



# The Extreme Programming Model

Damian Gordon

# Contents

1. Overview
2. Details
3. Advantages
4. Disadvantages
5. Interesting
6. Reflection
7. Review
8. Summary



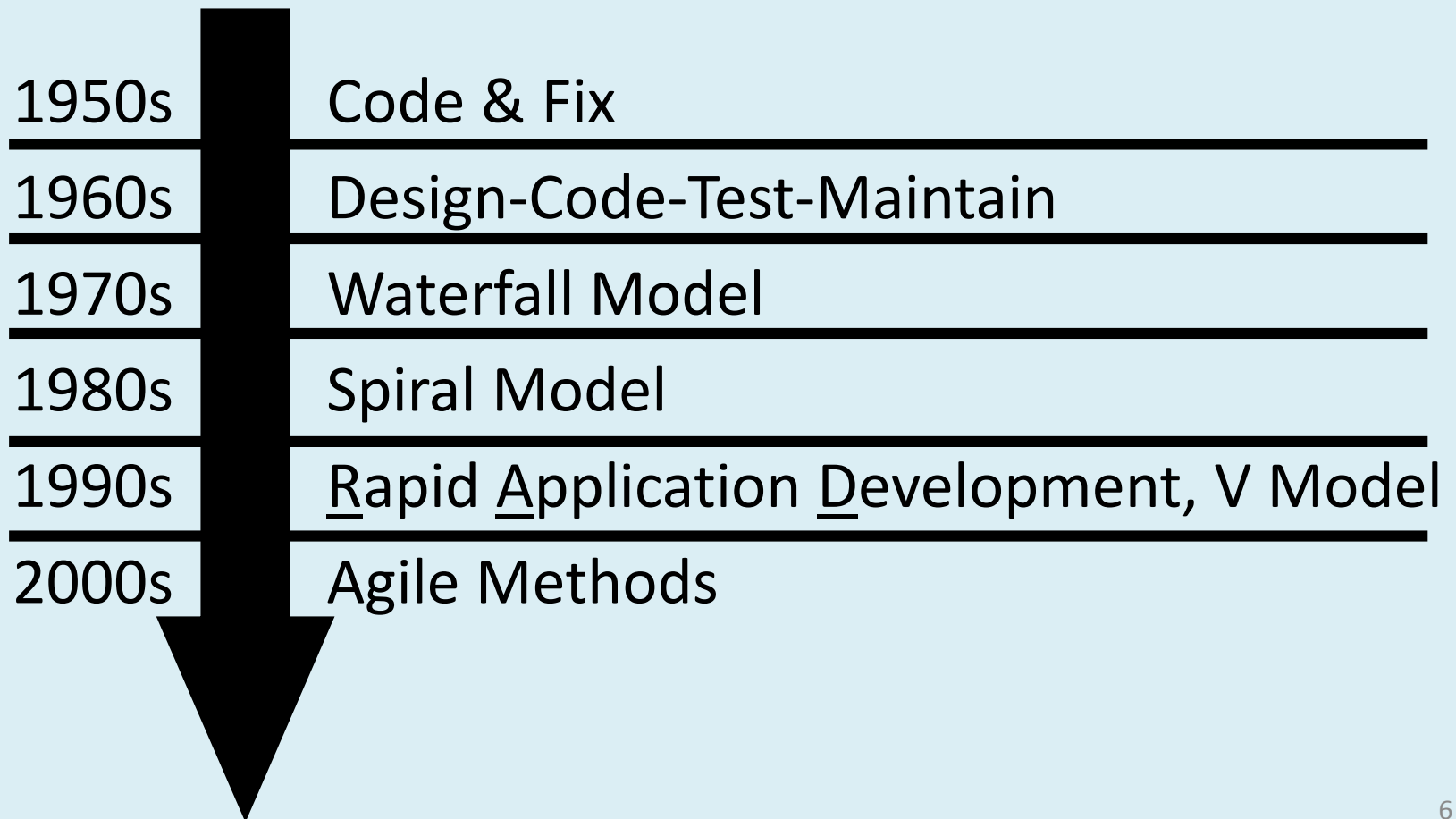


