Why Agile?

The Economics, Psychology, and Science of Agile's Success

Matthew Renze

@matthewrenze

Nebraska Code Camp 2014

Purpose

- Explain why Agile practices are so successful
- Insights from:
  - Economics
  - Psychology
  - Science
- Top 7 most important ideas
- Ideas that are not typically covered

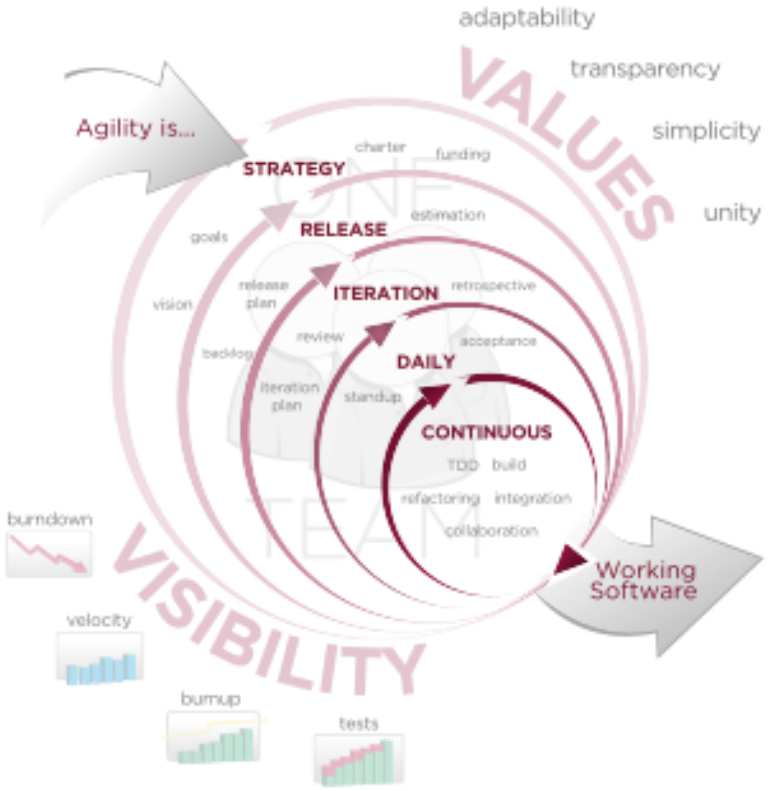
Overview

1. The World after Midnight
2. Inverted Constraints
3. Prioritizing Value
4. Embracing Change
5. Self-Organization
6. Efficient Communication
7. Feedback

A Brief Review of Agile

What is Agile?

- Started with the Agile Manifesto
  - 4 value propositions
  - 12 principles
- Common set of practices across several methodologies



Source: Wikipedia

What is Agile?

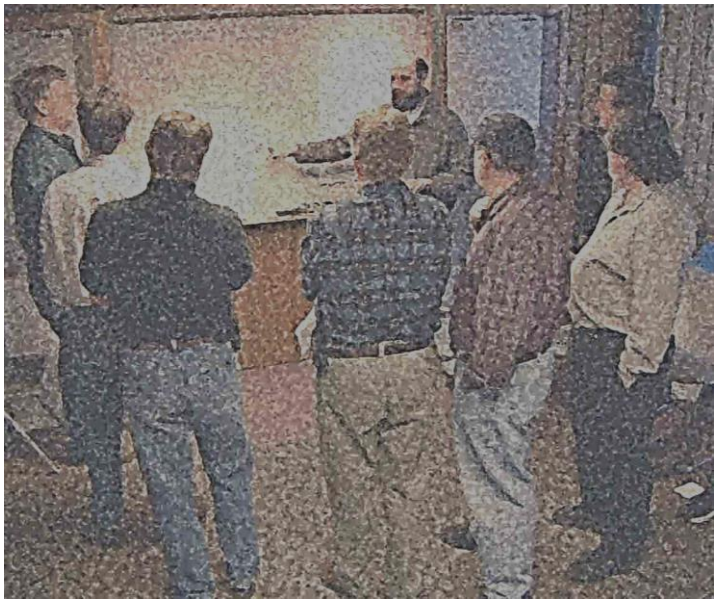
Agile is *not*:

- A software development methodology itself
- A silver bullet for all your software woes

Source: http://www.best-story.net/userfiles/silver-bullets.jpg

Agile Values

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



Source: http://agilemanifesto.org/

12 Principles of Agile

1. Continuous delivery of value
2. Embrace changing requirements
3. Frequent deployment
4. Customer collaboration
5. Motivated individuals
6. Face-to-face conversation

12 Principles of Agile

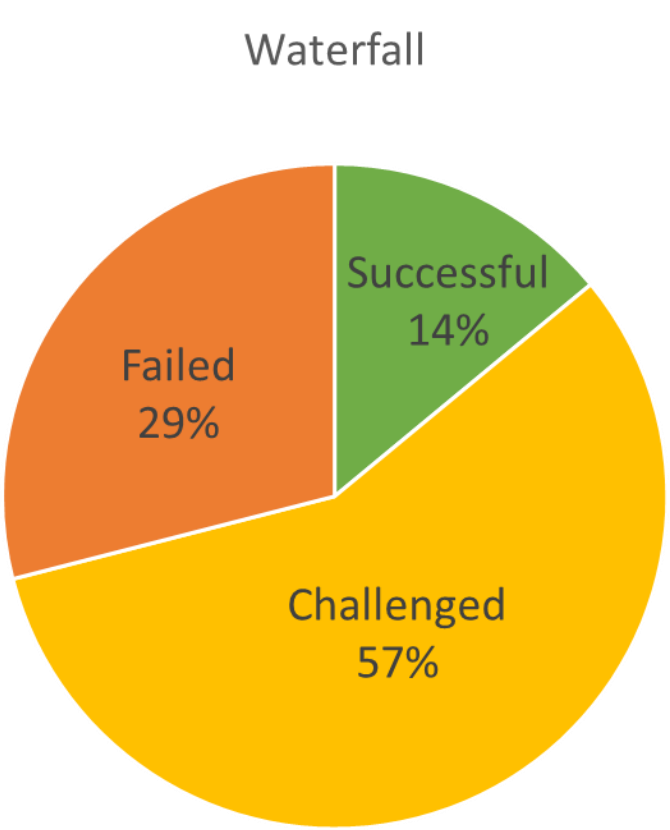
7. Working software as measure of progress
8. Sustainable development
9. Technical excellence
10. Simplicity
11. Self-organization
12. Continuous improvement

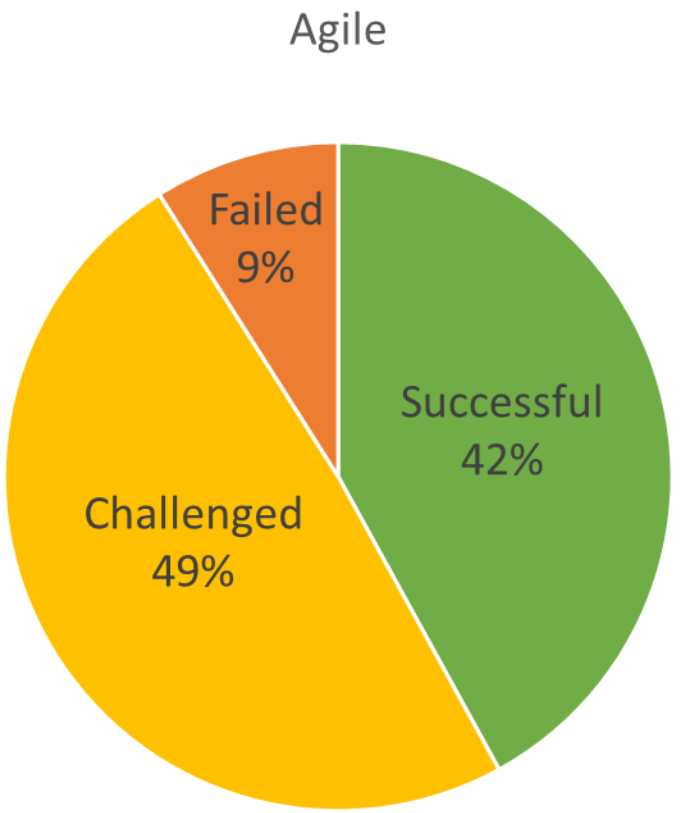
Agile Methodologies

- Scrum
- XP
- Kanban
- Lean
- And many more…

Source: http://parkertoddloesch.files.wordpress.com/2011/09/umbrella.jpg

Is Agile More Successful?





Original Source: The Standish Group, The CHAOS Report 2012

1. The World after Midnight

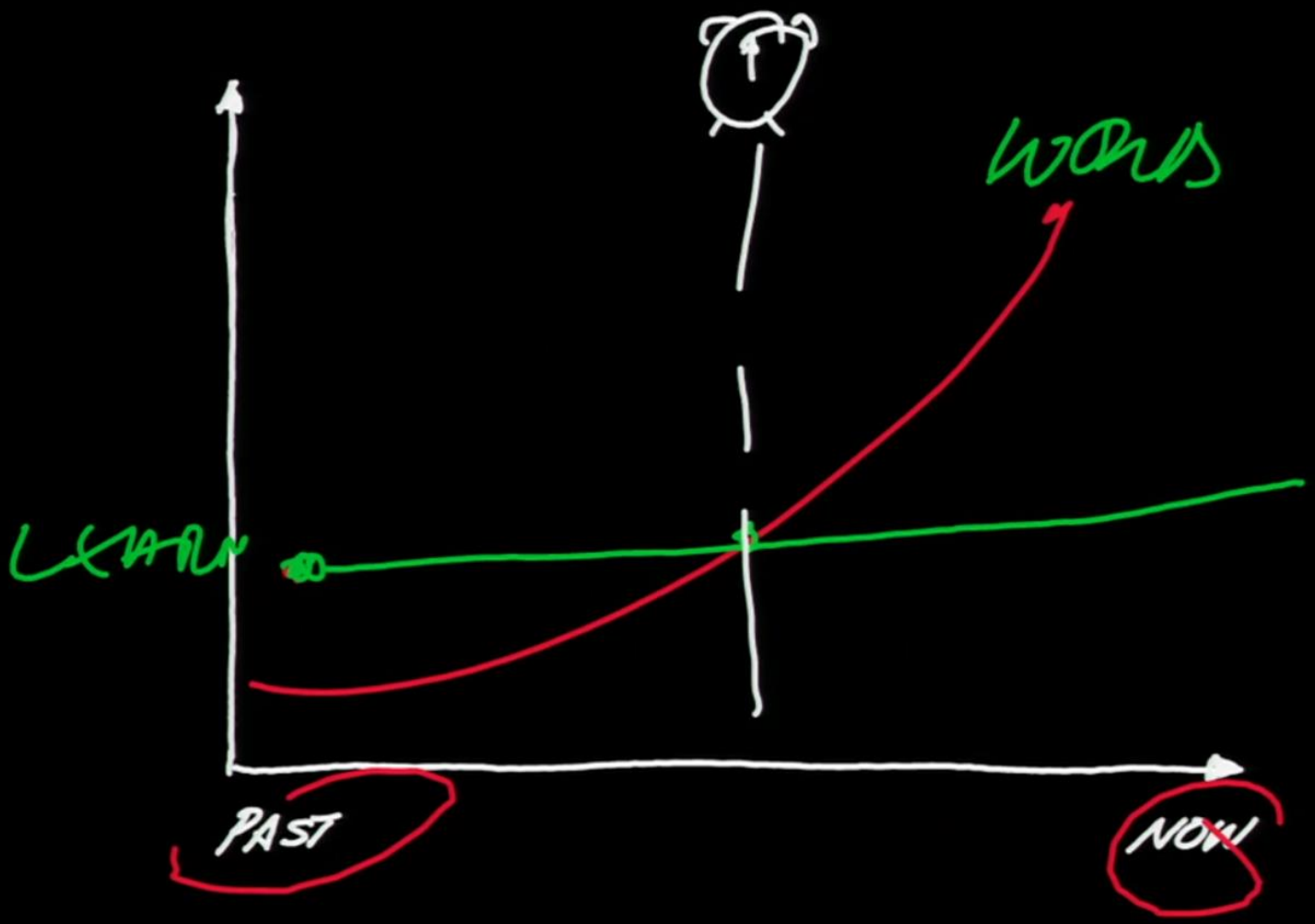
Source: www.ted.com

The World After Midnight

About fifteen years ago all the 'Rules' about how to run a business, organization, or government successfully, were changed or deleted and a completely new set of 'Rules' has been in operation ever since, which means that ***we keep acting rationally in response to a world we recognize and understand… but which no longer exists!***

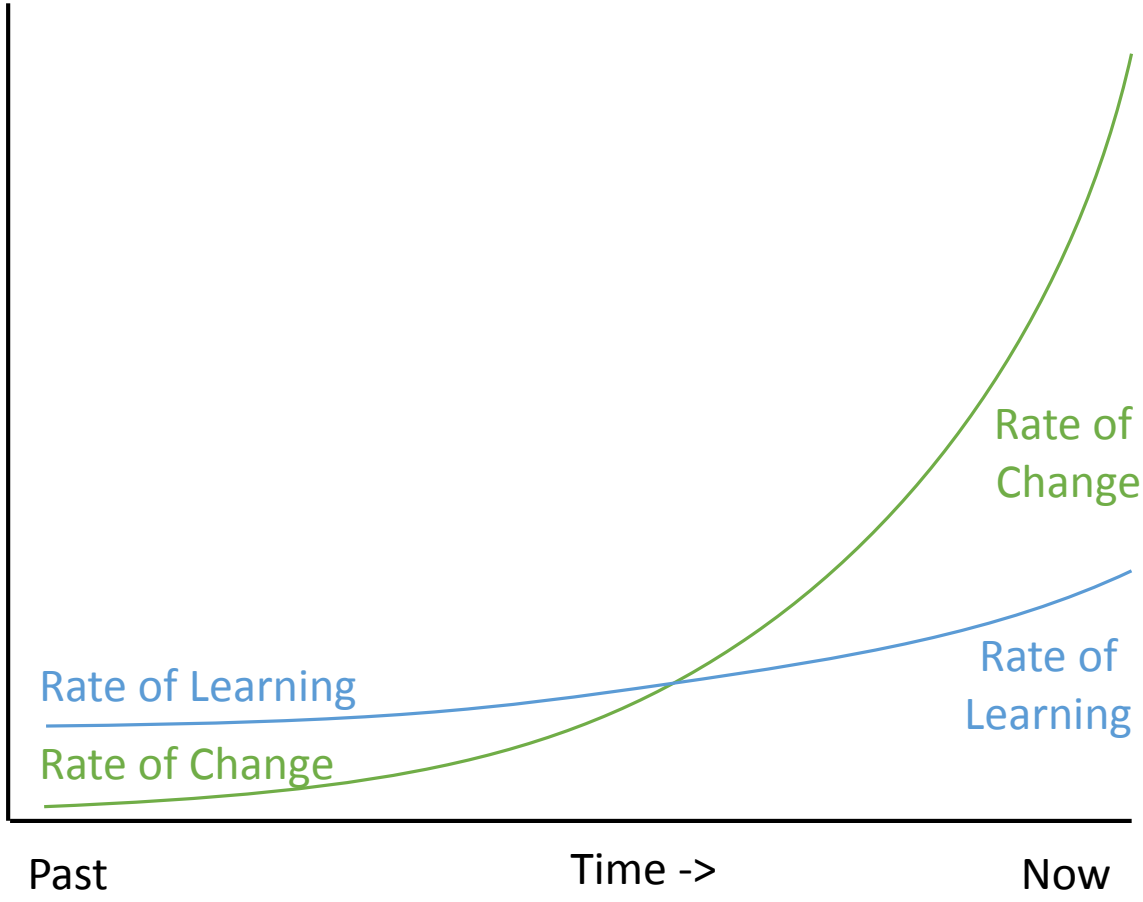
- Eddie Obeng

The World After Midnight



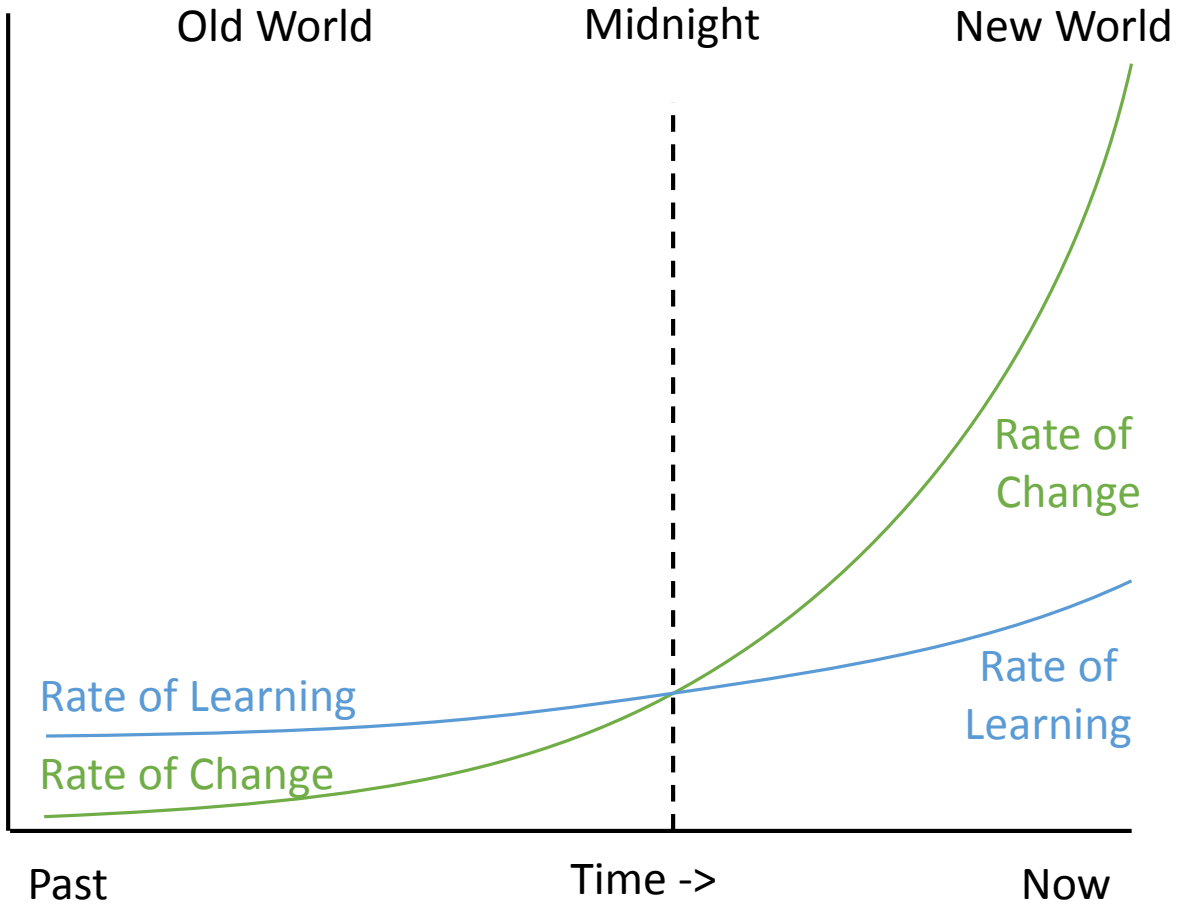
Source: http://www.ted.com/talks/eddie_obeng_smart_failure_for_a_fast_changing_world.html

