# Topic 04:
# Agile Development

# References

- [Pressman, 2010] Pressman, Roger S. Software Engineering: A Practitioner’s Approach. New York: McGraw-Hill Higher Education, 2010.
- [Sommerville] Sommerville, Ian, Software Engineering, 9th Edition, Pearson - Addison Wesley, England, 2011.

# AGILITY

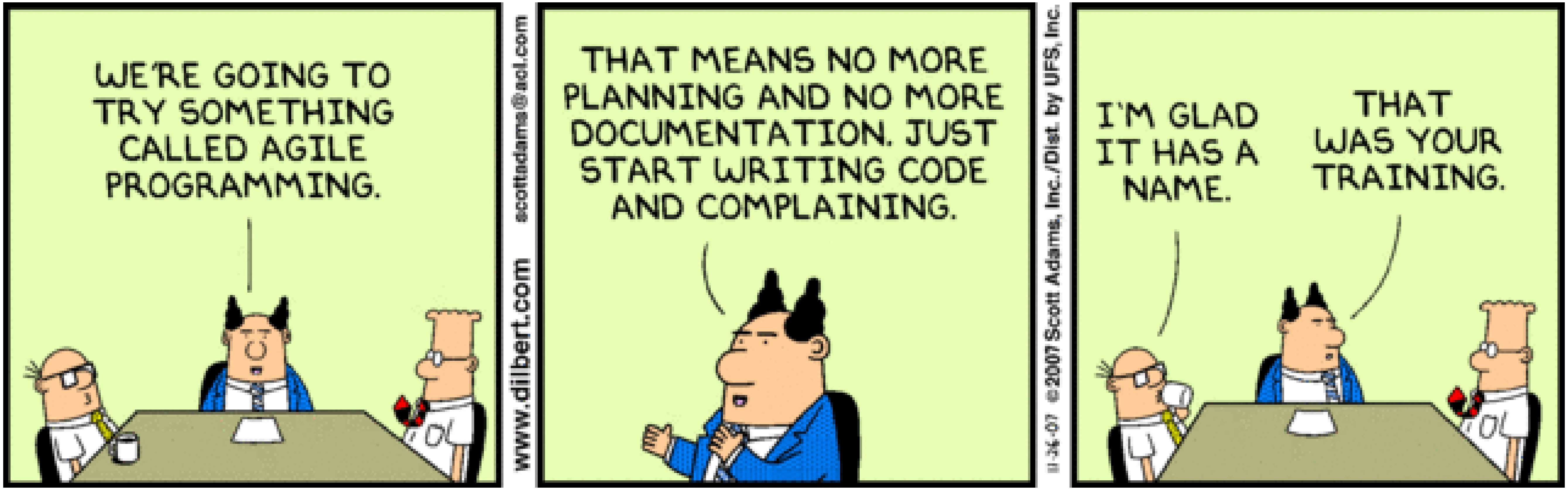
http://www.geekherocomic.com/2009/02/21/agile-development-explained/

FROM NOW ON WE'RE GOING TO CHANGE HOW WE DO THINGS HERE. WE NEED TO BE AGILE.
WHAT DOES IT MEAN?
NOTHING REALLY CHANGES, EXCEPT THAT NOW MANAGERS, DESIGNERS AND ARCHITECTS HAVE AN EXCUSE FOR CHANGING THE REQUIREMENTS ON FRIDAYS AT 4 PM.

#113 - "AGILE DEVELOPMENT, EXAPLAINED" - BY SALVATORE IOVENE, FEB. 21ST 2009

HTTP://WWW.GEEKHEROCOMIC.COM/

http://search.dilbert.com


WE'RE GOING TO TRY SOMETHING CALLED AGILE PROGRAMMING.
THAT MEANS NO MORE PLANNING AND NO MORE DOCUMENTATION. JUST START WRITING CODE AND COMPLAINING.
I'M GLAD IT HAS A NAME.
THAT WAS YOUR TRAINING.
www.dilbert.com scottadams@aol.com
11-26-07 © 2007 Scott Adams, Inc./Dist. by UFS, Inc.

# What is *“Agility”* ?

- Effective (rapid and adaptive) response to change
- Effective communication among all stakeholders
- Drawing the customer onto the team
- Organizing a team so that it is in control of the work performed
- Rapid, incremental delivery of software

# Plan-driven Approach v.s. Agility

[Sommerville]

- For some types of software, such as safety-critical control systems, where a complete analysis of the system is essential, a **plan-driven approach** is the right one.
- In a fast-moving business environment, the requirements are likely to change quickly and unpredictably due to external factors. If using a plan-driven approach, the software may then be out-of-date when it is delivered.

