Sample of an Emergent Practice

An Agile way to estimate a project in 3 steps ☺

- Story brainstorming
- Affinity estimation
- Velocity prediction

An Agile way to estimate a project in 3 steps ☺

- Story brainstorming
- Affinity estimation
- Velocity prediction

Story brainstorming

As a <*role*>,
I want <*goal/desire*>
so that <*benefit*>

The format is not the point

Story brainstorming

I *Independent*

S *Scalable (small sized)*

T *Testable*

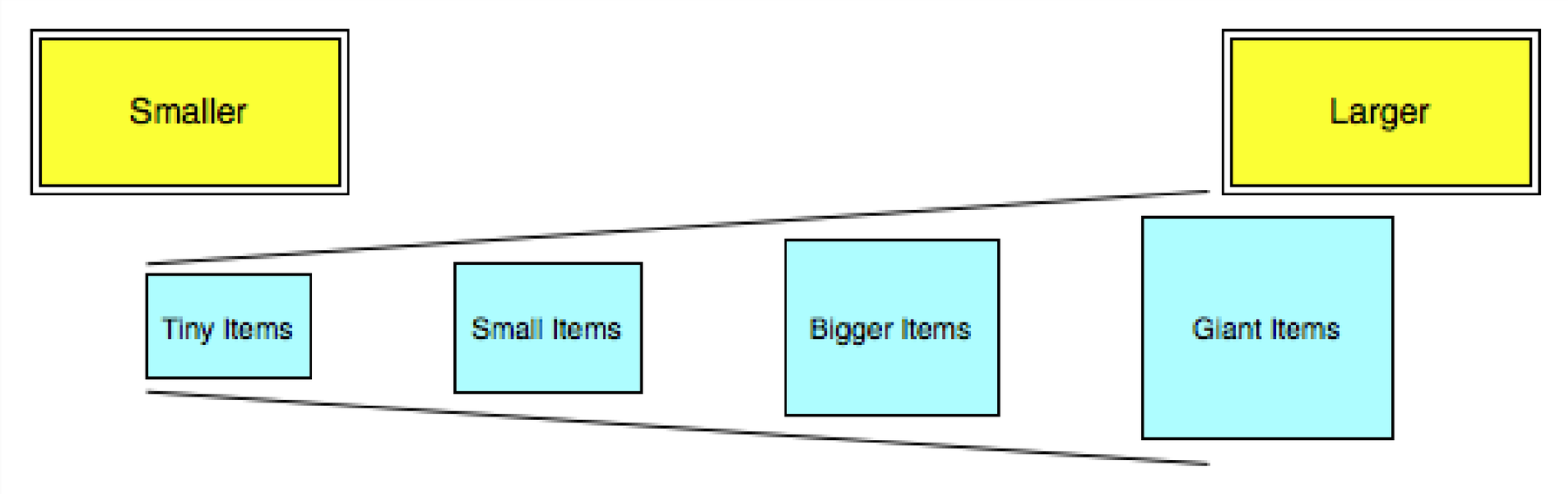
Story brainstorming

- *Step 1*
  - *Focus on quantity*
  - *10 minutes*
- *Step 2*
  - *Focus on quality*
  - *15 minutes*

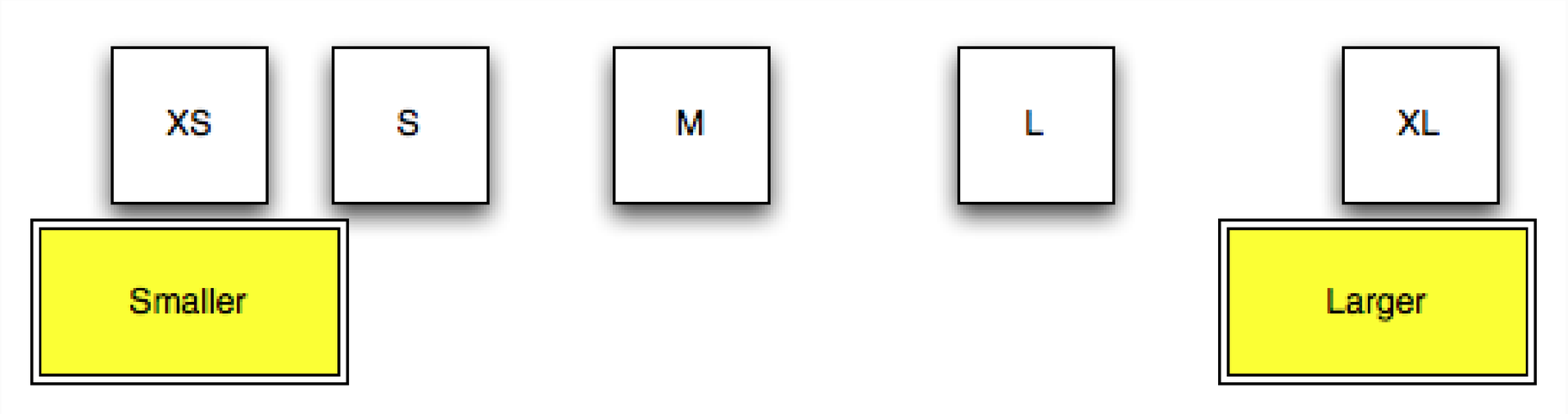
An Agile way to estimate a project in 3 steps ☺