The World after Midnight



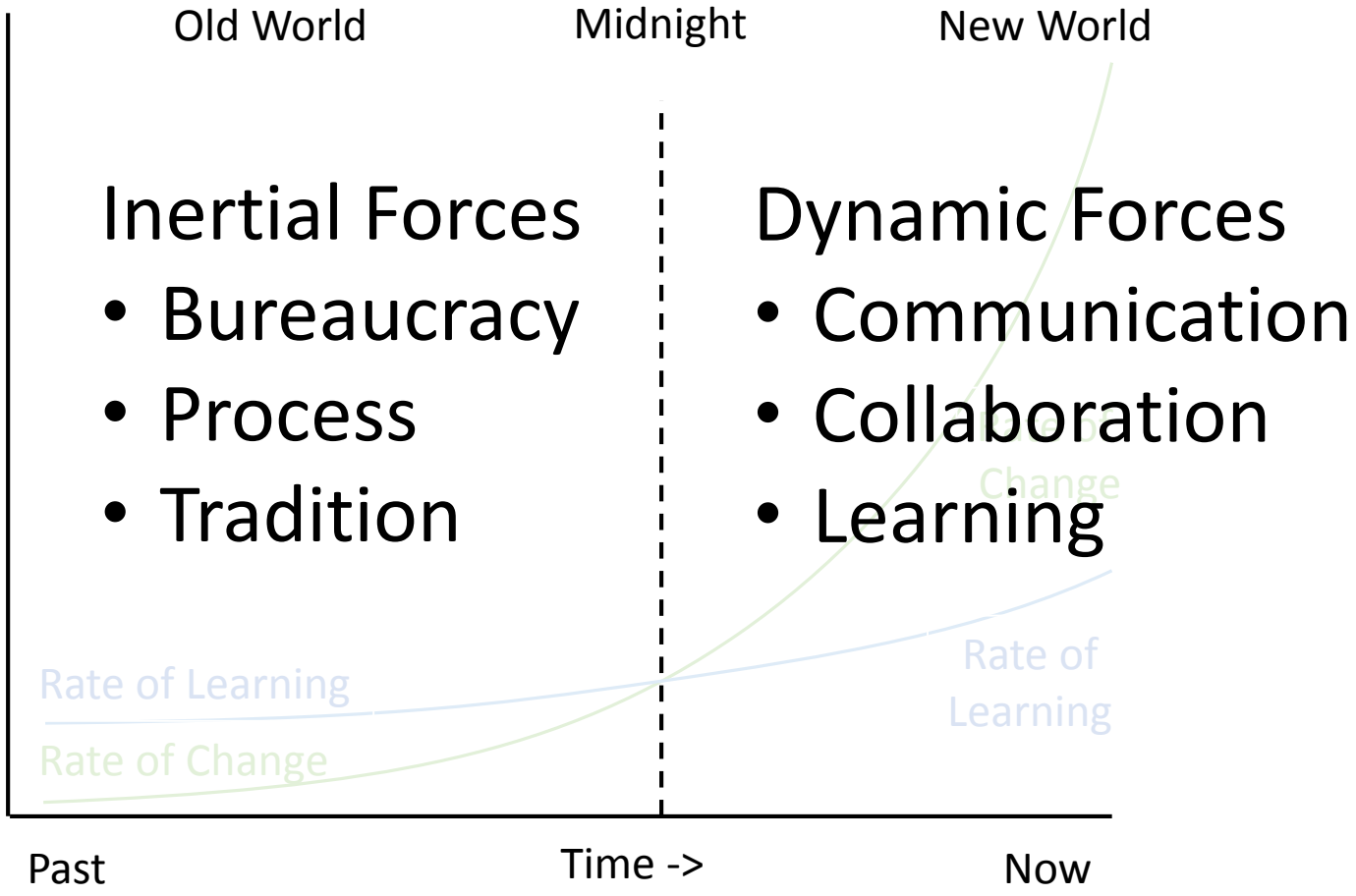
Original Source: http://pentacle.co.uk/Downloads/InnovationResources/Old%20New%20World.jpg

The World after Midnight



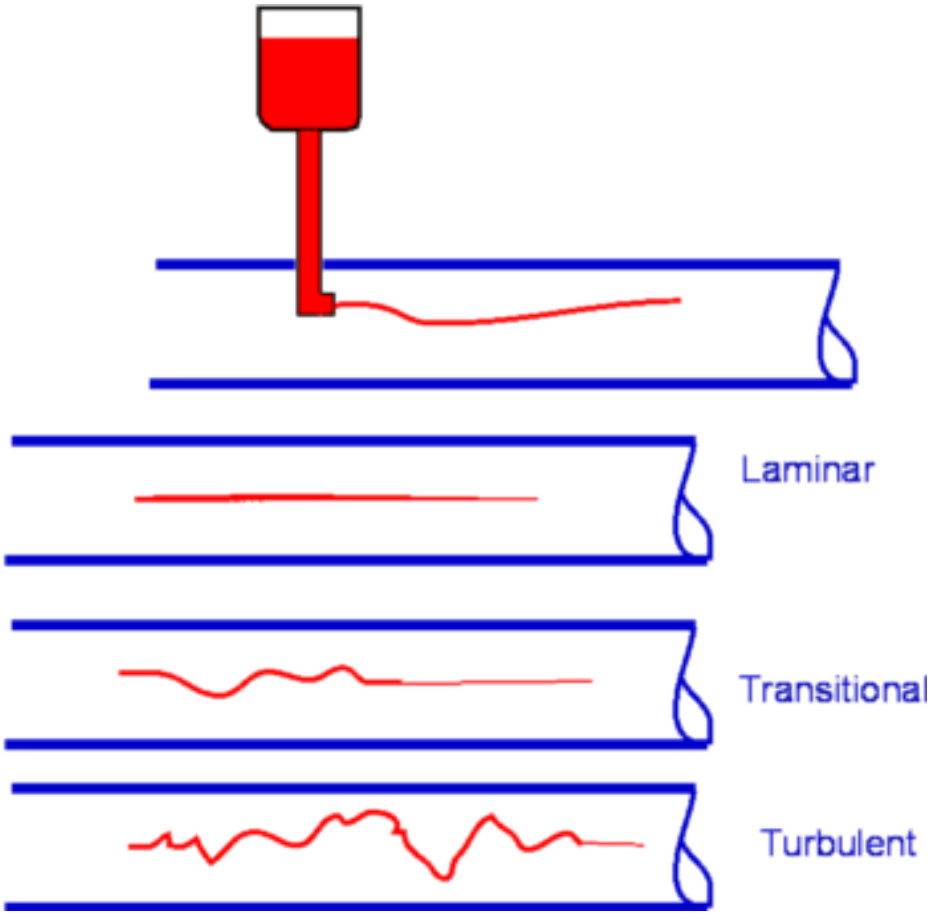
Original Source: http://pentacle.co.uk/Downloads/InnovationResources/Old%20New%20World.jpg

The World after Midnight



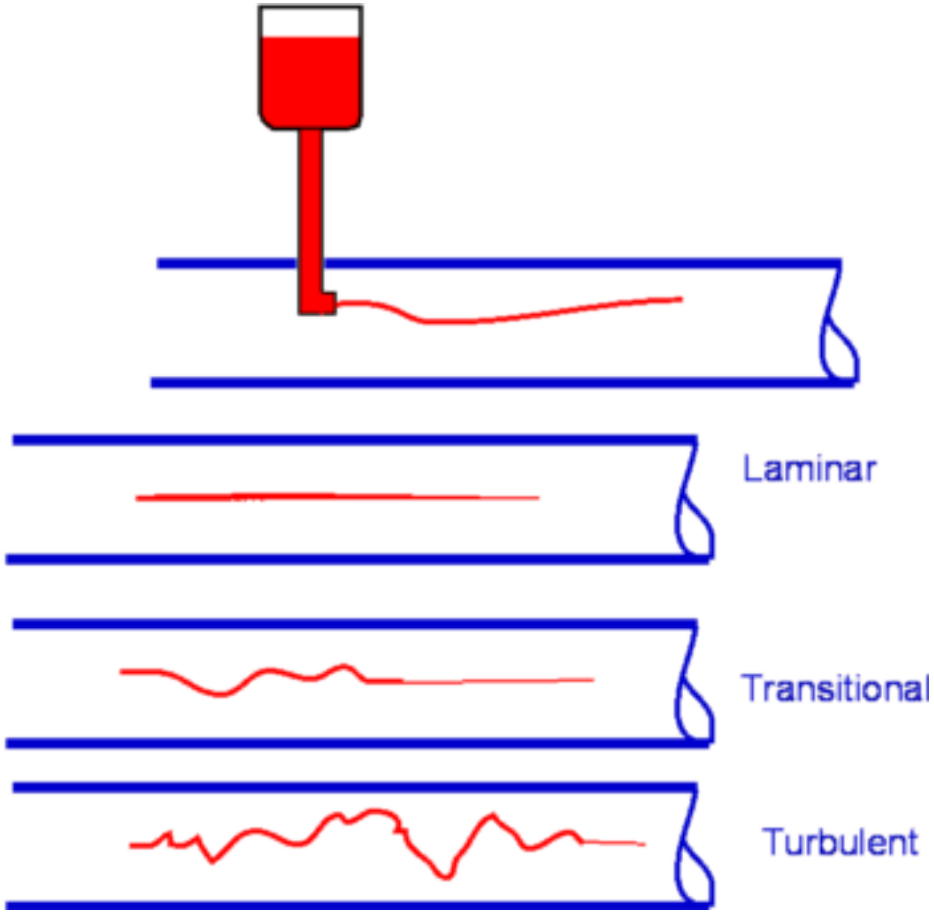
Original Source: http://pentacle.co.uk/Downloads/InnovationResources/Old%20New%20World.jpg

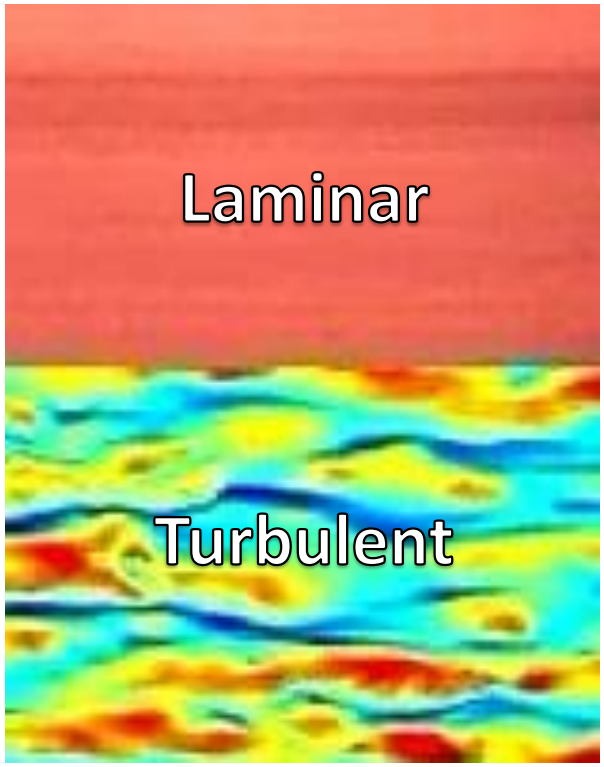
Laminar Flow vs. Turbulent Flow



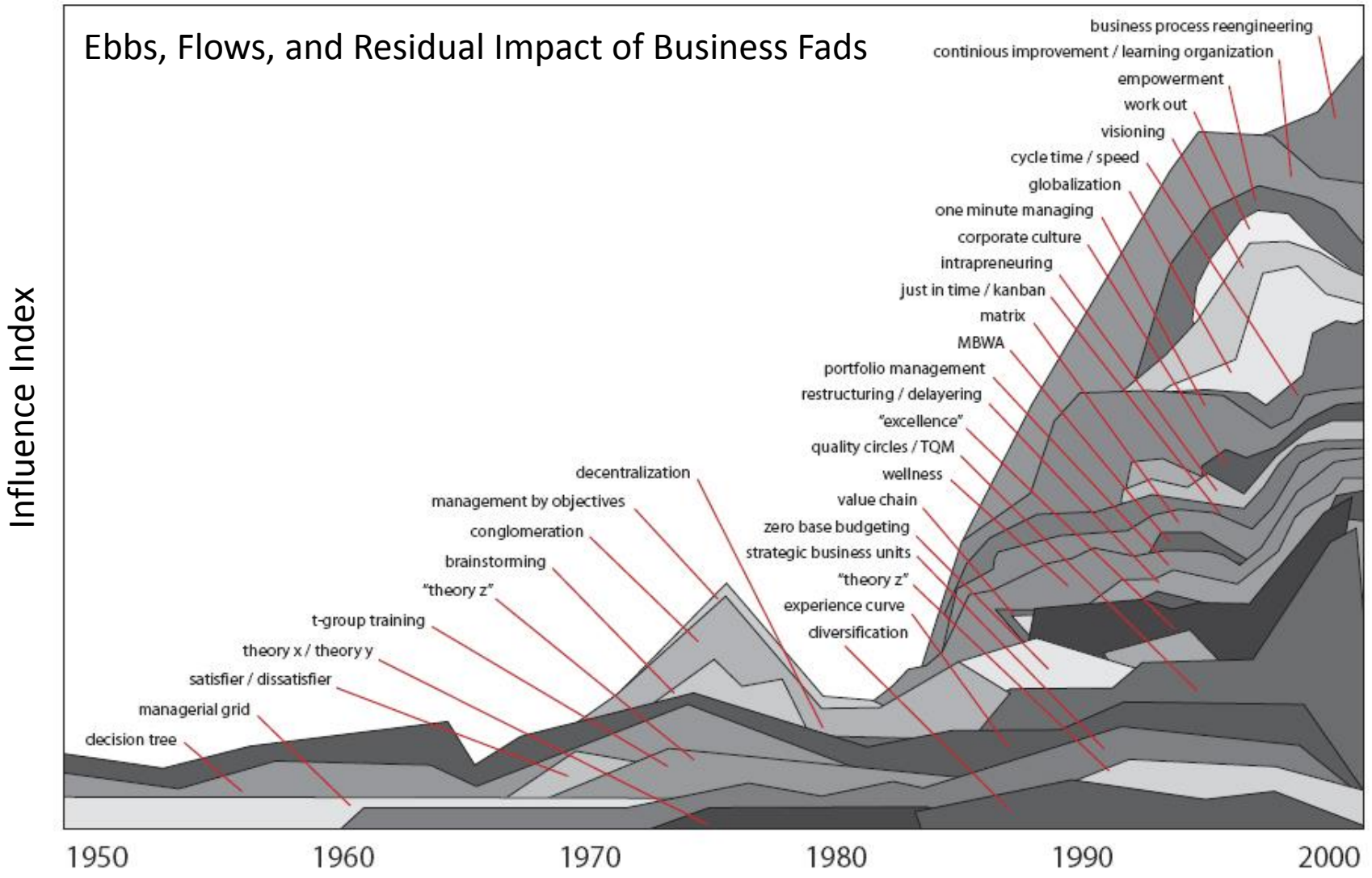
Source: http://www-mdp.eng.cam.ac.uk/web/library/enginfo/aerothermal_dvd_only/aero/fprops/pipeflow/node8.html

Laminar Flow vs. Turbulent Flow





Source: http://www-mdp.eng.cam.ac.uk/web/library/enginfo/aerothermal_dvd_only/aero/fprops/pipeflow/node8.html



Source: 'The Ebbs, Flows and Residual Impact of Business Fads 1950 – 1995' by R. Pascale

Why is this important?

Problem

- World has changed
- Markets change rapidly
- Requirements change rapidly
- High degree of uncertainty

Solution

- Adapt to new physics
- Faster time-to-market
- Better response to change
- Continuous and rapid feedback, validated learning, and smart failure

Agile is very well suited to operate in the physics of this new world!

