CSE 201: Introduction to Software Engineering

Hakam Alomari

Miami University Software Technology & Analysis Group (MUSTANG)
Computer Science & Software Engineering
Miami University, Oxford, Ohio, USA

AGENDA

- Science vs Engineering
- Software Development Processes
  - Waterfall Model
  - Incremental Development
- Agile Software Development

AGENDA

- Science vs Engineering
- Software Development Processes
  - Waterfall Model
  - Incremental Development
- Agile Software Development

SCIENCE VS ENGINEERING

- Science: “extends our knowledge of laws of nature”

- Engineers: “apply those laws of nature to build useful artifacts”

AGENDA

- Science vs Engineering
- Software Development Processes
  - Waterfall Model
  - Incremental Development
- Agile Software Development

Software Development Process

- "The process by which user needs are translated into a software product. The process involves translating user needs into software requirements, transforming the software requirements into design, implementing the design in code, testing the code, and sometimes, installing and checking out the software for operational use." [GLOSSARY]

CONTRASTING TWO PROCESSES

1. Waterfall Model
2. Incremental Development

- Predictive vs Adaptive

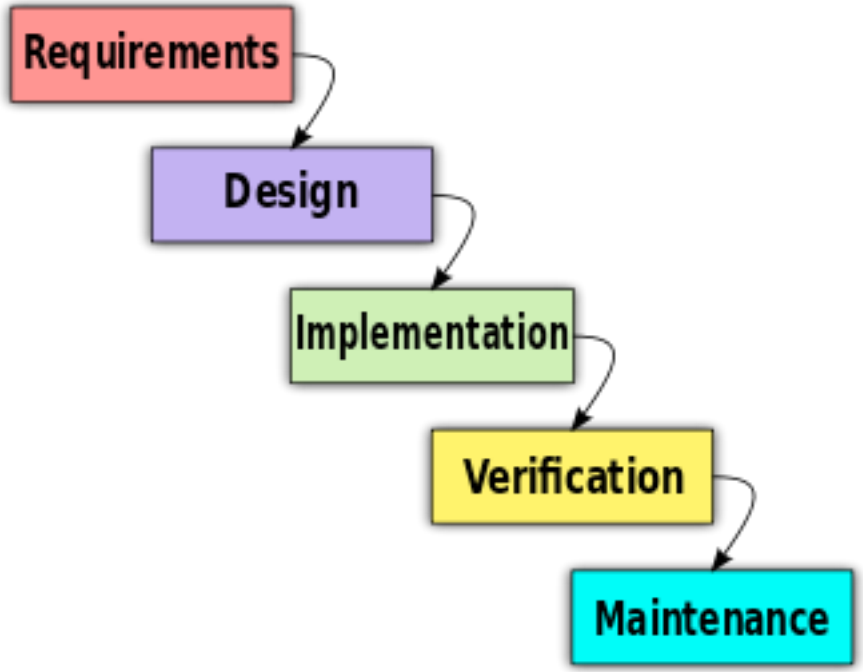
AGENDA

- Science vs Engineering
- Software Development Processes
  - Waterfall Model
  - Incremental Development
- Agile Software Development

WATERFALL MODEL

- "A model of the software development process in which the constituent activities, typically a concept phase, requirements phase, design phase, implementation phase, test phase, and installation and checkout phase, are performed in that order, possibly with overlap but with little or no iteration." [GLOSSARY]

WATERFALL MODEL



Criticisms of Waterfall

- "Many of the [system's] details only become known to us as we progress in the [system's] implementation. Some of the things that we learn invalidate our design and we must backtrack." - *David Parnas*
- On Average 45% of features in waterfall requirements are never used [1]
- Typical software projects experience 25% - 50% change in requirements [1]

- [1] pg18 of **Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition** (Read it on Safari Books Online http://bit.ly/W11KZc)