## Plan-driven Approach

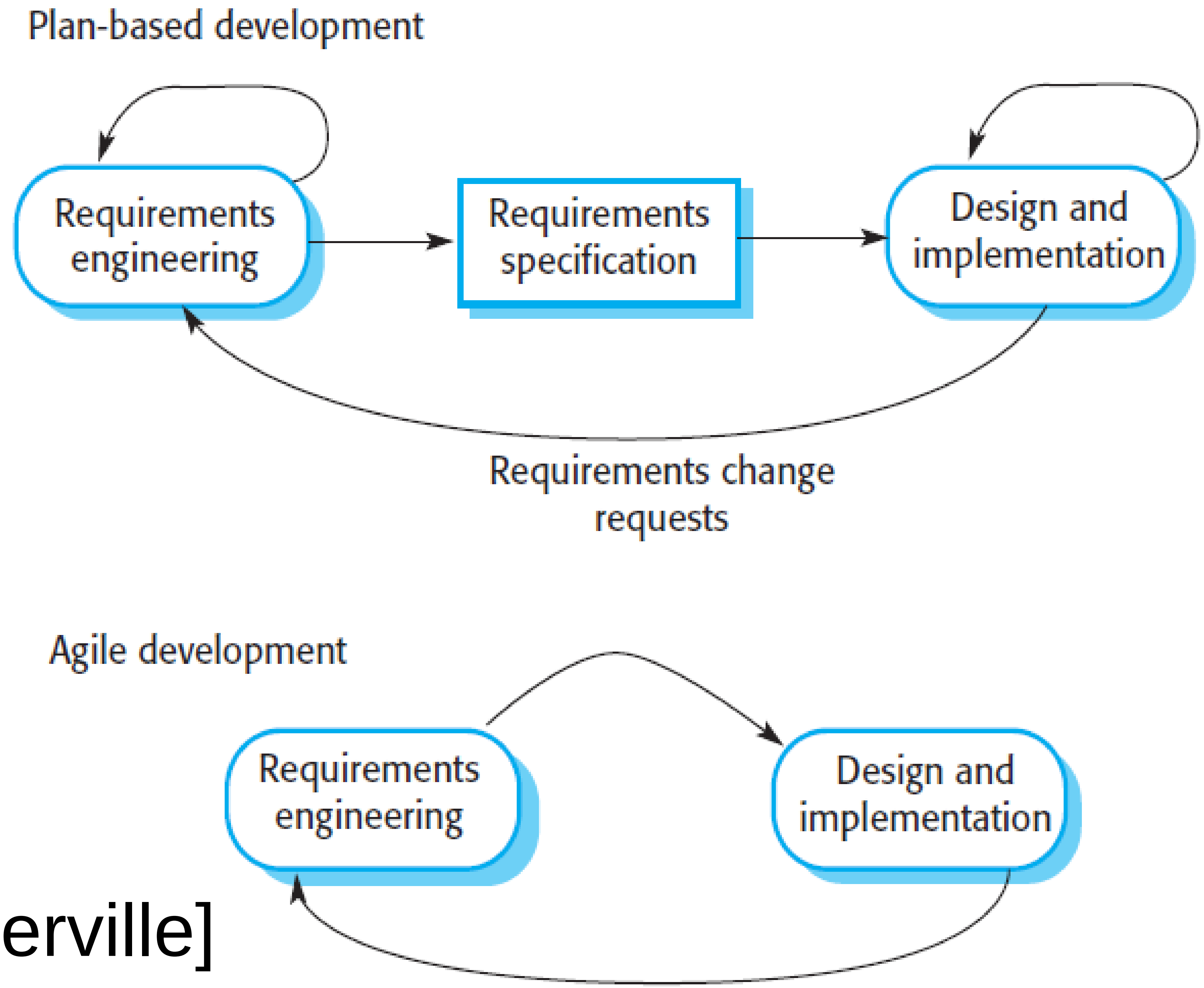
- Identifies separate stages in the software process with outputs associated with each stage.
- Iteration occurs within activities with formal documents used to communicate between stages of the process
  - Example: requirements will evolve until a requirements specification is produced, as an input to design and implementation

## Agile Approach

- Consider design and implementation to be the central activities in the software process
  - Incorporate other activities, such as requirements elicitation and testing, into design and implementation
- Iteration occurs across activities
  - requirements and design are developed together, rather than separately

[Sommerville]

# Iteration
# Plan-driven Approach v.s. Agility


Plan-based development
Requirements engineering
Requirements specification
Design and implementation
Requirements change requests
Agile development
Requirements engineering
Design and implementation

[Sommerville]

# An Agile Process

- Is driven by customer descriptions of what is required (scenarios)
- Recognizes that plan are short-lived
- Develops software iteratively with a heavy emphasis on construction activities
- Delivers multiple ‘software increments’
- Adapts as change occur

# The Manifesto for
Agile Software Development

"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

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

That is, while there is value in the items on the right, we value the items on the left more."

