

322 371 Software Engineering

Chapter 03: Agile Software Development - XP

By

Chitsutha Soomlek, Ph.D.

Department of Computer Science
Faculty of Science, KKU

Agile Manifesto

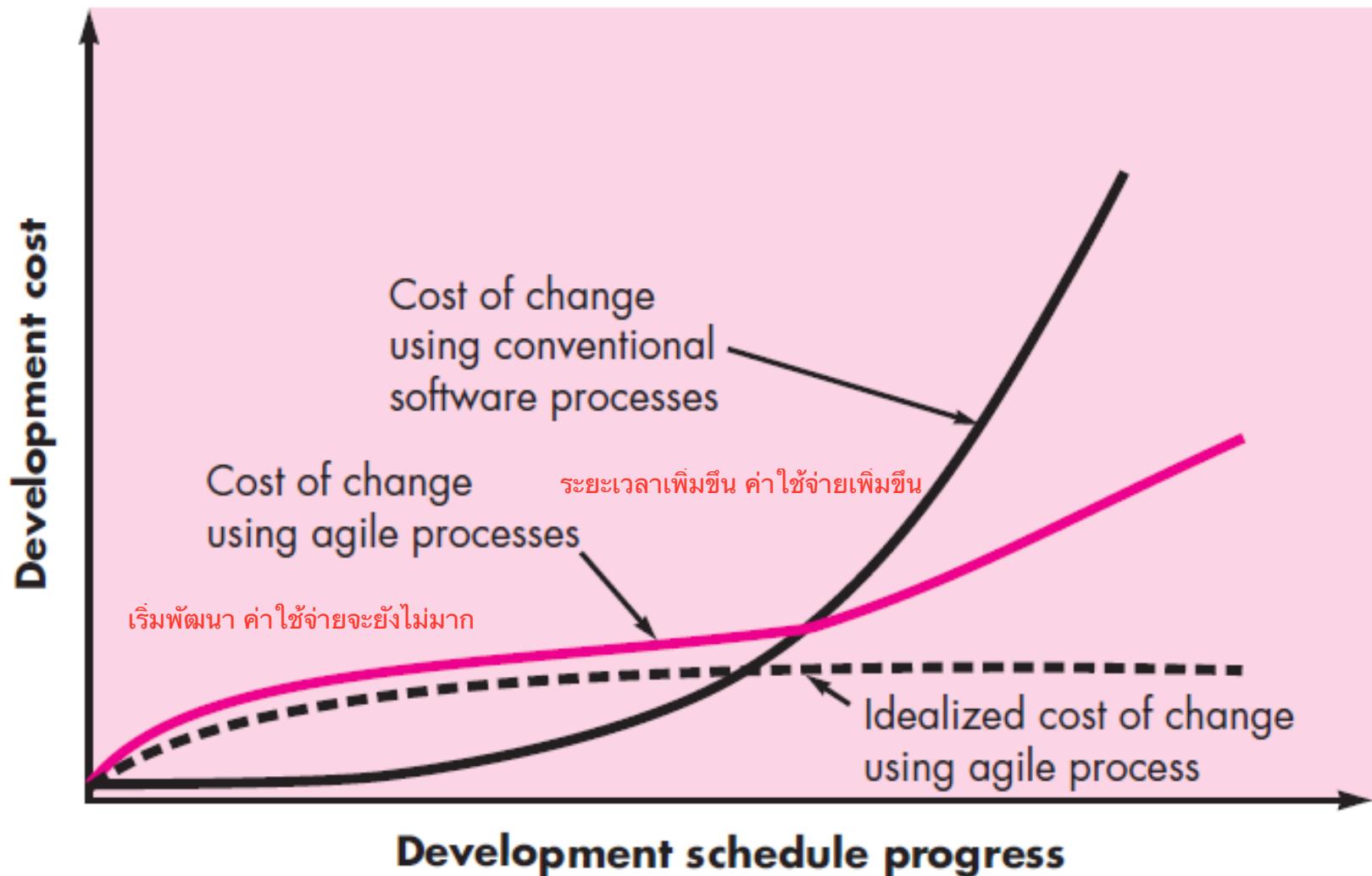
We are uncovering better ways of developing software by doing it and helping others do it.

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.

Agile and the Cost of Change



Agile Project Management

มีการวางแผนดังต่อไปนี้
คนวางแผนคือ Project Manager
จะเป็นคนวางแผนทั้งหมด
เลือกคน กำหนดเวลา
มีการเขียนแผน ติดตามแผน

Plan-Driven

- Software project manager is responsible for managing the project so that the software is delivered on time and within the planned budget for the project
- Managers draw up a plan for the project showing what should be delivered, when it should be delivered and who will work on the development of the project deliverables.

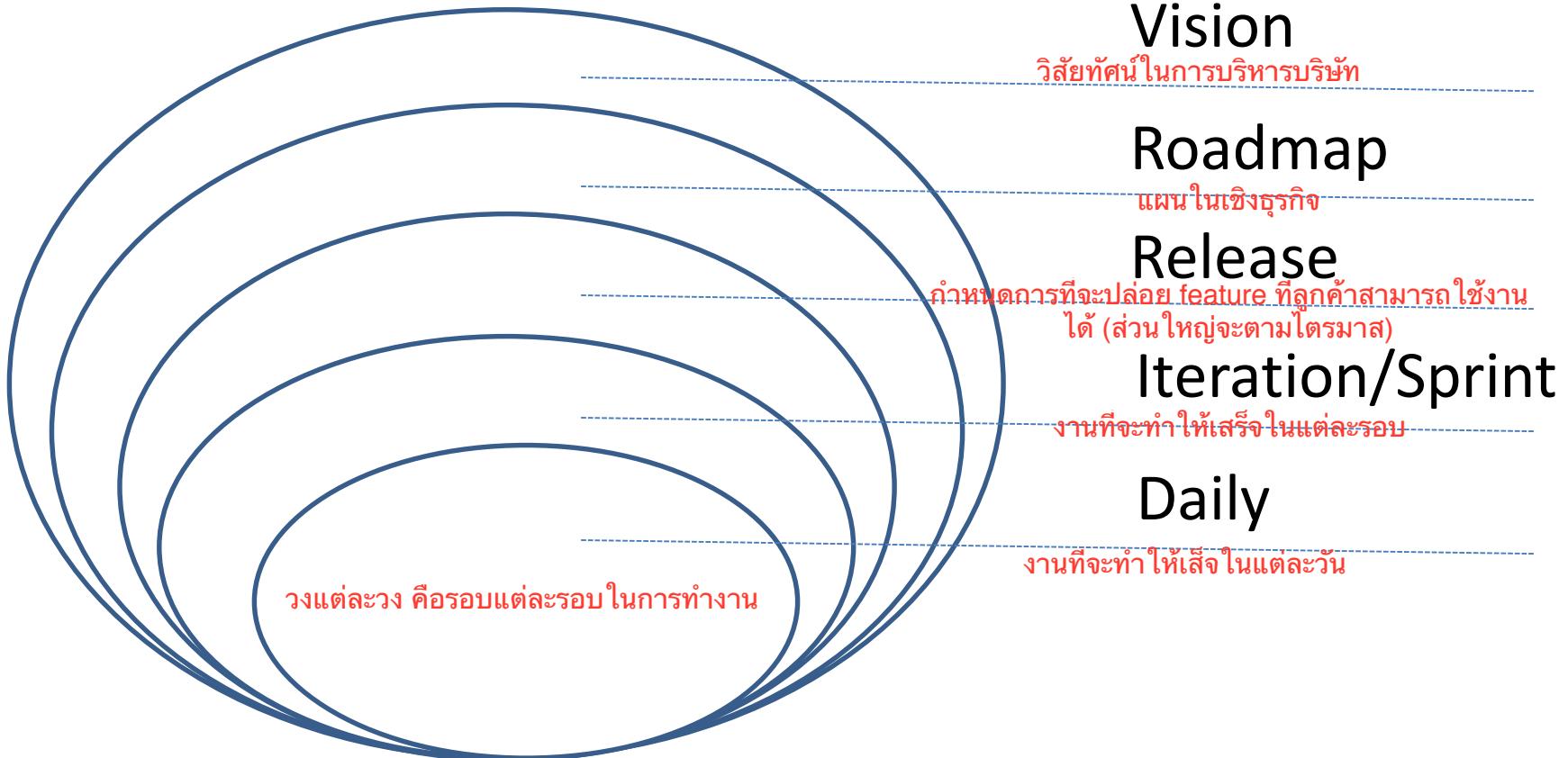
บริหารกันเป็นรอบๆ
ทีมมีการวางแผน
ไม่มี PM มาแพลนให้
ต้องดูว่างานที่เคยมาทำจะทำเสร็จหรือไม่