2. Inverted Constraints

Four Levers of Software Development

- Levers
  - Scope
  - Resources
  - Schedule
  - Quality
- Goals
  - Working software
  - Max value
  - Min cost

Source: http://farm6.staticflickr.com/5300/5521479079_36815225e4_z.jpg

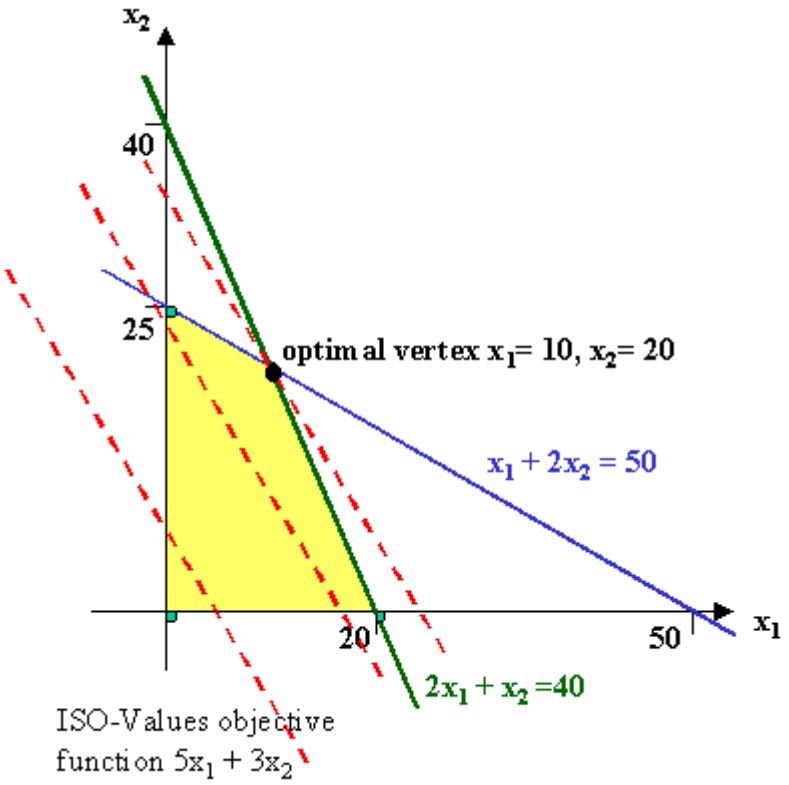
Constraints

- Restriction on a degree of freedom
- Prevent the system from achieving its goal
- Examples
  - Time
  - Money
  - Talent

Source: http://www.myspaceantics.com/image-myspace-graphic/funny-pictures/outfielder-wall-collision.jpg.html

Constrained Optimization

- Linear Programming
  - Dimensions = levers
  - Solid lines = constraints
  - Dash lines = value
  - Shaded = feasible
  - Solution = optimum

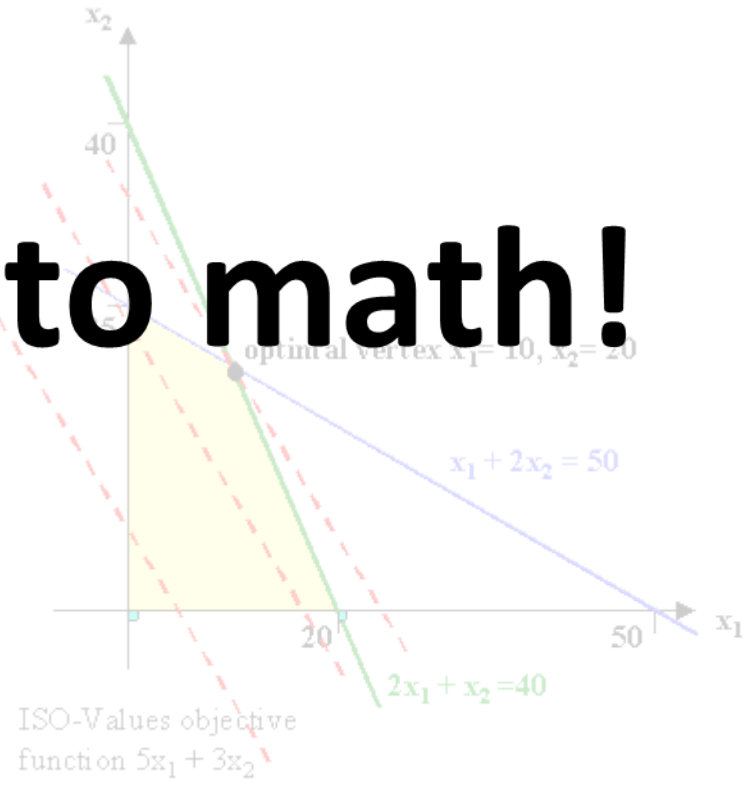


Source: http://home.ubalt.edu/ntsbarsh/business-stat/opre/partVIII.htm

Constrained Optimization

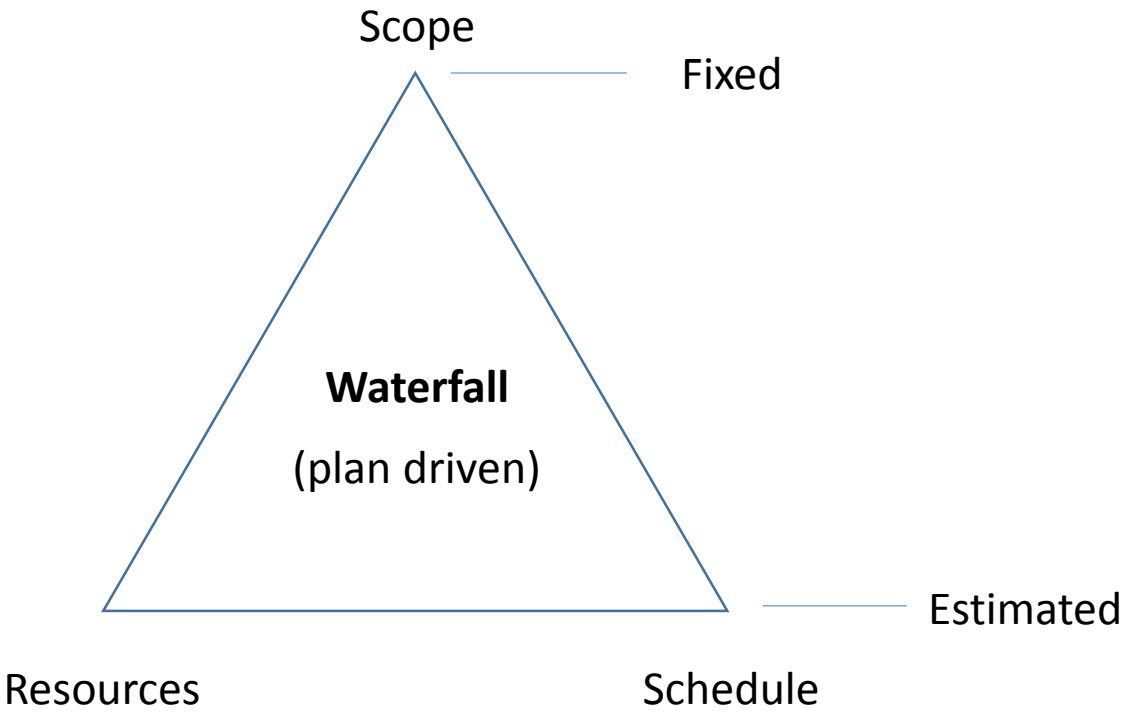
- Linear Programming
  - Dimensions = levers
  - Solid lines = constraints
  - Dashed lines = value
  - Shaded = feasible
  - Solution = optimum

Too much to math!

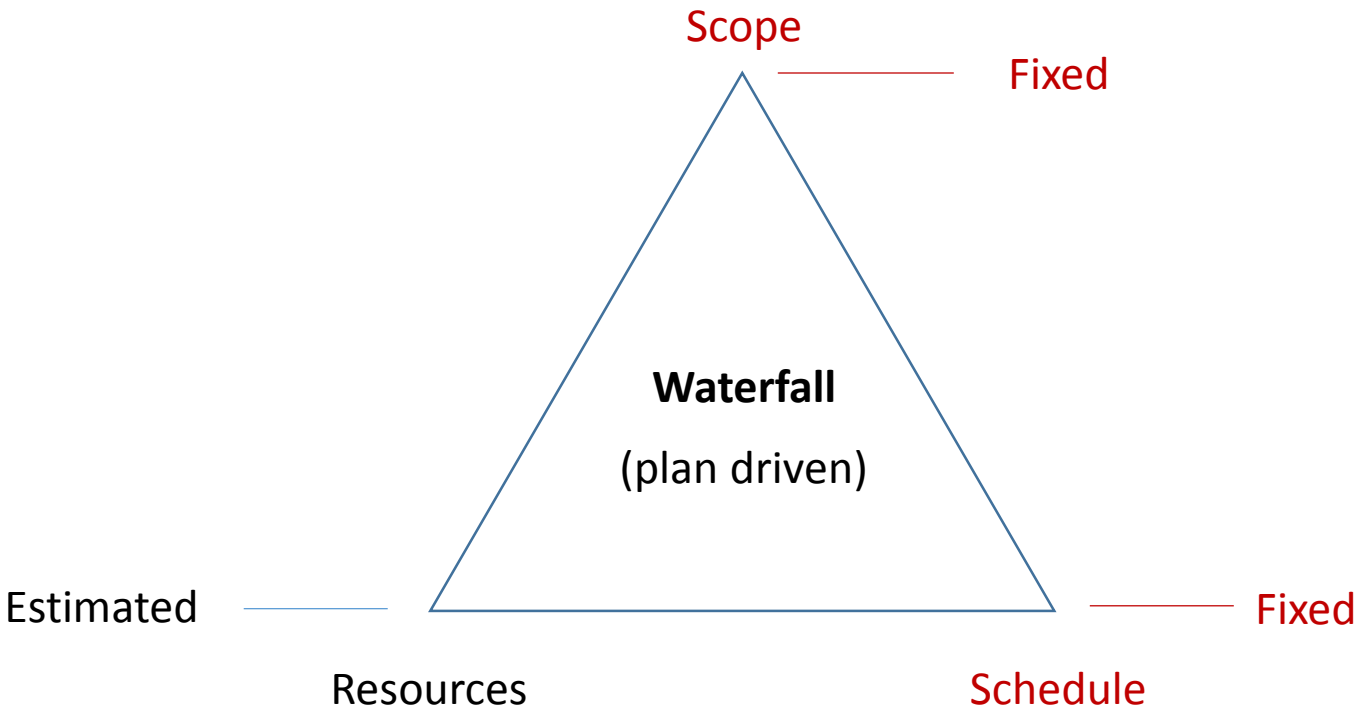


Source: http://home.ubalt.edu/ntsbarsh/business-stat/opre/partVIII.htm

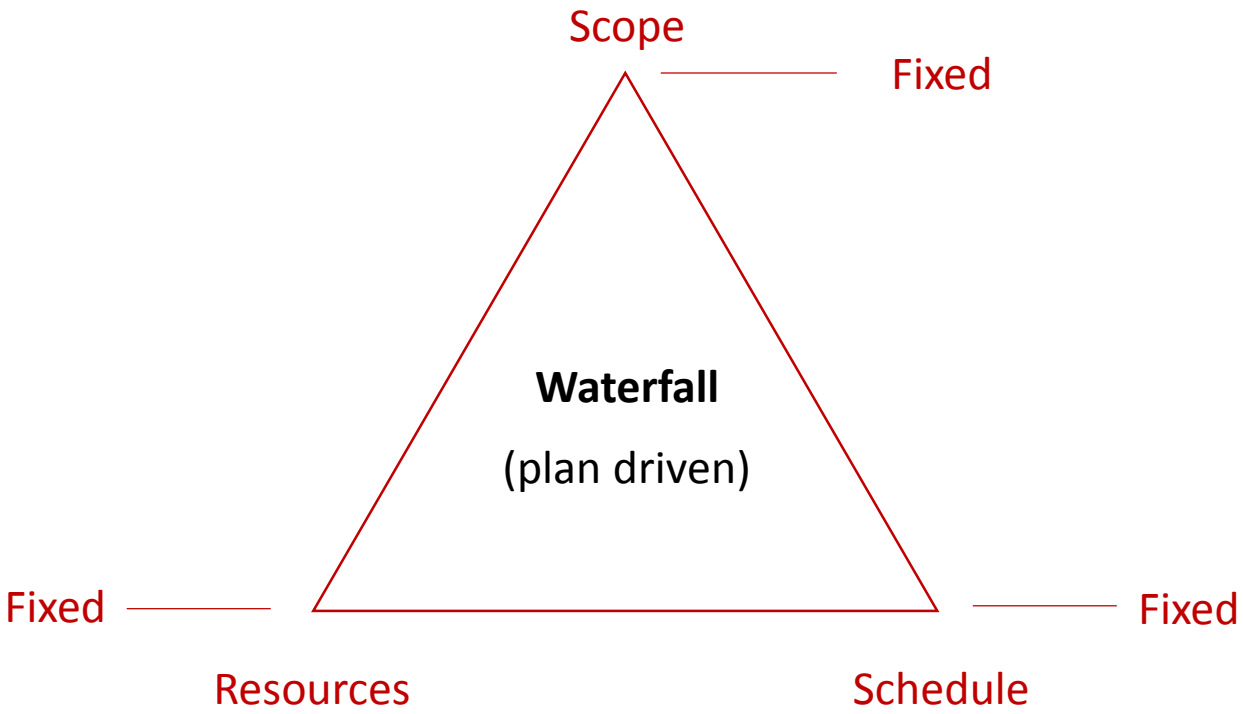
Waterfall Constraints



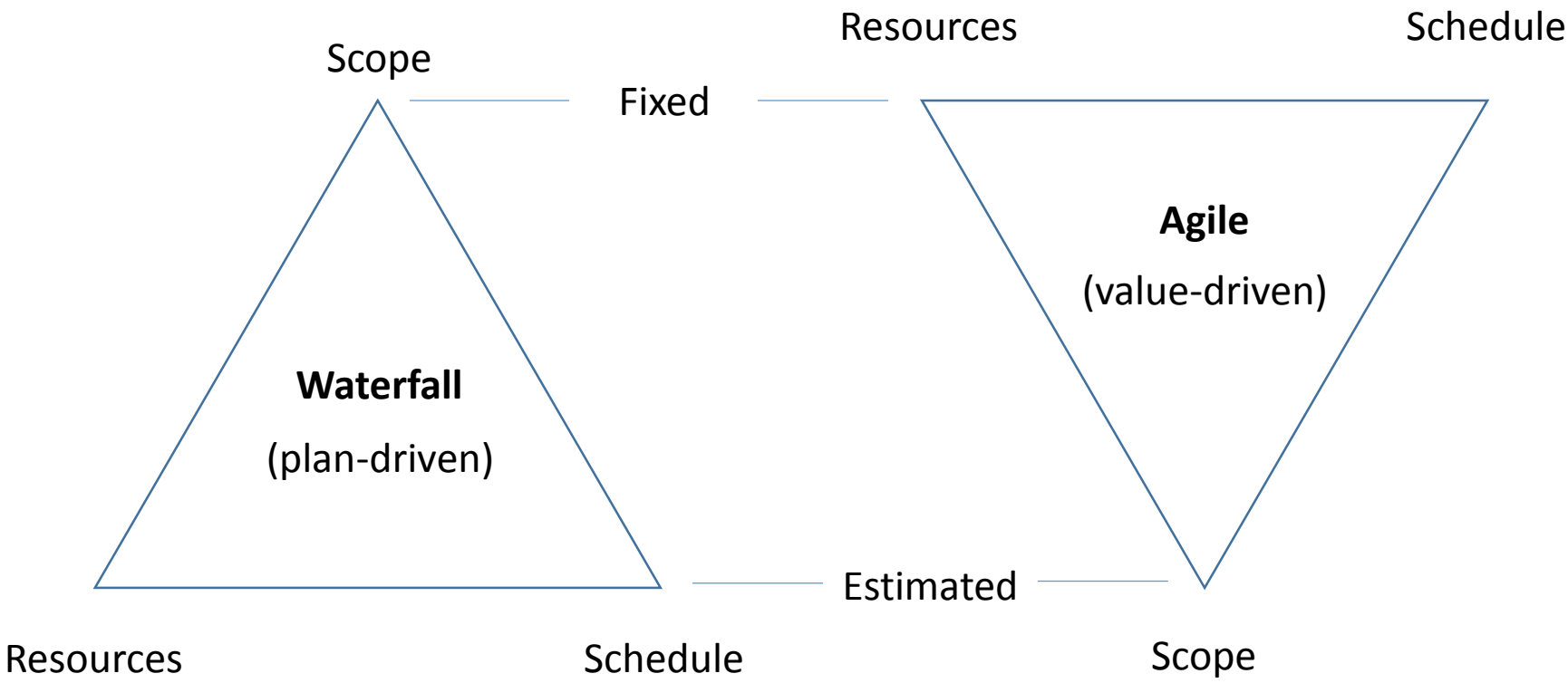
Waterfall Constraints



Waterfall Constraints

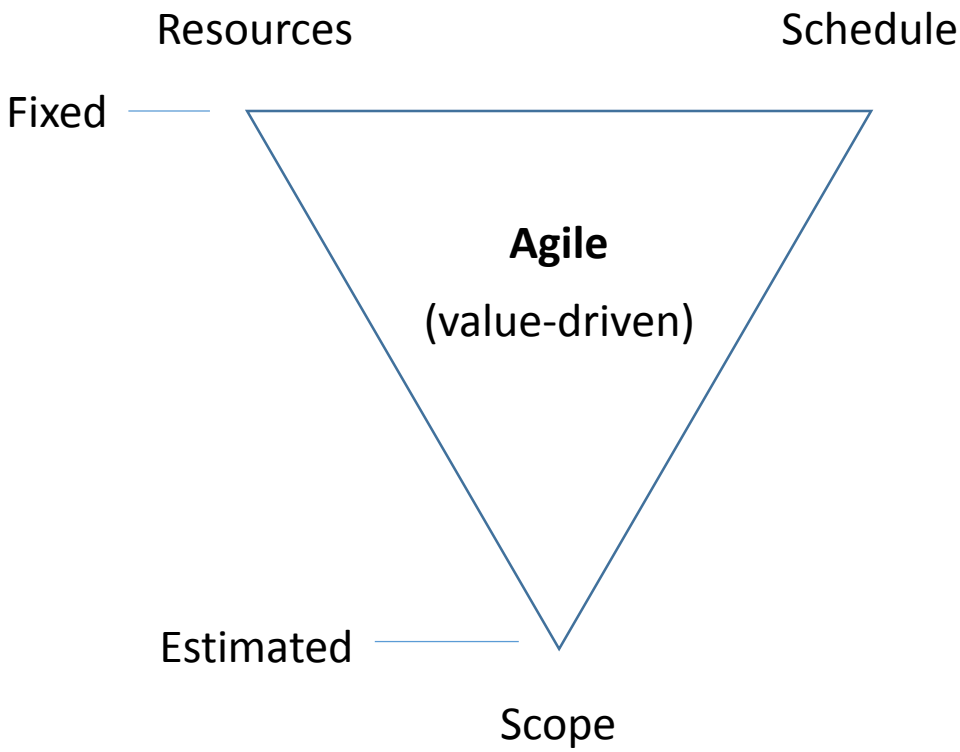


Agile Constraints



Agile Constraints

- Fixed team size
- Fixed releases
- Estimated features
- Team controls quality



Why is this important?