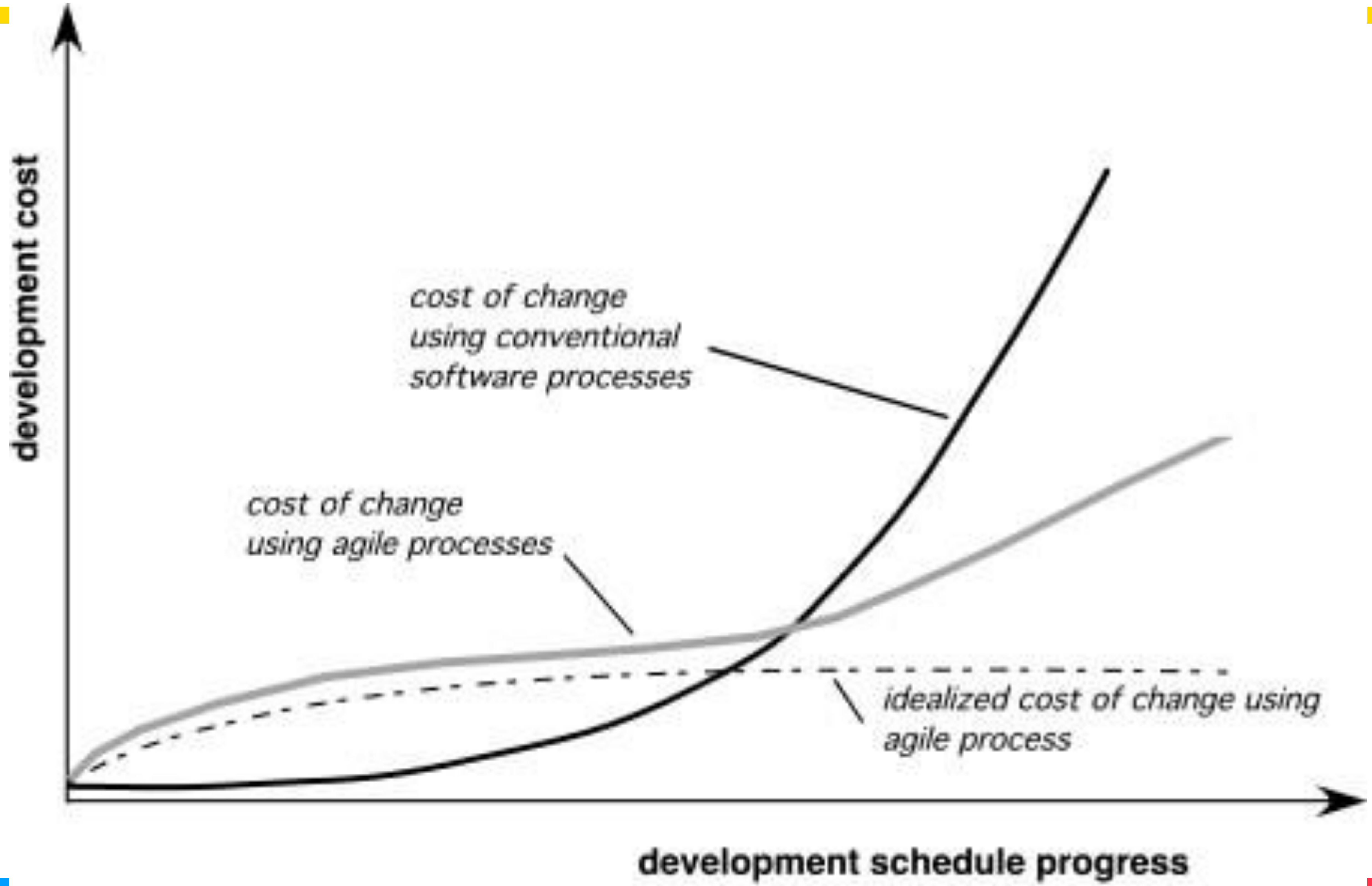
Kent Beck et al

# Principles of Agile Methods

[Sommerville]

- Customer involvement
  - Customers should be closely involved throughout the development process.
- Incremental delivery
- People, not process
  - The skills of the development team should be recognized and exploited
- Embrace change
- Maintain simplicity

# Agility and the Cost of Change



# Agility Principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.

# Agility Principles

5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

# Agility Principles

9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity - the art of maximizing the amount of work not done - is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

# Human Factors

- ***The process molds to the needs of the people and team***, not the other way around
- Key trait must exist among the people on an agile team and the team itself:
  - ❖ Competence
  - ❖ Common focus
  - ❖ Collaboration
  - ❖ Decision-making ability
  - ❖ Fuzzy problem-solving ability
  - ❖ Mutual trust and respect
  - ❖ Self organization

# Types of System for Agile Development

[Sommerville]

- small or medium-sized product for sale
- custom system development within an organization
  - where there is a clear commitment from the customer to become involved in the development process and
  - where there are not a lot of external rules and regulations that affect the software

