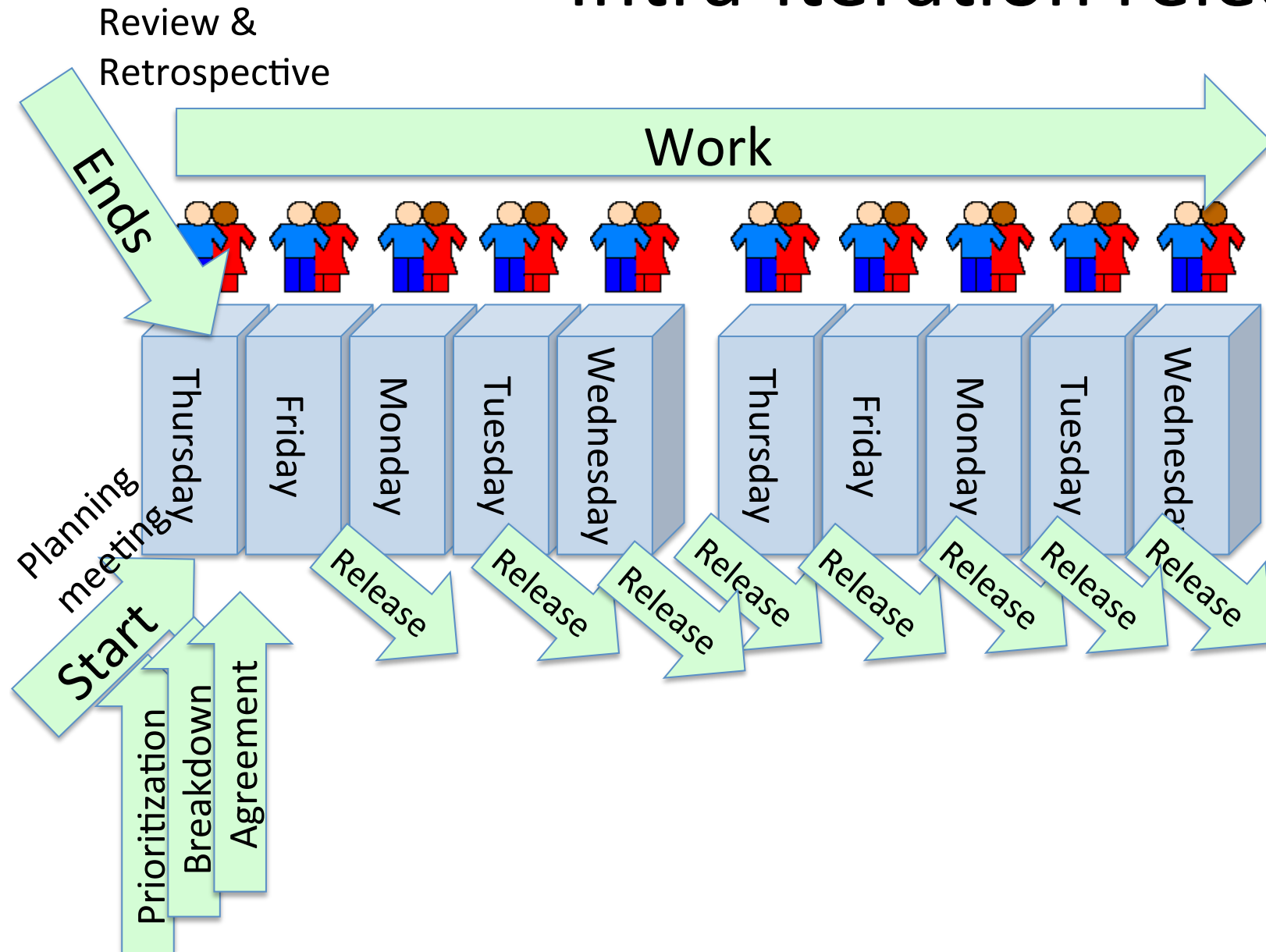




For most teams a release every 2 weeks  
seems an impossible goal

For the best teams a release every 2 weeks  
seems an impossibly long time to wait