Agile

- Agile project management matches with incremental development and the particular strengths of agile methods
- Managing iterative development rather than specific technical approaches to agile software engineering

Agile Planning

มีการวางแผนตั้งแต่ระดับใหญ่ ไปถึงเล็ก



Agile Planning: Roadmap

- More detail, more definite than vision planning
แผนที่บอกว่าจะมี Release ในแต่ละ Release จะดำเนินอะไรบ้าง
ต้องกำหนดชัดเจนว่า จะ Release เมื่อไหร่ อะไรบ้าง
- Define release dates and milestones
- Set high-level themes/feature sets
- Architecture map
- Product evolution over time

Agile Planning: Roadmap

แบ่ง Release ออกเป็น 4 ไตรมาส

แต่ละ Release มีเป้าหมายยังไง
มี Features อะไรที่ต้องทำบ้าง

Product Roadmap

Release: 1.0
Target: Q1 2012
Goal: Initial Market Entry

- Features:**
- Feature 1
 - Feature 2
 - Feature 3
 - Feature 4
 - Feature 5

Release: 1.1
Target: Q2 2012
Goal: Support European Market

- Features:**
- Feature 6
 - Feature 7
 - Feature 8
 - Feature 9
 - Feature 10

Release: 1.2
Target: Q3 2012
Goal: High-Availability Version

- Features:**
- Feature 11
 - Feature 12
 - Feature 13
 - Feature 14
 - Feature 15

Release: 2.0
Target: Q4 2012
Goal: Non-Linux platform variants

- Features:**
- Feature 16
 - Feature 17
 - Feature 18
 - Feature 19
 - Feature 20

Agile Planning: Release

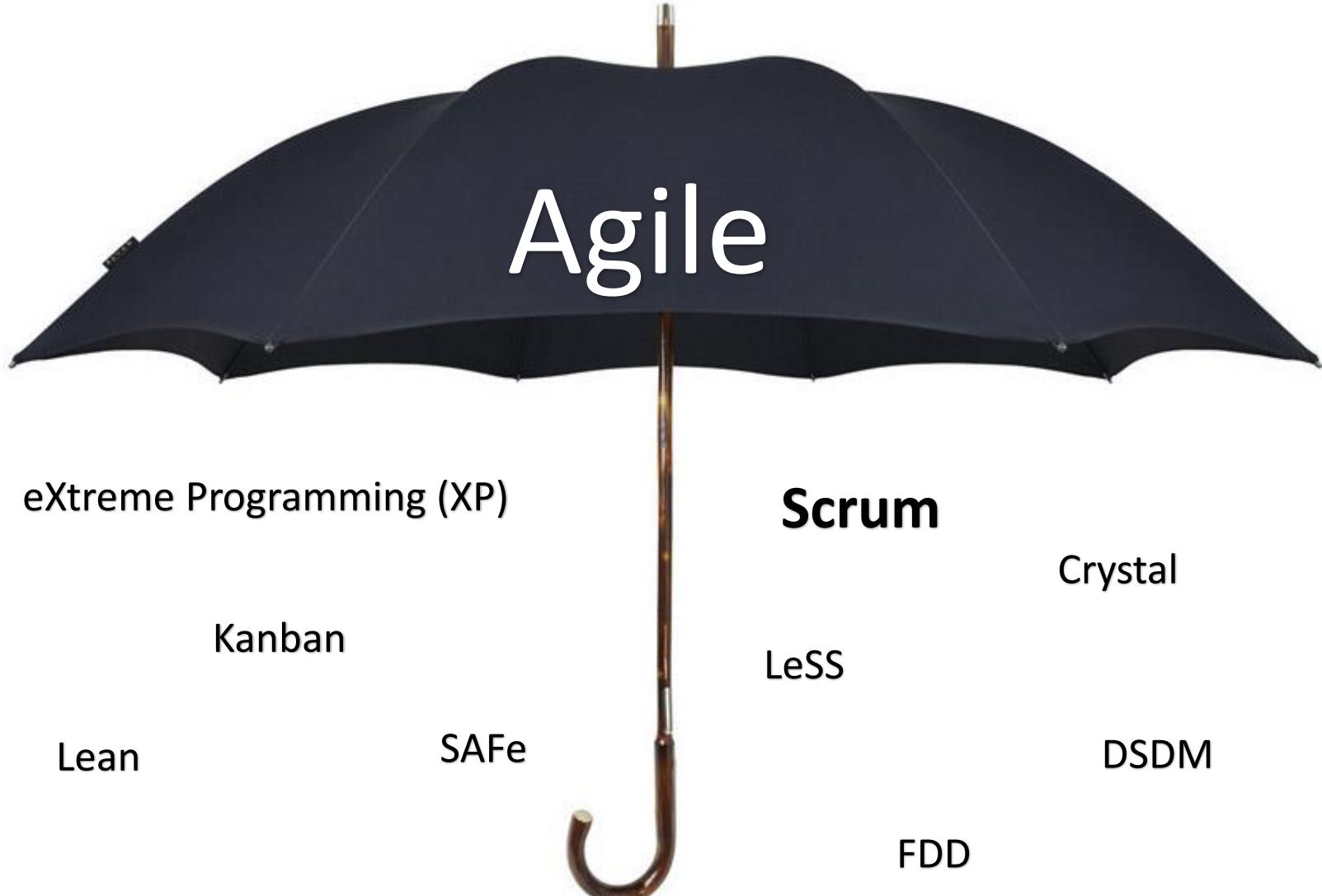
เบรี่ยนเหมือนการสร้าง Product Backlog ใน Scrum

- Set goals for release and known product backlog items
- Review current development state
- Predict the release dates
- Create a release schedule
- Issues/Concerns/Dependencies/Assumptions
- Create more stories and put into tentative sprints
- Prioritization
- Definition of done
- Communication plan
- Retrospective
- Example: <http://youtu.be/gMtv1zDUxvo>

Agile Planning: Sprint/Iteration

สิ่งที่ส่งให้ทีมคือ Product Backlog

- Review big picture
กำหนดเป้าหมายที่จะทำให้เสร็จในแต่ละ Sprint
- Review product backlog
- Set a sprint goal
- Select high priority product backlog items based on velocity
- Discuss stories to create tasks
- Estimate tasks
- Ask: “Can we commit to this?”
- Repeat until no more can be committed



eXtreme Programming (XP)

ลักษณะของ XP ก็จะทำเป็นรูปๆ เมื่อนัก

- Kent Beck (1996, 2000)
ทุกอย่างทำในระดับชั่วโมง นาที กำหนดรายละเอียดที่ลึกมาก
- XP is about social change.
- XP focuses on excellent application of programming techniques, clear communication, and teamwork.
Text
- XP aims to produce higher quality software, and higher quality of life for the development team.