# 1. Overview

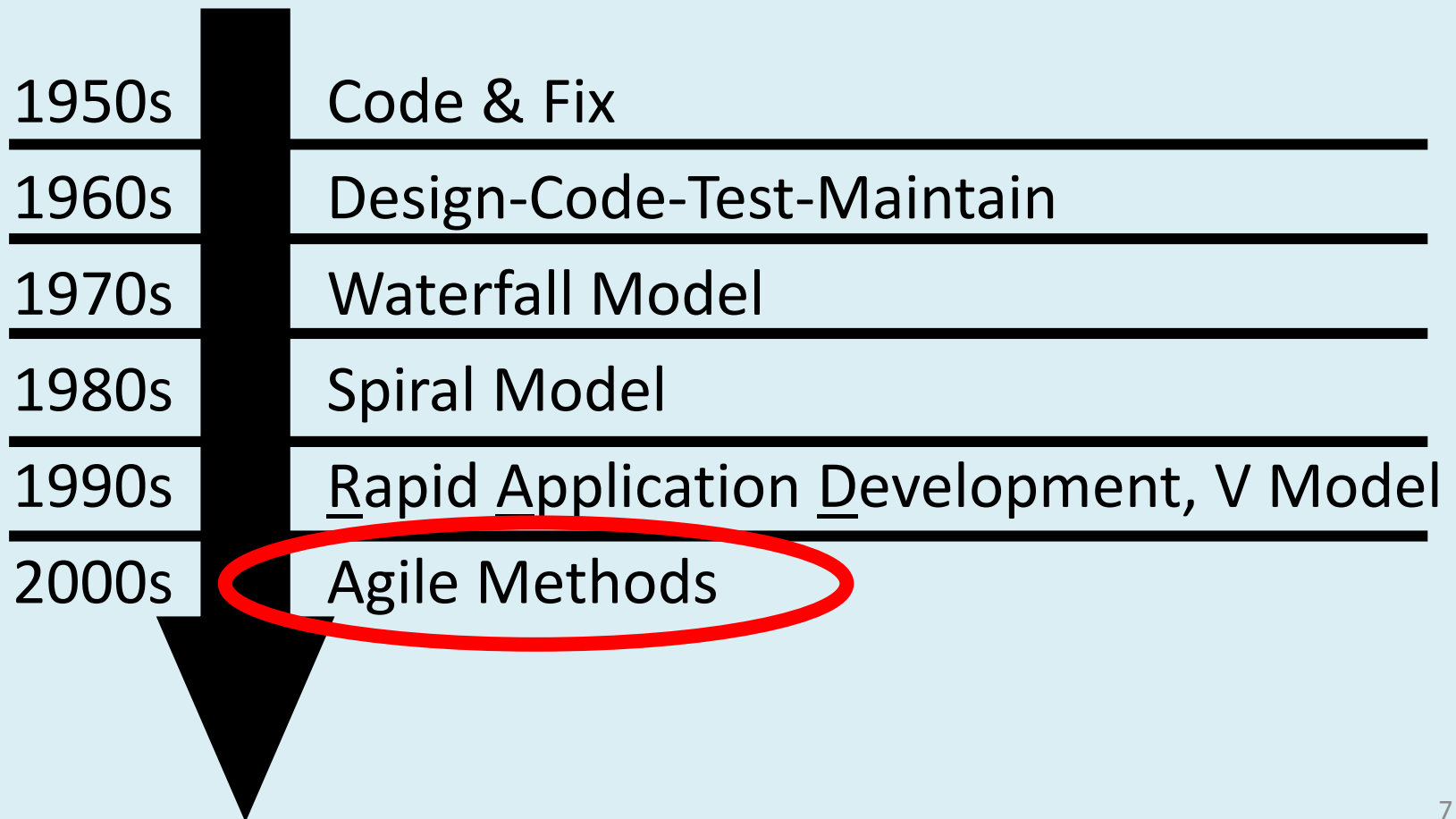
# Overview

- The “Extreme Programming (XP) Model” is a model that represents one method as to how software can be developed.

