Introduction Agile

Allan Kelly

22 November 2006

http://www.allankelly.net

http://blog.allankelly.net

What is Agile?

- Source in business literature
- What is means is...
  - Business that recognises change
  - Business that responds to change
  - Advantage over competitors by being flexible

Agile for software?

"Agile processes promise to react flexibly to changing requirements, thus providing the highest business value to the customer at any point in time" Jutta Eckstein (2004)

- Software Agile is business focused
- Don't do anything the business doesn't want

Agile manifesto (abridged)

- Highest priority is to satisfy the customer
  - through early and continuous delivery of valuable software.
- Welcome changing requirements
  - even late in development.
  - harness change for the customer's competitive advantage.
- Deliver working software frequently
- Business people and developers work together every day

Agile manifesto 2

- Build projects around motivated individuals
- Give them the environment and support they need
  - trust them to get the job done.
- Face-to-face conversation is the most efficient and effective means of conveying information
- Working software is the primary measure of progress
- Promote sustainable development.
  - Seek to maintain a constant pace indefinitely.

Agile manifesto 3

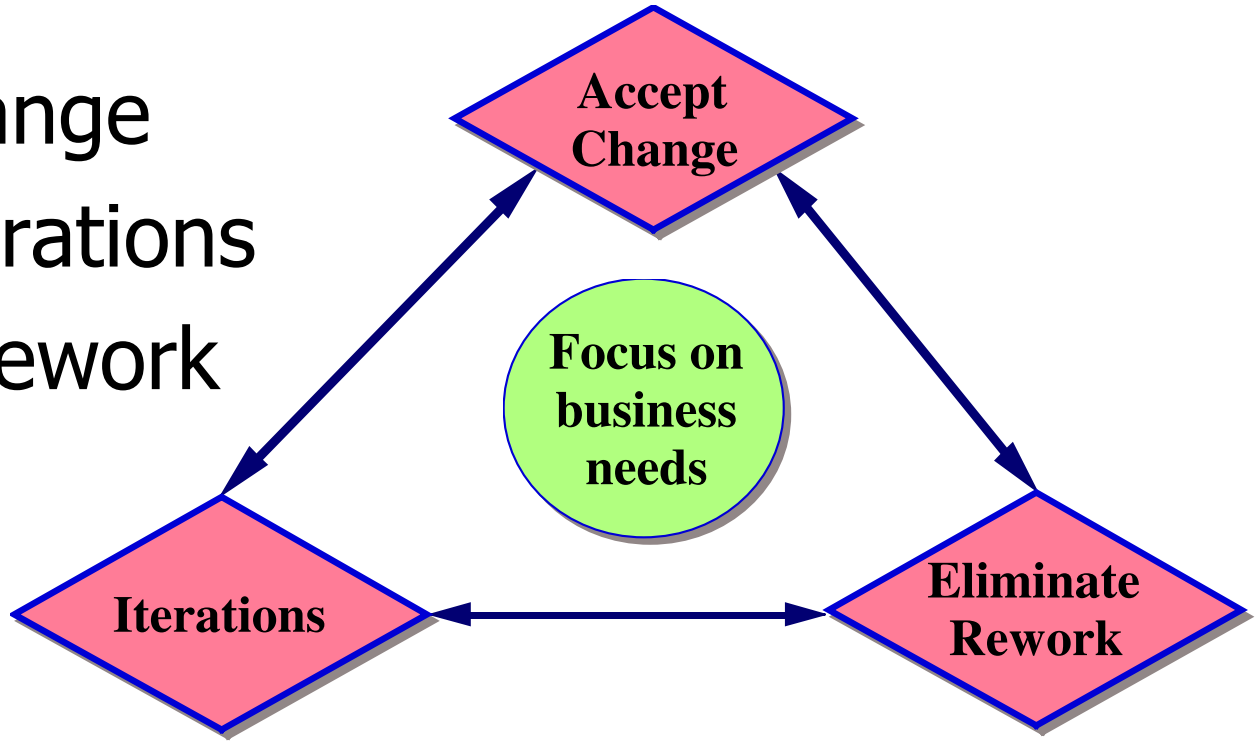
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity the art of maximizing the amount of work not done
- The best architectures, requirements, and designs emerge from self-organizing teams.
- Reflects at regular intervals on how to become more effective: tunes and adjust accordingly

Agile methodologies

- Extreme Programming (XP)
- SCRUM
- Dynamic Systems Development Method (DSDM)
- Crystal (Clear, Orange, Web)
- Lean
- Adaptive Software Development (ASD)
- Feature-Driven Development (FDD)
- Blue-White-Red
- ...