# AGILE DEVELOPMENT PROCESS MODELS

# Extreme Programming (XP)

The most widely used agile process, originally proposed by Kent Beck

http://search.dilbert.com

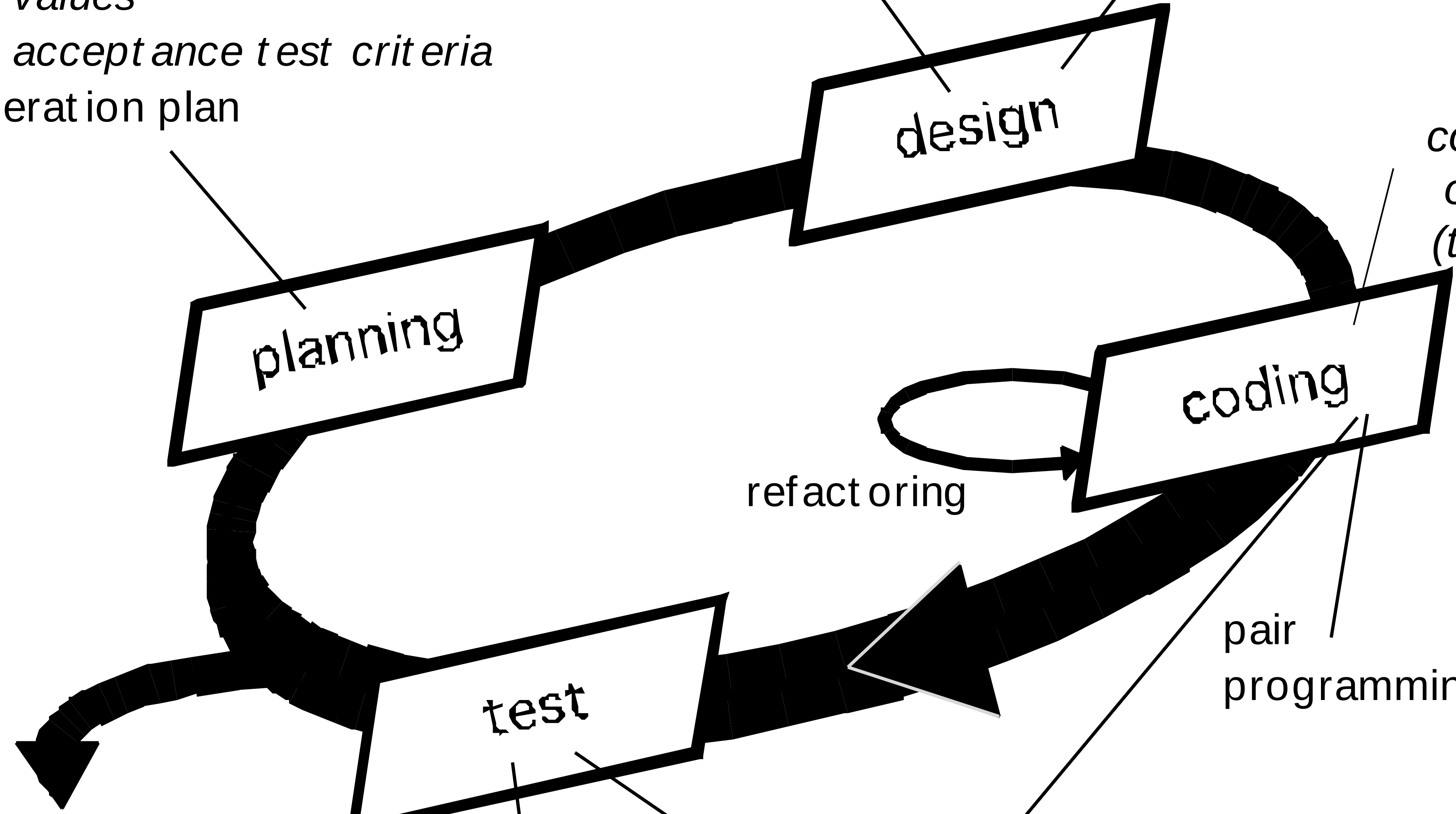
WE'RE GOING TO TRY SOMETHING CALLED EXTREME PROGRAMMING.
scottadams@aol.com
www.dilbert.com
FIRST, PICK A PARTNER. THE TWO OF YOU WILL WORK AT ONE COMPUTER FOR FORTY HOURS A WEEK.
© 2002 United Feature Syndicate, Inc.
1/9/03
THE NEW SYSTEM IS A MINUTE OLD AND I ALREADY HATE EVERYONE.

# Extreme Programming (XP)

http://search.dilbert.com

EXTREME PROGRAMMING
I CAN'T GIVE YOU ALL OF THESE FEATURES IN THE FIRST VERSION.
scottadams@aol.com
www.dilbert.com
AND EACH FEATURE NEEDS TO HAVE WHAT WE CALL A "USER STORY."
© 2002 United Feature Syndicate, Inc.
1/10/03
OKAY, HERE'S A STORY: YOU GIVE ME ALL OF MY FEATURES OR I'LL RUIN YOUR LIFE.

# Extreme Programming (XP)



simple design
*CRC cards*

spike solutions
*prototypes*

user stories
*values*
*acceptance test criteria*
iteration plan

*construction of unit test (test cases)*

pair programming

unit test
continuous integration

acceptance testing

Release

software increment
*project velocity computed*

# Extreme Programming (XP)

- XP Planning
  - ❖ Begins with the creation of ***“user stories”***
  - ❖ Customer assisns a ***value*** (i.e. priority) for each story
  - ❖ Agile team assesses each story and assigns a ***cost*** (measured in development weeks)
  - ❖ Stories are grouped for a ***deliverable increment***
  - ❖ A ***commitment*** is made on delivery date
  - ❖ After the first increment, ***“project velocity”*** is used to help define subsequent delivery dates for other increments

# Extreme Programming (XP)

- XP Design
  - ❖ Follows the ***KISS principle***
  - ❖ Encourage the use of ***Class Responsibility Collaboration (CRC) cards***
  - ❖ For difficult design problems, suggests the creation of ***"spike solution"*** - a design prototype
  - ❖ Encouraging ***"refactoring"*** - an iterative refinement of the internal program design

# Extreme Programming (XP)

[http://search.dilbert.com](http://search.dilbert.com)