เปลี่ยน

Requirement ได้ทุกวัน

- Project ที่มีความเสี่ยงสูง

- ใช้กับทีมพัฒนาซอฟต์แวร์ที่มีขนาดเล็ก 15-150 คน โดยที่ทีมทำงานอยู่ที่เดียวกัน

- ระบบการทำงานแบบ Scrum, XP, ตัวอย่าง Support Automation

eXtreme Programming (XP)

- XP pushes recognized good practice, such as iterative development, to ‘extreme’ levels
- Several new versions of a system may be developed by different programmers, integrated and tested in a day.
- XP is the most specific of the agile frameworks regarding appropriate engineering practices for software development.

When is XP applicable?

- When dealing with dynamically changing software requirements
- When there are risks caused by fixed time projects using new technology
- Small, co-located extended development team
- Automated unit and functional tests are possible

When is XP not applicable?

ซอฟต์แวร์ที่ไม่ใช้ชิ้นใหญ่ๆ หากมีการแก้ไขแล้วค่าสกัดใช้ในการแก้ไขไม่มาก

- Concurrent middleware development
- OS kernels and device drivers
- Safety critical systems where change has to be managed very carefully to preserve safety
- Legacy systems
- When the whole project is making expensive-to-change decisions based on the software
 - e.g. changing (refactoring) an application that works well on distributed boxes to one that requires one big box after the hardware has been bought

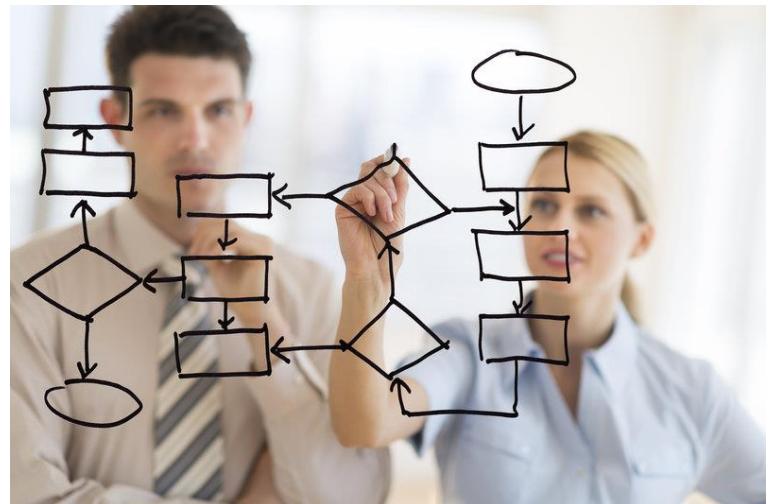
ส่วนประกอบ
XP value - สิ่งที่ทีมให้ความสำคัญ
2.XP Principle
สิ่งที่ทีมทำตามหลัก

XP Values

- Communication
- Simplicity
- Feedback
- Courage
- Respect

Communication

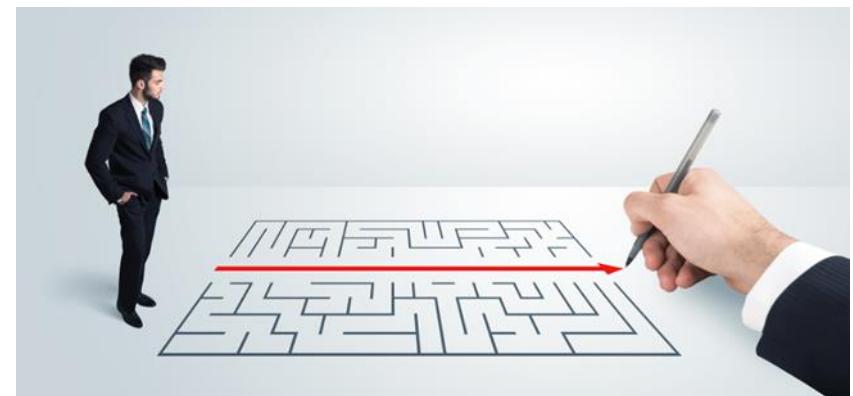
- Communicate face-to-face daily
- Face-to-face discussion with the aid of a white board or other drawing mechanism
- Work together on everything from requirements to code
- Create the best solution to our problem that we can together



เวลาเริ่มทำงานให้เริ่มจากง่ายๆ
วิธีการที่ง่ายที่สุดที่จะทำงานขึ้นฟันเสร็จคืออะไร
ไม่ต้องคิดเพื่อ เพราะ Requirement เปลี่ยนได้ตลอดเวลา

Simplicity

- “What is the simplest thing that will work?”
- Do what is needed and asked for, but no more
- Avoid waste and do only absolutely necessary things such as keep the design of the system as simple as possible so that it is easier to maintain, support, and revise.
- Address only the requirements that you know about; don’t try to predict the future



Feedback

- Demonstrate the committed software early and often then listen carefully and make any changes needed
- Gather feedback on your design and implementation, and then adjust your product going forward
- Teams can identify areas for improvement and revise their practices

“Tell us what
you think!”