Problem

- Mythical man-month
- Slipping release dates
- Scope creep
- Technical debt

Solution

- Limit team size
- Fix schedule
- Estimate scope
- Protect quality

Agile is more flexible

3. Prioritizing Value

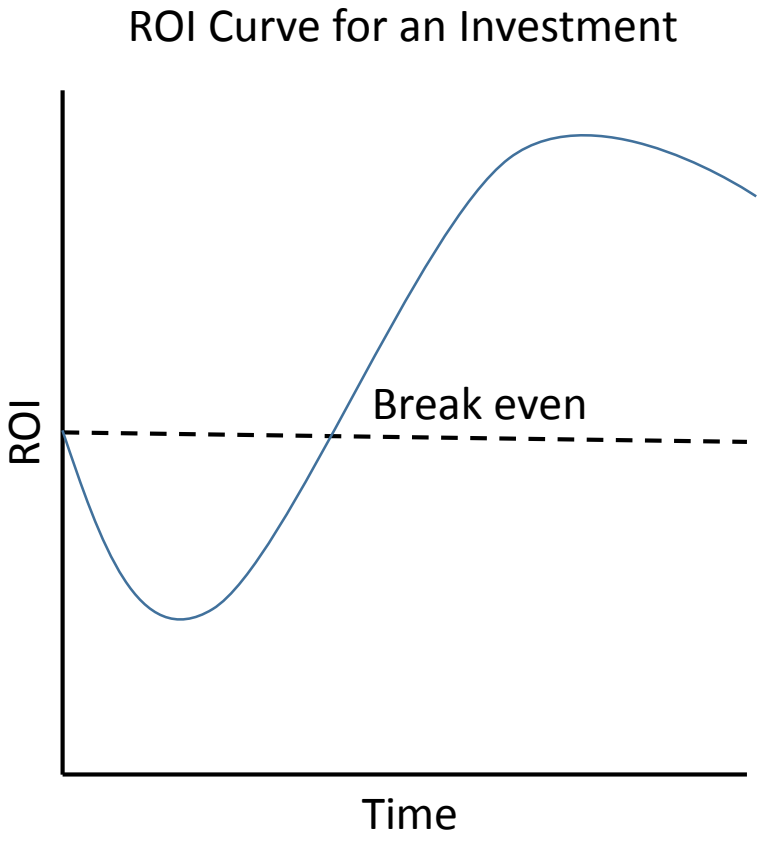
Quick Lesson in Economics

1. Return on Investment
2. Pareto Principle
3. Opportunity Cost

Source: http://myhomeworkhelp.com/economics-homework-help/

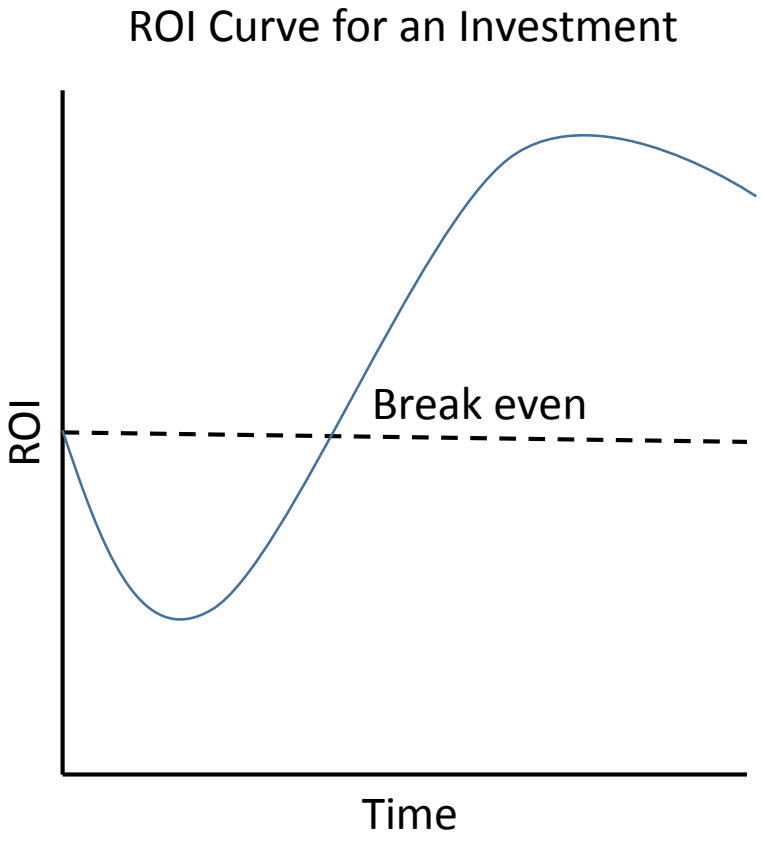
Return on Investment

- $ROI = \frac{Value - Cost}{Cost}$
- High ROI => lots of value
- Low ROI => some value
- Neg. ROI => lost value



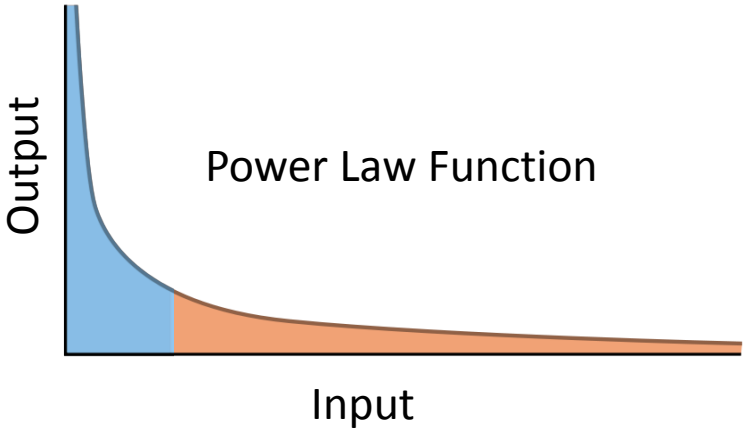
Return on Investment

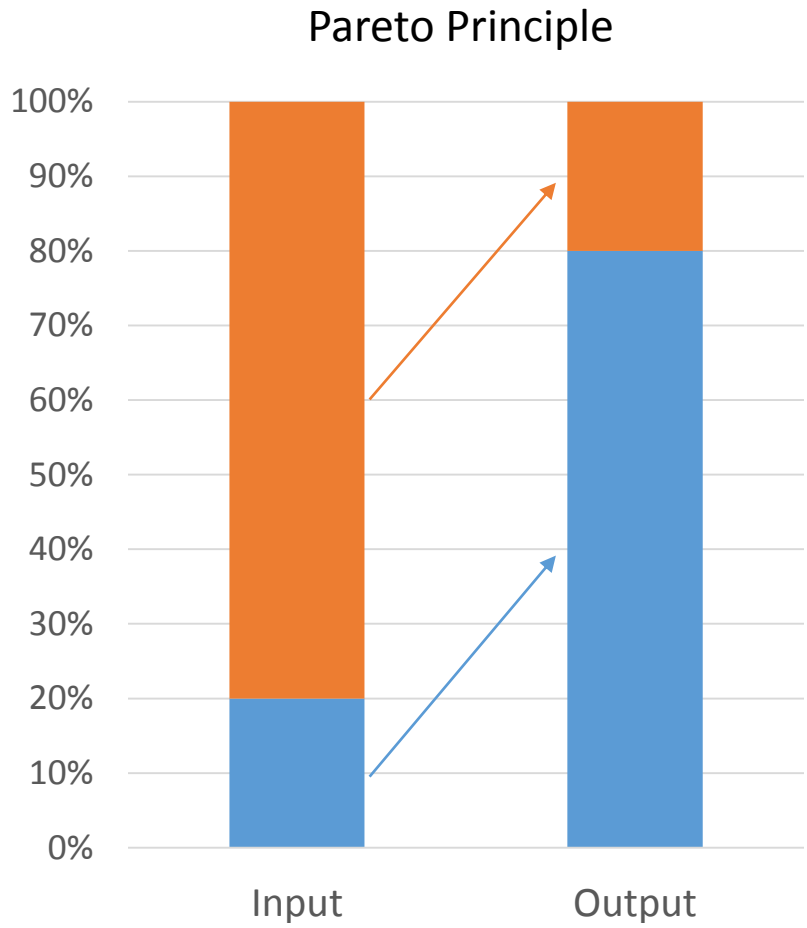
- Each feature has ROI
  - Cost to develop
  - Value to business
- Project ROI is sum of all feature ROIs
- Goal is to maximize ROI of the project



Pareto Principle

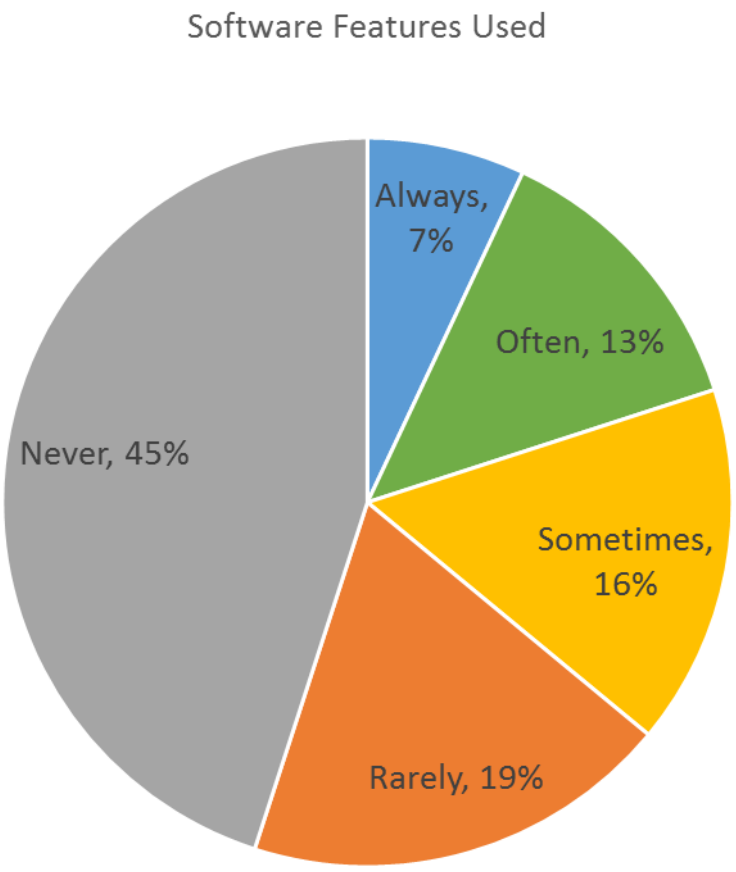
- 80/20 rule
- 80% of effects
- 20% of causes
- Power law function
- Diminishing marginal returns





Pareto Principle of Software Feature Usage

- Features:
  - 20% of features
  - 80% of value
- Traditional software is:
  - 20% high-value features
  - 80% low-value features



Source: Standish Group

Opportunity Cost

Source: http://www.ethicurean.com/2009/03/03/free-lunch-program-in-new-england/

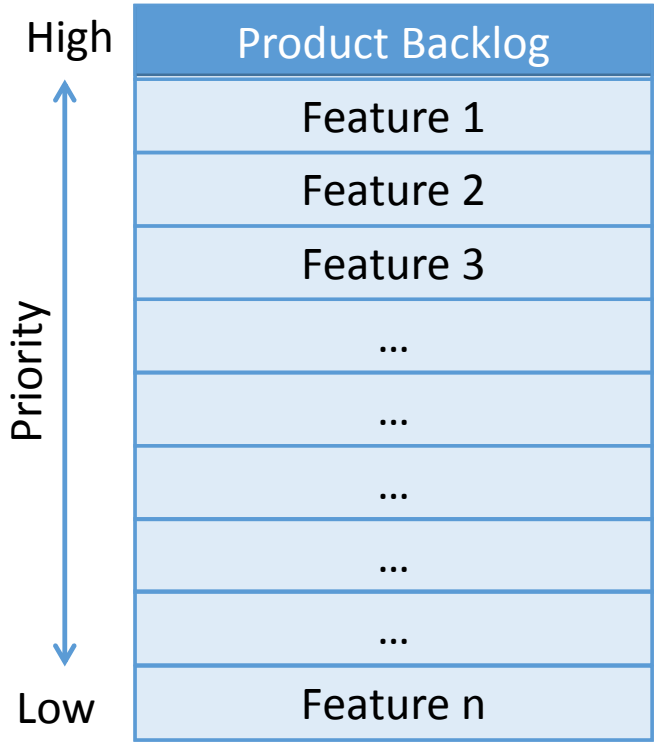
Opportunity Cost

- Cost of foregone alternative options
- True cost = explicit cost + implicit cost
- Must be included in cost-benefit analysis

Source: http://www.stus.com/

Prioritizing Features by Business Value

- Product backlog
- List of features
- Ordered by business value
- Highest priority on top
- Create and deliver features in order



Why is this important?

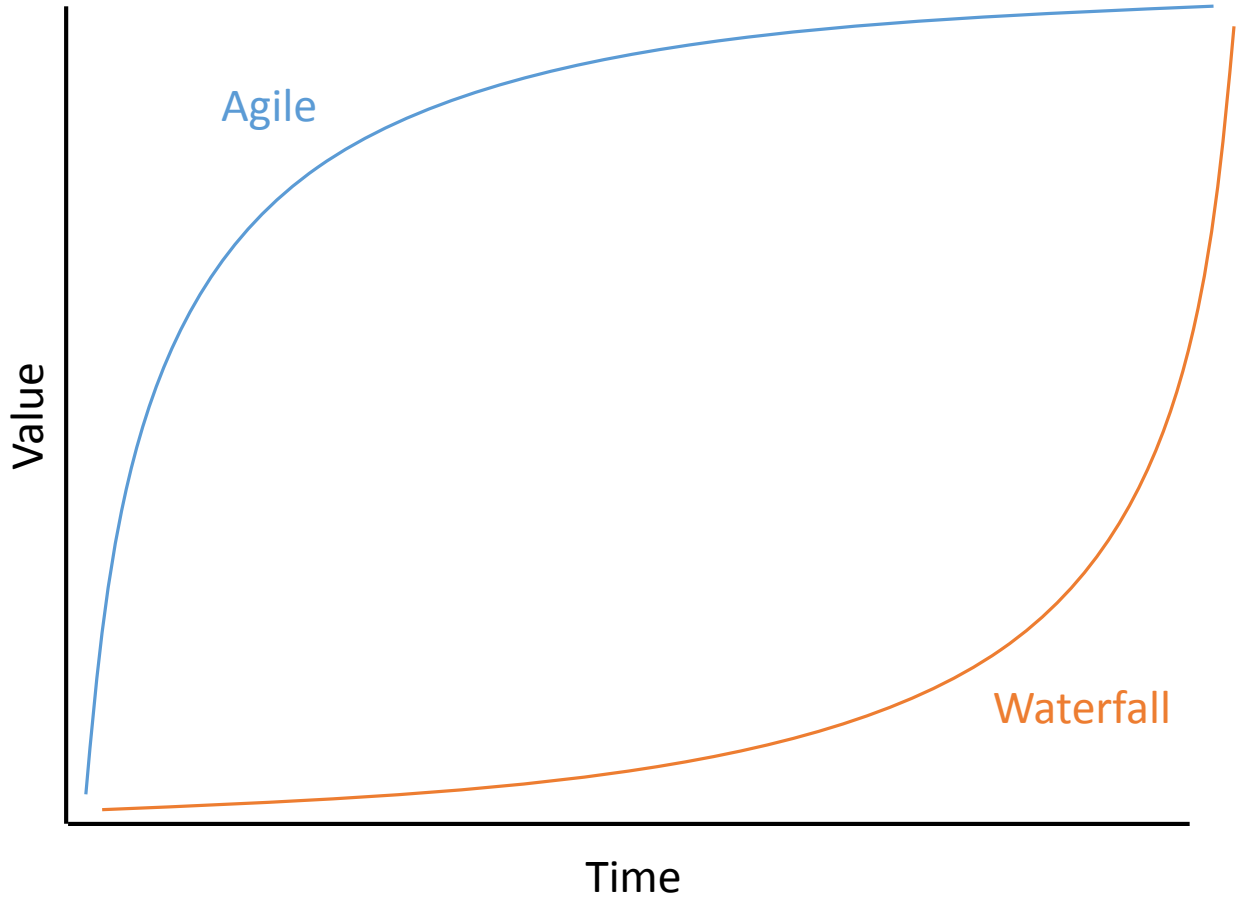
Problem

- Need to maximize ROI
- Need to reduce low-value features (80/20)
- Need to consider opportunity cost

Solution

- Prioritize features according to ROI
- Deliver highest-value features first
- Prioritize features relative to one another