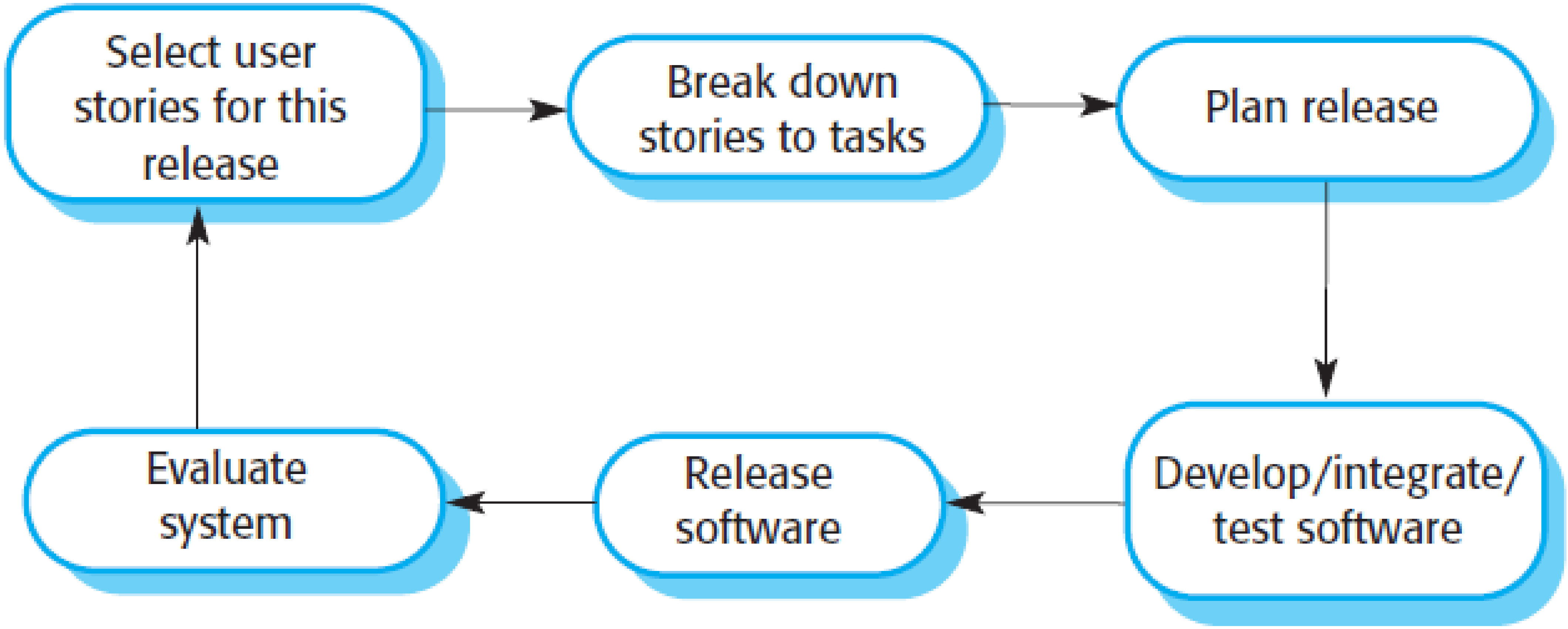
EXTREME PROGRAMMING
THE TWO OF YOU WILL BE A CODE-WRITING TEAM.
scottadams@aol.com
www.dilbert.com
STUDIES PROVE THAT TWO PROGRAMMERS ON ONE COMPUTER IS THE MOST PRODUCTIVE ARRANGEMENT.
© 2002 United Feature Syndicate, Inc.
SOMETIMES I CAN WHISTLE THROUGH BOTH NOSTRILS. I'VE SAVED A FORTUNE IN HARMONICAS.

# Extreme Programming (XP)

- XP Coding
  - ❖Recommends the ***construction of a unit test*** for a store before coding commences
  - ❖Encourages ***"pair programming"***
- XP Testing
  - ❖All ***unit tests are executed daily***
  - ❖***"Acceptance tests"*** are defined by the customer and executed to assess customer visible functionality