กล้ายอมรับ Feedback ที่ได้รับมา ไม่ว่าดี หรือไม่ดี

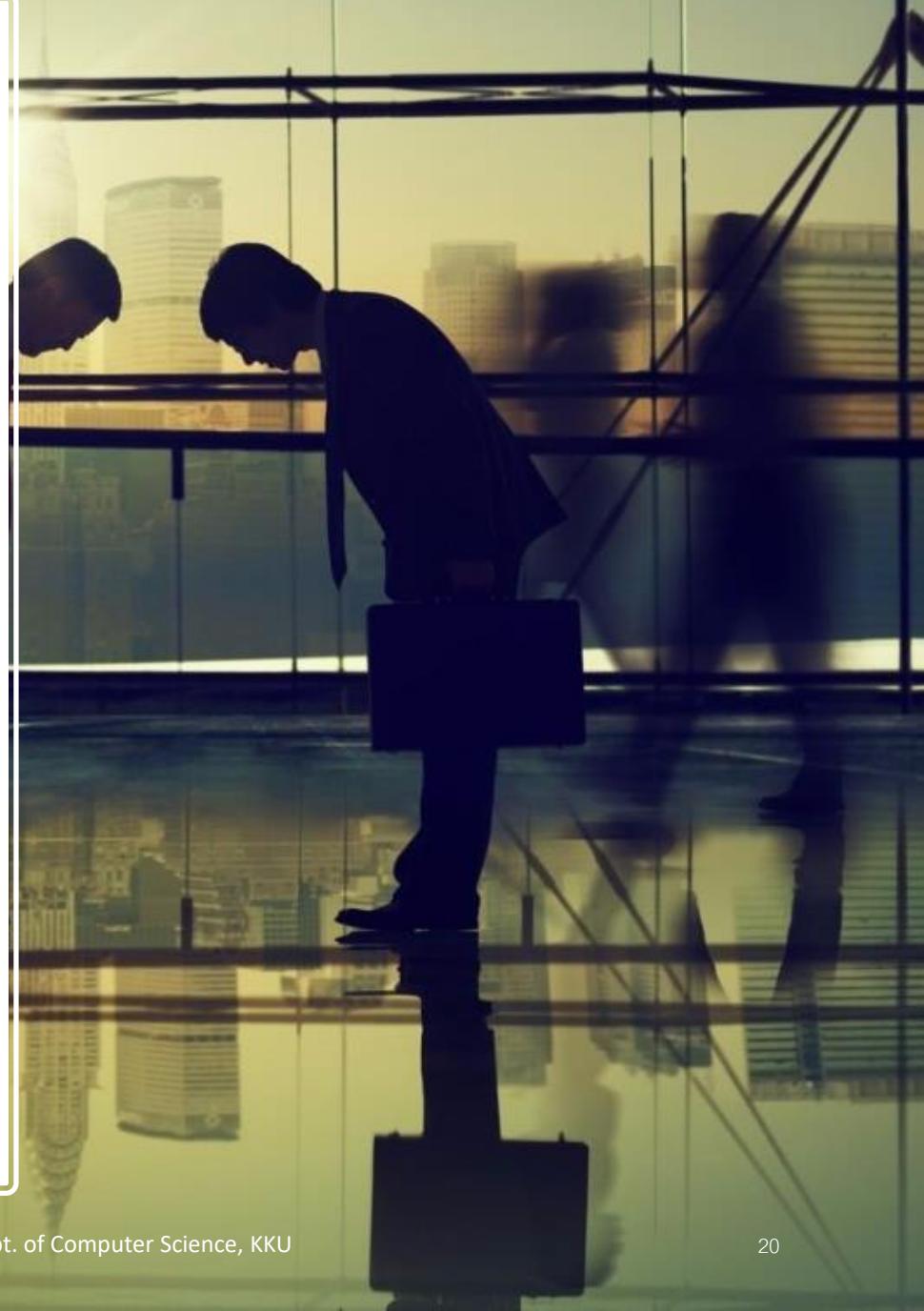
Courage

- “Effective action in the face of fear”
- Team needs courage to stop doing something that doesn’t work and try something else.
- Team needs courage to accept and act on feedback, even when it’s difficult to accept.
- Team will adapt to changes when ever they happen.



Respect

- Everyone gives and feels the respect they deserve as a valued team member.
- The members of your team need to respect each other in order to
 - communicate with each other,
 - provide and accept feedback that honors your relationship, and
 - to work together to identify simple designs and solutions.



XP Principles

- Humanity
- Economics (Risks)
- Mutual Benefit
- Self Similarity
- Improvement
รับ feedback ปรับปรุงอยู่เสมอ
- Diversity
- Reflection
- Flow
- Opportunity
บางอย่างควรทำข้ามอย่าง เพื่อให้ได้ประสิทธิภาพ เช่น การแทนบอยๆ
- Redundancy
- Failure
นำส่ง Software ที่มีคุณภาพเสมอ // จะไม่เขียน โค้ดคนเดียว // pair programming
- Quality
- Baby steps
- Accepted Responsibility

ไม่มี Tester, Programmer ถ้าหยิบไปทำคือต้องทำทุกส่วน Design , implement , test เอง

XP Practices: Primary Practices

ควรทำหลักนี้ให้ชินก่อน

นั่งทำงานร่วมกัน

- **Sit together**

ไม่มีใครแยกแยกออกจากทีม

- **Whole team**

- **Informative Workspace**

การแบ่งเวลาทำงาน พัก ให้

- **Energized work**

- **Pair programming**

- **Pairing and Personal Space**

- **(User) Stories**

ให้เขียน Requirement ทุกอย่าง ให้อยู่ในรูป User story

วันแรกของงาน จะต้องทำงานกับลูกค้า ว่า ลูกค้าอยากรื้อแบบไหนตอนลิ้นลับดา�์

- **Weekly cycle**

Release Planning // วางแผน Release Planning ทุกไตรมาส

- **Quarterly cycle**

- **Slack**

- **Ten-minute build**

- **Continuous integration**

- **Test-first programming**

- **Incremental design**

ไม่ต้อง Design เพื่อ เอาแค่ตามลูกค้าบอก

XP Practices: Corollary Practices

- Real customer involvement
- Incremental deployment

ถ้าทีมนั้น โวเคล้า ไม่ต้องฟอร์มใหม่ ให้ทีมนั้นรับผิดชอบงานต่อไปเรื่อย

- Team continuity

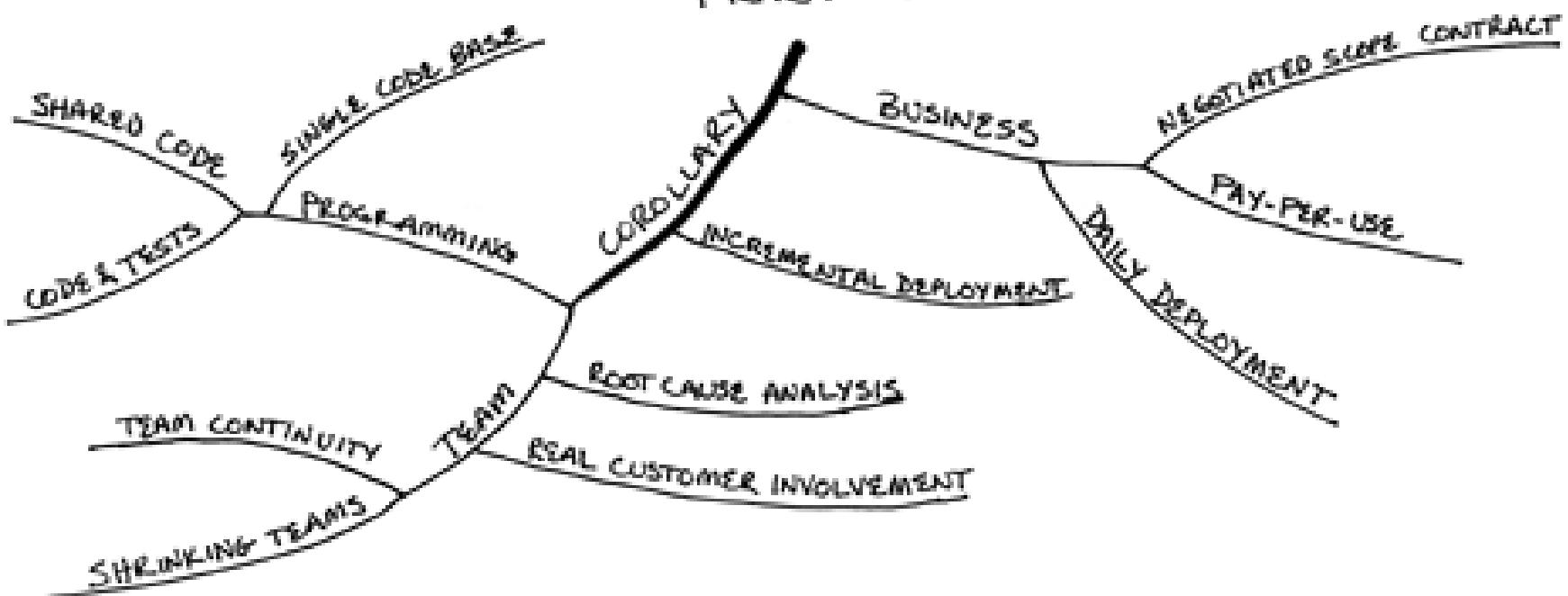
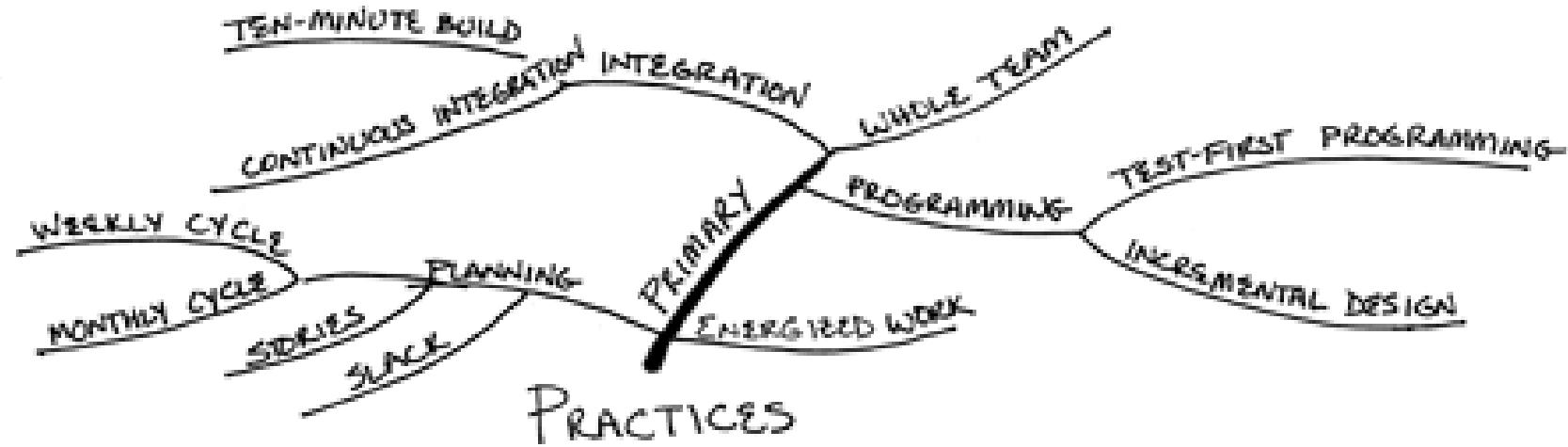
ถ้าเริ่มคล่องแล้ว ให้ลดขนาดทีมลง แต่ให้ได้งานปริมาณเท่าเดิม