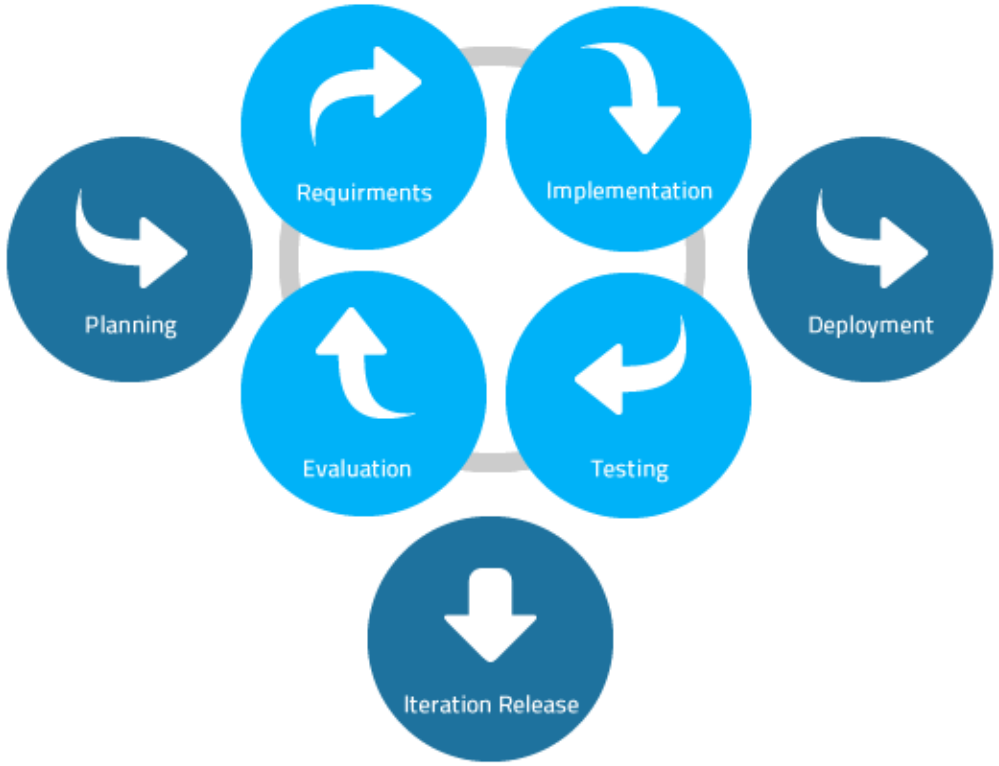
AGENDA

- Science vs Engineering
- Software Development Processes
  - Waterfall Model
  - Incremental Development
- Agile Software Development

INCREMENTAL DEVELOPMENT

- "A software development technique in which requirements definition, design, implementation, and testing occur in an overlapping, iterative (rather than sequential) manner, resulting in incremental completion of the overall software product." [GLOSSARY]

INCREMENTAL DEVELOPMENT



The Long, Dismal History of Software Project Failure

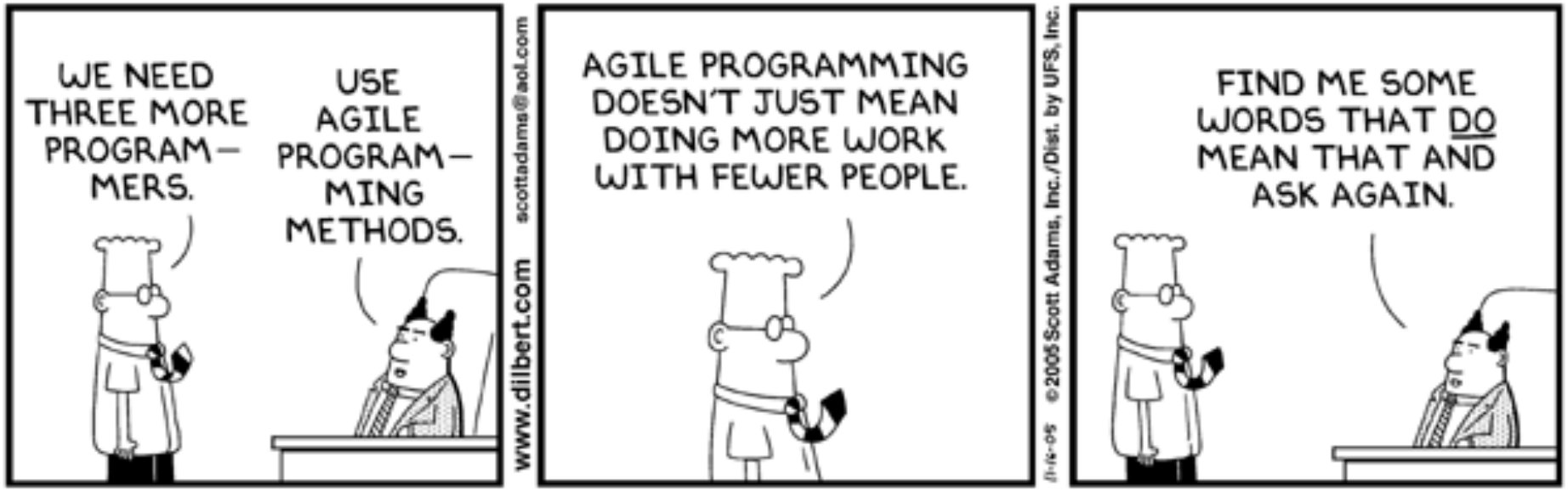
- From 1994 to 2004 the number of successful (on time and on budget) software projects improved by 100%:
  - From 16% to 32%
- "The primary reason is the projects have gotten a lot smaller. Doing projects with iterative processing as opposed to the waterfall method, which called for all project requirements to be defined up front, is a major step forward."
- Read the following article:
  - https://blog.codinghorror.com/the-long-dismal-history-of-software-project-failure/