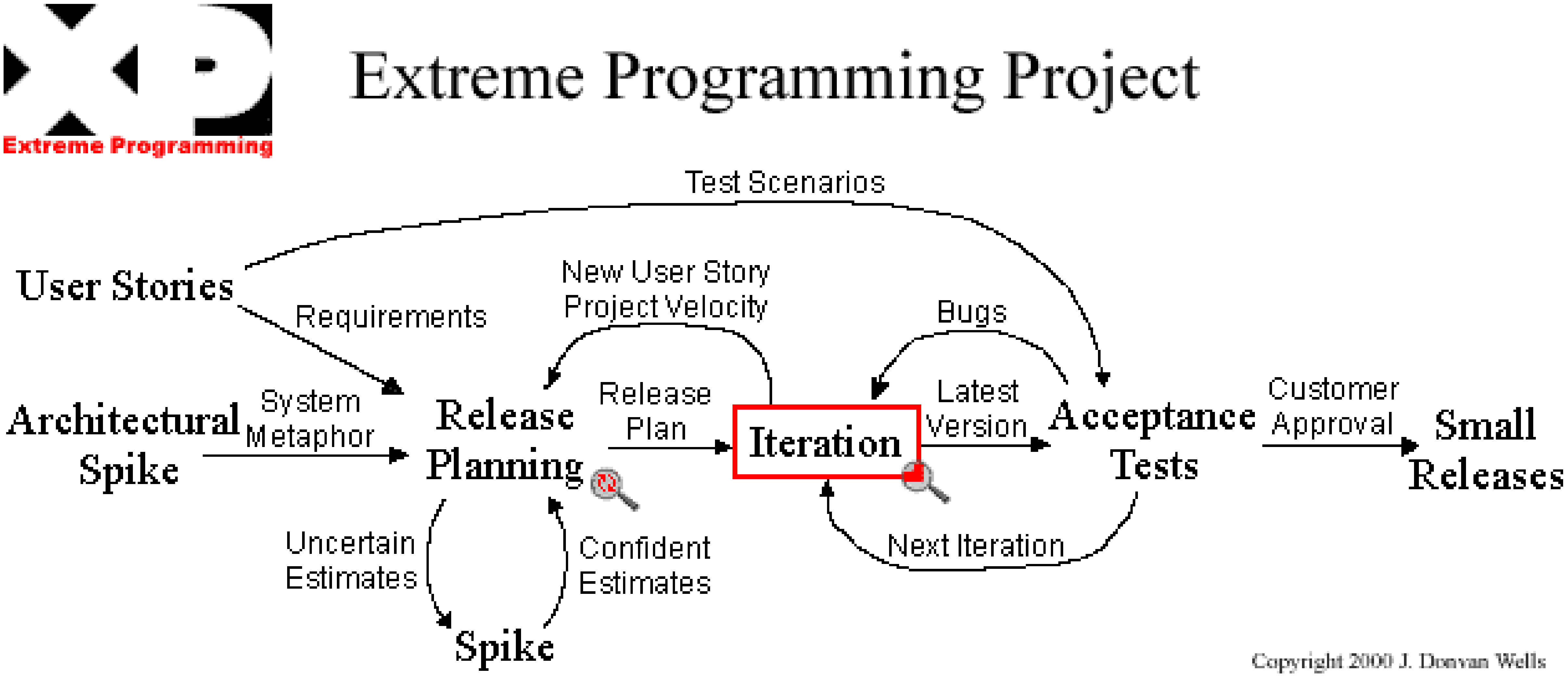
# XP Release Cycle

[Sommerville]


Select user stories for this release
Break down stories to tasks
Plan release
Develop/integrate/ test software
Release software
Evaluate system

# Extreme Programming (XP)

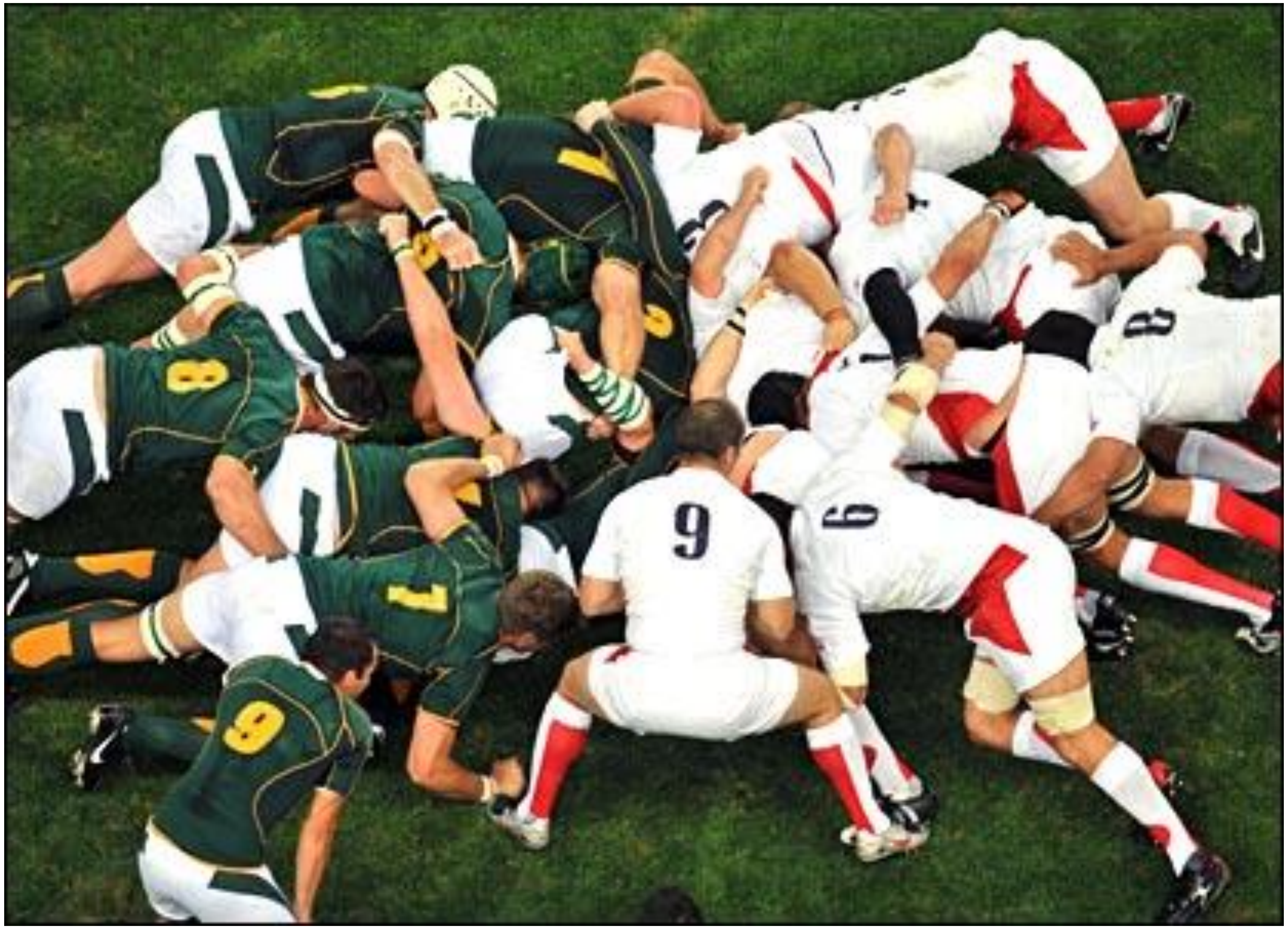
http://www.extremeprogramming.org/map/project.html