- Shrinking teams

เวลาเกิดปัญหา ให้วิเคราะห์หาต้นตอของปัญหา

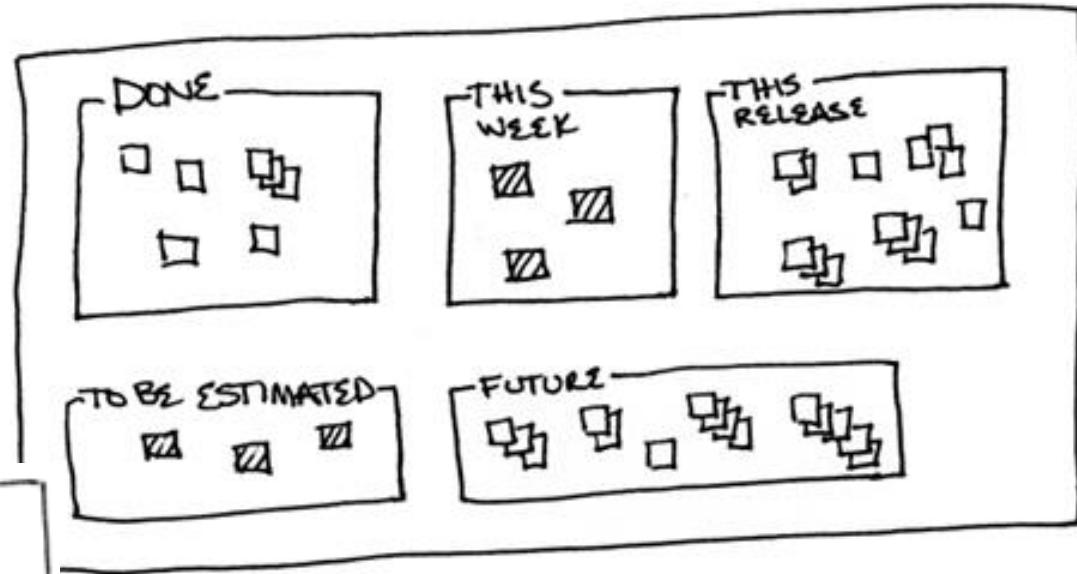
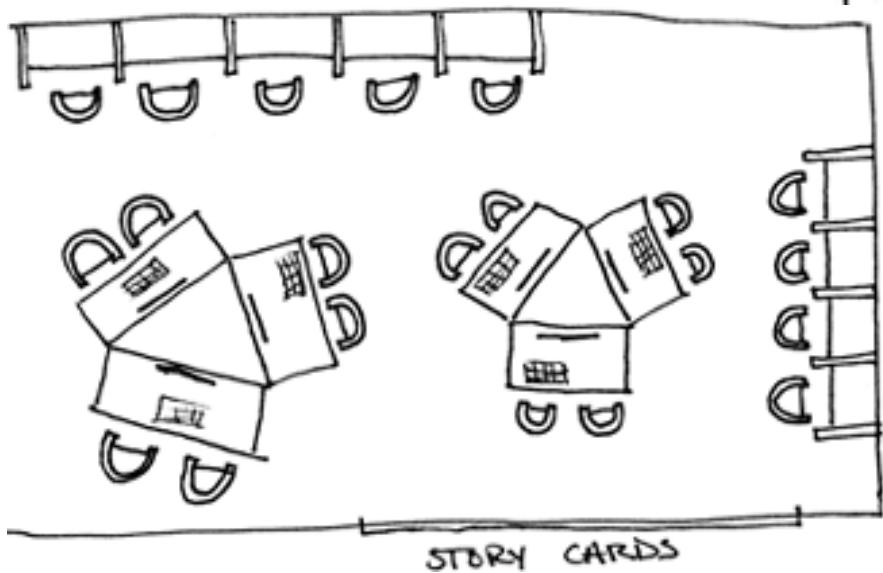
- Root-cause analysis

- Shared code
เก็บโค้ดและ tes ไปด้วยกัน
- Code and tests
- Single code base
กำหนดเวลาในการ Deploy.Code
- Daily deployment
Scope เวลาในการส่งงาน
- Negotiated scope contract
- Pay-per-use
จ่ายเงินเมื่อได้งานที่ต้องการ



Informative Workspace

Big and visible charts providing up-to-date progress



Workspace provides human needs, comfort, and encourage positive social interactions

Pair Programming



- Two people
- One computer
- Working on the same task



จะมี Navigator และ Pilot

- Navigator จะคิดอัลกอริทึมทำงานยังไง
- Pilot จะเขียนตามที่สั่ง ใช้ Syntax อะไร

User Story (Story Card)

SAVE WITH COMPRESSION

8 HRS

- CURRENTLY THE COMPRESSION OPTIONS ARE IN A DIALOG SUBSEQUENT TO THE SAVE DIALOG. MAKE THEM PART OF THE SAVE DIALOG ITSELF.