AGENDA

- Science vs Engineering
- Software Development Processes
  - Waterfall Model
  - Incremental Development
- Agile Software Development

WHAT IS "AGILE" DEVELOPMENT

- Dictionary: "having a quick resourceful and adaptable character"
- Manifesto for Agile Software Development



MANIFESTO FOR AGILE SOFTWARE DEVELOPMENT

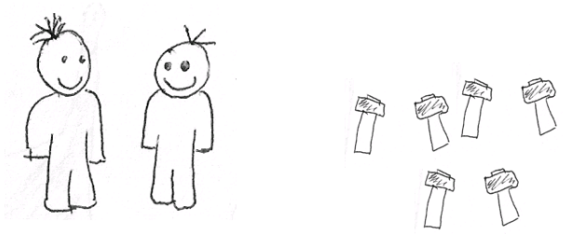
- We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

individuals and interactions	process and tools
working software	comprehensive documentation
customer collaboration	contract negotiation
responding to change	following a plan

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

MANIFESTO FOR AGILE SOFTWARE DEVELOPMENT

Agile Manifesto



Individuals and interactions over processes and tools

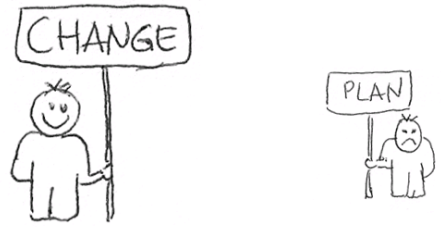
Agile Manifesto

Working software over comprehensive documentation

Agile Manifesto

Customer collaboration over contract negotiation

Agile Manifesto



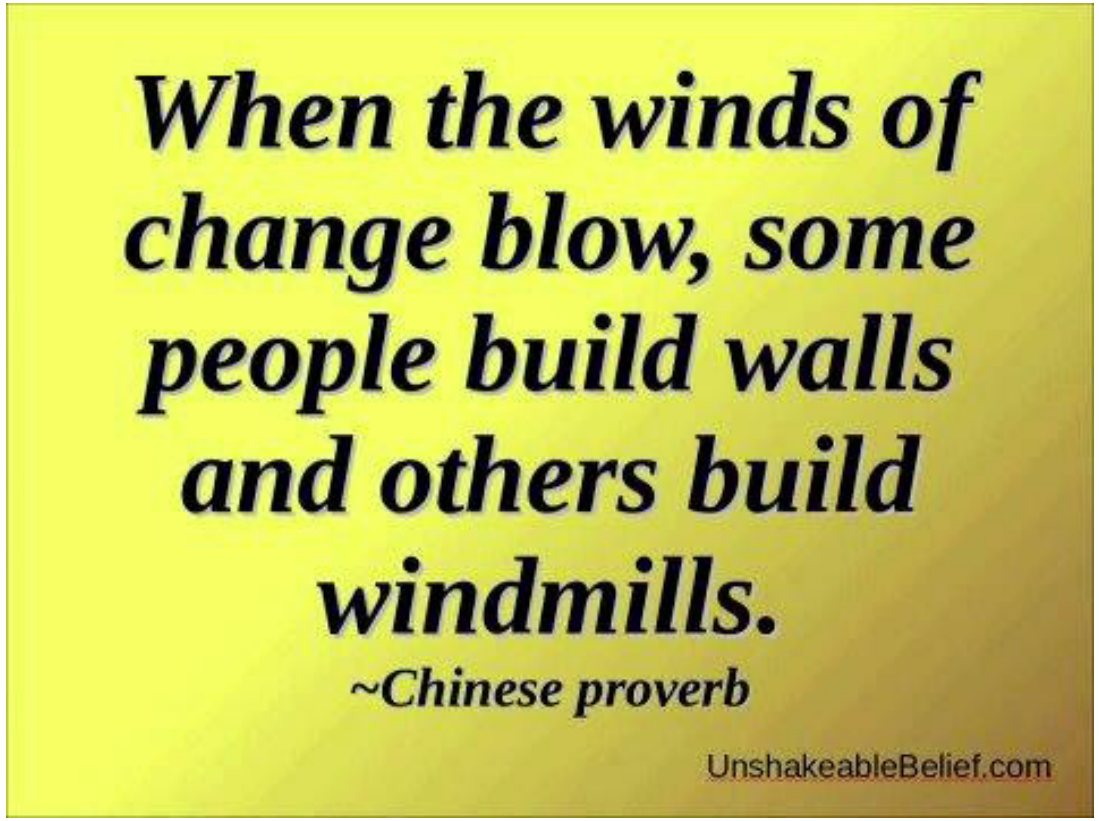
Responding to change over following a plan