Agile Produces More Value



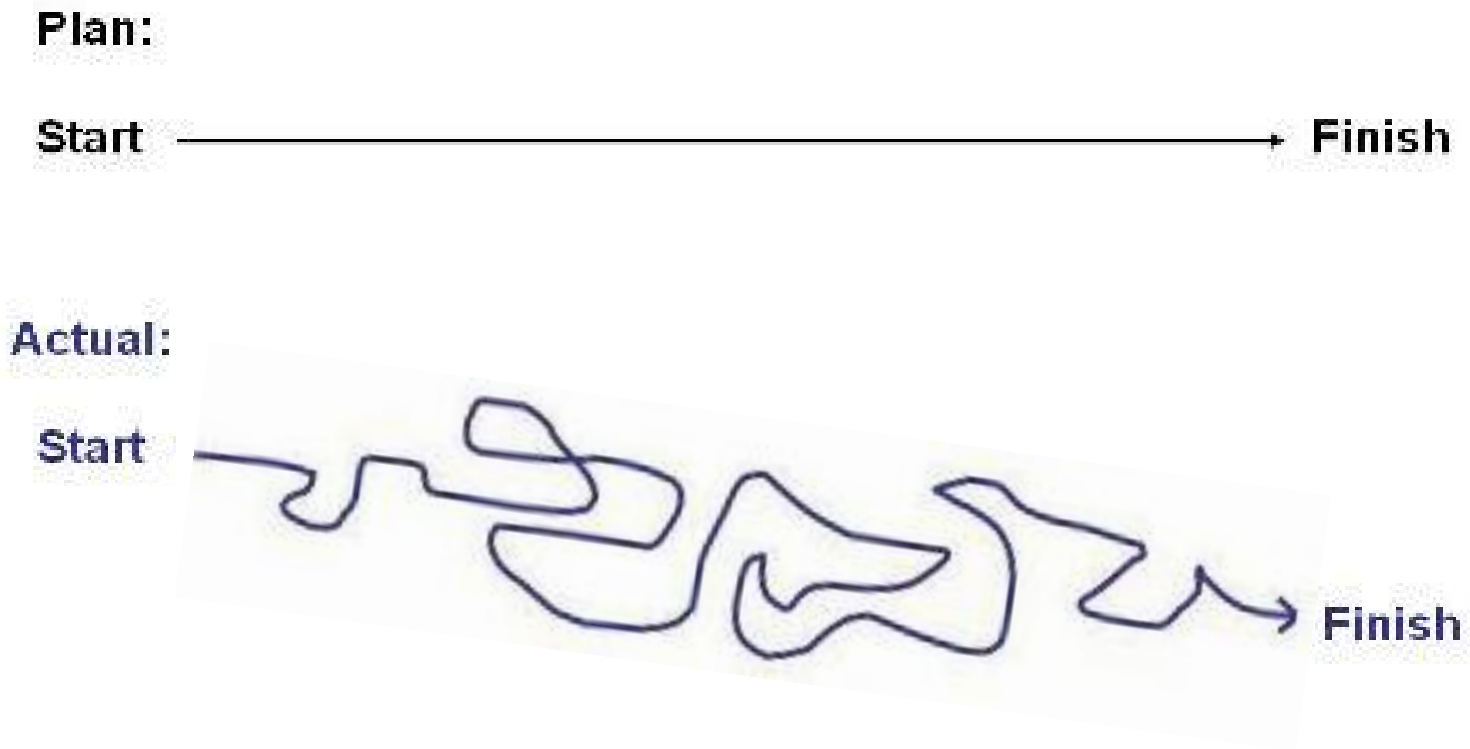
Original Source: http://www.versionone.com/Agile101/Agile-Software-Development-Benefits/

4. Embracing Change

Waterfall Assumes that Things Will Go According to a Plan

Source: Doug DeCarlo – eXtreme Project Management

Waterfall assumes that everything will go according to plan



Source: Doug DeCarlo – eXtreme Project Management

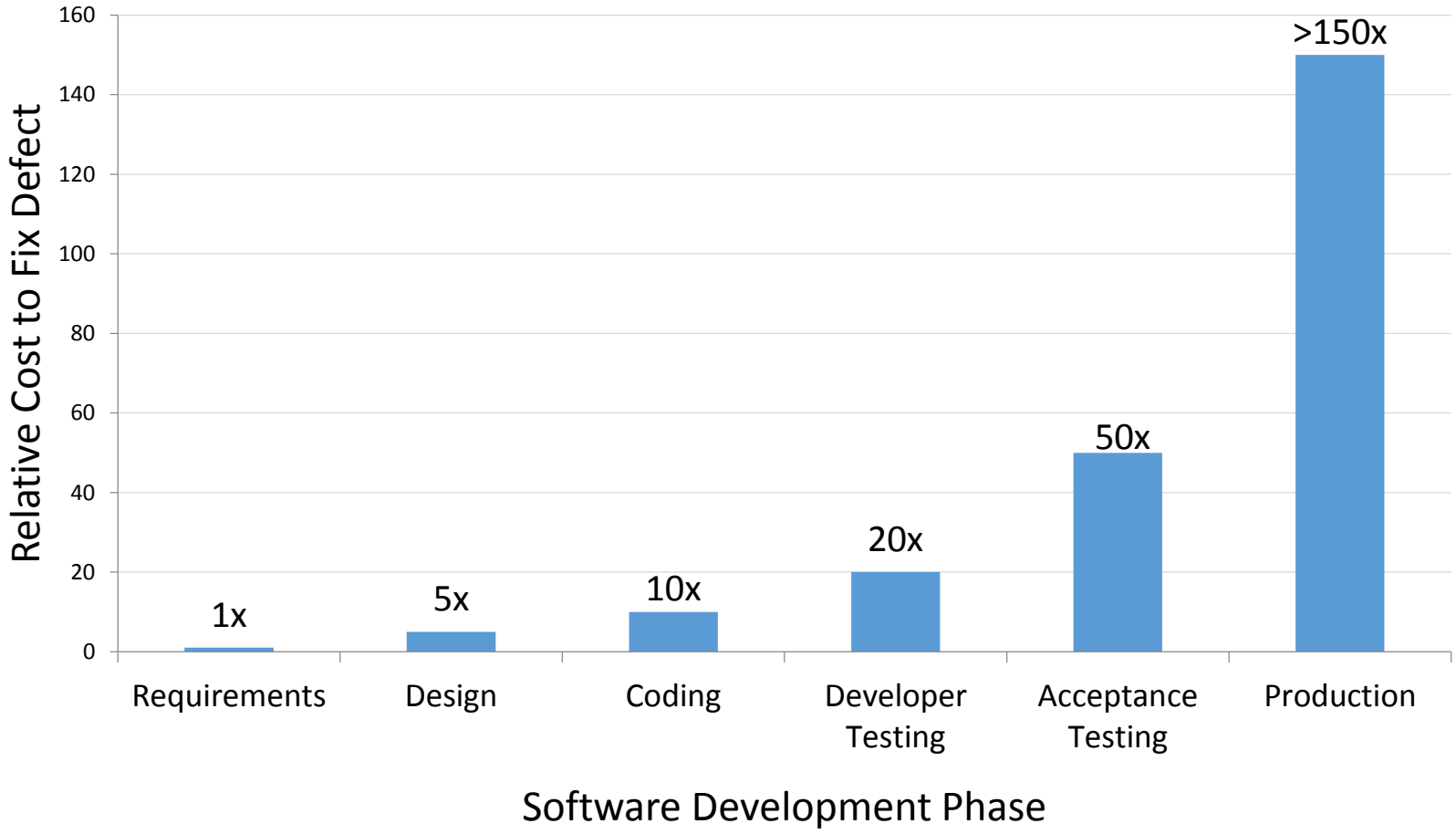
Waterfall

- Waterfall assumes:
  - Users actually know what they want
  - Markets will not change during development
  - There is nothing new or unknown
  - Technology is stable and mature
  - All of the pieces will fit together in the end

Waterfall

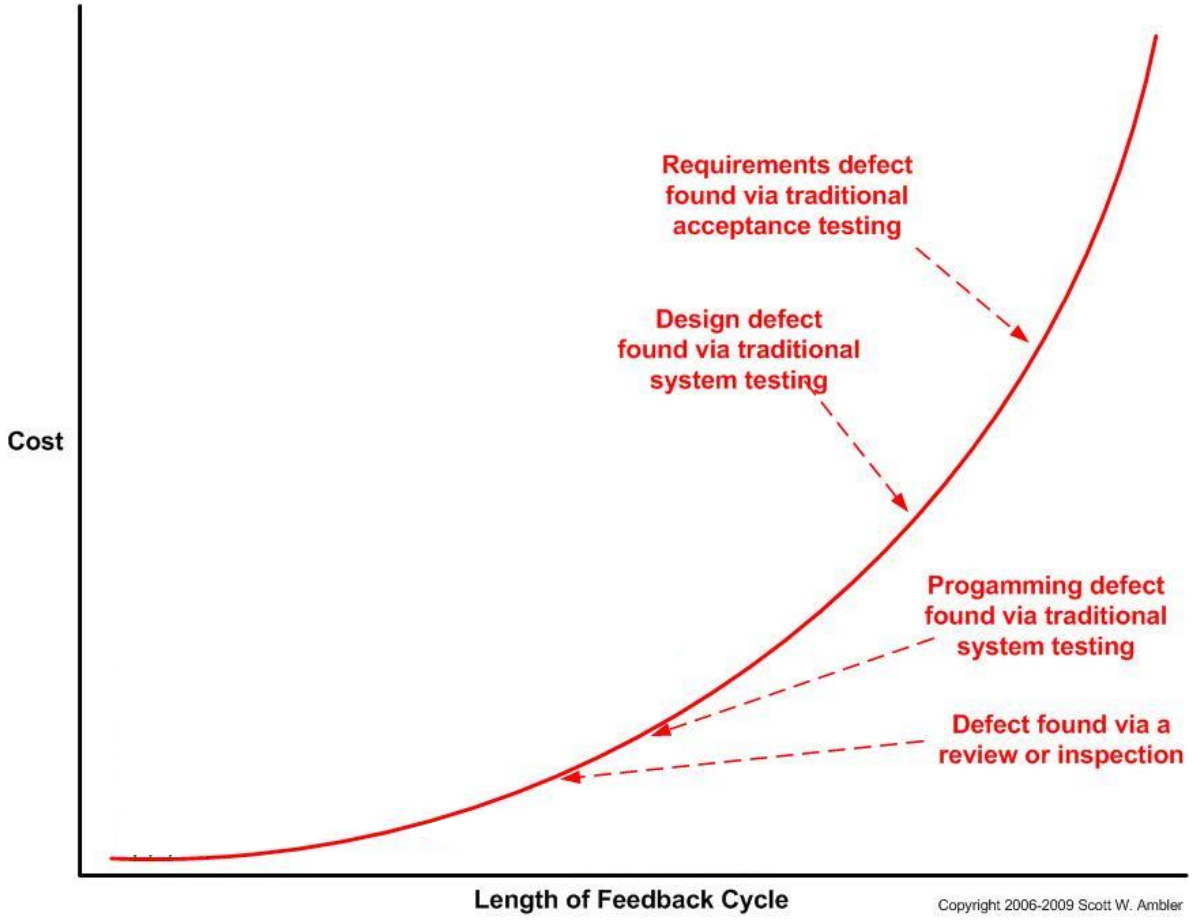
- Reality for most projects is:
  - Requirements are not stable
  - Requirements are just assumptions

Cost of Fixing Defects in Waterfall



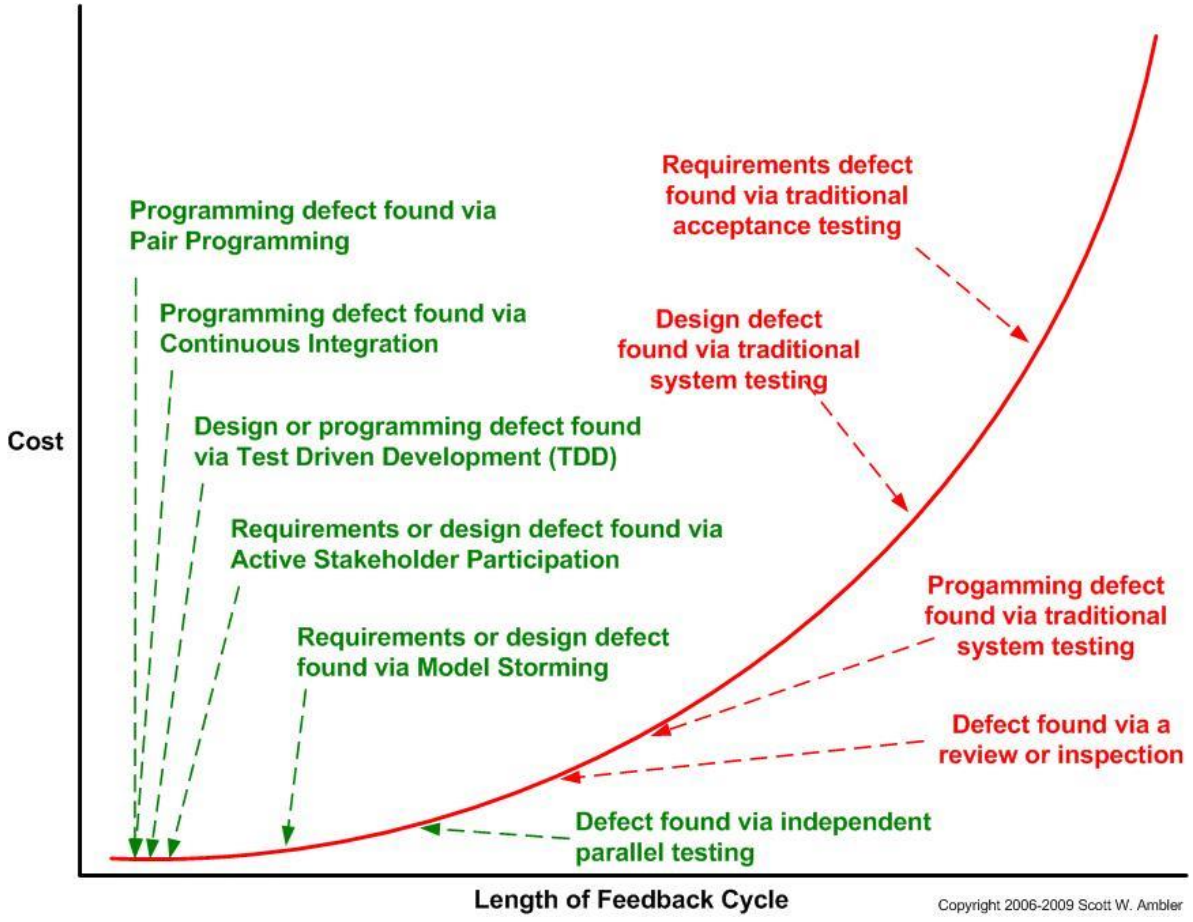
Original Source: Barry Boehm, “Equity Keynote Address” March 19, 2007

Finding Defects in Waterfall



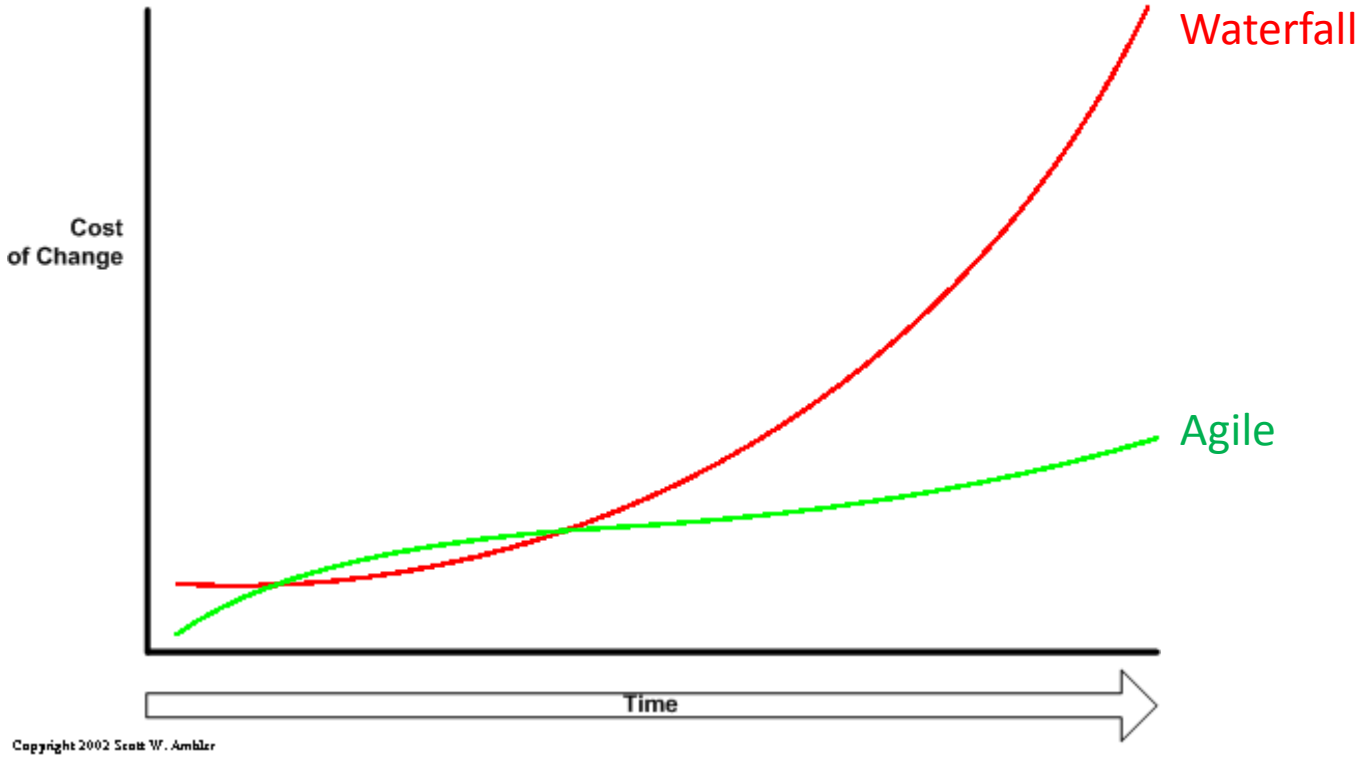
http://www.agilemodeling.com/essays/costofchange.htm

Finding Defects in Agile



http://www.agilemodeling.com/essays/costofchange.htm

Cost of Change in Agile



Source: http://www.agilemodeling.com/essays/costOfChange.htm

Why is this important?