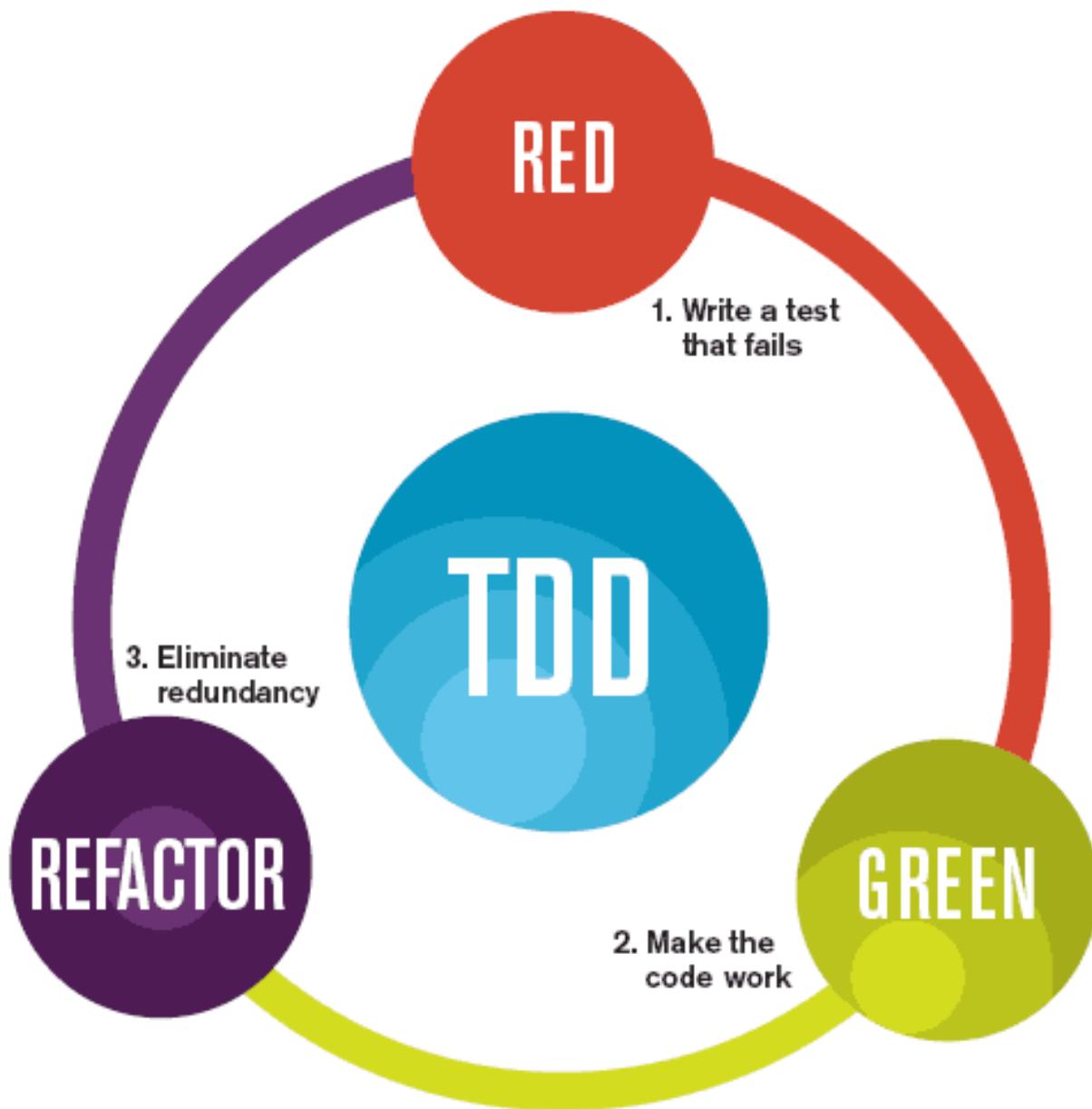
As a Game Player,
I want my Rocket to move back and forth when I press left and right arrows
so that I can avoid asteroids

Test-First Development

เขียนโค้ด และ tesl ไปเรื่อยๆ ทำแบบจากหน่วยเล็กที่สุดก่อน และเพิ่มไปเรื่อยๆ

- Also known as Test-Driven Development (TDD)
- Write a unit test before start writing a code
- Write a code that is enough to pass the test
- Verify the code against the test
- Continuously integrate the passed code together
- Can be performed together with pair programming

เขียน Unit test ก่อนเริ่มการเขียนโค้ด



The mantra of Test-Driven Development (TDD) is “red, green, refactor.”

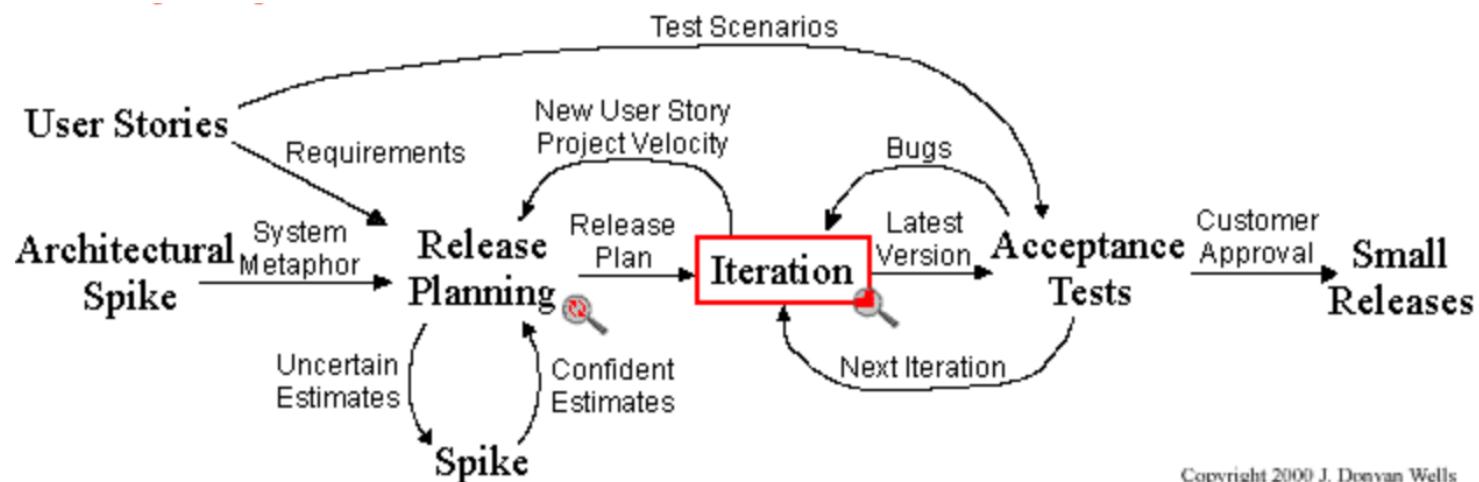
The Rules of Extreme Programming

วางแผน ต้องมีการเขียน User stories มีการทำ Release Planning จะ Release อะไรบ้าง ทำ Release เล็กๆ