# Intra-iteration releases



# Iterations & Flow

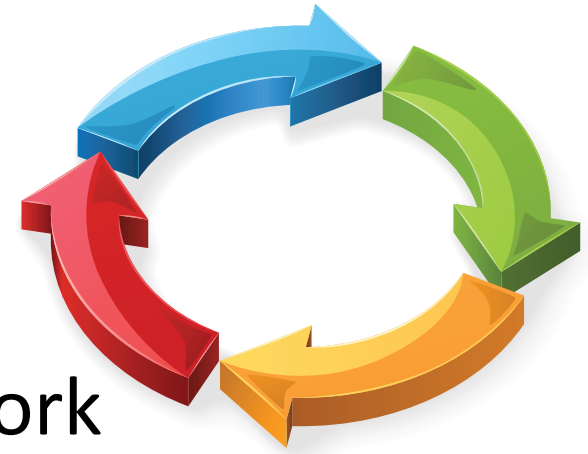
Iterations bring structure

**But**

Strict iterations break flow



# Iterations & Flow



- Stories spanning sprints levels work
  - Break down stories to tasks
  - Break down is design
  - Tasks only counted when completed
  - When all tasks done, Story done
- *3 Strikes and you are out!*

# Unplanned work



## **Unplanned but urgent and valuable**

- Seek value
- Nothing wrong with late work
  - Late does not mean it is less valuable
  - Late breaking may be more valuable



# Planned & Unplanned work

- Planned work: planned in planning meeting
- Unplanned work at any time
  - Tag it, e.g. Yellow card
- At end of the iteration analyze & understand



# Stories: 2 Golden Rules

1

**As a**  
**I want to**  
**So that**

*Role or Persona*  
*Do a Something*  
*Objective*

2

Story should benefit business  
(Story should have value \$s & €s)

*Bang!*

Story should be small –  
deliverable in days; max 2  
weeks

# Breakdown

- In planning meeting
- Part
  - Software Design
  - Requirements elicitation
  - Opportunity to reduce scope
  - Estimation exercise



Image from Paul Goyette, Creative Commons License  
[http://commons.wikimedia.org/wiki/File:Wrecking\\_ball.jpg](http://commons.wikimedia.org/wiki/File:Wrecking_ball.jpg)