Common points

- Accept Change
- Work in iterations
- Eliminate rework



- Continual learning

Accept Change

- Change the way we do design...
  - Stop trying to "Plan for change"
  - RUFD over BUFD
- Cuts both ways
  - Refactoring - part of eliminating rework
- Show and tell
  - Give the business a chance to change its mind

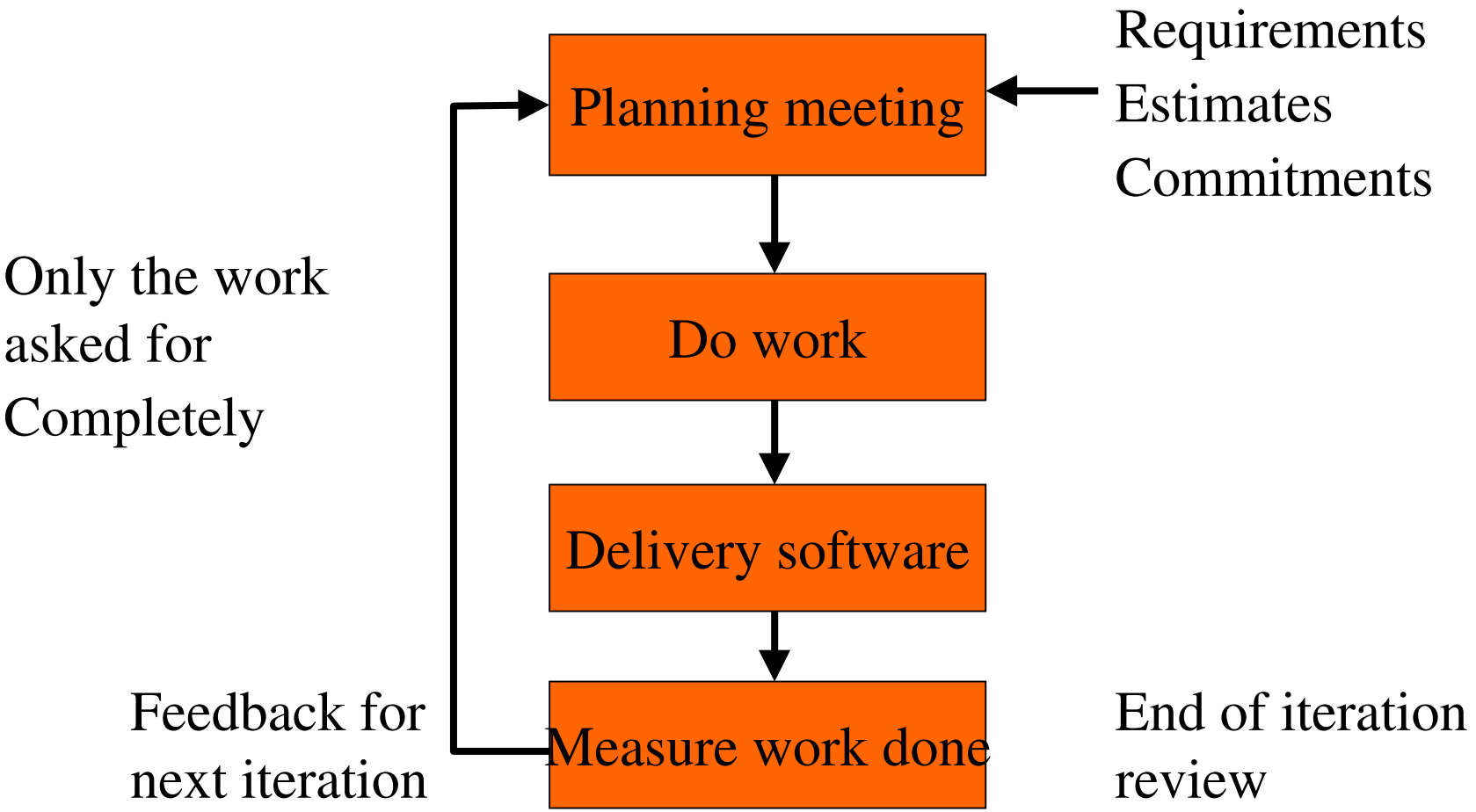
Accept Change 2

- Cut out bureaucracy: impedes change
  - Just enough documentation
- Time-boxing: work in fixed time slots
  - Within slot things are fixed
- Planning meetings: start of the box
- Iterations: Time-boxed piece of work
  - Deliverable at the end