# Timeline of Methodologies



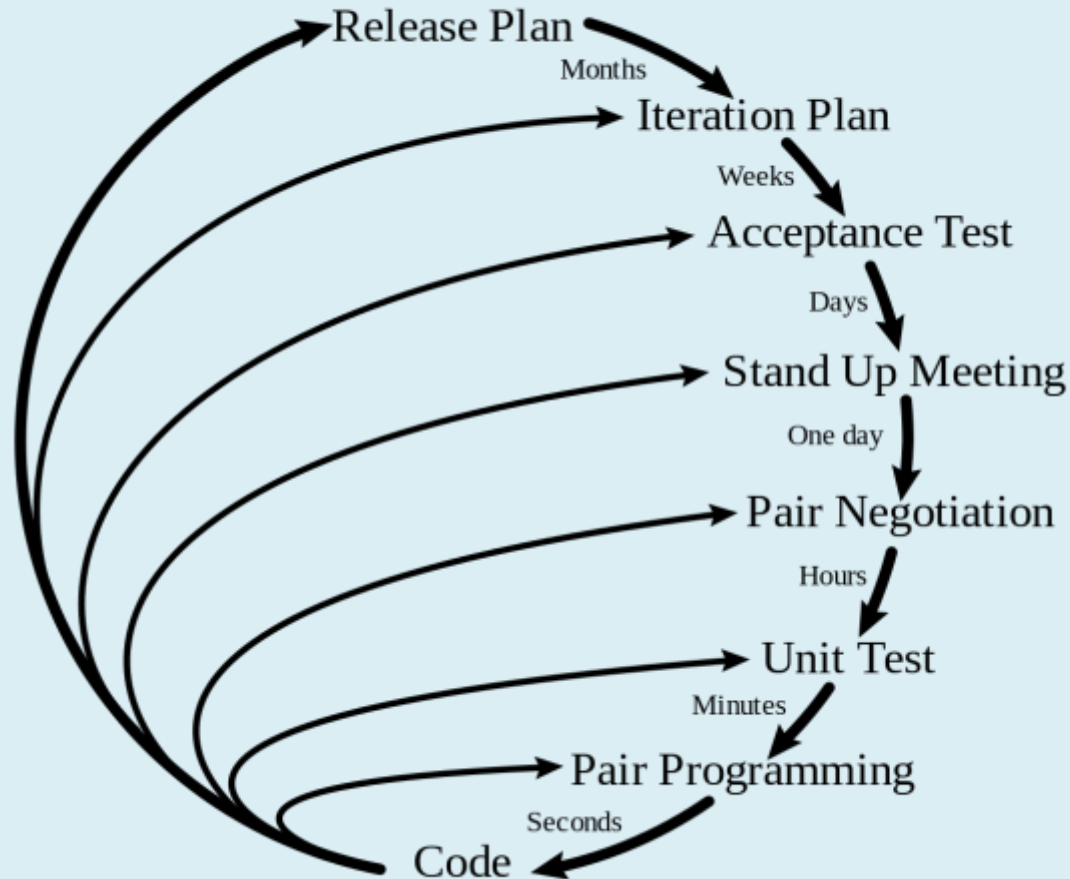
# Timeline of Methodologies





# Overview

## Planning/Feedback Loops





# Overview

- Extreme Programming (XP) was created by Kent Beck during his work on the C3 project.
- Beck became the C3 project leader in 1996 and began to refine the development methodology used in the project
- He wrote a book on the methodology, published in October 1999, called *Extreme Programming Explained*.

# Reference

- Beck, K. (1999) *“Extreme Programming Explained: Embrace Change”*. Addison-Wesley, ISBN 978-0321278654.

# Kent Beck

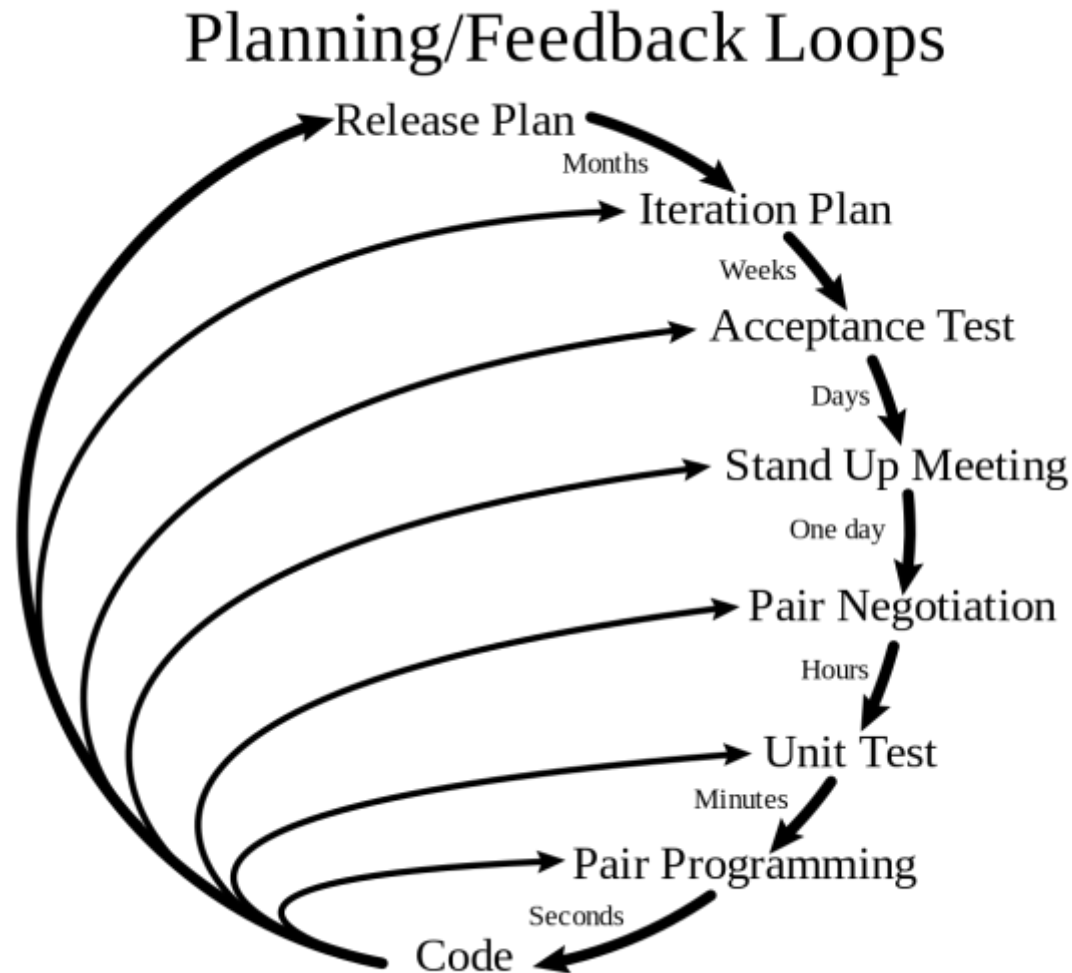
- Born in 1961.
- an American software engineer and the creator of Extreme Programming
- was one of the 17 original signatories of the Agile Manifesto
- the leading proponent of Test-Driven Development