- Story brainstorming
- Affinity estimation
- Velocity prediction

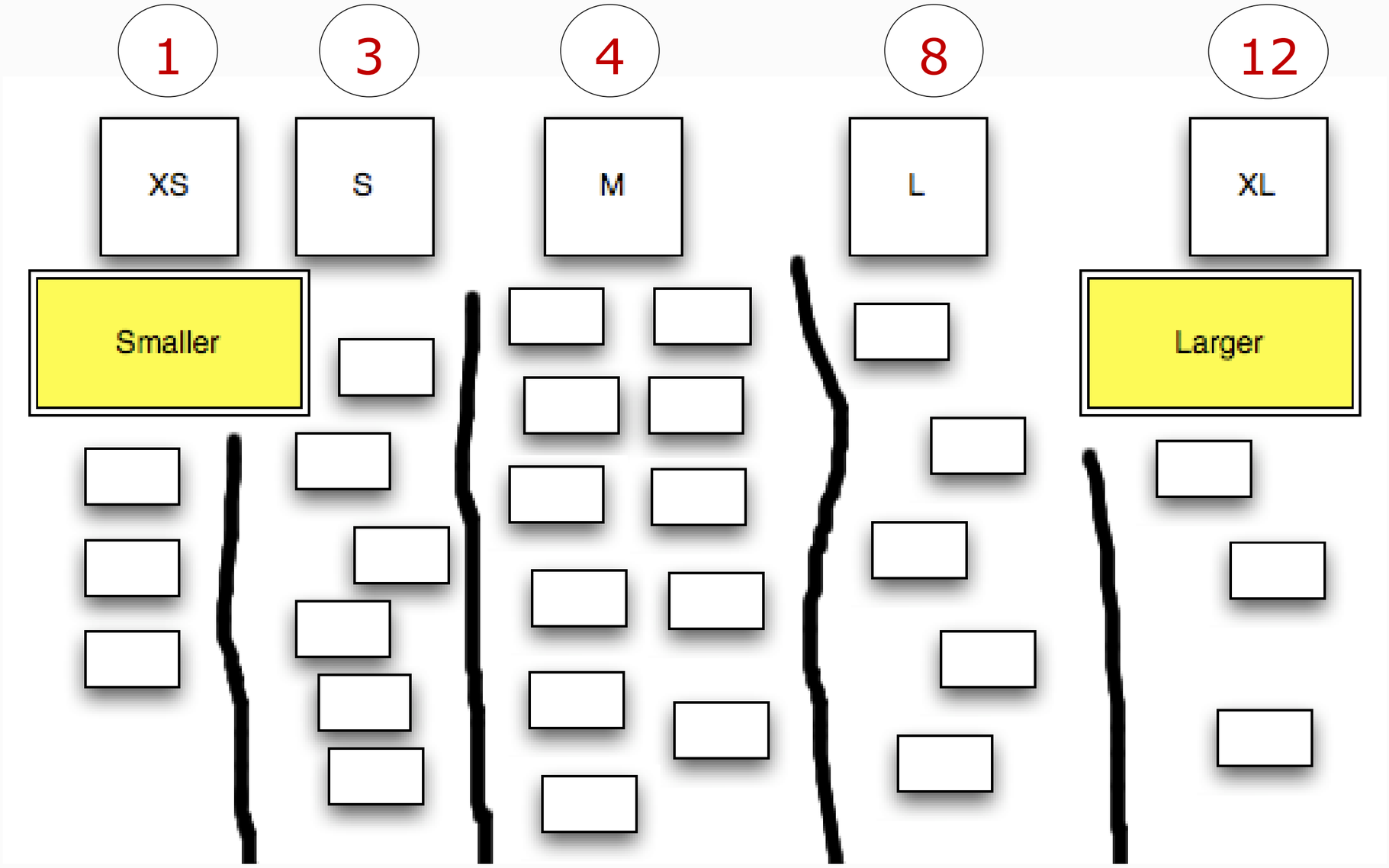
Affinity estimation



Affinity estimation



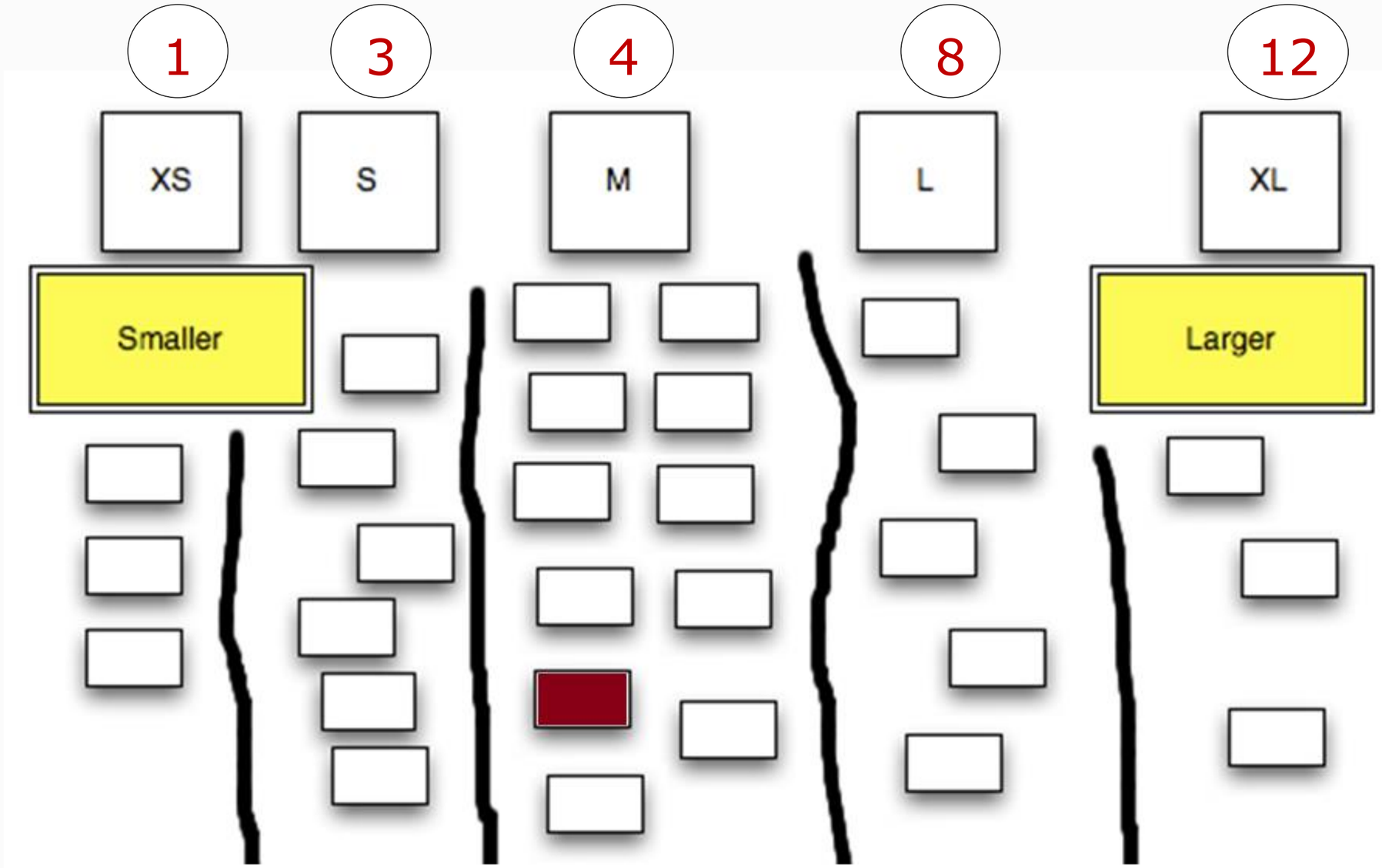
Affinity estimation



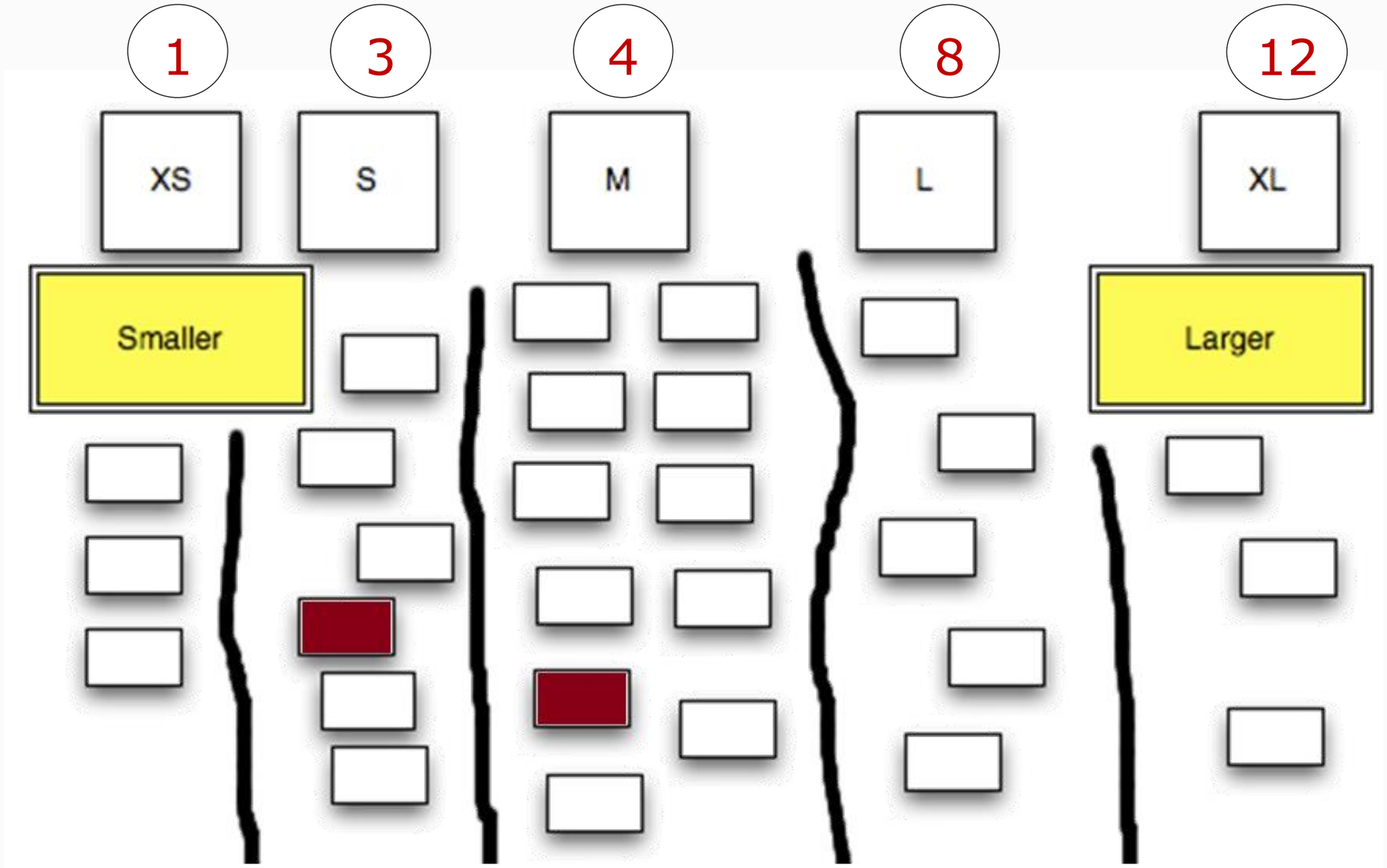
An Agile way to estimate a project in 3 steps ☺