- Planning
- Managing
- Designing
- Coding
- Testing

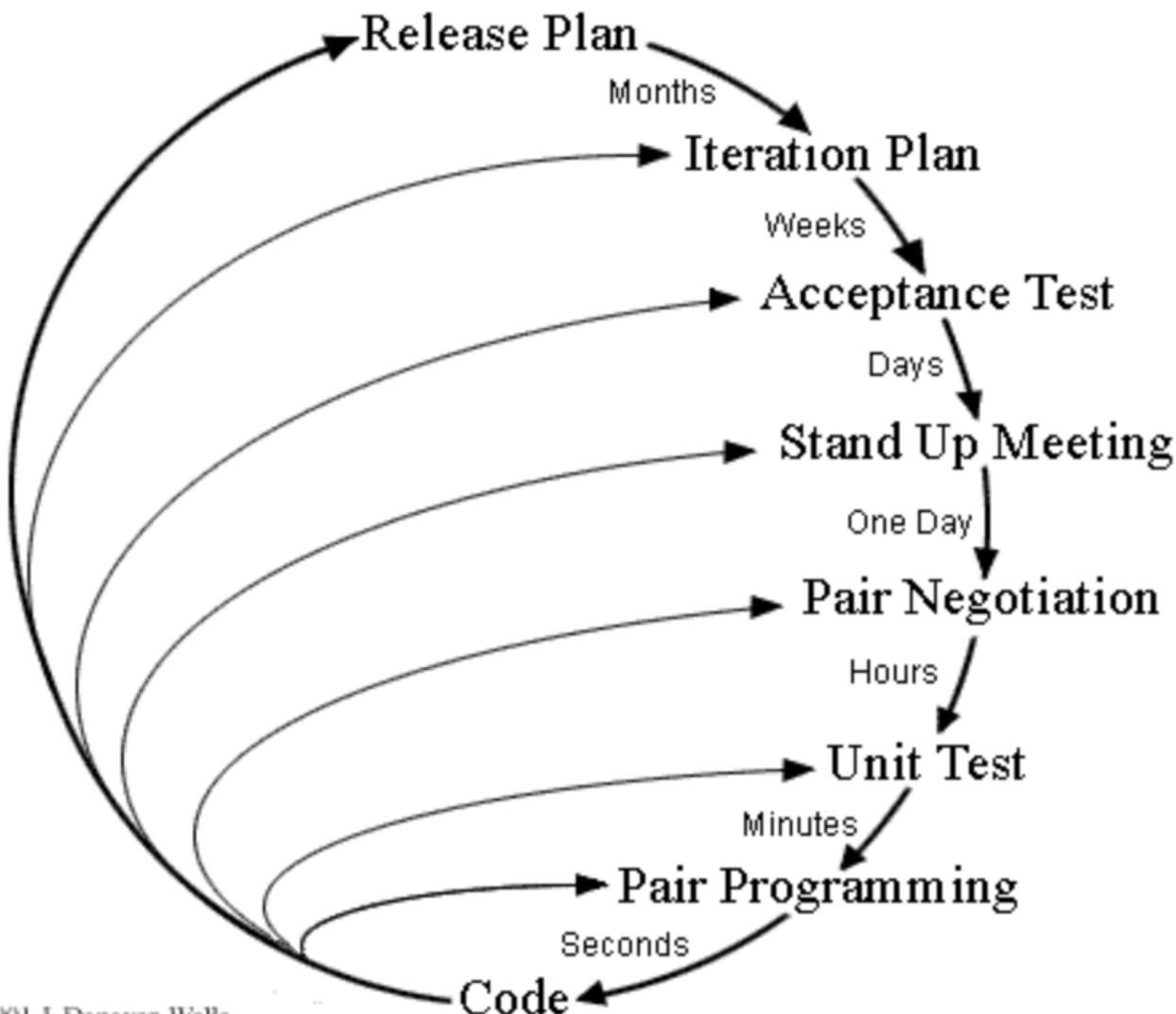
Planning

- User stories are written
- Release planning creates the release schedule
- Make frequent small releases
- The project is divided into iterations
- Iteration planning starts each iteration



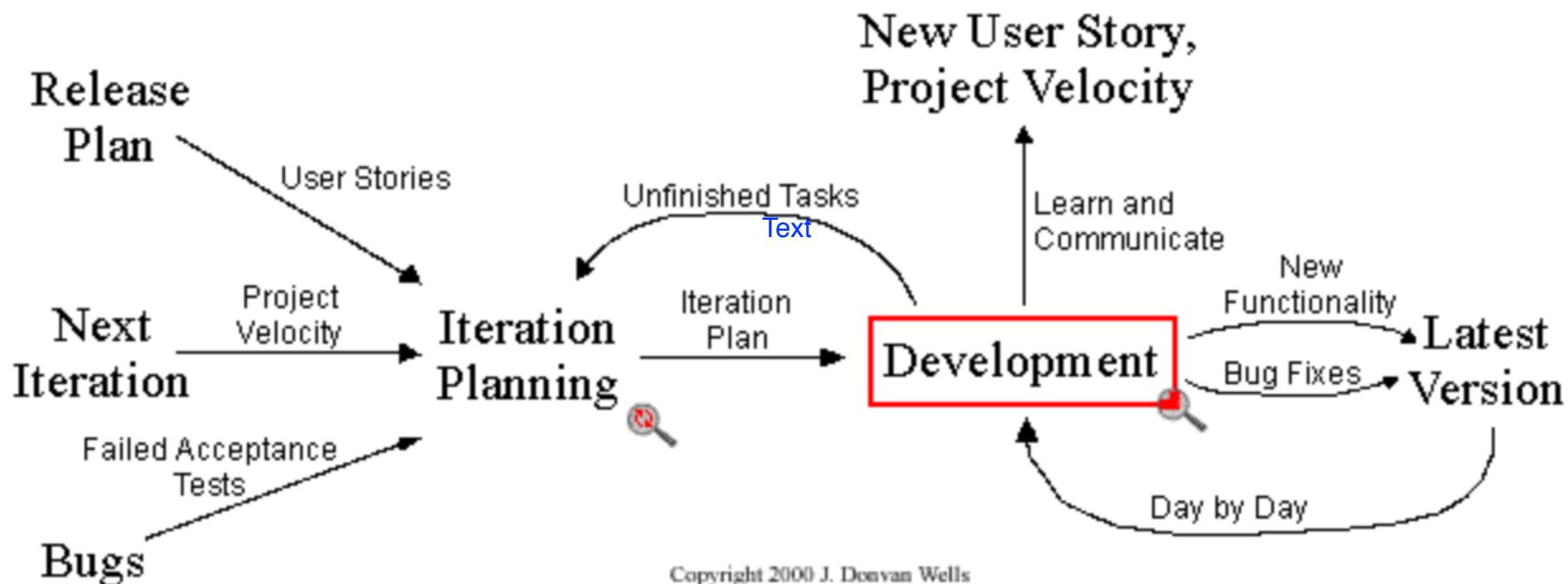
Copyright 2000 J. Donvan Wells

Planning/Feedback Loops



Copyright 2001 J. Donovan Wells

Iteration



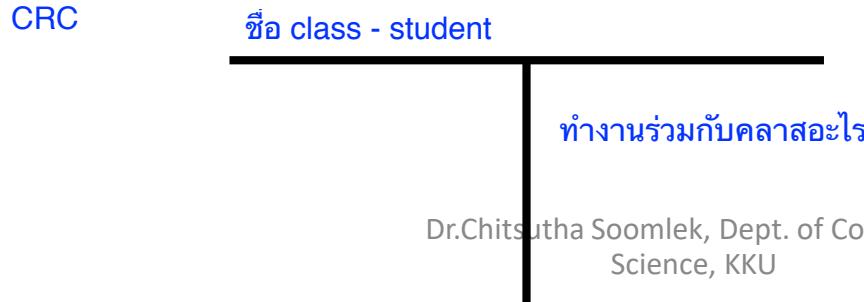
Managing

- Give the team a dedicated **open work space**
- Set a **sustainable pace**
มีพื้นที่ให้ Stand up meeting กันทุกวัน
- A **stand up meeting** starts each day
วัดอัตราเร็วในการผลิตซอฟต์แวร์ของทีม
- The **Project Velocity** is measured
- Move people around
- Fix XP when it breaks