Value but too  
big to deliver  
soon

# EPIC

Should be  
business

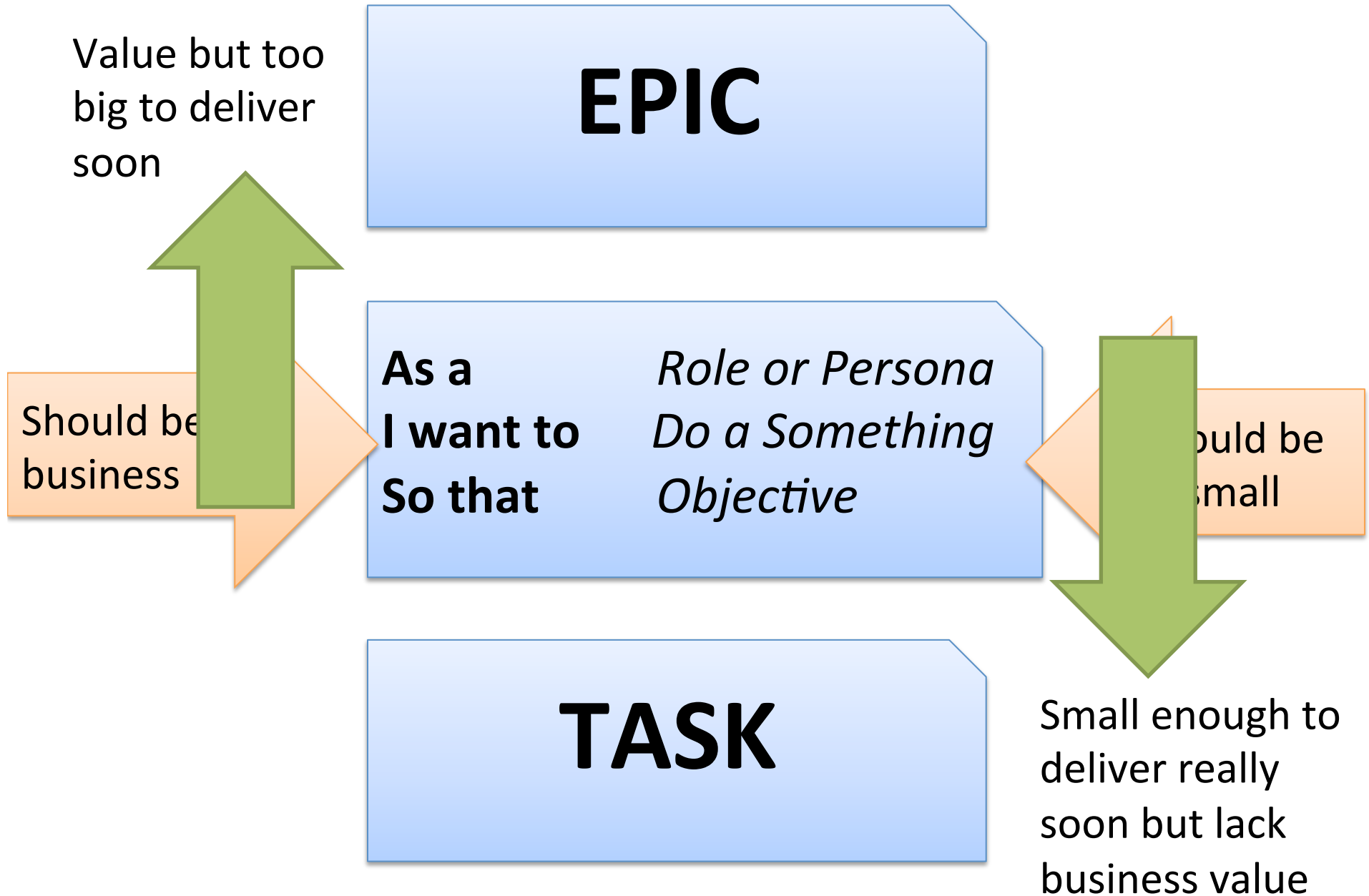
**As a**  
**I want to**  
**So that**

*Role or Persona*  
*Do a Something*  
*Objective*

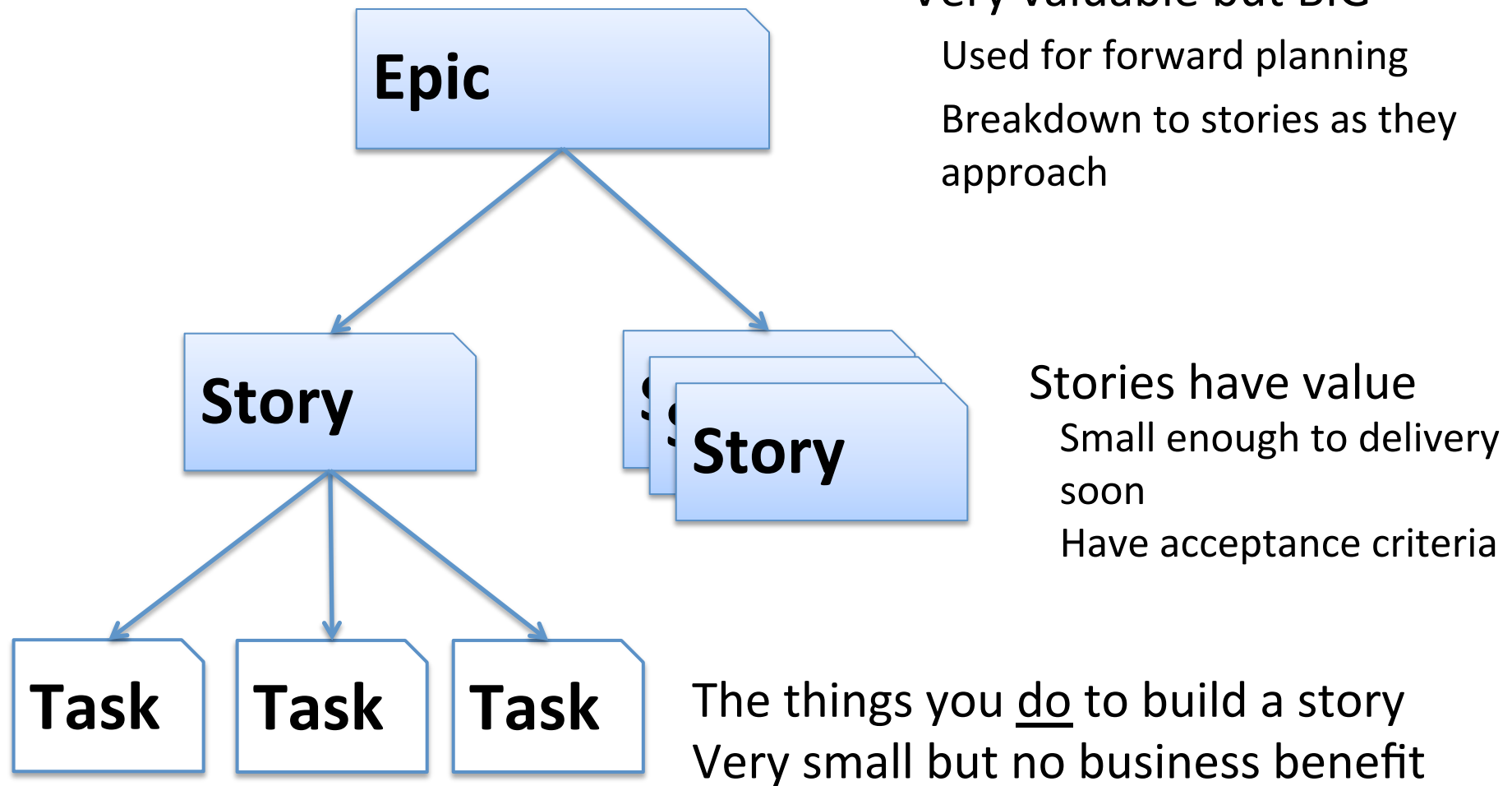
Should be  
small

# TASK

Small enough to  