# SCRUM

http://newsimg.bbc.co.uk/media/images/44181000/jpg/_44181384_eng_sa_scrum416.jpg

Scrum in a game of rugby



# SCRUM

- Originally proposed by Schwaber and Beedle
- Scrum - distinguishing features
  - ❖ Development work is partitioned into ***“packets” called product backlog***
  - ❖ ***Testing and documentation are on-going*** as the product is constructed
  - ❖ Work occurs in ***“sprints”*** and is derived from a “backlog” of existing requirements
  - ❖ ***Meeting are very short*** and sometimes conducted without chairs
  - ❖ ***“Demos”*** are delivered to the customer with the time-box allocated

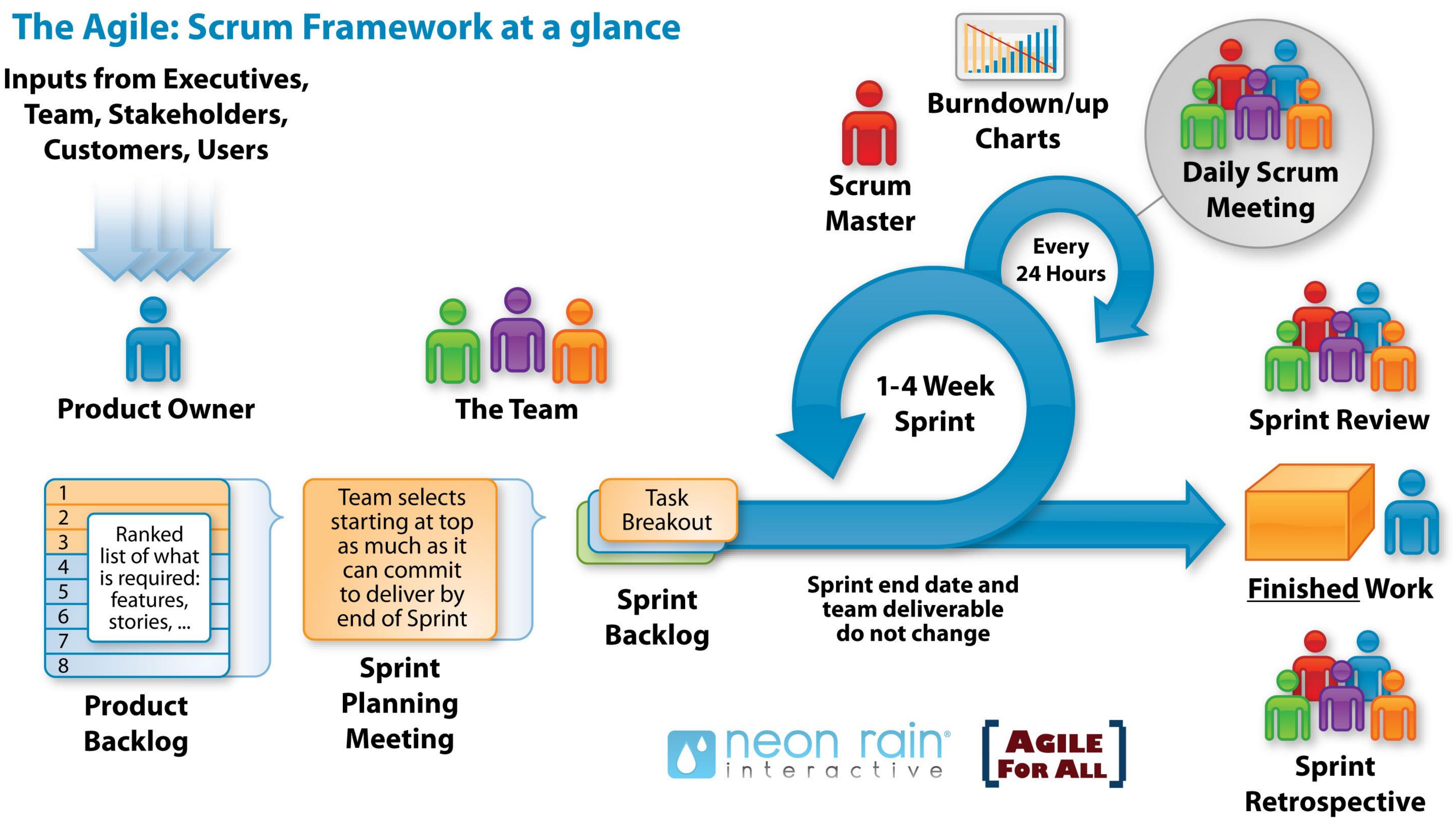
# Phases in Scrum

- Outline Planning
  - ❖establish the general objectives for the project and design the software architecture
- Series of Sprint Cycles
  - ❖each cycle develops an increment of the system
- Project Closure
  - ❖wraps up the project, completes required documentation such as system help frames and user manuals and assesses the lessons learned from the project

# Sprint Cycle

- Sprints are fixed length, normally 2–4 weeks
- The starting point for planning is the product backlog
  - ❖Assessment phase: backlog is reviewed, priorities and risks are assigned
- Selection
  - ❖Each project team select features and functionality to be developed during the sprint
- Short daily meetings
- At the end of the sprint, the work done is reviewed and presented to stakeholders. The next sprint cycle then begins.