## 2. Details

# Extreme Programming (XP)

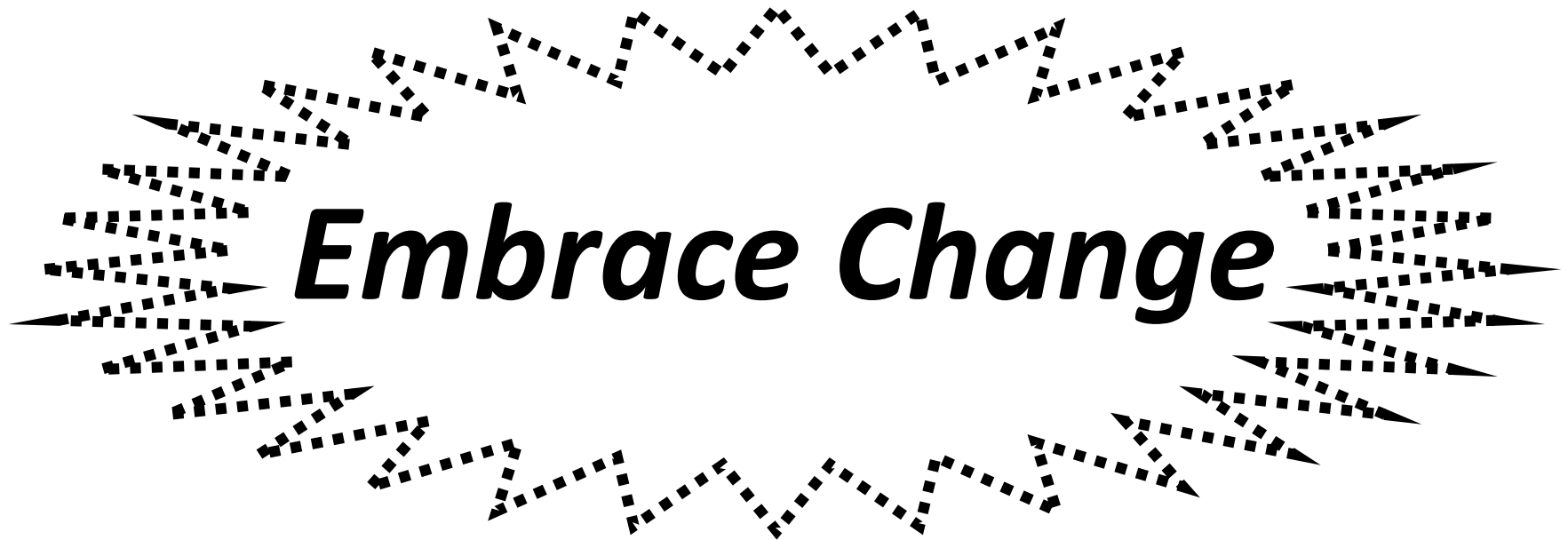


# Extreme Programming (XP)

- The key assumption of XP is:

# Extreme Programming (XP)

- The key assumption of XP is:





# Extreme Programming (XP)

- Start with the *Planning Game*:
- The game is a meeting that occurs once per iteration, typically once a week. The planning process is divided into two parts:
  - *Release Planning*: This is focused on determining what requirements are included in which near-term releases, and when they should be delivered. The customers and developers are both part of this.
  - *Iteration Planning*: This plans the activities and tasks of the developers. In this process the customer is not involved.

# Extreme Programming (XP)

- This is done by:
  - Emphasis on continuous feedback from the customer
  - Short iterations
  - Design and redesign
  - Coding and testing frequently
  - Eliminating defects early, thus reducing costs
  - Keeping the customer involved throughout the development
  - Delivering working product to the customer

# Extreme Programming (XP)

- **Management-Practices**

- **On-Site Customer:** A central customer contact must always be accessible in order to clarify requirements and questions directly.
- **Planning Game:** Projects, in accordance with XP, run iteratively (repeatedly) and incrementally (gradually build on each other). The contents of the next step are planned before each iteration. All project members (incl. the customer) participate.
- **Short Releases:** New deliveries should be made at short intervals. Consequently, customers receive the required functions quicker and can therefore give feedback on the development quicker.

# 12 Practices of XP

# Extreme Programming (XP)

- **Management-Practices**

- **On-Site Customer:** A central customer contact must always be accessible in order to clarify requirements and questions directly.
- **Planning Game:** Projects, in accordance with XP, run iteratively (repeatedly) and incrementally (gradually build on each other). The contents of the next step are planned before each iteration. All project members (incl. the customer) participate.
- **Short Releases:** New deliveries should be made at short intervals. Consequently, customers receive the required functions quicker and can therefore give feedback on the development quicker.

# Extreme Programming (XP)

- **Team-Practices**

- **Metaphor:** Only a few clear metaphors should describe the system being developed so that the nitty-gritty of the system is clear to all of the project members.
- **Collective Ownership:** The whole team is responsible for the system, not individuals. Each developer must have access to all lines of code so that each developer is able to take over the task of another developer.
- **Continuous Integration:** All changes to the system are integrated promptly so that not too many dependencies between changes occur.
- **Coding Standards:** Regarding the common responsibility for the code, there should be a given common standard for writing the code.
- **Sustainable Pace:** XP builds on the creativity of the individual project members. This creativity cannot be achieved if the project team constantly works overtime. Overtime is to be avoided.