PRINCIPLES BEHIND THE AGILE MANIFESTO

- "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."

PRINCIPLES BEHIND THE AGILE MANIFESTO

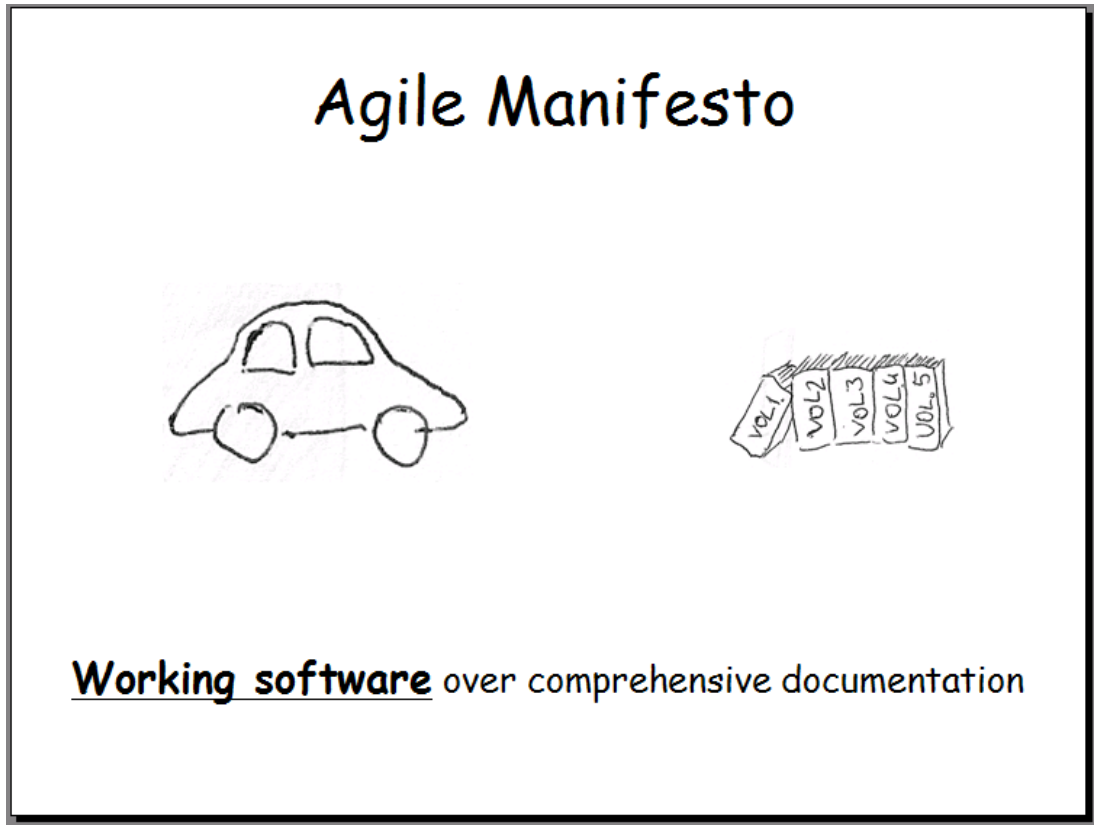
- "Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage."



PRINCIPLES BEHIND THE AGILE MANIFESTO

- "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale."

- "Working software is the primary measure of progress."