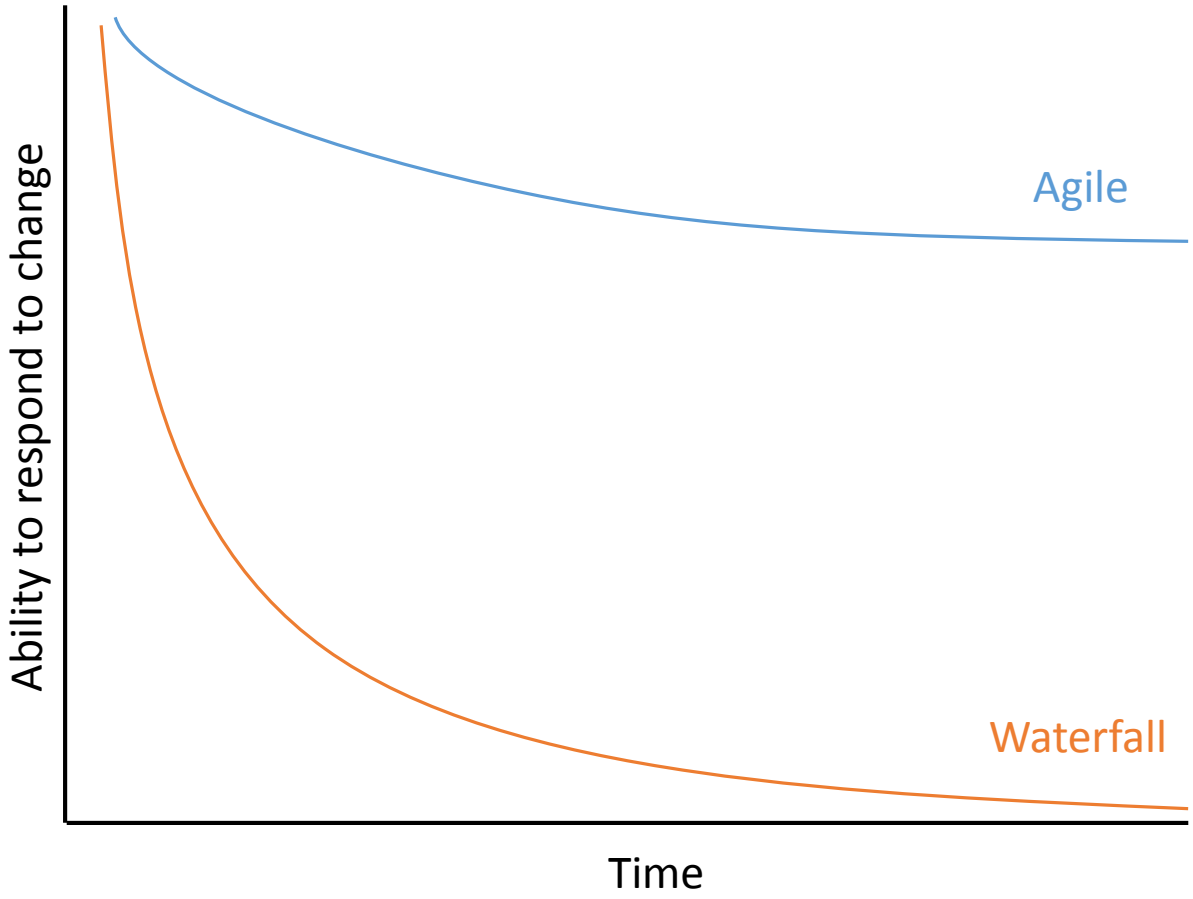
Problem

- Requirements change
- Defect in production are costly
- Late changes in requirements are costly

Solution

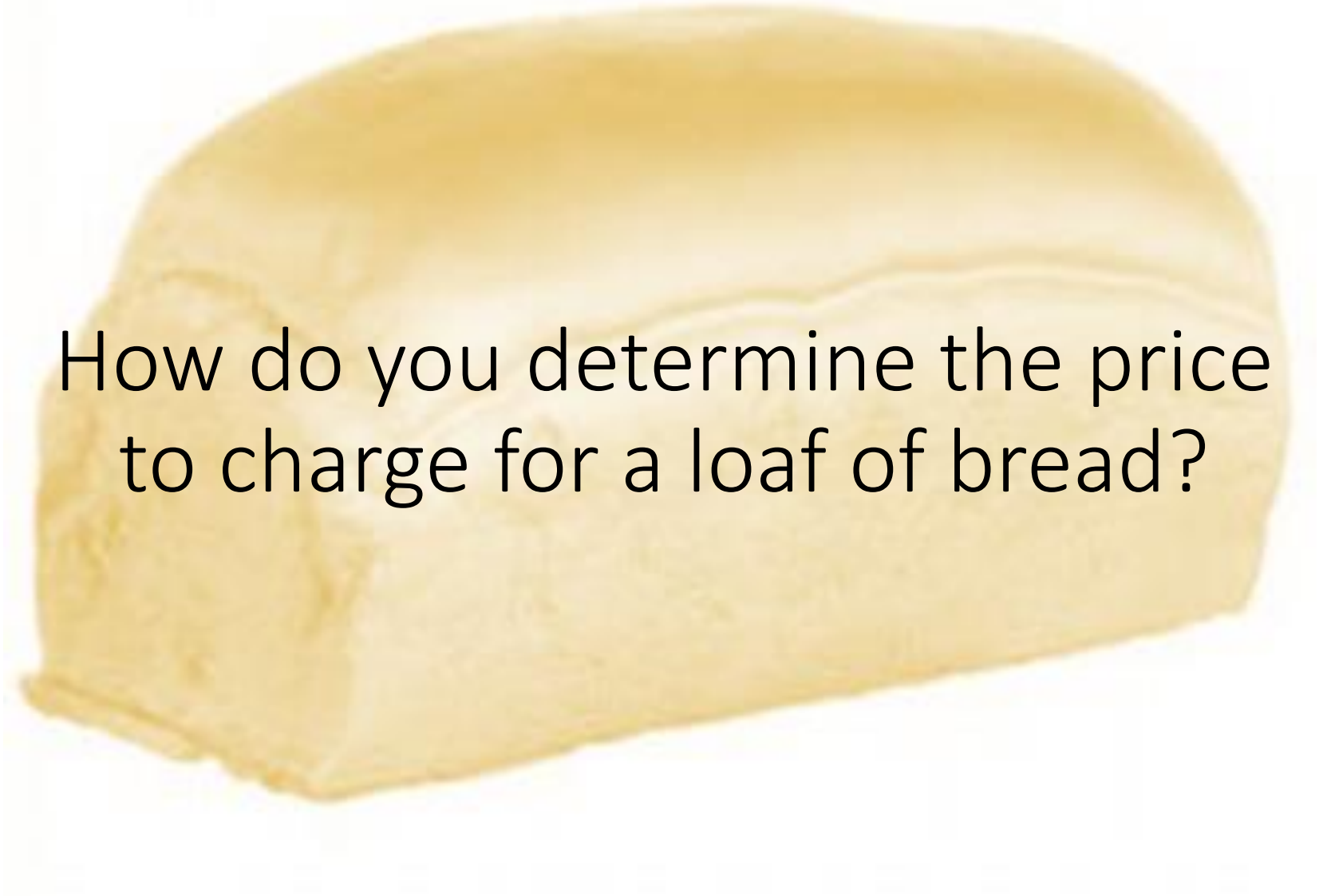
- Embrace change
- Find and fix defects early
- Build flexibility into your code and process

Agile is More Adaptable



Original Source: http://www.versionone.com/Agile101/Agile-Software-Development-Benefits/

5. Self-Organization



How do you determine the price to charge for a loaf of bread?

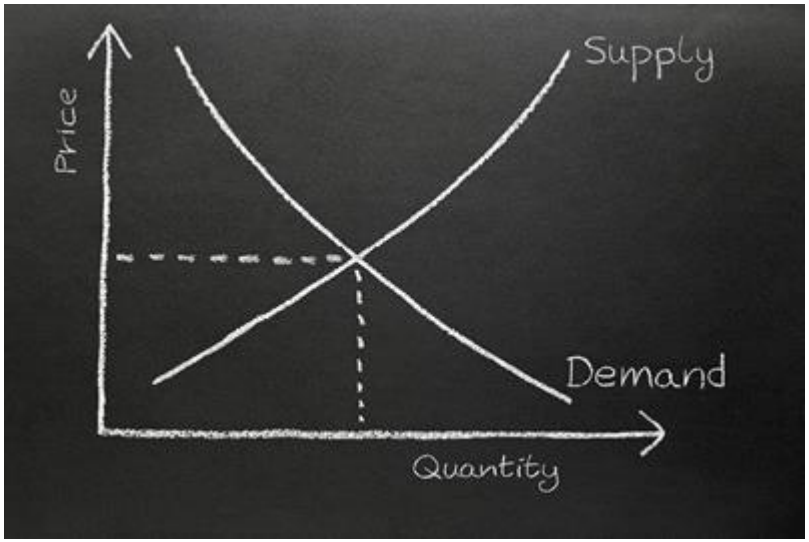
Market Economy

- Market makes decisions
  - Individuals
  - Interactions
- Produces & Consumers
  - Supply
  - Demand
- Millions of decisions

Source: Britannica

Market Economy

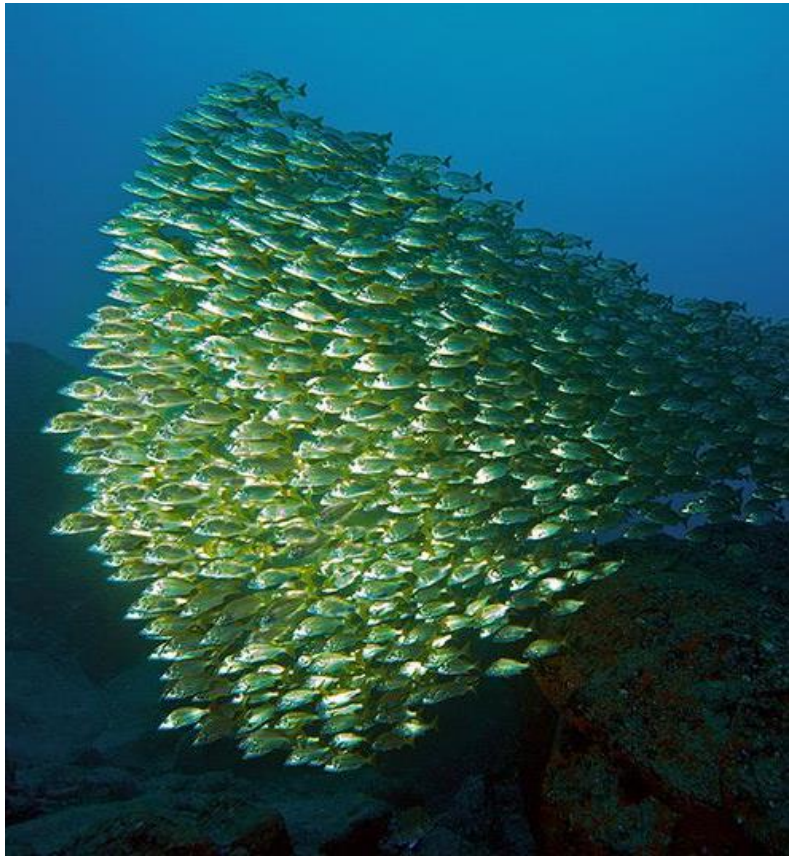
- Goal: Maximize Social Welfare
- Competitive Market Equilibrium
- Extremely efficient
- “Chaotic success”



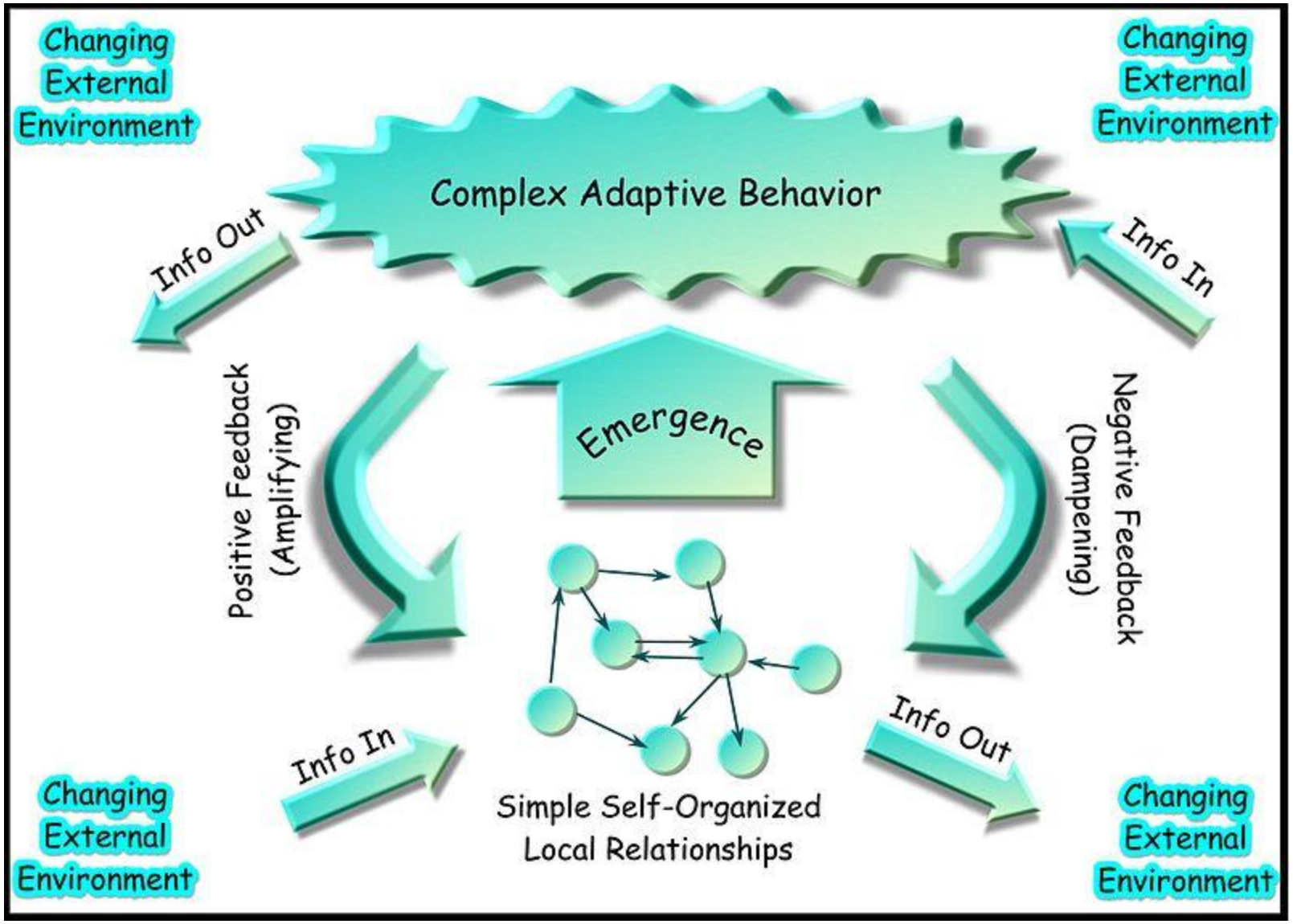
Source: https://content.dodea.edu/VS/HS/DVHS_Courses/Economics/syllabus.html

Complex Adaptive Systems

- System
  - collection of interconnected things
- Complex
  - dynamic network of interactions
- Adaptive
  - changes in response to environment
  - to increase survivability



Source: http://integral-options.blogspot.com/2013/03/peter-fryer-brief-description-of.html



Source: Wikipedia

Inversion of Control

- Top-down
- Command and Control
- Bureaucracy

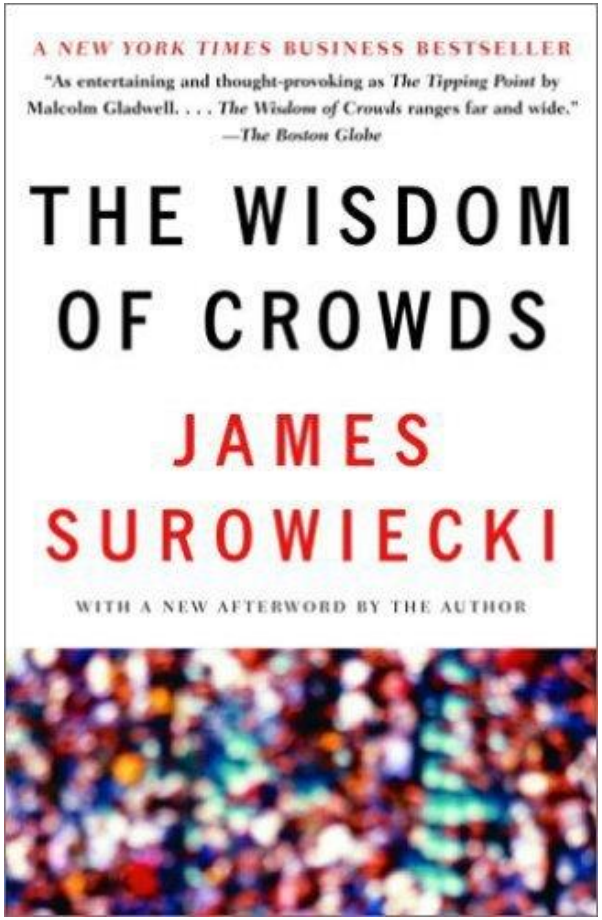
  vs.