Iterations

- Short period of work
  - 1 week, 2 weeks, 1 month
- Time boxed
- Clearly defined work
- Clear priorities
- Working software at the end
  - Which delivers business value
- Completely self contained

Iterations 2



Queues

- What is a queue?
  - A list of work to do
- Queues are predictable - statistically
  - Think of your local Post Office
- What causes queues?
  - Too much work - too few people
  - Variability of work

Queues 2

- Some things
  - take longer than others
  - take longer than you think
- But.... some things
  - take less time than others
  - take less time than you think
- Why don't they balance?
  - They balance below 76% commitment

What does this mean?

- Queues (back log of work) are caused by
  - Variability of work
  - Trying to do everything - working flat out
  - Work is unpredictable
- So... you can have:
  - Predictable Schedules, or
  - 100% work usage
- How do we work with that?

Predictable schedules

- Business wants predictable schedules
  - Customers want to know when they will get their web-sites
- Managers don't want you lazing about
- So,
  - we have working around 76% law
- Solution: plan to drop work
  - Prioritise in planning meeting

Eliminate Rework

- Poor quality costs more - rework
  - Reporting bugs
  - Fixing bugs
  - Managing the bugs
- What if there were no bugs?
  - How much time would we save?
  - How predictable would schedules be?
  - How much happier would we all be?

Eliminate Rework 2

- If we eliminate rework...
  - We can show the software at any time
  - We can stop development at any time
  - We can deliver it at any time
  - We can deliver it in increments
    - Some business benefit now, more tomorrow
  - No Big Test & Fix at the end
  - Everything is more predictable

How to eliminate rework

- Ideas please...

- Conscientious approach
- Automated unit tests
- Continual build and integration
- Code review
  - Pair programming
- Refactoring

Emergent Design

- Refactoring
  - Can't work with shoddy code
  - Quality code ->
    - Easier to work work
    - Quality products
- The deal.... Developers:
  - Give up BUFD
  - Get the right to improve code at any time

Lean - eliminate waste

- Roots in Toyota Production System
- Just in Time - No buffer stock
  - Saves space, saves money
  - Shows bottleneck and problems
- Quality control
  - Anyone can stop the line
- Work as a team

7 Kinds of Waste

- Inventory
- Extra processing
- Overproduction
- Transportation
- Waiting
- Motion
- Defects