# Extreme Programming (XP)

- **Programming-Practices**
  - **Testing:** All developments must be tested.
  - **Simple Design:** The system should be designed as simply as possible so that it is easier to understand, modify and test.
  - **Refactoring:** As soon as it becomes necessary to alter the structure of the system, it should be implemented.
  - **Pair Programming:** There are always two developers sitting in front of a computer in order to increase the quality and transfer the knowledge better.



# Extreme Programming (XP)

- Project are broken down into 1-2 week iterations.
- If changes occur in the middle of an iteration, the team are capable of reacting to them, as long as the team hasn't started work on a particular feature.
- Extreme Programming teams work in a strict priority order. Features to be developed are prioritized by the customer.

# Extreme Programming (XP)

- Design
  - Writing unit tests before programming and keeping all of the tests running at all times. The unit tests are automated and eliminates defects early, thus reducing the costs.
  - Starting with a simple design just enough to code the features at hand and redesigning when required.

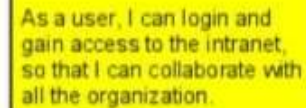
# Extreme Programming (XP)

- Design: User Stories

## The 3 C's

### 1. Card

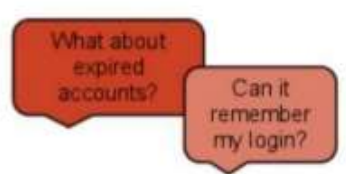
Written on a **card**



As a user, I can login and gain access to the intranet, so that I can collaborate with all the organization.

### 2. Conversation

Details captured in **conversations**

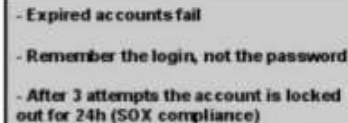


What about expired accounts?

Can it remember my login?

### 3. Confirmation

Acceptance criteria **confirm** that the story is Done.



- Expired accounts fail
- Remember the login, not the password
- After 3 attempts the account is locked out for 24h (SOX compliance)

*Source: XP Magazine 8/30/01, Ron Jeffries*

# Extreme Programming (XP)

- Development
  - Programming in pairs (called pair programming), with two programmers at one screen, taking turns to use the keyboard. While one of them is at the keyboard, the other constantly reviews and provides inputs.
  - Integrating and testing the whole system several times a day.

# Extreme Programming (XP)



## Pair Programming

- two programmers work together at one workstation.
- One, the ***driver***, writes code while the other, the ***observer*** or ***navigator***, reviews each line of code as it is typed in.
- The two programmers switch roles frequently.

# Extreme Programming (XP)



## Pair Programming

- While reviewing, the observer also considers the "strategic" direction of the work, coming up with ideas for improvements and likely future problems to address.
- This frees the driver to focus all of their attention on the "tactical" aspects of completing the current task, using the observer as a safety net and guide.

# Extreme Programming (XP)



## Pair Programming

- Pair programming increases the man-hours required to deliver code compared to programmers working individually from up to between 15% and 100%.
- However, the resulting code has about 15% fewer defects.



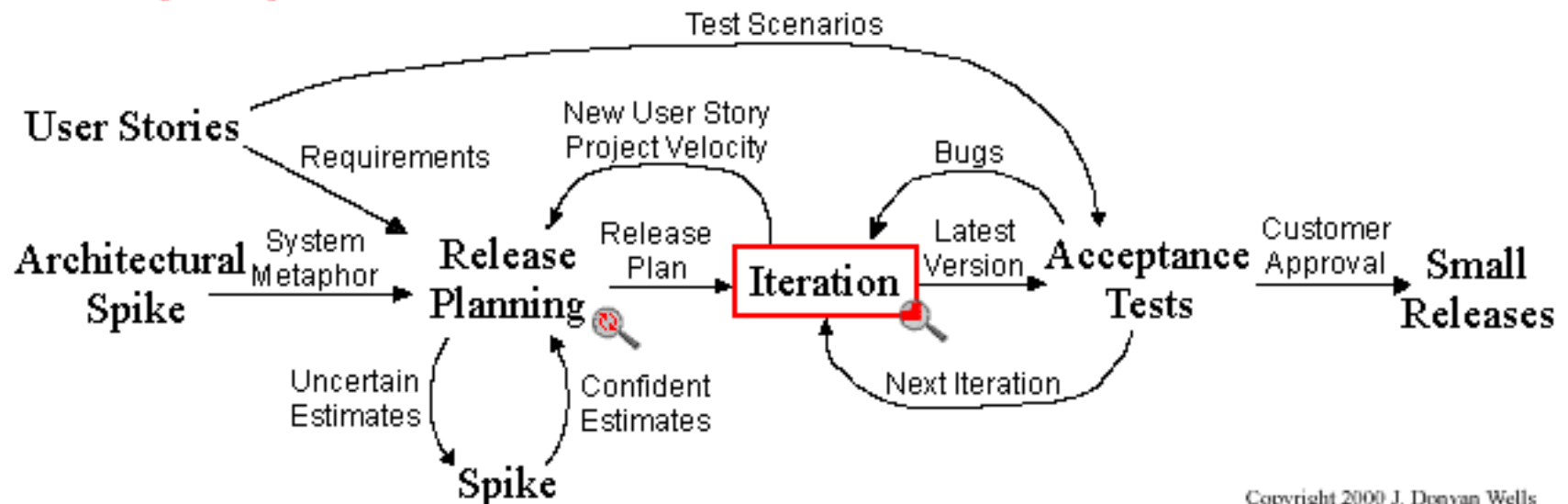
# Extreme Programming (XP)

- Production
  - Putting a minimal working system into the production quickly and upgrading it whenever required.
  - Keeping the customer involved all the time and obtaining constant feedback.