- Story brainstorming
- Affinity estimation
- Velocity prediction

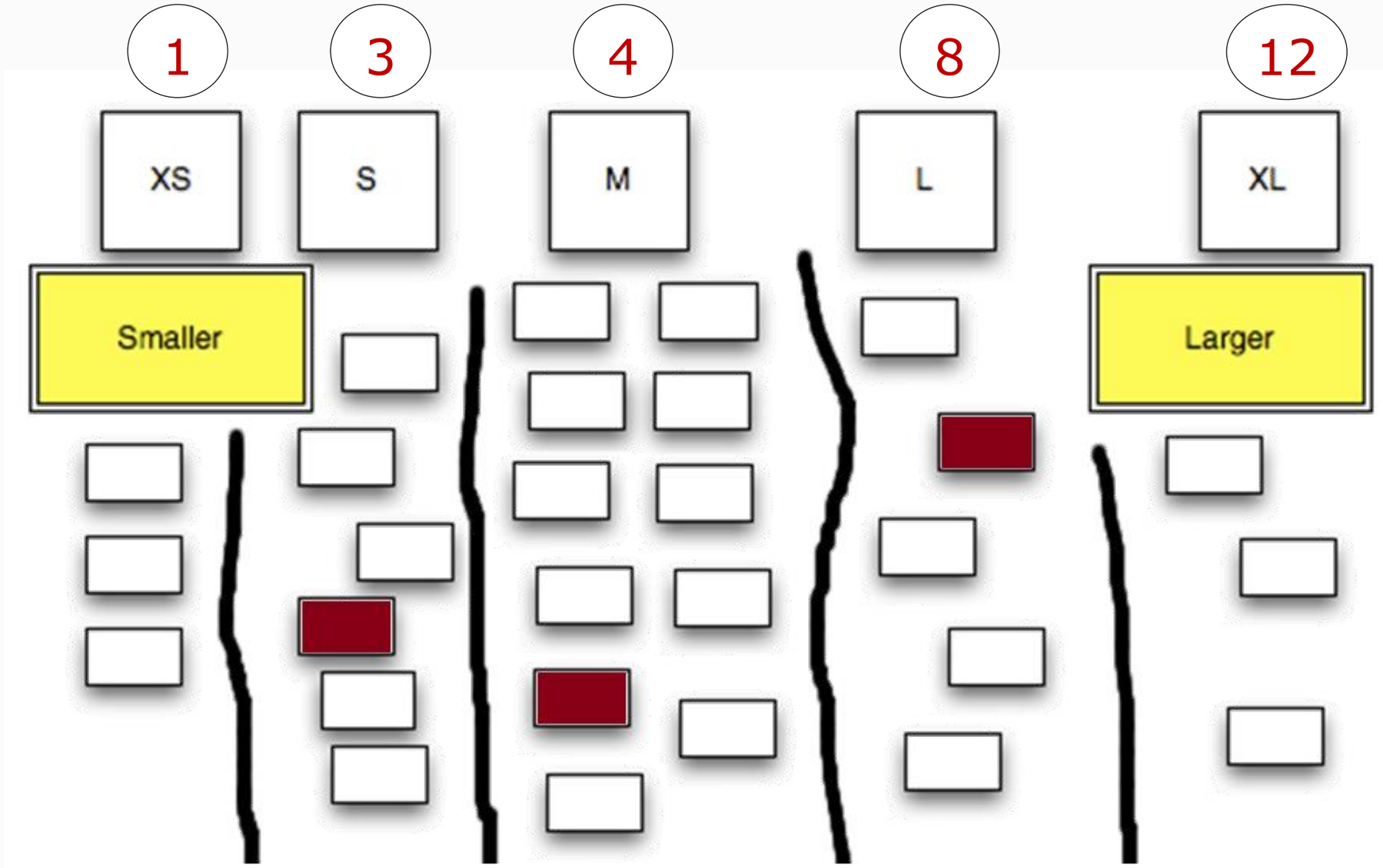
Velocity prediction



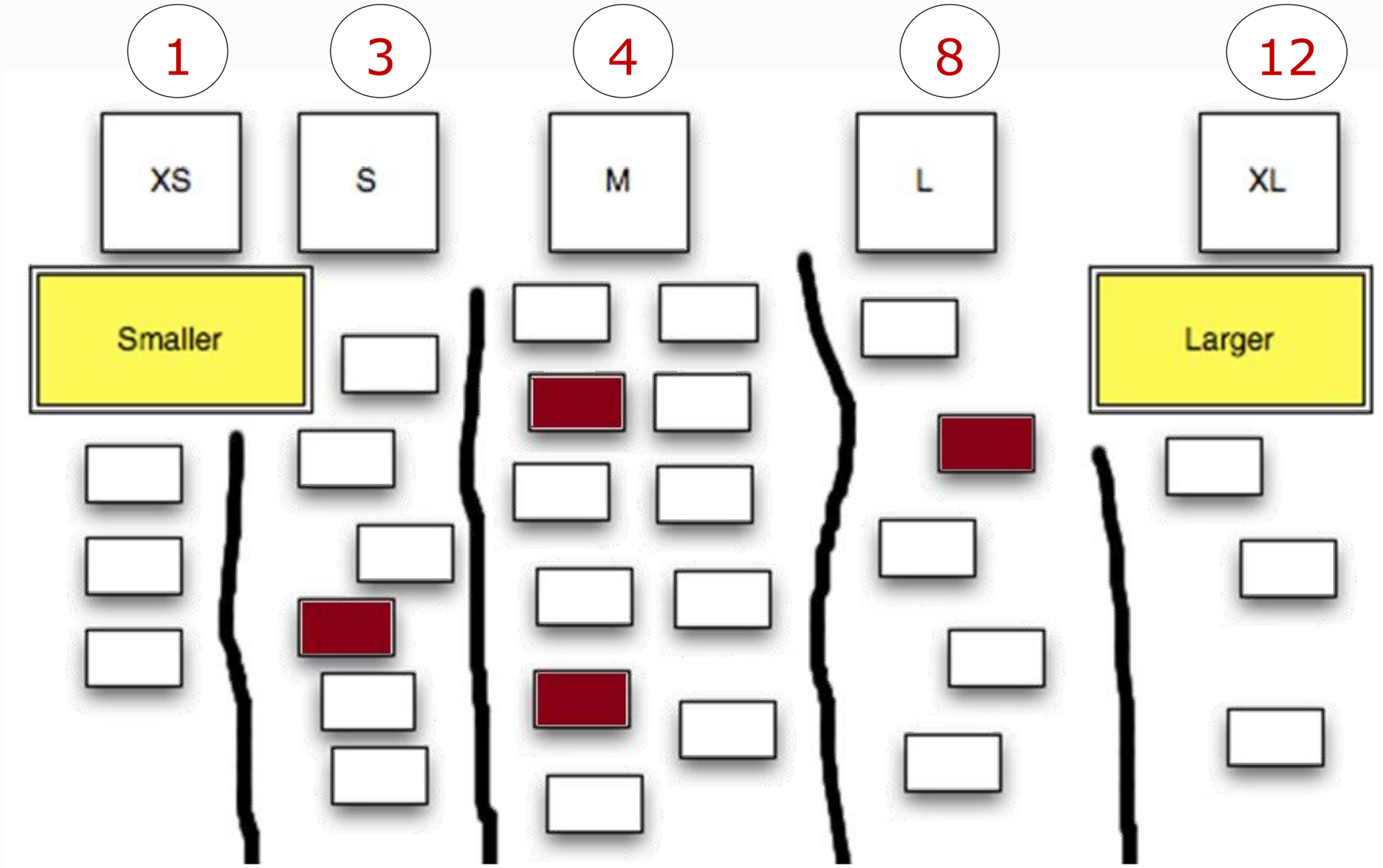
Velocity prediction



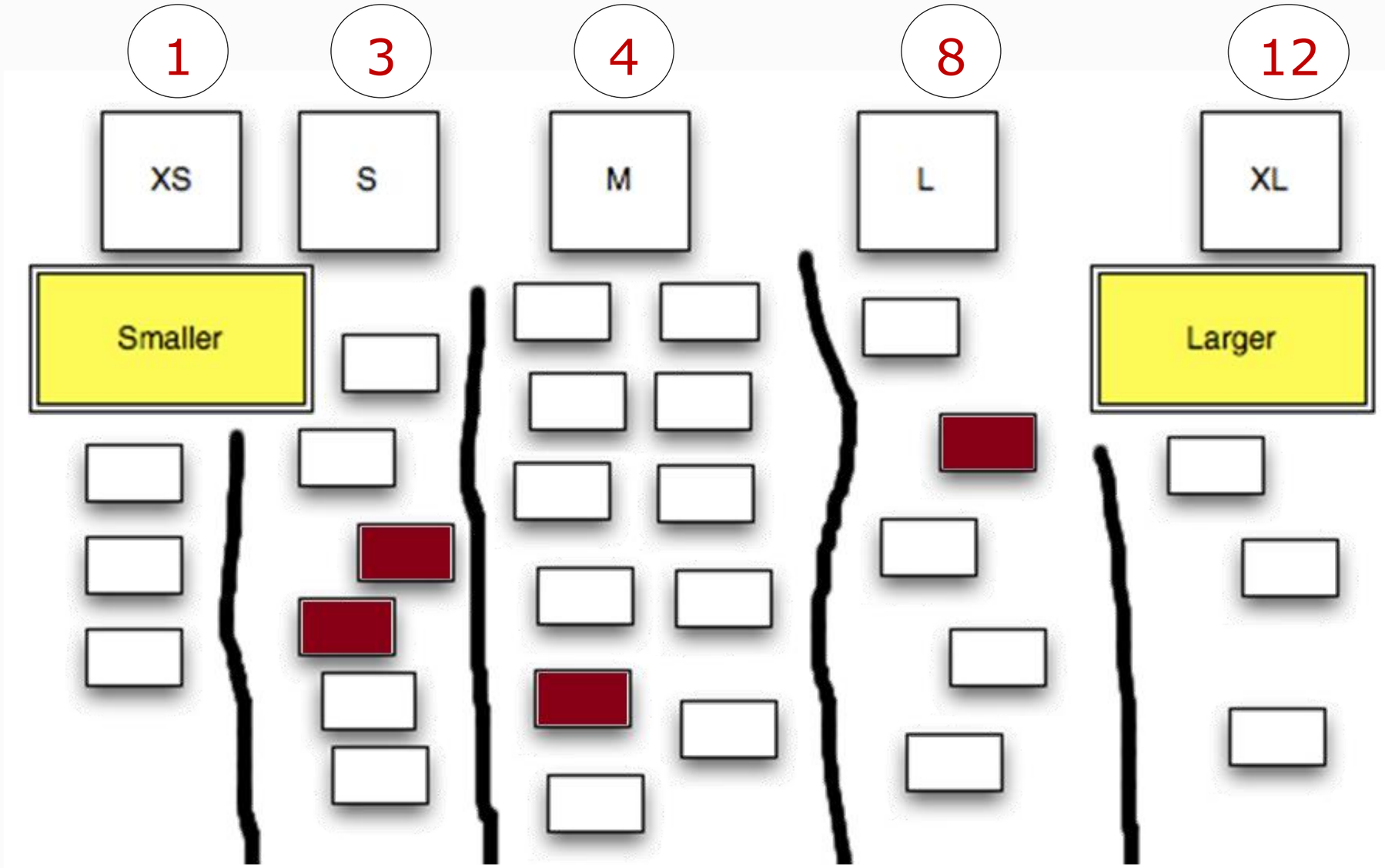
Velocity prediction



Velocity prediction



Velocity prediction



Velocity prediction

18

Extreme programming

XP

5

XP is not…

XP is not…

XP is not…

XP is not…

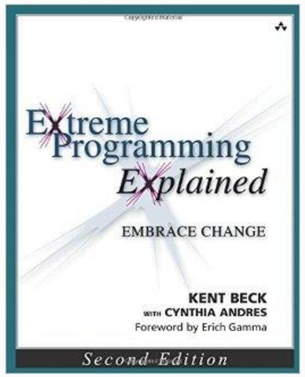
XP is…

…. a lightweight software development methodology for small to medium sized teams developing software in the face of t vague or rapidly changing requirements”