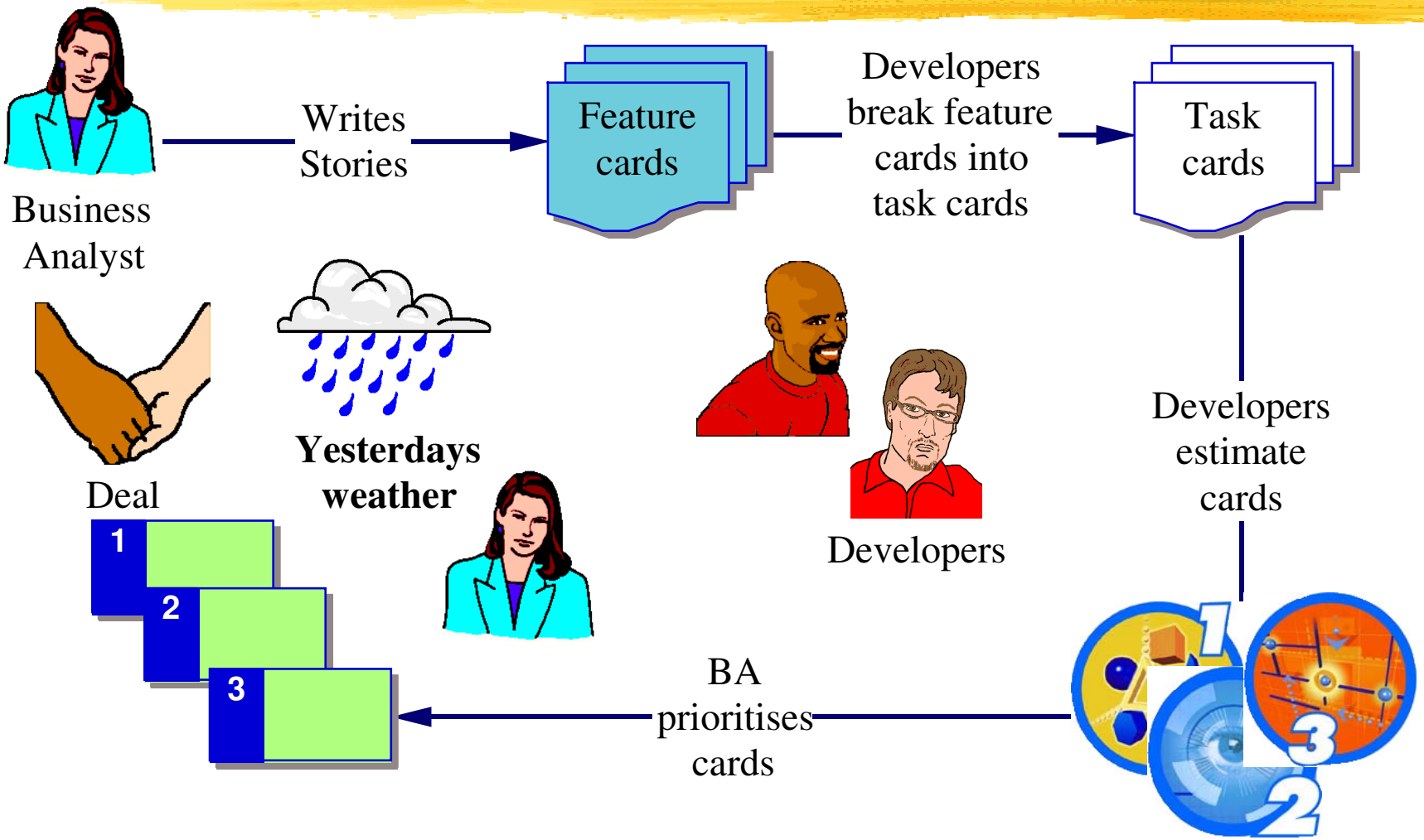
- Partially done work
- Extra processing
- Extra features
- Task switching
- Waiting
- Motion
- Defects

And in software...