- Bottom-up
- Self-organization
- Adhocracy

Source: http://funnyasduck.net/post/10458

Wisdom of the Crowd

- Take collective guesses of the crowd
- Aggregated answer is better than expert in many cases
- Works well for:
  - Quantity estimation
  - General knowledge
  - Spatial reasoning
- Not all crowds are wise!



Why is this important?

Problem

- Command and control is slow and inefficient
- Poor information flow in top-down structures
- Ineffective decisions

Solution

- Self-organizing teams
- Invert control structure
- Wisdom of the Crowds

Self-organizing Agile teams are more efficient

6. Efficient Communication

Cost of Poor Communication

- Cost is enormous
- Hard to quantify
- Hidden cost
- Expense is real

Source: http://www.cathy.willman.com/2012/06/what-boys-need.html

Cost of Poor Communication

- 17.5 hrs / person / week
- Top 5 issues identified:
  1. Waiting for information
  2. Unwanted communication (noise)
  3. Inefficient coordination among team members
  4. Barriers to collaboration
  5. Customer complaints

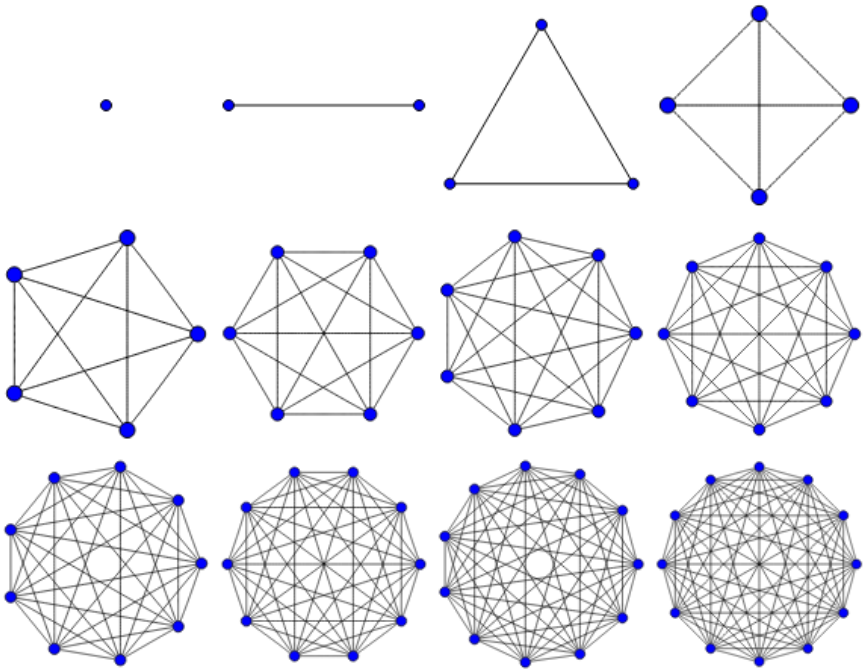
Source: http://thoughtleadership.sismarketresearch.com/industrial-b2b-journal/2009/3/10/smb-communications-pain-study-white-paper-uncovering-the-hid.html

Total estimated annual cost of poor communication per enterprise knowledge worker: **$50,562**

Source: http://thoughtleadership.sismarketresearch.com/industrial-b2b-journal/2009/3/10/smb-communications-pain-study-white-paper-uncovering-the-hid.html

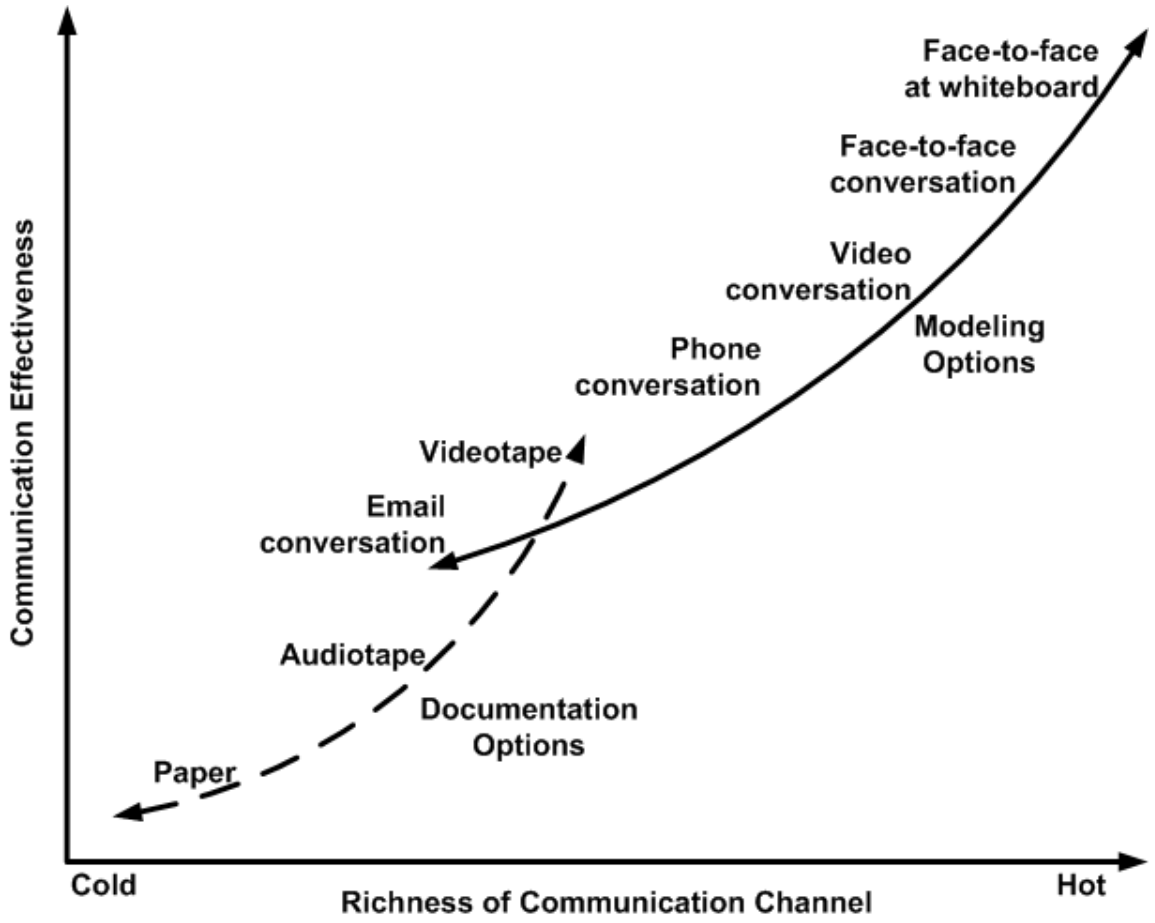
Communication Structures - Complete Graph

- Communication network modeled as a complete graph
  - Nodes = people
  - Edges = channels
- Edges increase by $O(n^2)$ in the number of nodes
- Becomes extremely inefficient very fast



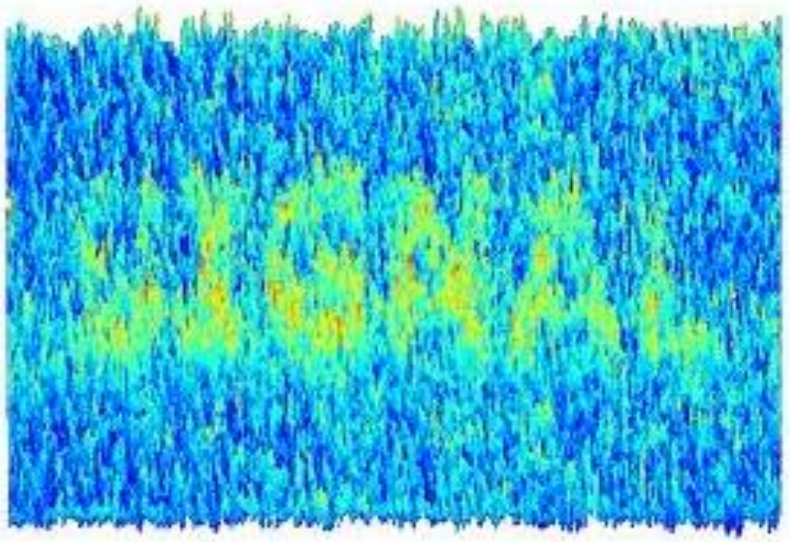
Source: Wikipedia

Effectiveness of Communication



Signal-to-Noise Ratio

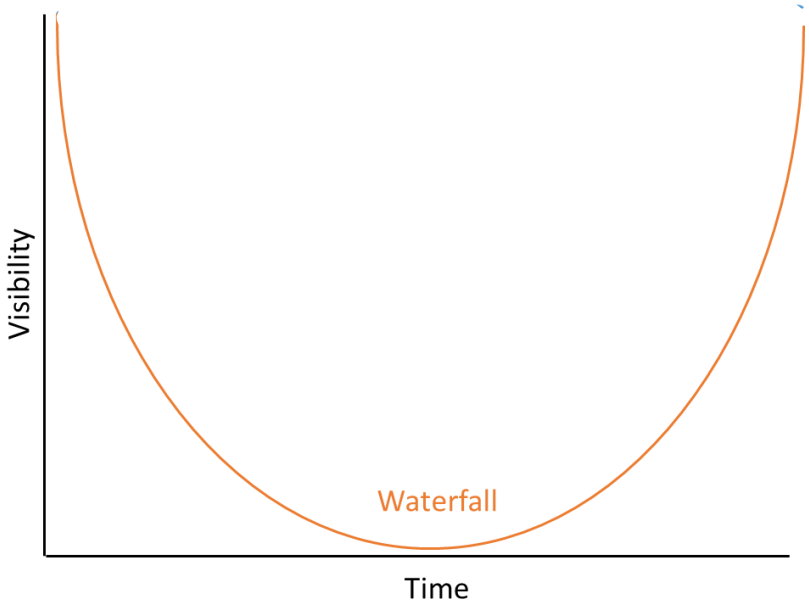
- SNR = P(signal) / P(noise)
- Signal = message
- Noise = everything else
- Goal is to maximize signal-to-noise ratio



Source: http://uber.la/2012/05/signal-to-noise/

Visibility

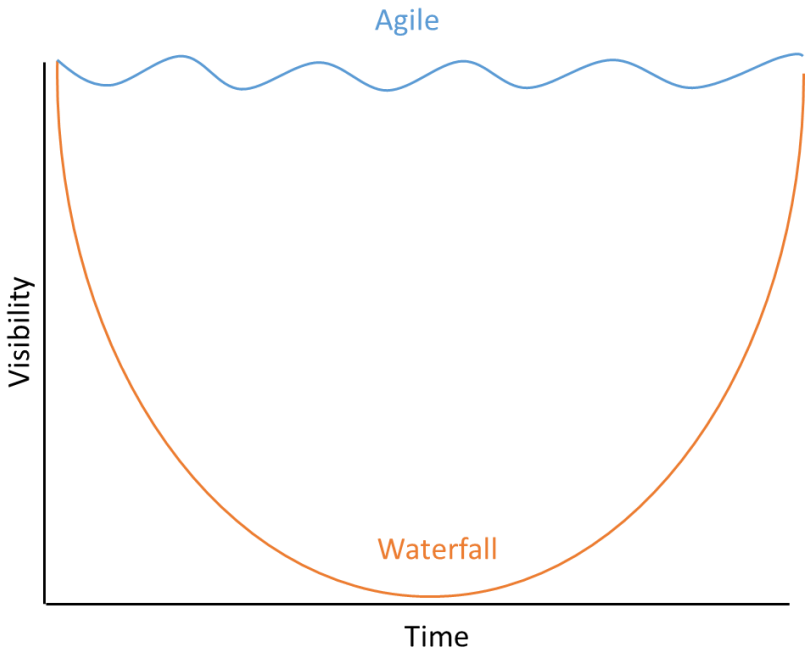
- Waterfall tends to hide many problems
- High visibility in the beginning
- Low visibility in the middle
- High visibility in the end



Original source: http://www.versionone.com/Agile101/Agile-Software-Development-Benefits/

Visibility

- Agile provides visibility:
  - Information radiators
  - Regular inspection and adaptation
  - Frequent delivery of working software
- Agile is on the surface with project visibility
- Problems have no where to hide



Original source: http://www.versionone.com/Agile101/Agile-Software-Development-Benefits/

Why is this important?

Problem

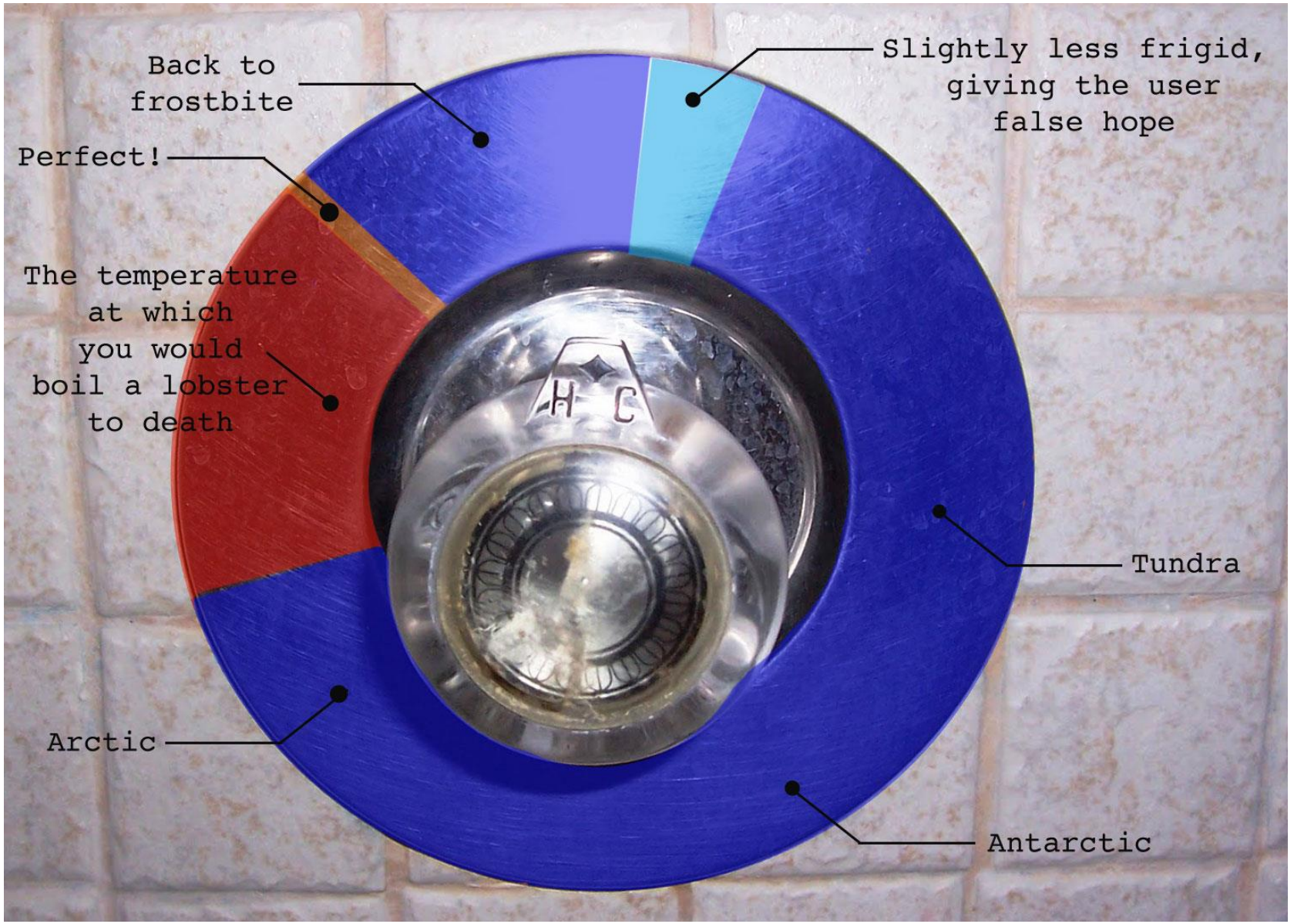
- Communication overload
- High cost of Miscommunication
- Lack of transparency