deliver really  
soon but lack  
business value

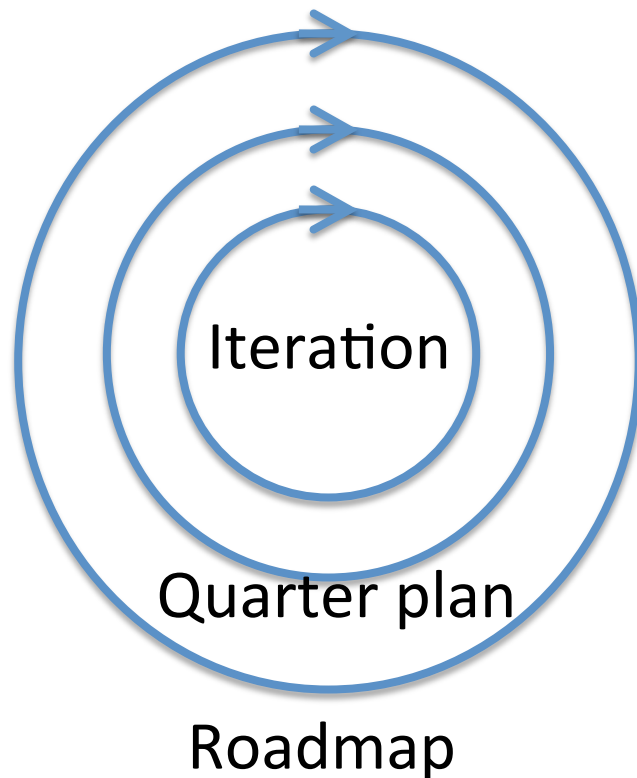


# Epic-> Story-> Task





# 3 Planning Horizons



## Iteration (Sprint)

- 2-4 weeks ahead

## Quarter plan (Release)

- Next quarter
- 2-4 releases ahead
- (2-8 Iteration)