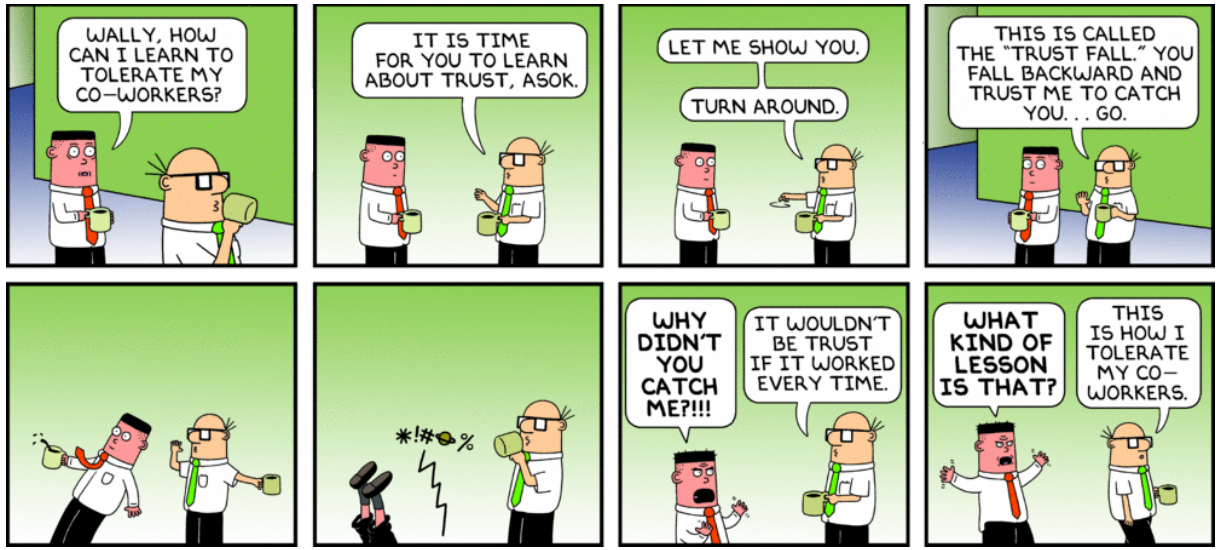


PRINCIPLES BEHIND THE AGILE MANIFESTO

- "Business people and developers must work together daily throughout the project."

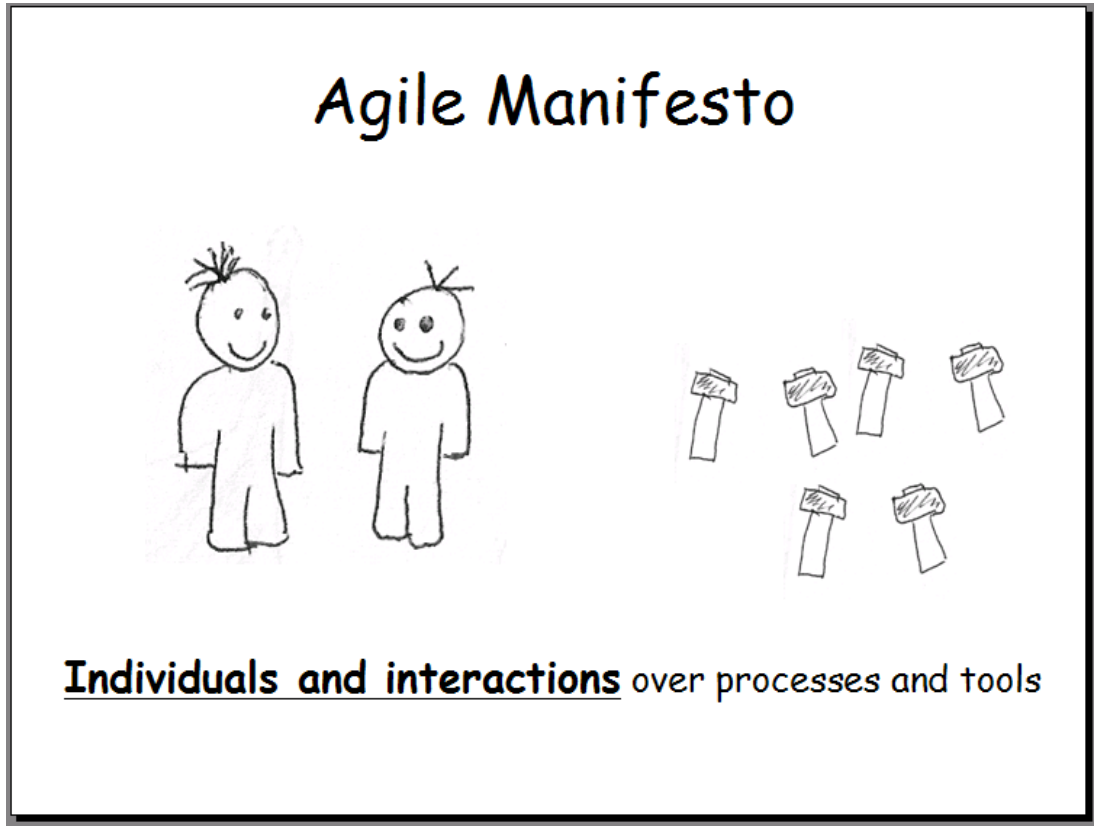
PRINCIPLES BEHIND THE AGILE MANIFESTO

- "Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done."