Kent Beck

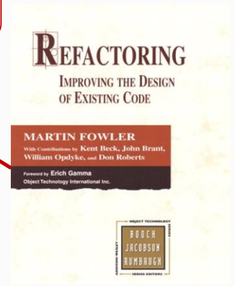
XP is...

.... a lightweight software development methodology

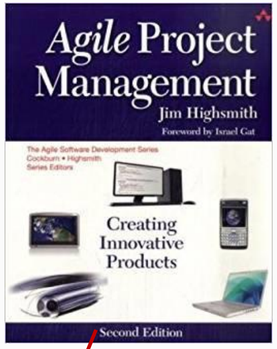


XP

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

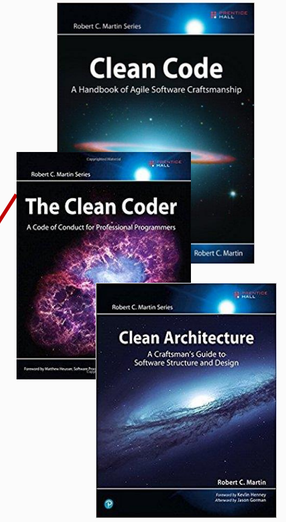


Crystal



James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

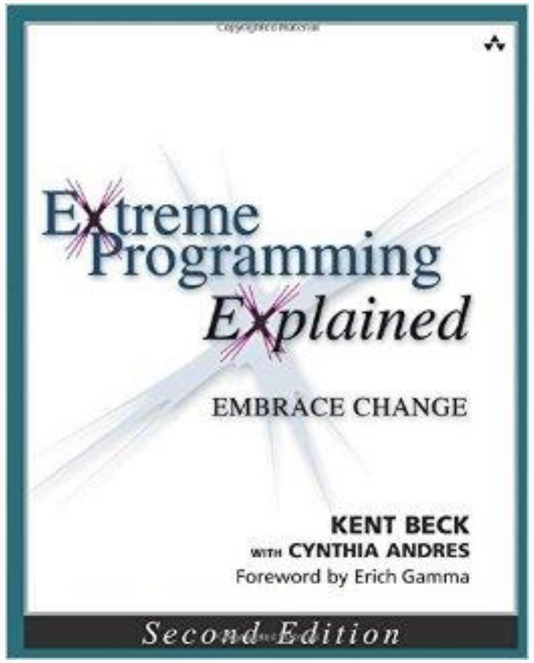


Scrum

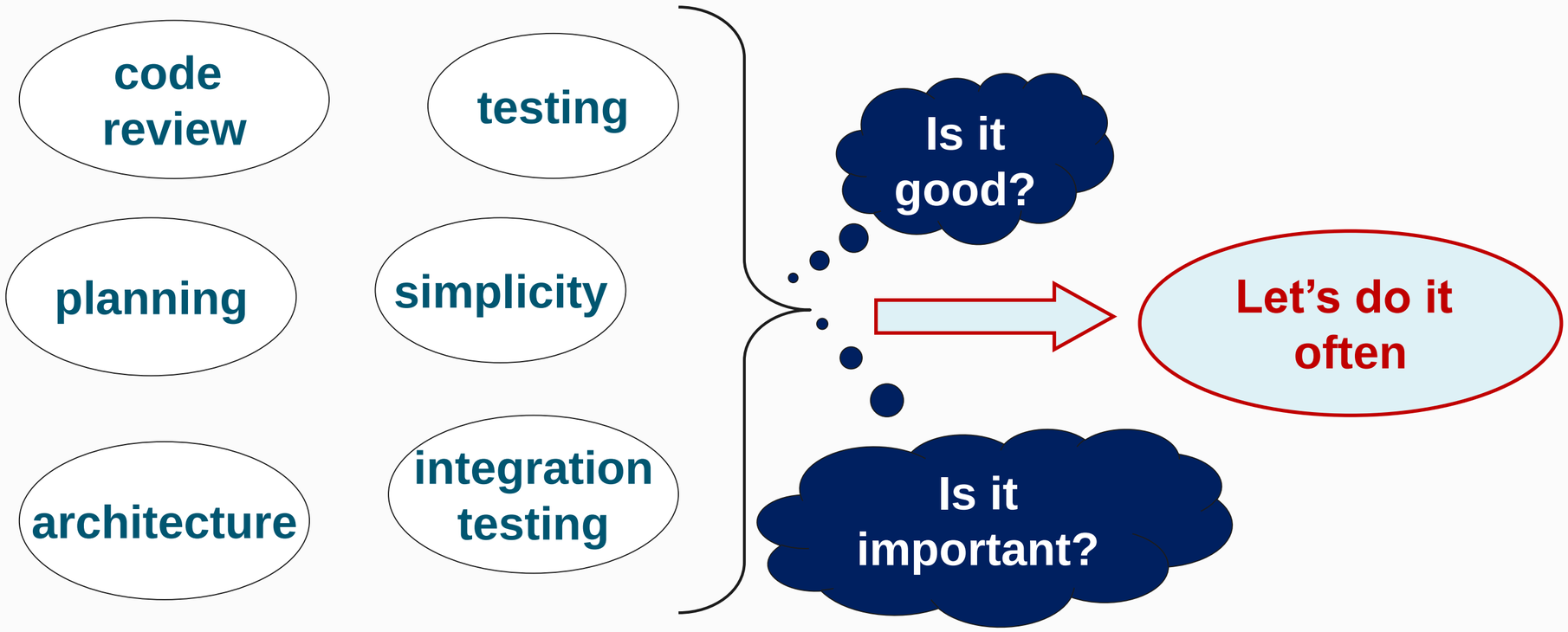
Model Driven Architecture

XP is…

…. a lightweight software development methodology



Extreme?

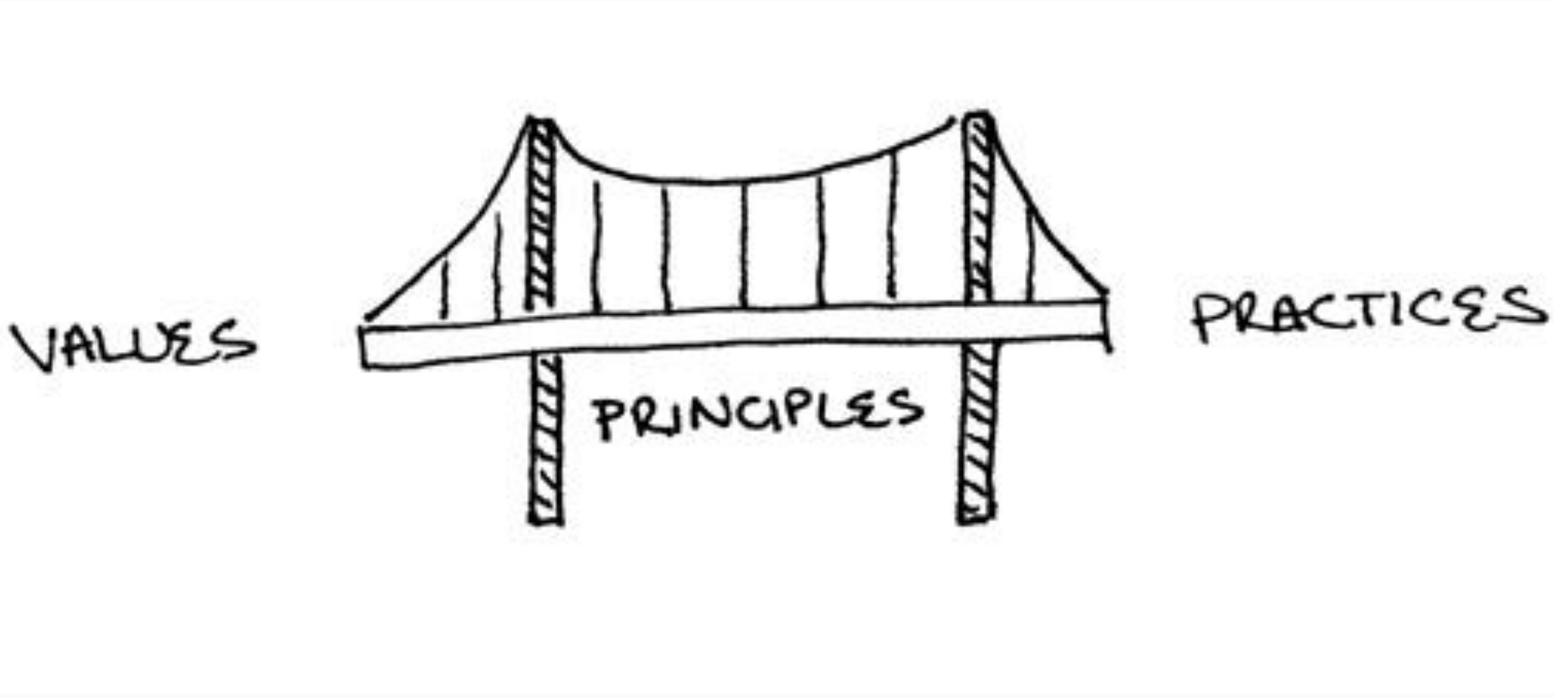

code review
testing
planning
simplicity
architecture
integration testing
Is it good?
Let's do it often
Is it important?