## Roadmap

- 1-2 years by quarter
- 2-5 year ahead

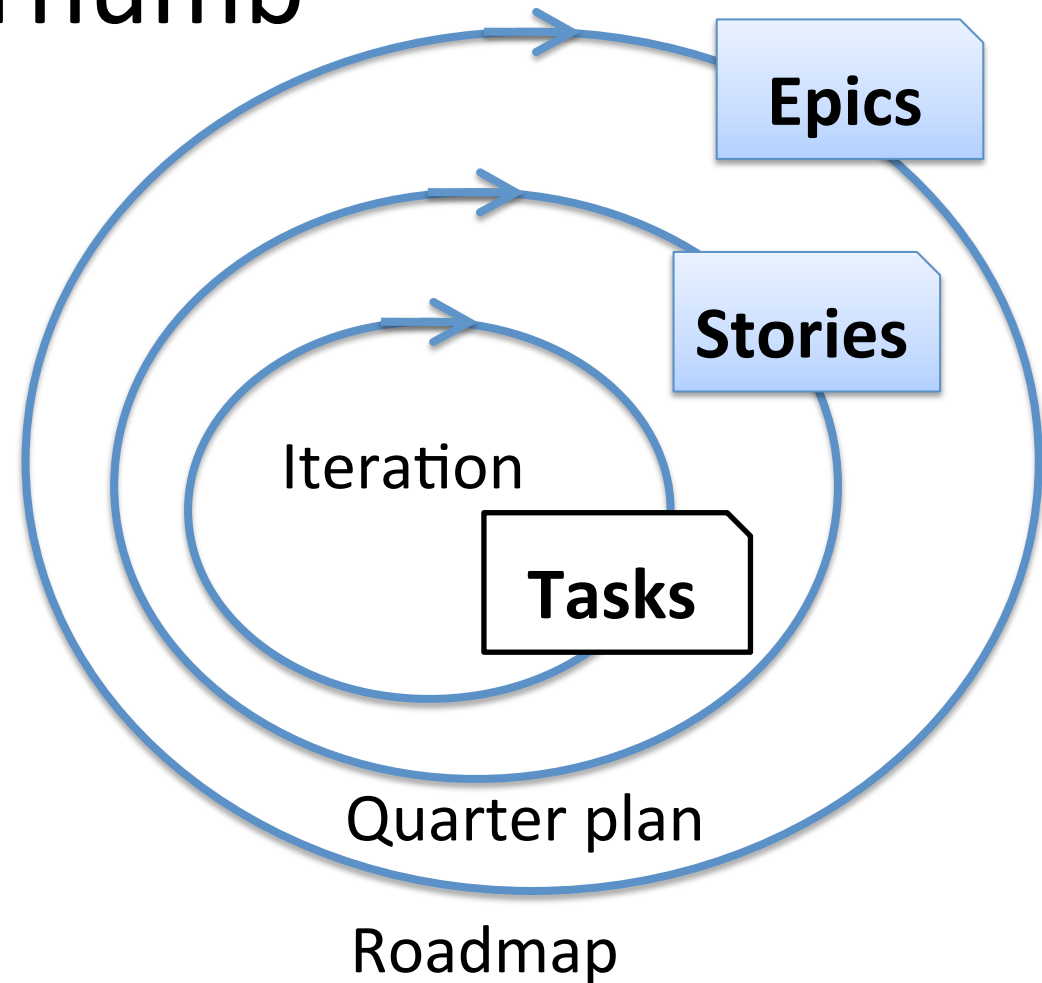


# Rule of Thumb

Iteration plan with  
Task level

Quarter / Release  
plan with Stories

Roadmap plan with  
Epics



# Focus on Value not The End

Ask not, “When will the software be done?”

But ask: “When will the software deliver value next?”



Think: Stream of Value  
(which might stop one day)  
Not: An end date



*Which brand of Cola  
are you drinking?*



allan kelly

[www.allankelly.net](http://www.allankelly.net)

[allan@allankelly.net](mailto:allan@allankelly.net)

Twitter: [@allankellynet](https://twitter.com/allankellynet)

48hr ½price code:  
**Malmo2016**

<http://leanpub.com/xanpan>

Printed: Lulu.com [tinyurl.com/zf5hke4](http://tinyurl.com/zf5hke4)