Solution

- Small teams
- Maximize signal-to-noise ratio
- Increase visibility

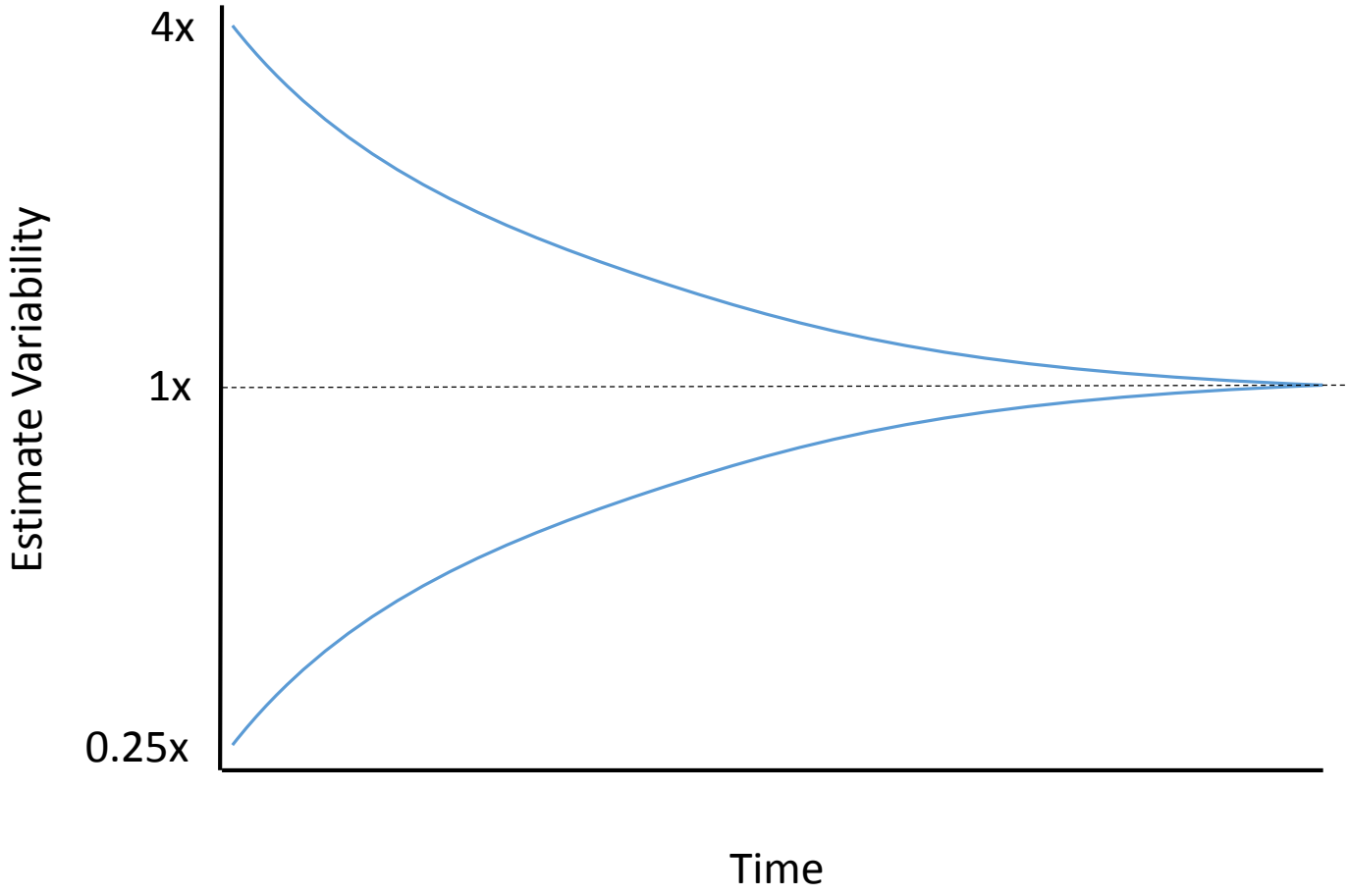
Agile teams communicate more effectively

7. Feedback



Source: http://www.letterstobuffoons.com/wp-content/uploads/2012/09/ShowerHandle.jpg

Cone of Uncertainty



Original Source: Barry Boehm, Software Engineering Economics (1981)

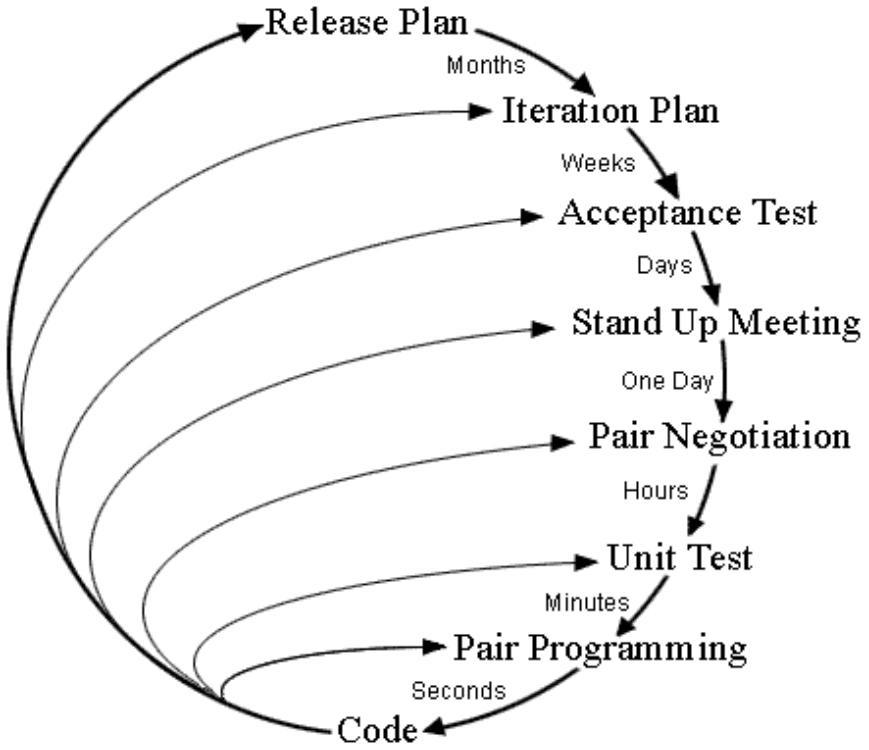
Feedback and Learning

- We reduce uncertainty by learning
- Feedback is necessary for learning
- Continuous and rapid feedback allows us to learn more effectively

Source: http://www.icanhascheezburger.com

Agile Feedback

- Continuous and rapid feedback
- Multiple timescales
- Powerful for:
  - Learning
  - Reducing risk
  - Eliminating Uncertainty



Source: http://www.agile-process.org/communicate.html

Smart Failure

- Short and frequent experiments
  - Lots of small failures
  - Lots of small successes
- Low cost and high value
- Old world vs. new world
- Requires mindset change
  - It’s ok to fail small
  - It’s ok to fail smart
  - However…

Source: http://craftfail.com/2011/08/cookie-monster-cupcake-fail/

It’s Not OK to Fail BIG!

Source: http://t4toby.files.wordpress.com/2008/07/epicfail1.jpg/

Know When to Pivot

- Pivot = change direction
- When our assumptions are incorrect we pivot
- Pivot early, not late
- Minimize cost to pivot

Source: http://thesalespivot.com/wp-content/uploads/2011/07/left-turn-sign.jpg

Why is this important?

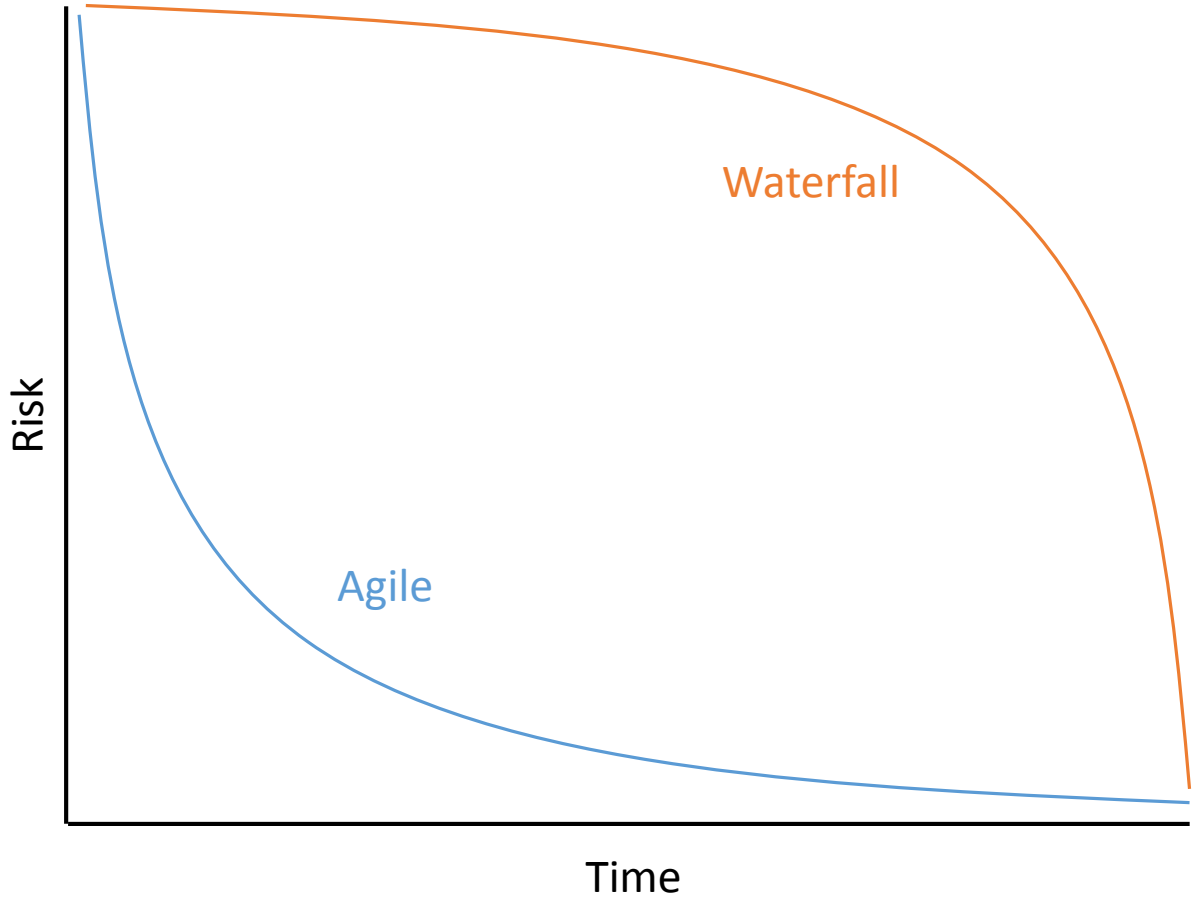
Problem

- Cone of Uncertainty
- Epic Failure
- Difficulty changing course once invested

Solution

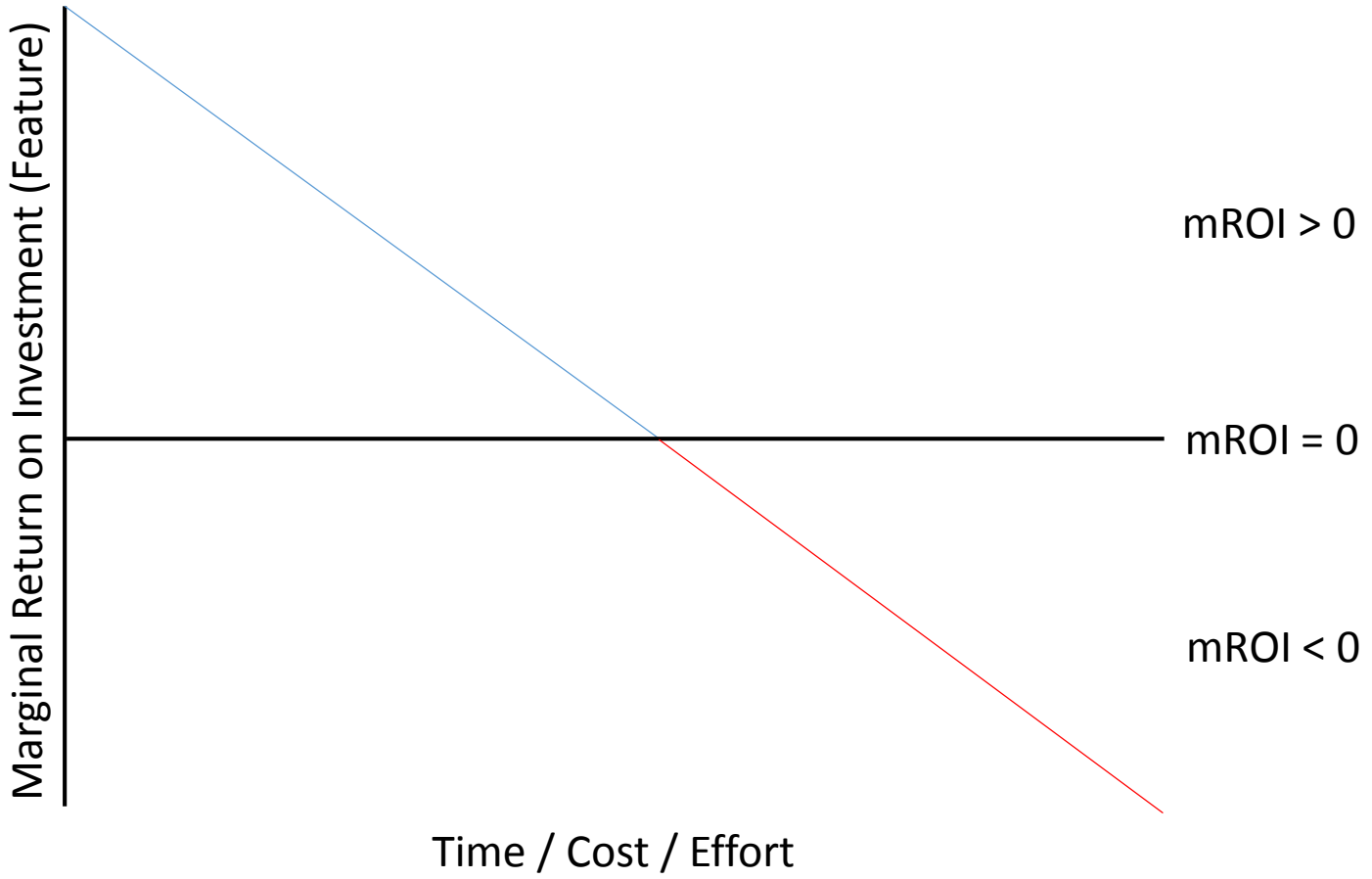
- Feedback
- Embrace Smart Failure
- Minimize cost to learn and then pivot often

Agile Teams Use Feedback to Reduce Risk



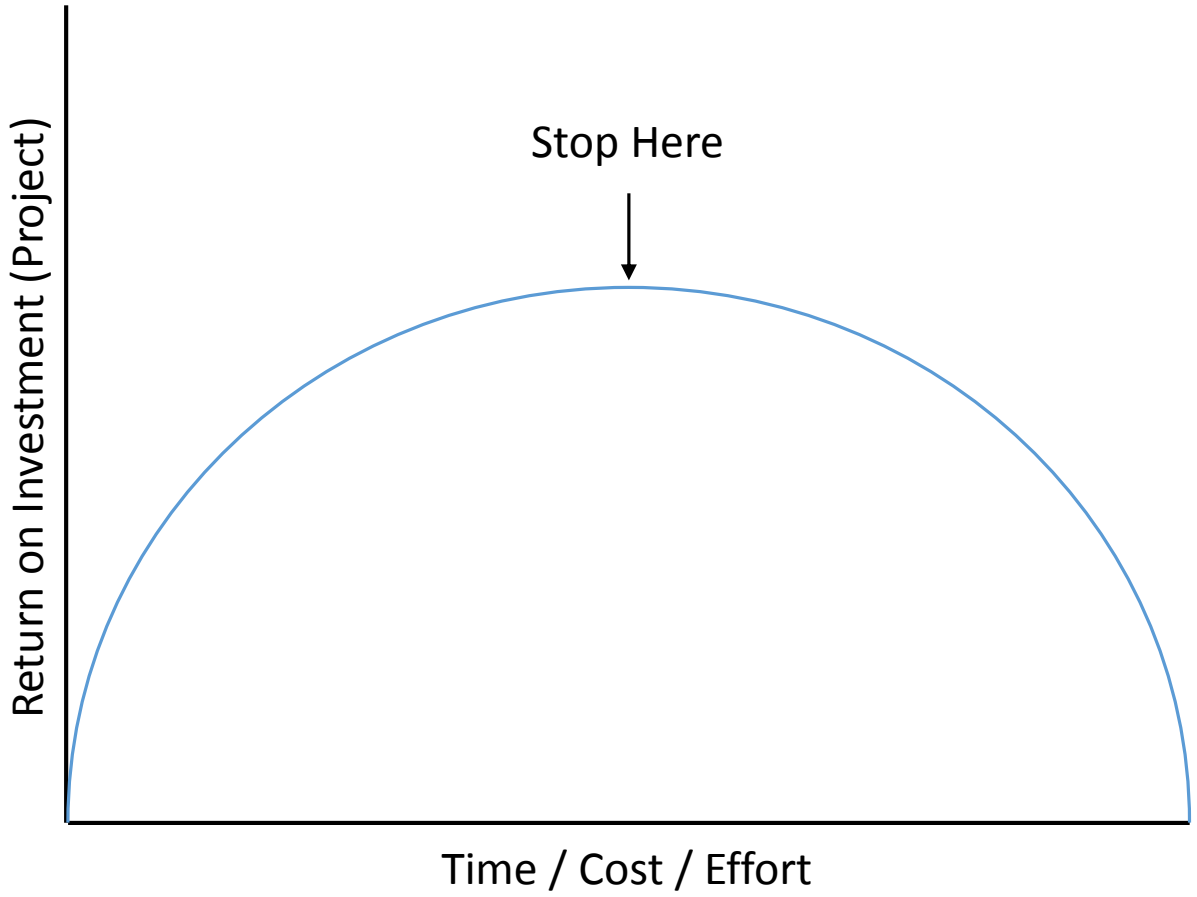
Source: http://www.versionone.com/Agile101/Agile-Software-Development-Benefits/

Know When to Stop



Original Source: http://www.versionone.com/Agile101/Agile-Software-Development-Benefits/

Know When to Stop



Original Source: http://www.versionone.com/Agile101/Agile-Software-Development-Benefits/

Know When to Stop

- Everything else:
  - The Cost of Complexity
  - Eliminating Waste
  - Inventory Hides Problems
  - Metrics Have Consequences
  - Embracing Human Factors
  - Information Gain / Entropy
  - Embedded Documentation
  - Kanban and Queuing Theory
  - TDD, Dopamine, and Crack
  - Sustainable Development
  - Agile is an Emergent Property
  - and much more…

Source: http://www.rounds.com/blog/wp-content/uploads/2010/11/stop-hammertime.png

Conclusion

Why is Agile So Successful?

1. It is well adapted to the world after midnight.
2. It inverts the project management constraints.
3. It prioritizes features by business value.
4. It embraces change vs. following a plan.
5. It utilizes the efficiencies of self-organization.
6. It produces more effective communication.
7. It provides continuous and rapid feedback.

Questions?

Feedback

- One thing you liked
- One thing you would change

Contact Info

Matthew Renze

- Twitter: @matthewrenze
- Email: matthew@renzeconsulting.com
- Website: www.matthewrenze.com

Renze Consulting

www.renzeconsulting.com