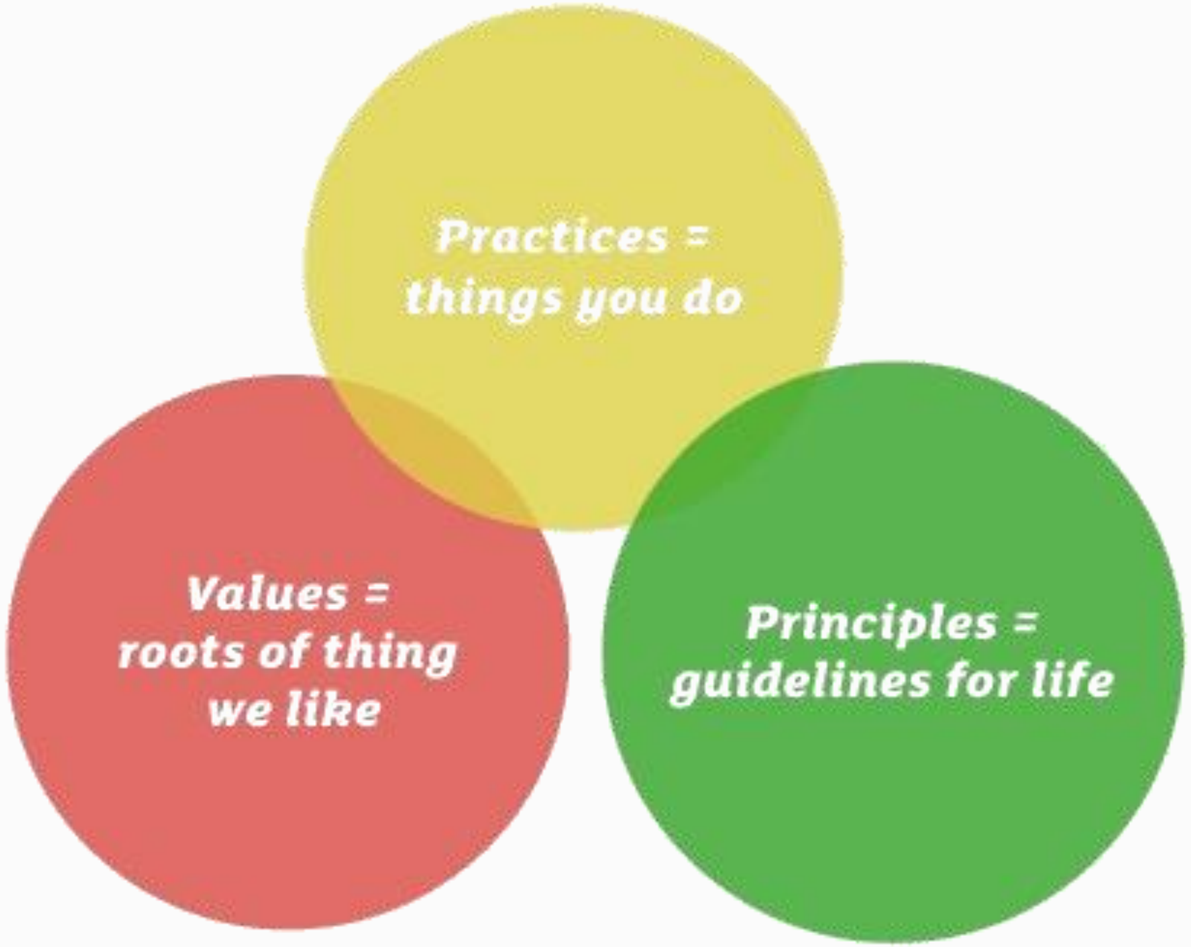
XP Paradigm

Stay aware.

Adapt.

Change.


VALUES
PRINCIPLES
PRACTICES



XP = Outstanding software

XP - Values

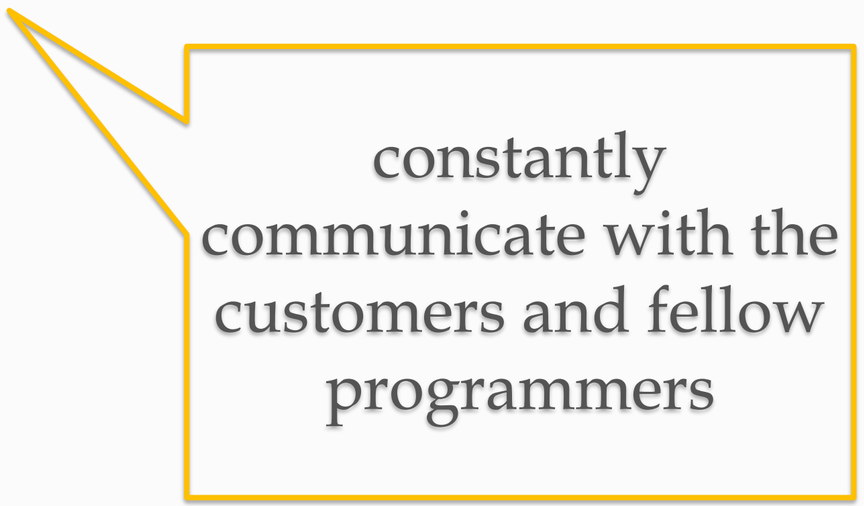
Communication

Simplicity

Feedback

Courage

Respect



XP - Values

Communication

Simplicity

Feedback

Courage

Respect

keep design
simple and clean

XP - Values

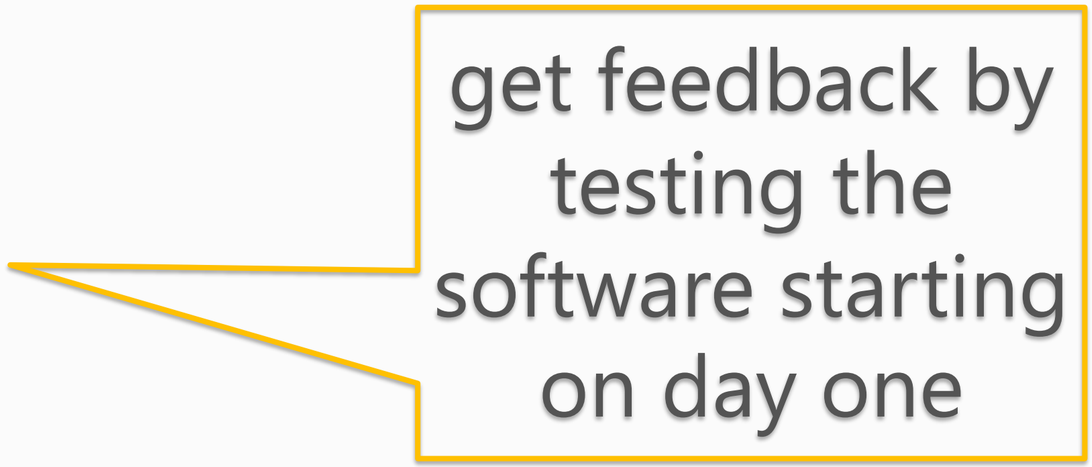
Communication

Simplicity

Feedback

Courage

Respect



XP - Values

Communication

Simplicity

Feedback

Courage

Respect

deliver the system to the customers as early as possible and implement changes as suggested