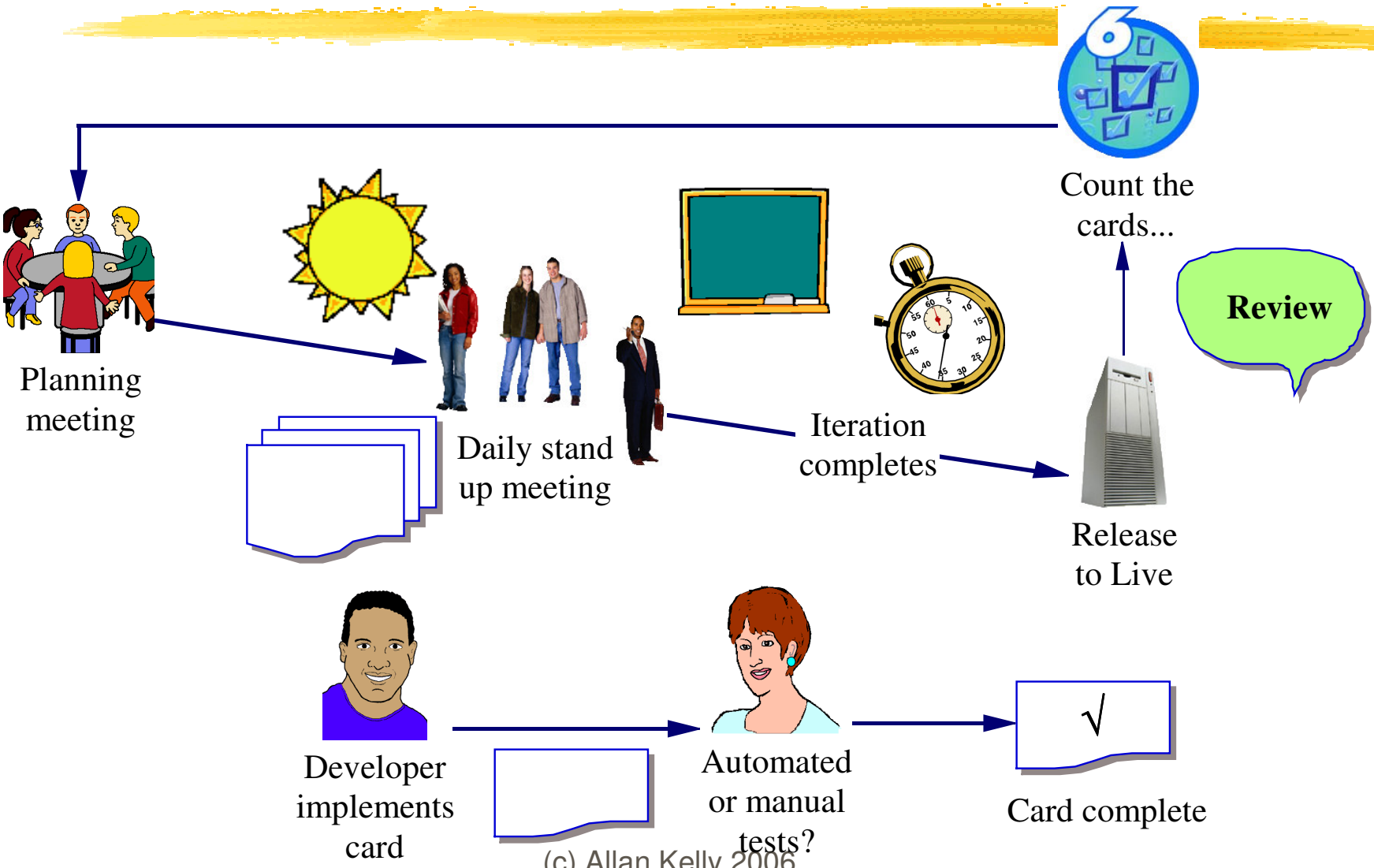
How does it all work?

- Blue-White-Red
  - You will need:
    - White board & pens
    - Blue index cards
    - White index cards
    - Red index cards (just in case)