PRINCIPLES BEHIND THE AGILE MANIFESTO

- "The most efficient and effective method of conveying information to and within a development team is face-to-face conversation."



PRINCIPLES BEHIND THE AGILE MANIFESTO

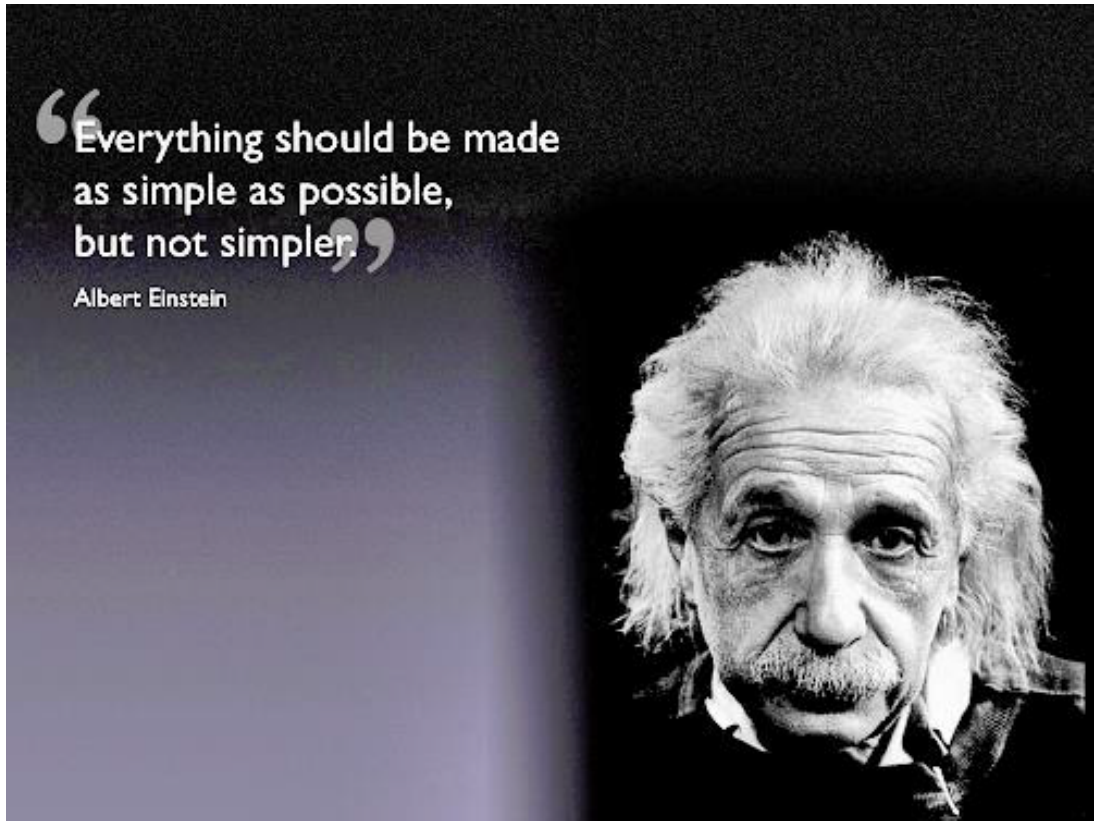
- "Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely."

PRINCIPLES BEHIND THE AGILE MANIFESTO

- "Continuous attention to technical excellence and good design enhances agility."

PRINCIPLES BEHIND THE AGILE MANIFESTO

- "Simplicity -- the art of maximizing the amount of work not done -- is essential."



PRINCIPLES BEHIND THE AGILE MANIFESTO

- "The best architectures, requirements, and designs emerge from self-organizing teams."

- "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly."

Agile Examples

- Scrum
- Extreme Programming (XP)
  - Pair programming
- Under Canvas >
  - Modules > Agile examples - Scrum