XP - Values

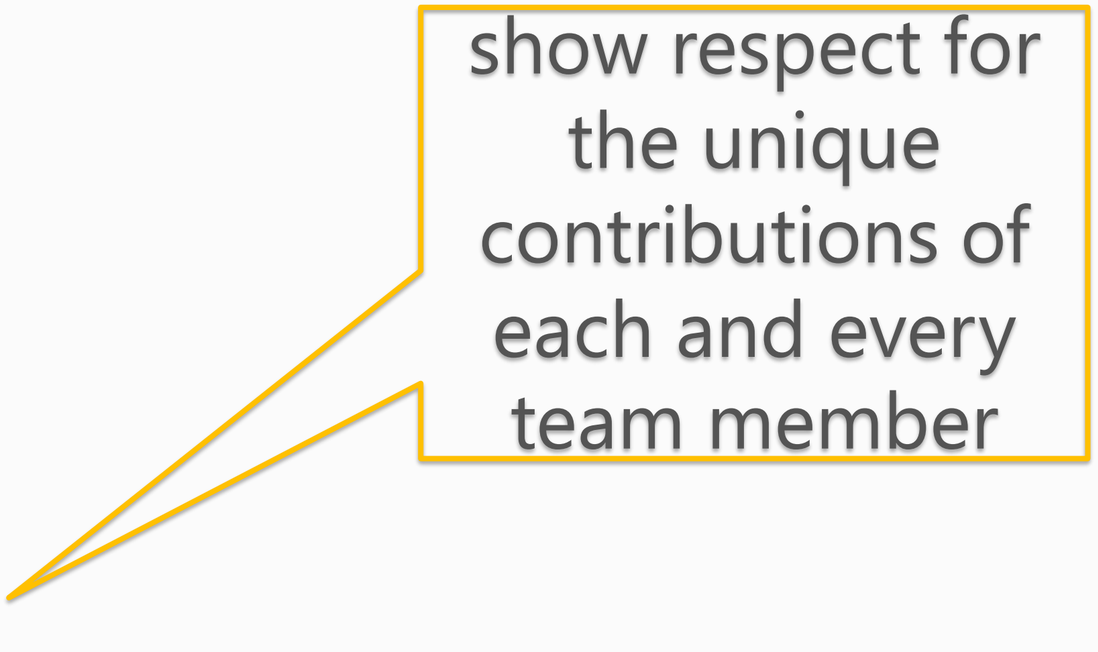
Communication

Simplicity

Feedback

Courage

Respect



XP - Principles

Rapid feedback

Assume simplicity

Incremental change

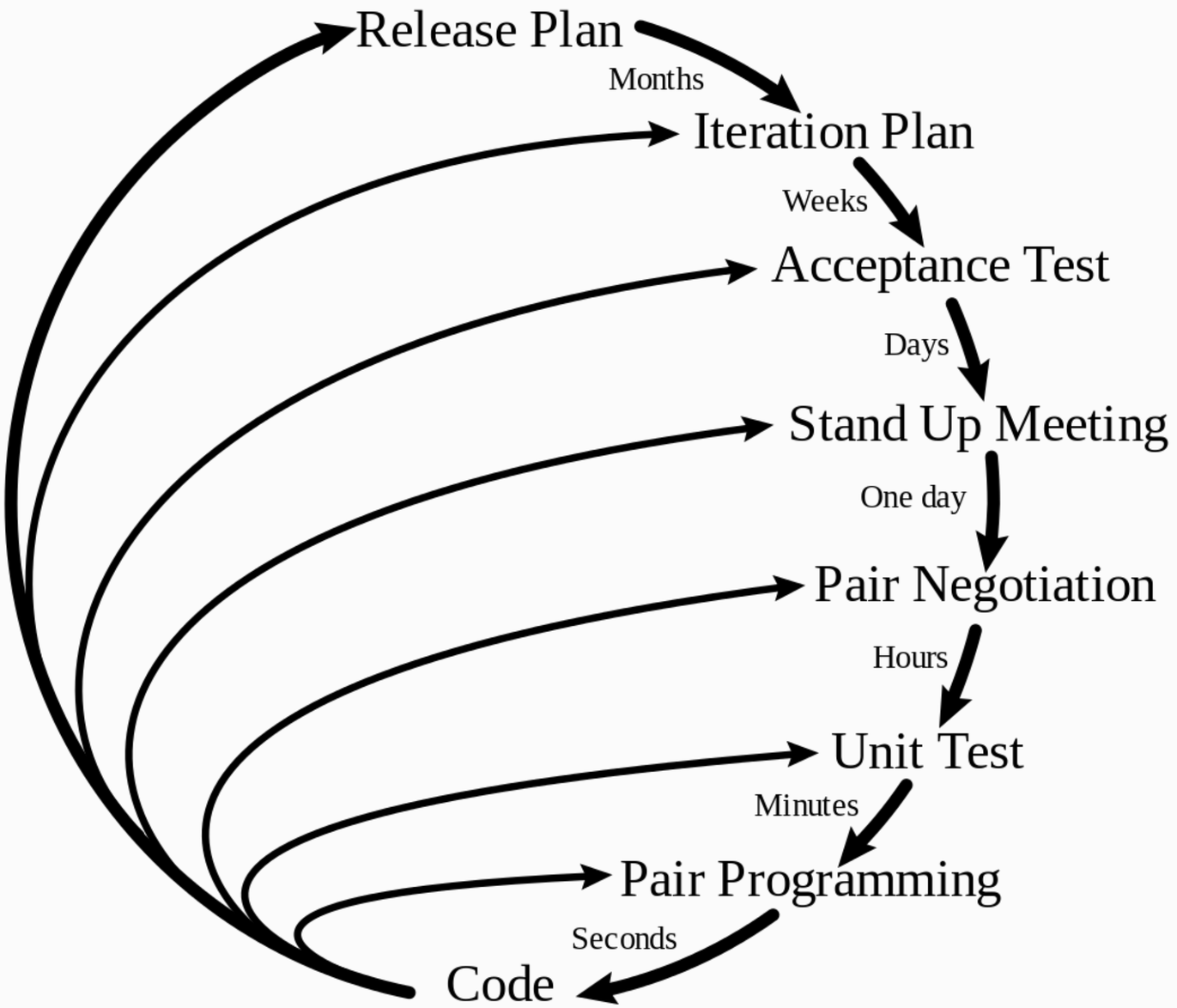
Embracing change

Quality work

get the feedback, understand it, and put the learning back into the system as quickly as possible

- Works as a catalyst for change
- Indicates progress
- Gives confidence to the developers that they are on the right track

Planning/Feedback Loops



XP - Principles

Rapid feedback

Assume simplicity

Incremental change

Embracing change

Quality work

to treat every problem as if it can be solved with simplicity

''Do the simplest thing that could possibly work''
KISS (“Keep It Simple, Stupid”
YAGNI (“You Aren’t Going to Need It”)

XP - Principles

Rapid feedback

Assume simplicity

Incremental change

Embracing change

Quality work

any problem is solved with a series of the smallest change that makes a difference

- The design changes a little at a time.
- The plan changes a little at a time.
- The team changes a little at a time.