Blue-White-Red: Planning

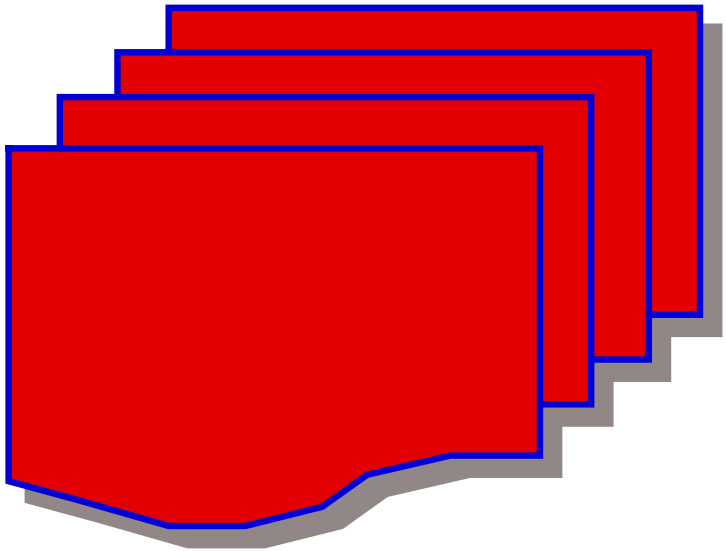


Blue-White-Red: Cycle

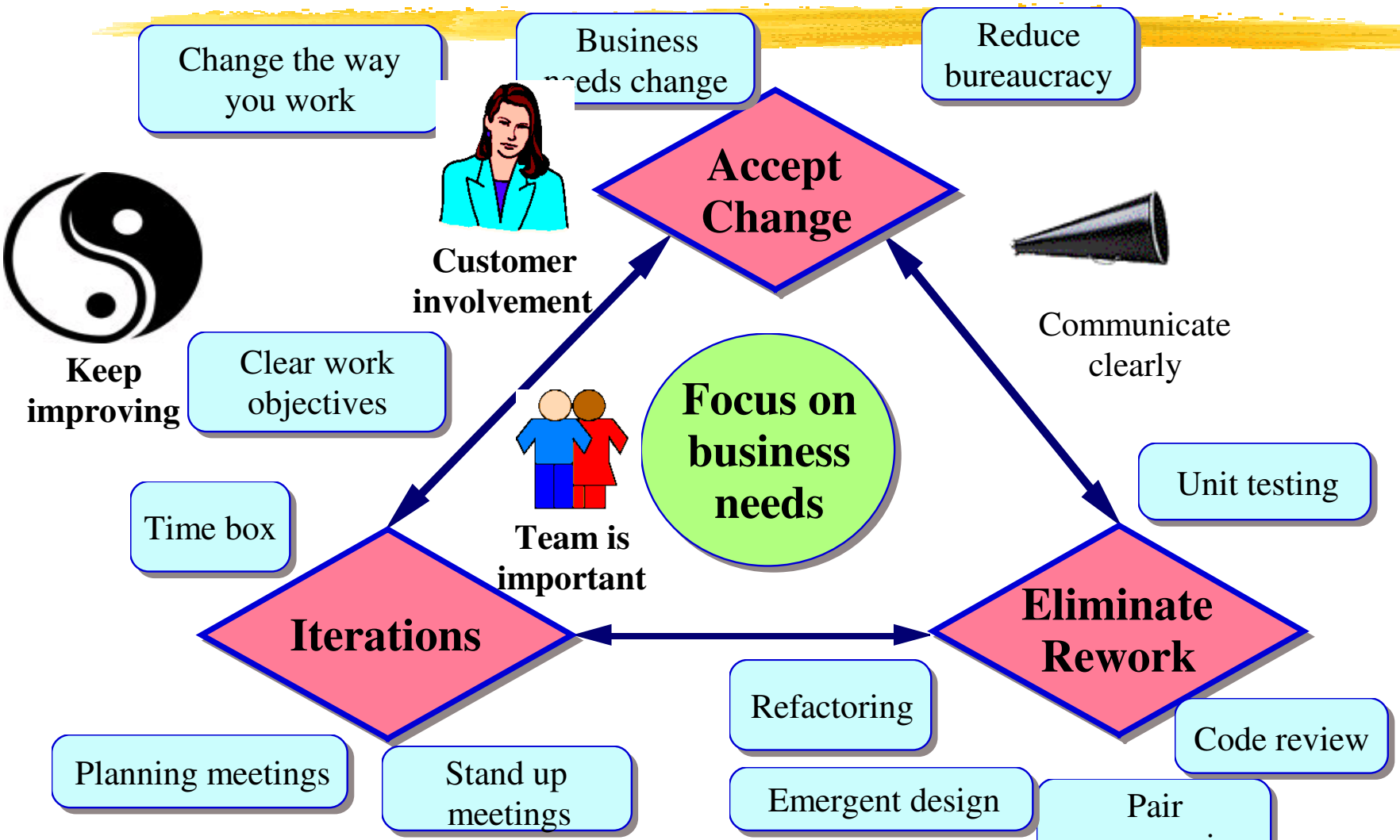


Blue-White-Red: Red cards

- Hope you never need
- Red cards are fault cards
- Red cards trump others
- Project with many Red cards on the board is not healthy



Putting it all together



Further reading

- Beck - *Extreme programming explained*
- Jeffried, Anderson & Hendrickson - *Extreme Programming Installed*
- Poppendieck & Poppendieck - *Lean Software Development*
- *Cockburn - Agile Software Development*
- *Highsmith - Agile Software Development Ecosystems*

Web sites

- Agile Manifesto - agilemanifesto.org
- SCRUM - www.controlchaos.com
- Lean - www.poppendieck.com/
- Alistair Cockburn - alistair.cockburn.us
- Extreme Programming
  - www.extremeprogramming.org
  - www.xprogramming.com

The End.

Questions?