# Extreme Programming (XP)



## Extreme Programming Project





### 3. Advantages

# Advantages

- Large project are divided into manageable amounts

# Advantages

- Reduced costs and time required for project realization.

# Advantages

- Extreme Programming teams save lots of money because they don't use too much documentation.

# Advantages

- Simplicity is another advantage of Extreme Programming projects.



# Advantages

- XP reduces the risks related to programming. Normally programming depends a lot on individuals key team members. In XP, Using module structure, and pair programming XP spreads the risk and mitigate the dependence on individuals

# Advantages

- In software simplicity always brings quality and contributes the robustness. By bringing simplicity, XP manages to create faster software with less defects.

Test driven development at the coding stage and the customer UAT validation leads to successful development completion.



## 4. Disadvantages

# Disadvantages

- Extreme Programming is focused on the code rather than on design.

# Disadvantages

- XP requires a detailed planning from the start due to changing costs & scope

# Disadvantages

- XP doesn't measure/plan Quality Assurance of coding.

# Disadvantages

- To convince developers to accept this tough practice is not always easy. It requires more discipline in the team and devotion of your customers. Project management might experience difficulties related with the practice that changes during the life cycle.

# Disadvantages

- XP is practiced with pair programming which might usually lead to too much duplication of codes and data.





## 5. Interesting

# Interesting

- XP engineering practices include things like test-driven development, automated testing, pair programming, simple design, and refactoring.

# Interesting

- XP says that the only truly important product of the system development process is code.

# Interesting

- Coding can also help to communicate thoughts about programming problems. A programmer dealing with a complex programming problem, or finding it hard to explain the solution to fellow programmers, might code it in a simplified manner and use the code to demonstrate what she means.

# Interesting

- System-wide integration testing is encouraged, initially, as a weekly activity, for early detection of incompatible interfaces, to reconnect before the separate sections diverged widely from coherent functionality.

# Interesting

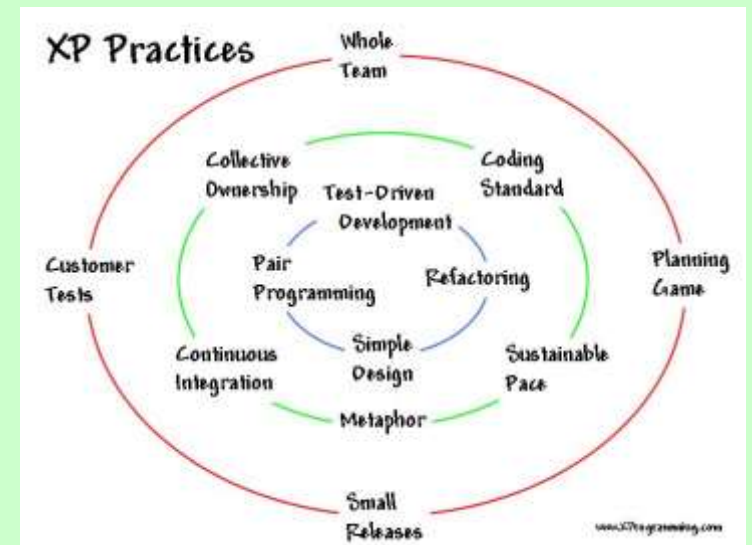
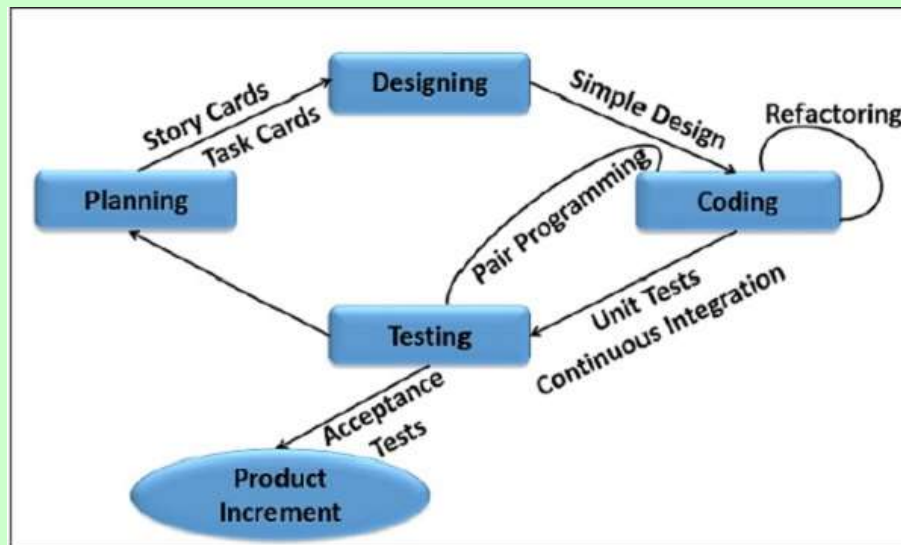
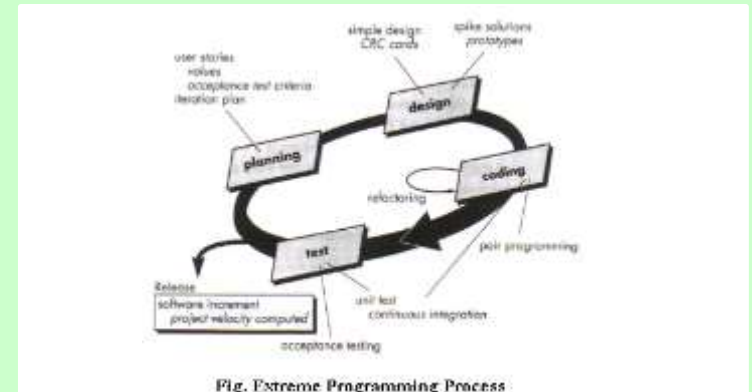
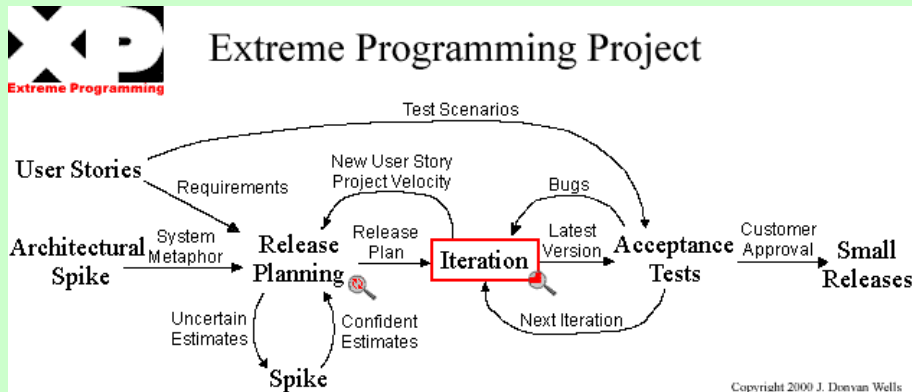
- Listening is a key element of XP, programmers must listen to what the customers need the system to do, what "business logic" is needed.