XP - Principles

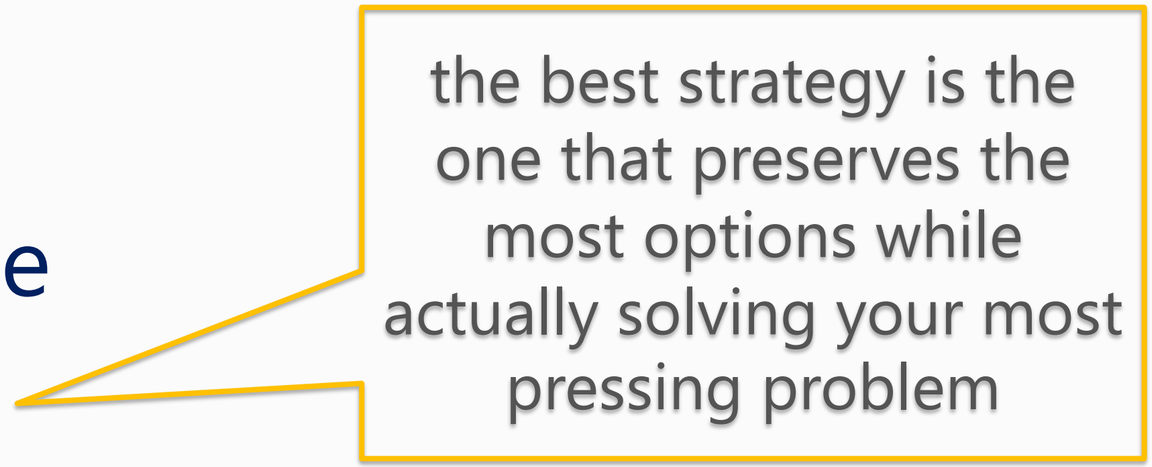
Rapid feedback

Assume simplicity

Incremental change

Embracing change

Quality work



XP - Principles

Rapid feedback

Assume simplicity

Incremental change

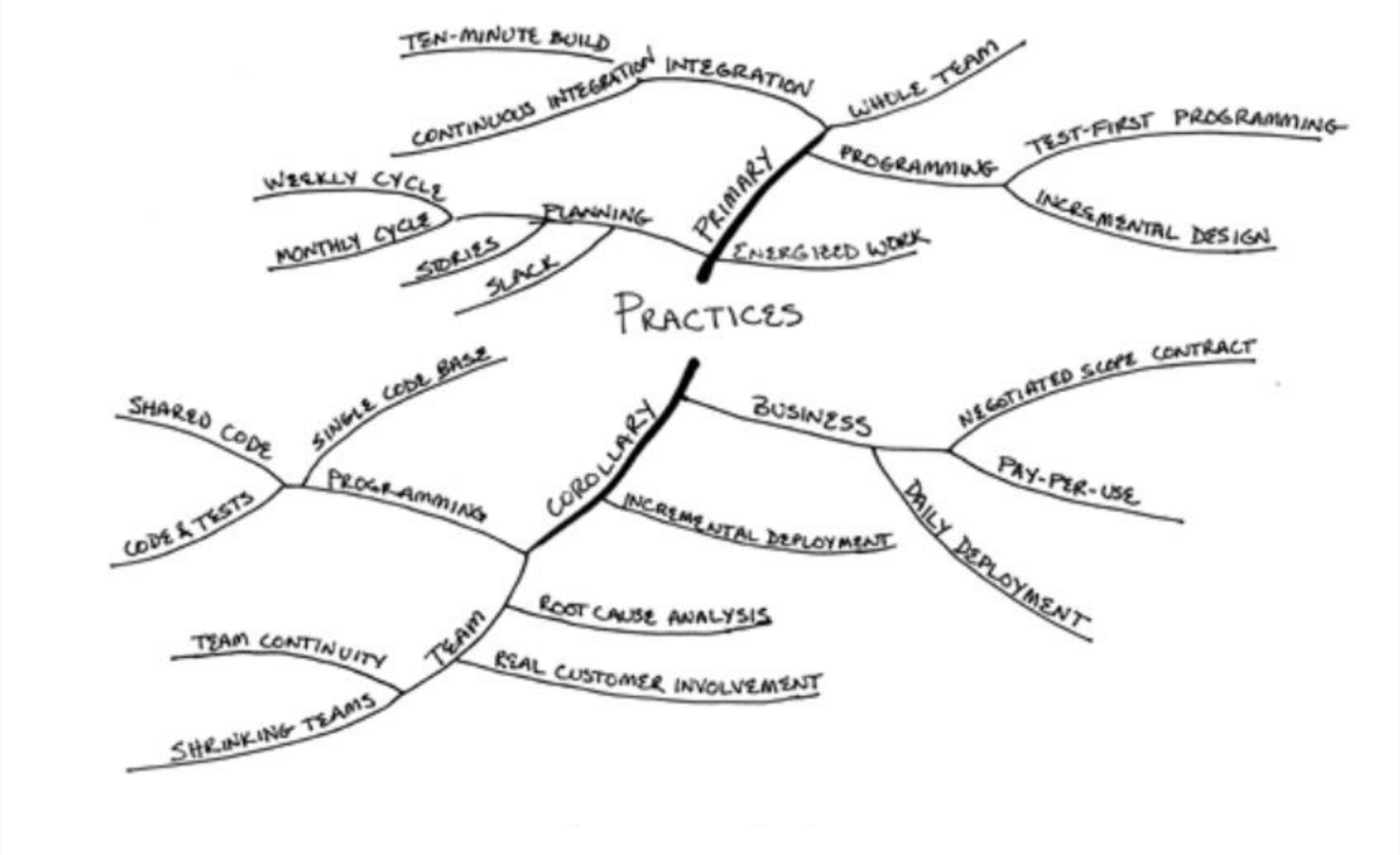
Embracing change

Quality work

The team members try to produce the quality that they are proud of

The team:
Works well
Enjoys the work
Feels good in producing a product of value

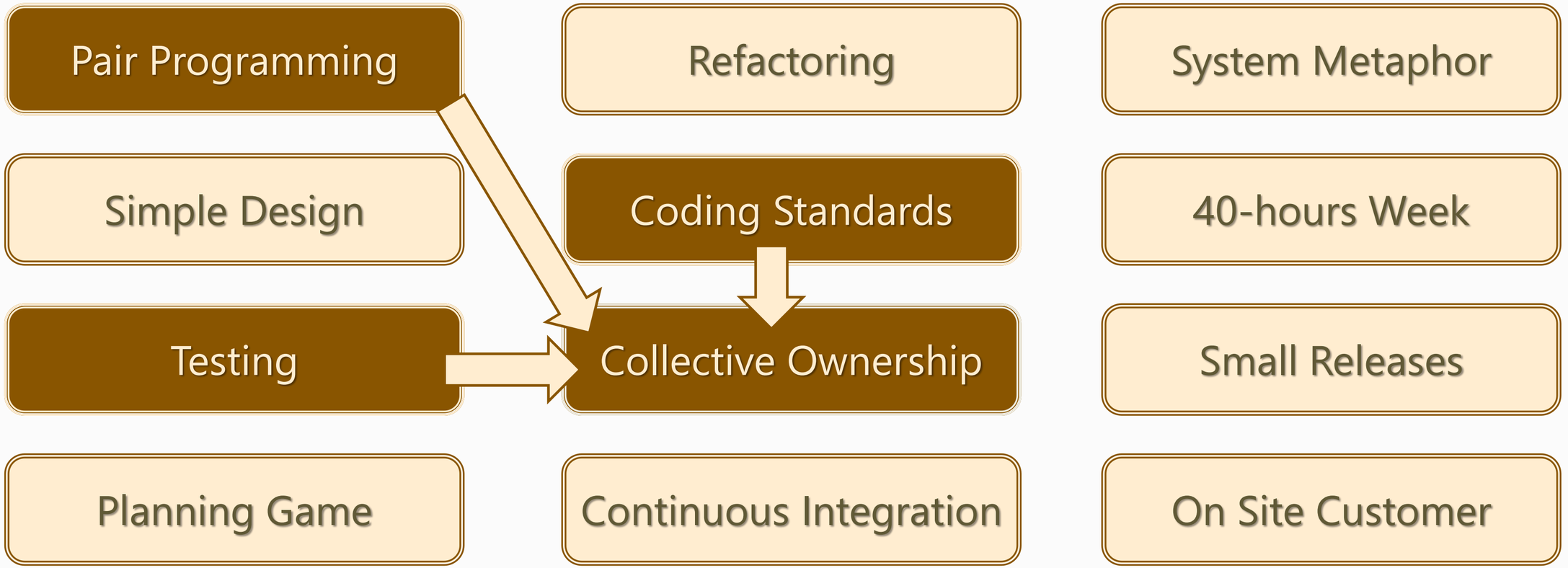
XP - Practices



12 Original **XP** Practices

Pair Programming	Refactoring	System Metaphor
Simple Design	Coding Standards	40-hours Week
Testing	Collective Ownership	Small Releases
Planning Game	Continuous Integration	On Site Customer