ลองสร้าง Prototype มาก่อน

Designing

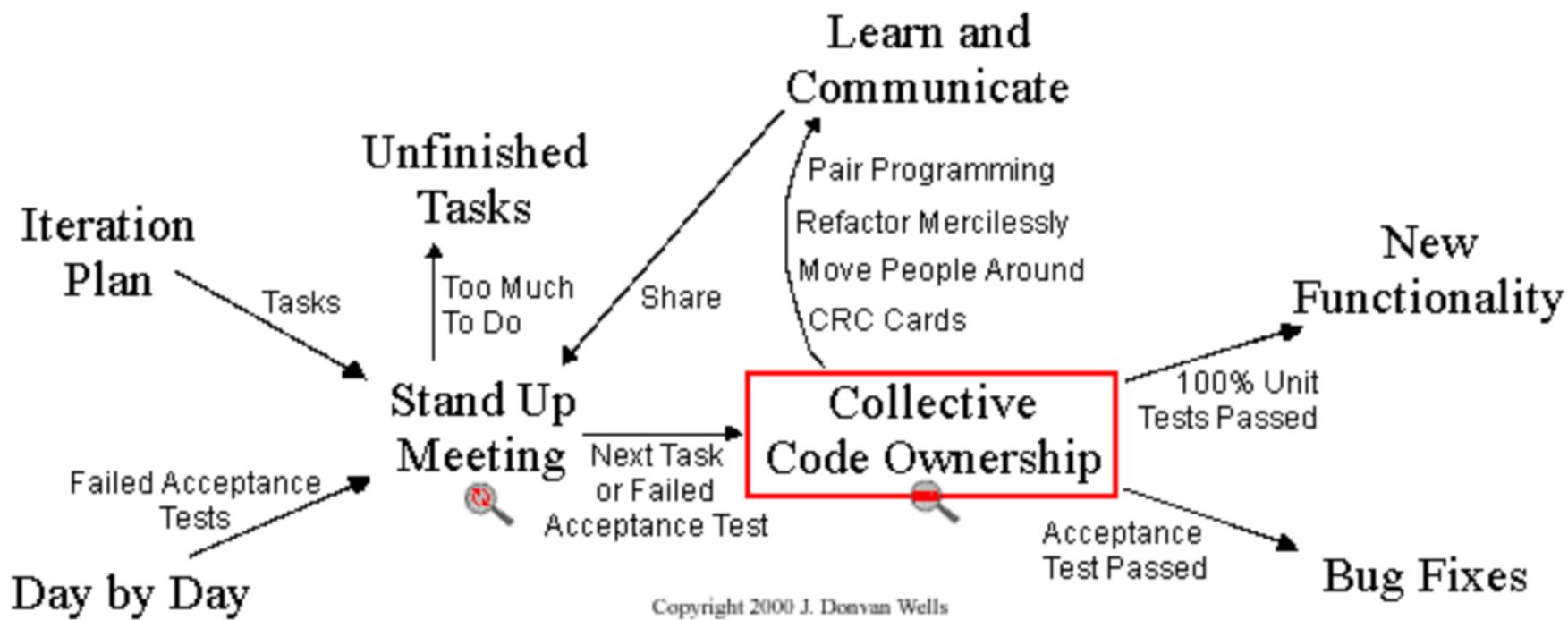
- Simplicity
- Choose a **system metaphor**
- Use **CRC cards** for design sessions
- Create **spike solutions** to reduce risk
- No functionality is **added early**
- **Refactor** whenever and wherever possible



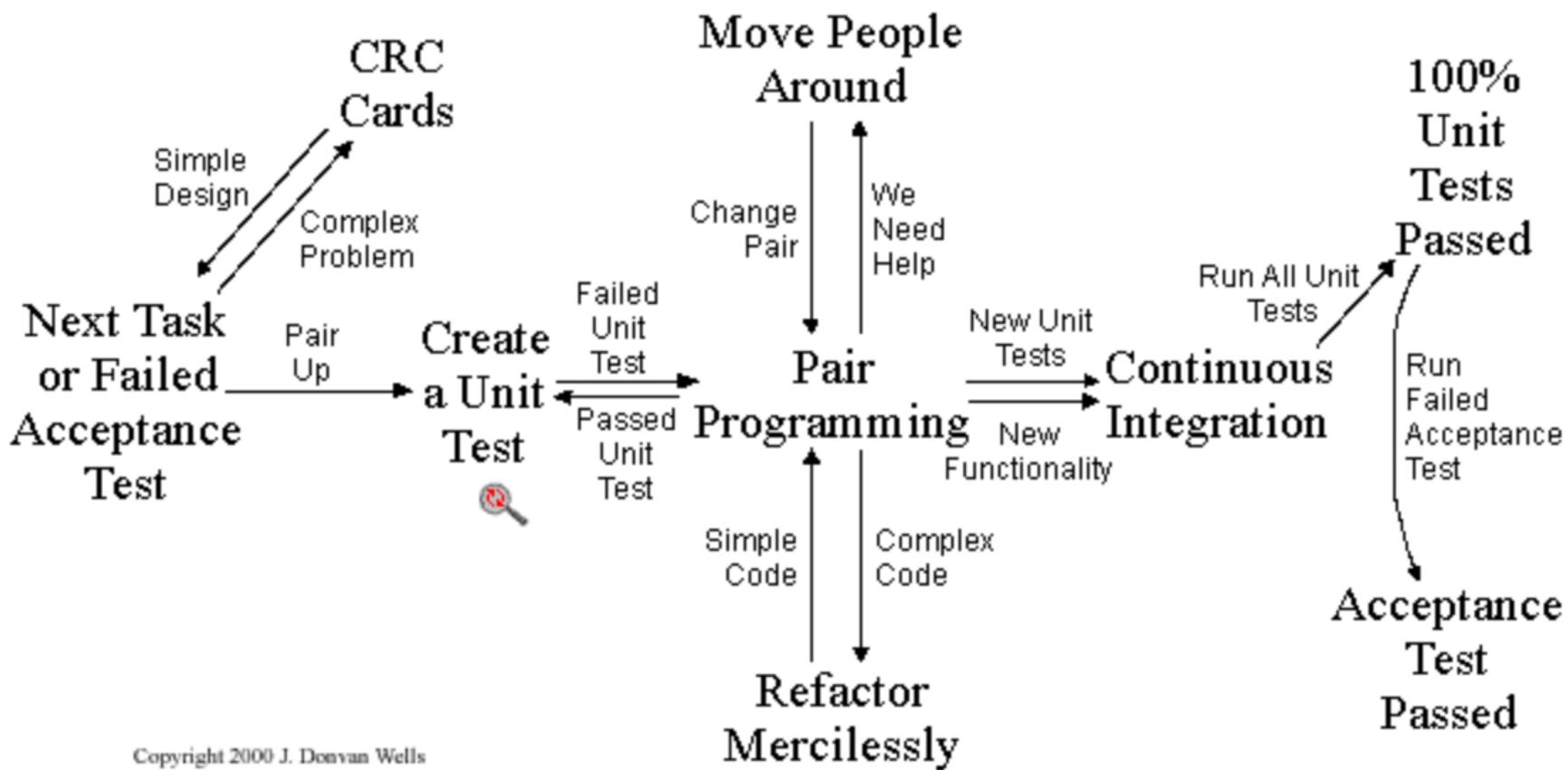
Coding

- The customer is **always available.**
กำหนดวิธีเขียนให้เป็นมาตรฐานเดียวกันก่อนเริ่มการเขียน
- Code must be written to **agreed standards.**
- Code the **unit test first.**
- All production code is **pair programmed.**
- Only one pair **integrates code at a time.**
- **Integrate often**
- Set up a dedicated **integration computer**
- Use **collective ownership**

Development



Collective Ownership



Testing

- All code must have **unit tests**
- All code must pass all **unit tests** before it can be released
- When **a bug is found** tests are created to guard against it coming back
- **Acceptance tests** are run often and the score is published

ทดสอบตามความต้องการของลูกค้า

Simplified Version of XP

- Ron Jeffries (2011)
- Practical sets of practices
 - The Planning Game
 - Small Releases
 - Metaphor
 - Simple Design
 - Testing
 - Refactoring
 - Pair Programming
 - Collective Ownership
 - Continuous Integration
 - 40-hour week
 - On-site Customer
 - Coding Standard
- Roles in XP
 - Customer
 - Developer
 - Tracker
 - Coach

Simplified Version of XP