# http://35qk152ejao6mi5pan29erbr9.wpengine.netdna-cdn.com/wp-content/uploads/2013/05/Scrum_Framework.jpg



## The Agile: Scrum Framework at a glance

Inputs from Executives, Team, Stakeholders, Customers, Users

Product Owner

The Team

Scrum Master

Burndown/up Charts

Daily Scrum Meeting

Every 24 Hours

1-4 Week Sprint

Sprint Review

1
2
3
4
5
6
7
8

Ranked list of what is required: features, stories, ...

Product Backlog

Team selects starting at top as much as it can commit to deliver by end of Sprint

Sprint Planning Meeting

Task Breakout

Sprint Backlog

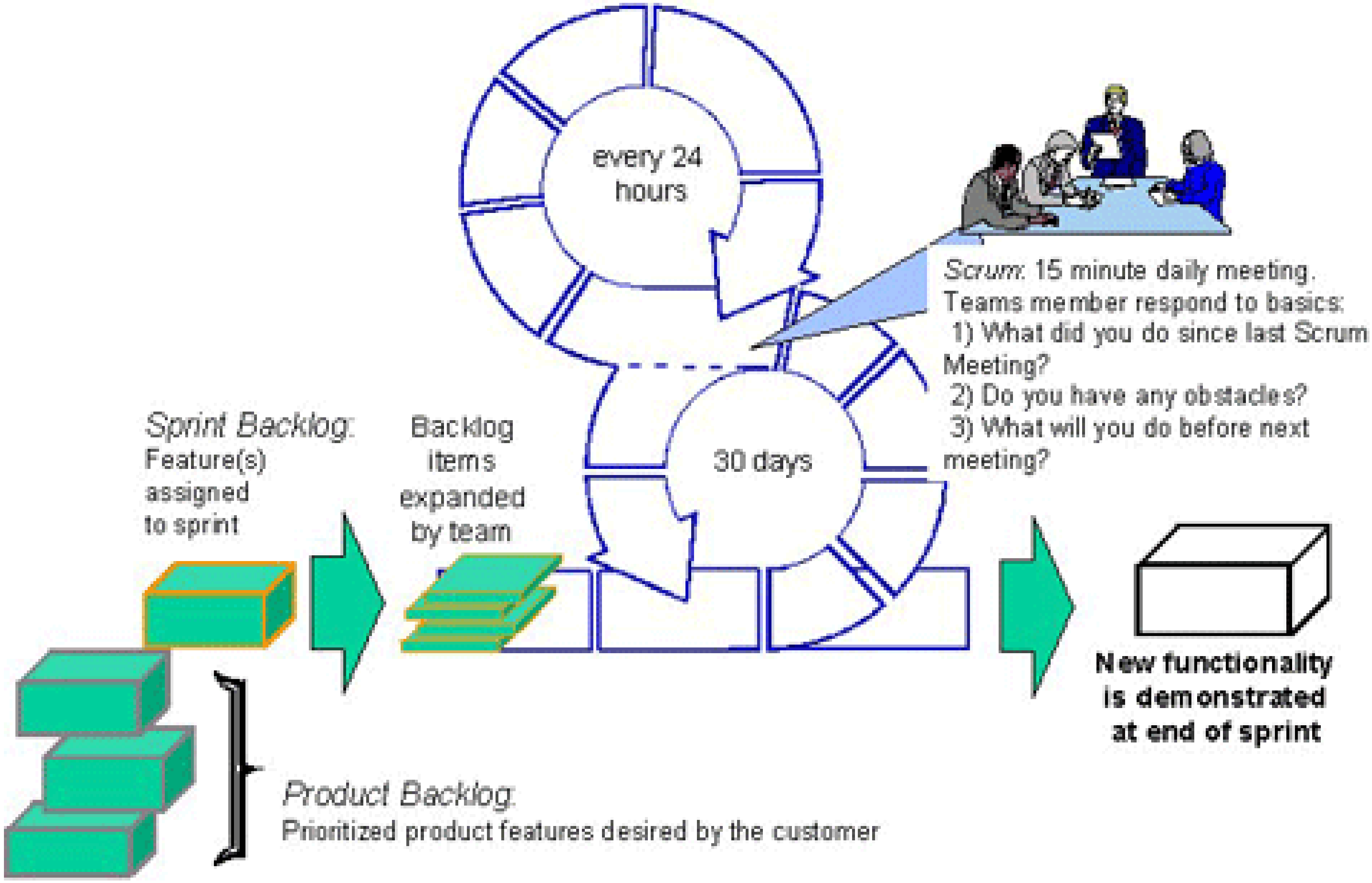
Sprint end date and team deliverable do not change

Finished Work

Sprint Retrospective

# SCRUM

http://www.kathan.at/snipsnap/space/snipsnap-index/development/methodologies+%26+processes/scrum.gif


every 24 hours
30 days
Backlog items expanded by team
New functionality is demonstrated at end of sprint

*Sprint Backlog:* Feature(s) assigned to sprint

*Scrum:* 15 minute daily meeting.
Teams member respond to basics:
1) What did you do since last Scrum Meeting?
2) Do you have any obstacles?
3) What will you do before next meeting?

*Product Backlog:* Prioritized product features desired by the customer

# Other Agile Process Models

- Adaptive Software Development
- Dynamic System Development Method
- Crystal
- Feature Driven Development
- Agile Modeling
- Agile Unified Process

# Q & A