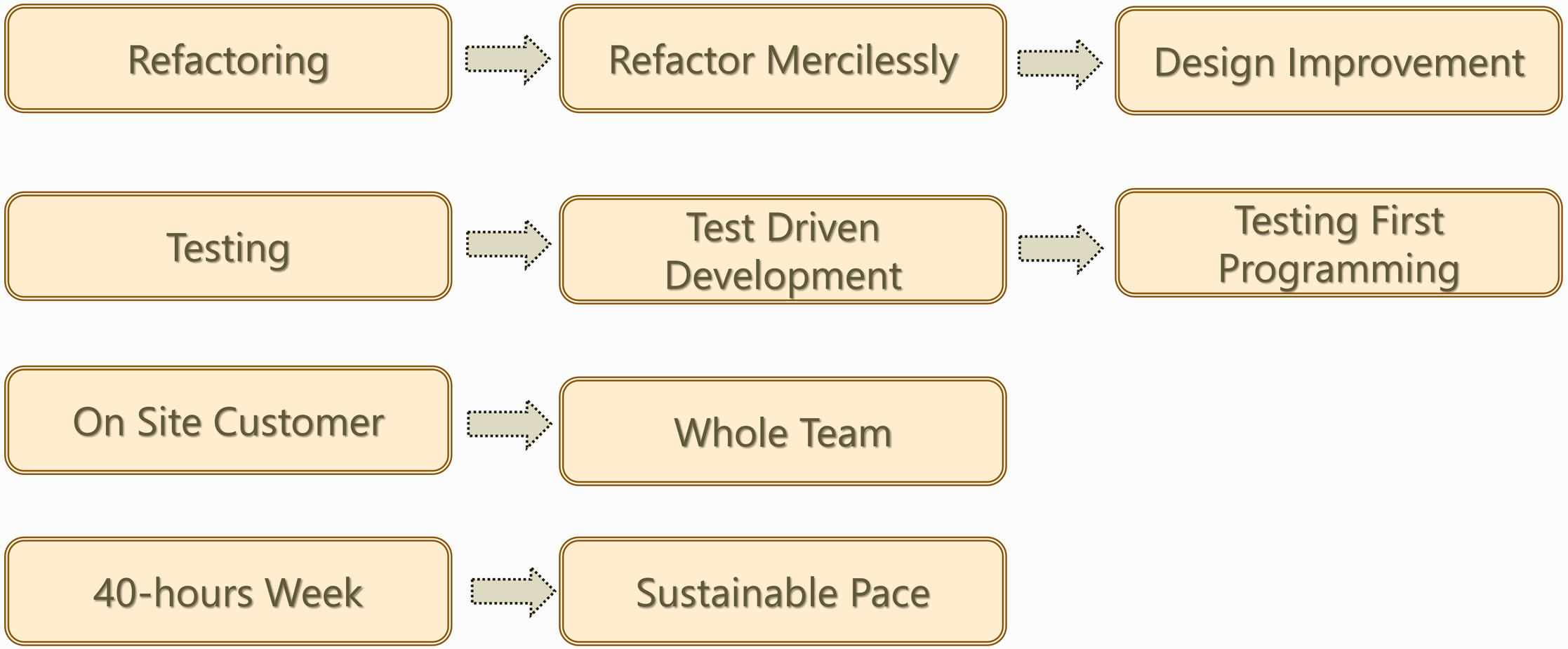
12 Original **XP** Practices

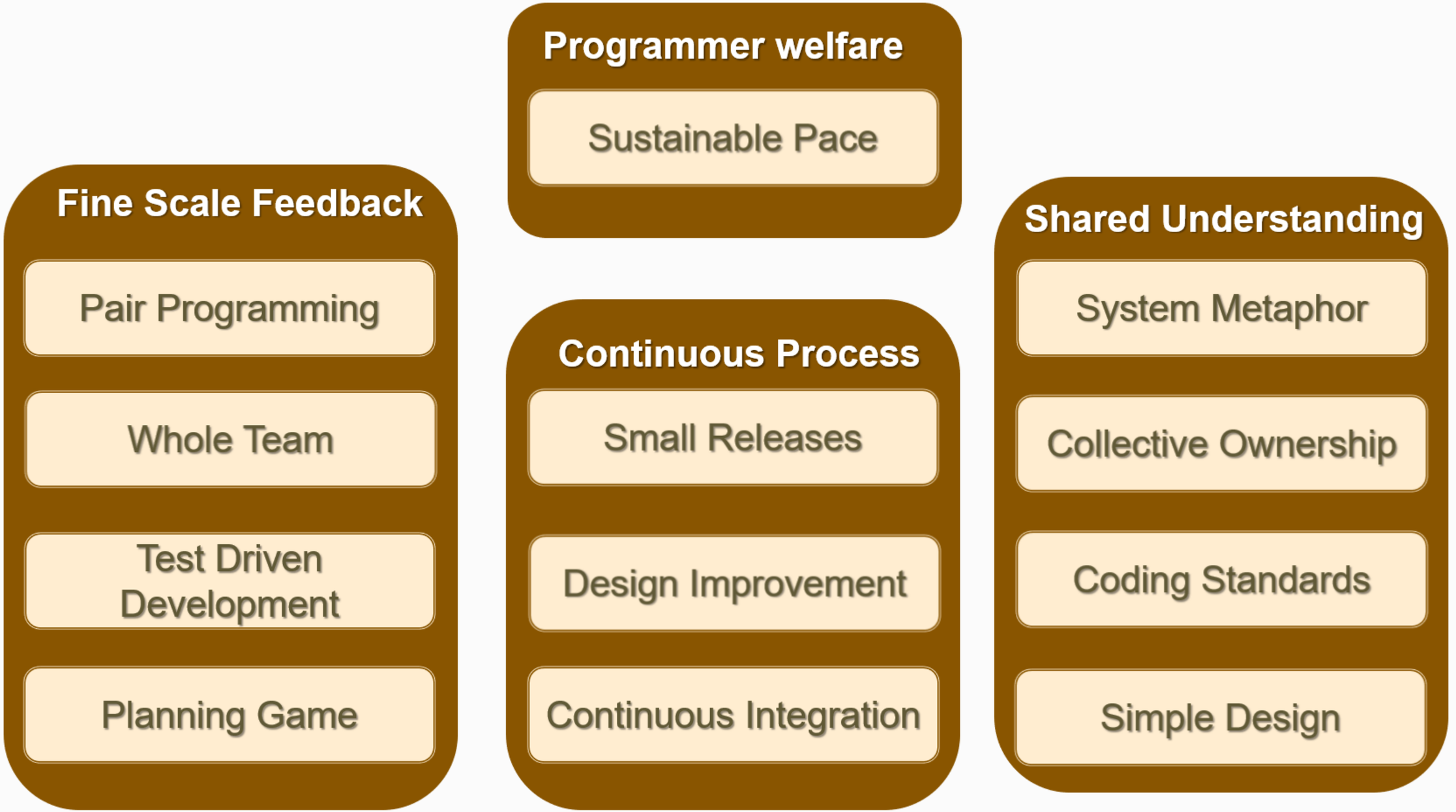


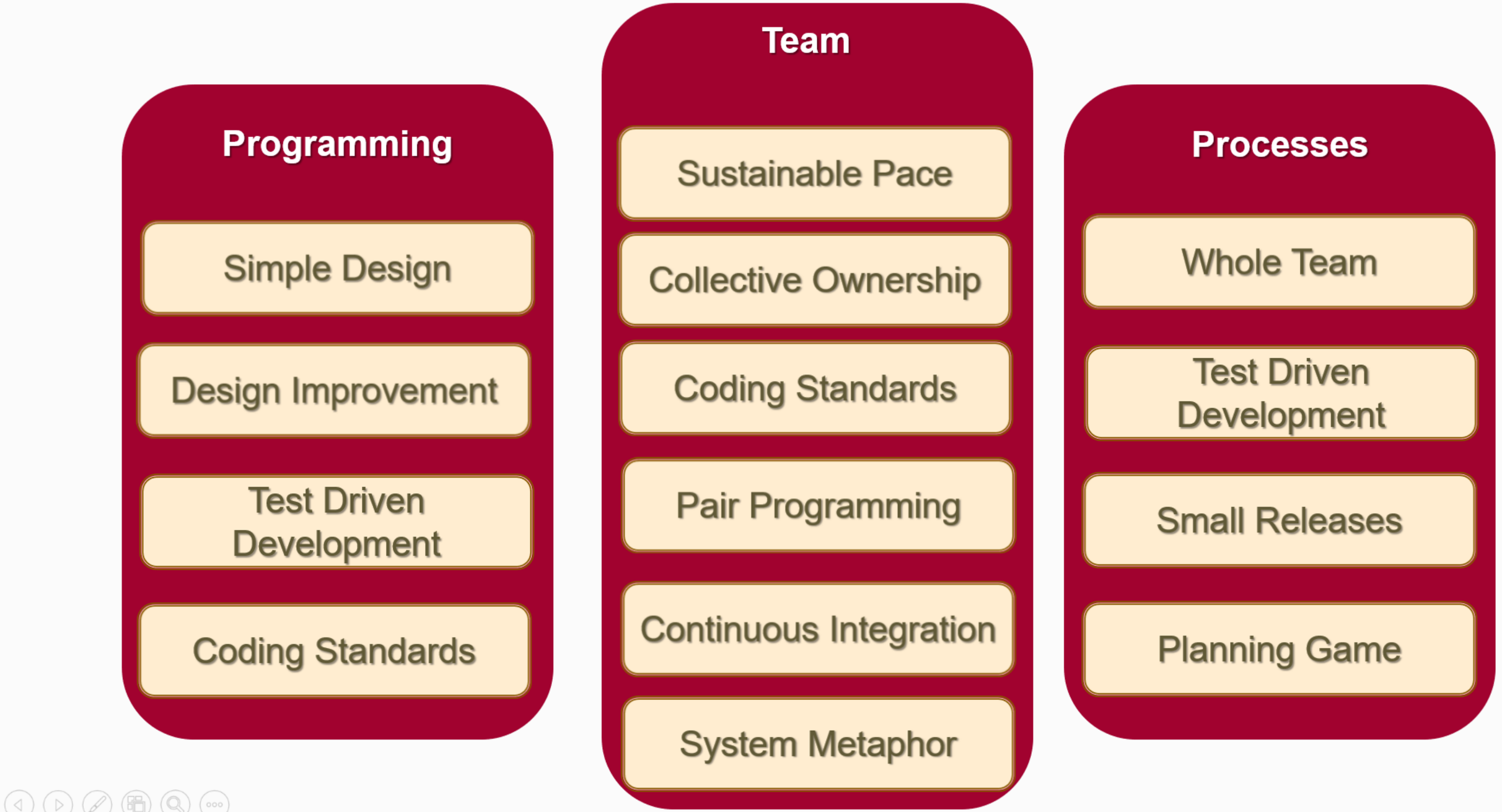
Some practices, if applied in isolation, could bring chaos

12 Original **XP** Practices

Evolution of some practices in time




Programmer welfare
Sustainable Pace
Fine Scale Feedback
Pair Programming
Whole Team
Test Driven Development
Planning Game
Continuous Process
Small Releases
Design Improvement
Continuous Integration
Shared Understanding
System Metaphor
Collective Ownership
Coding Standards
Simple Design


Programming
Simple Design
Design Improvement
Test Driven Development
Coding Standards
Team
Sustainable Pace
Collective Ownership
Coding Standards
Pair Programming
Continuous Integration
System Metaphor
Processes
Whole Team
Test Driven Development
Small Releases
Planning Game

Pair Programming

- Technique that requires 2 people, 1 computer

DRIVER:

- Controls the keyboard
- Writes the code and tests
- Tactics (how?)

NAVIGATOR:

- Has the role of reviewer
- Guides the driver
- Strategy (what?)

Important: they switch roles!!!

When to use it?

- When mentoring new hires
- For extremely high-risk tasks
- At the start of a new project
- When adopting a new technology

Benefits

- Better code, design
- Fewer bugs
- Higher morale
- Shared perspectives and knowledge
- Better time management
- Higher productivity

!!! All without sacrificing productivity

How to do it better?

- Have a well-defined task
- Define a goal at a time
- Rely, support and synchronize with your partner
- Pair with everyone in the team
- Be physically comfortable
- Give everyone a chance to be an expert

Important: communicate!!!

Proxemix

= the study of human use of space

Intimate distance (*for embracing, touching or whispering*)

15 cm - 46 cm

Personal distance (*for interactions among good friends or family*)

46 to 122 cm

Social distance (*for interactions among acquaintances*)

1.2 - 3.7 m

Public distance (*used for public speaking*)

3.7 to 7.6 m