# Interesting

- XP is still evolving, assimilating more lessons from experiences in the field. In the second edition of *Extreme Programming Explained*, five years after the first edition, Beck added more values and practices and differentiated between primary and corollary practices.

# Interesting

There are other versions of the Model:







## 6. Reflections

# Reflections

- XP is very useful when we have:
  - Dynamically changing software requirements
  - Risks caused by fixed time projects using new technology
  - Small, co-located extended development team
  - The technology you are using allows for automated unit and functional tests

# Reflections

- XP sees communication as a vital aspect of development, and most people agree that face to face conversation is the best form of communication, so XP suggests that the team sit together in the same space without barriers to communication, such as cubicle walls.

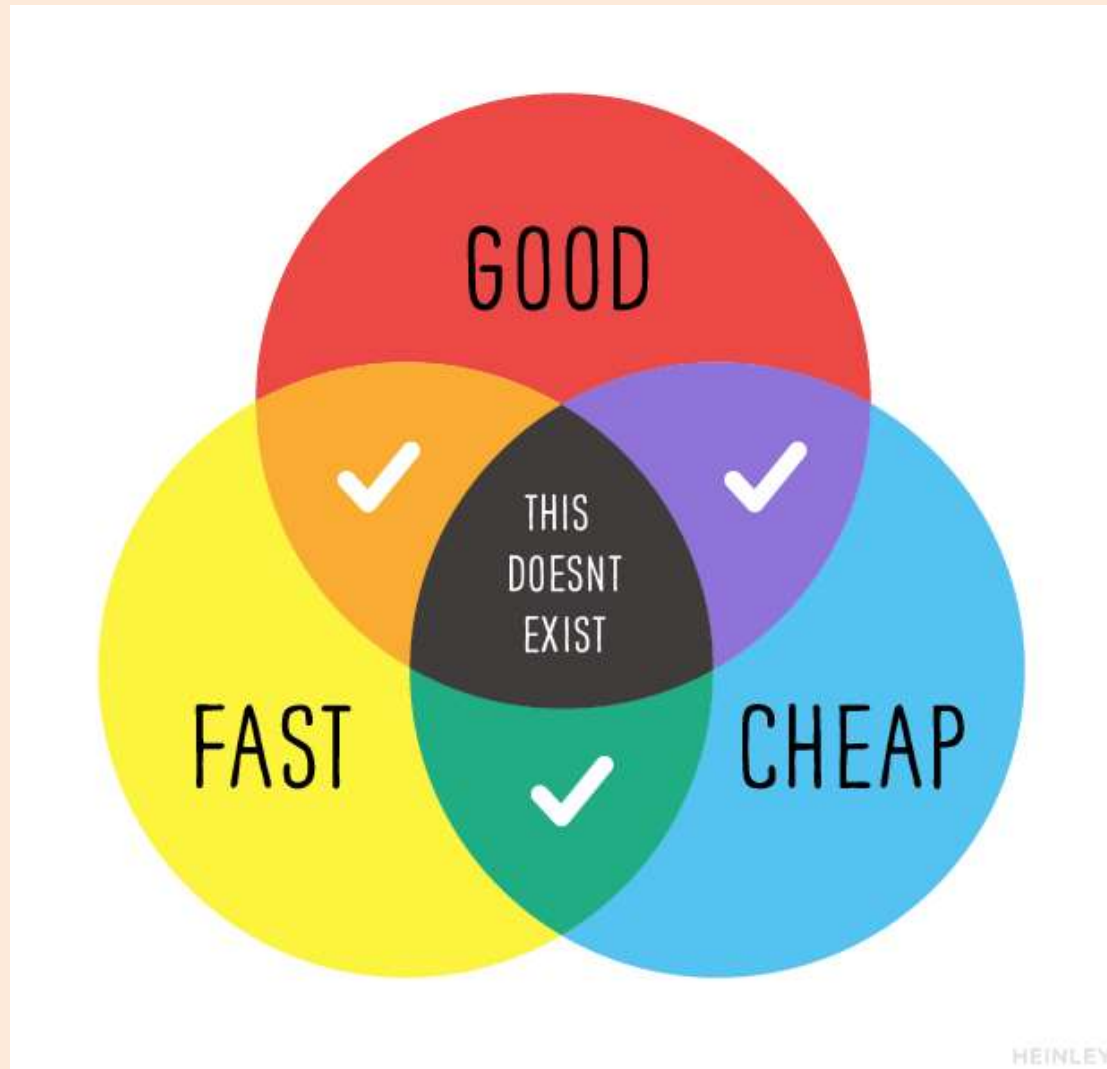
# Reflections

- XP tells you to build in some slack, the idea behind slack is to add some low priority tasks or stories in your weekly and quarterly cycles that can be dropped if the team gets behind on more important tasks or stories.

# Reflections

- XP focuses on practice excellence. The method prescribes a small number of absolutely essential practices and encourages teams to perform those practices as good as they possibly can, almost to the extreme.

# Reflections





## 7. Review

# Review

- What did we learn?

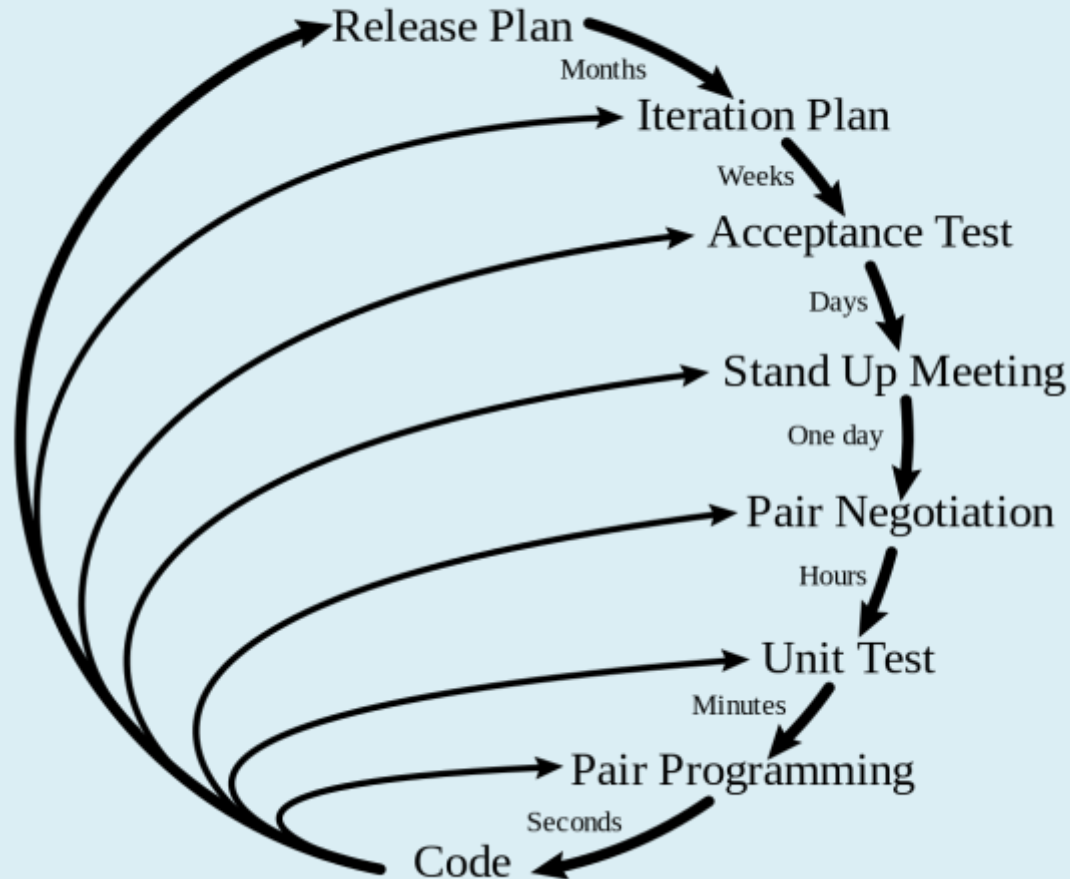




## 8. Summary

# Summary

## Planning/Feedback Loops





Windows is shutting down...