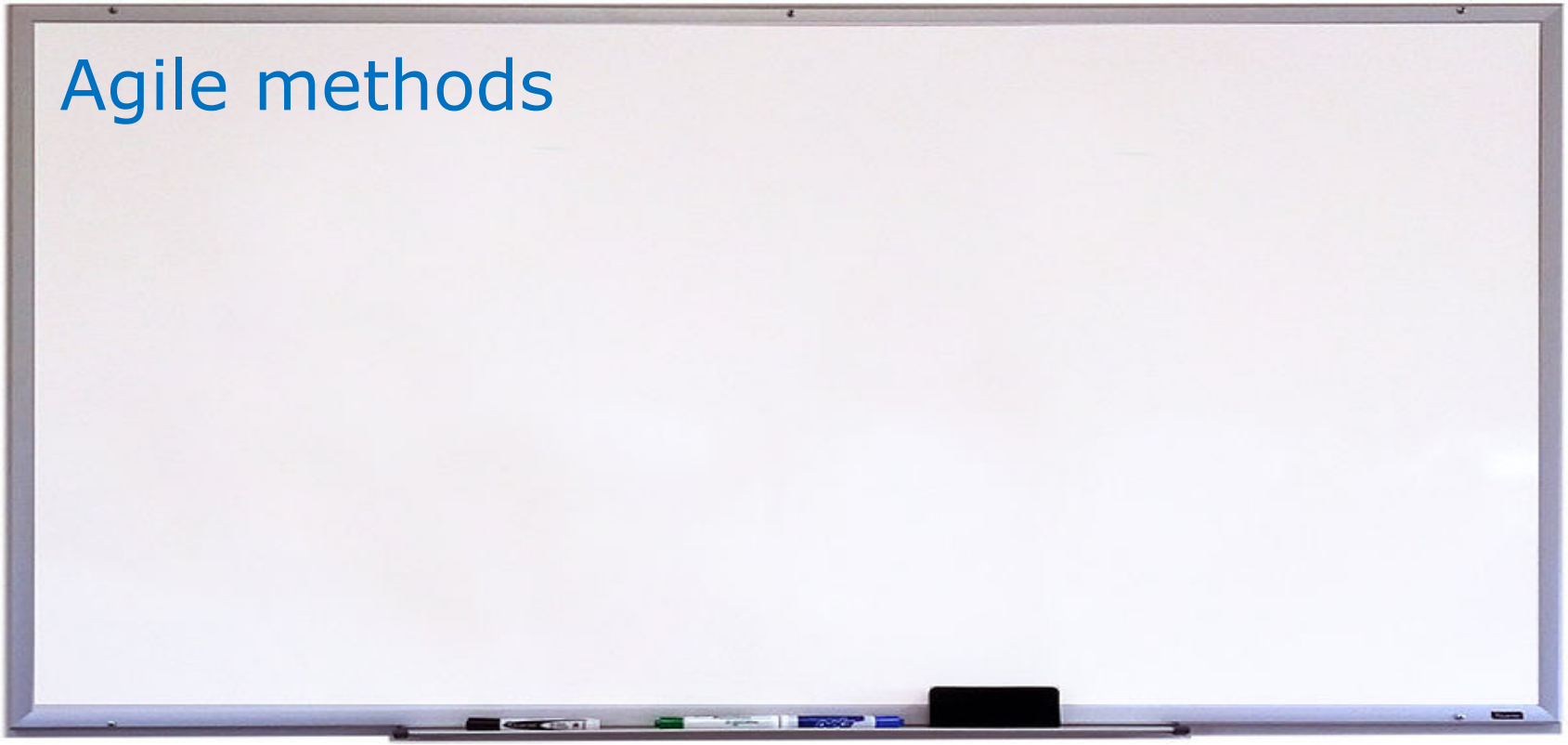
Agile Software Engineering

Antonio Brogi

Department of Computer Science
University of Pisa

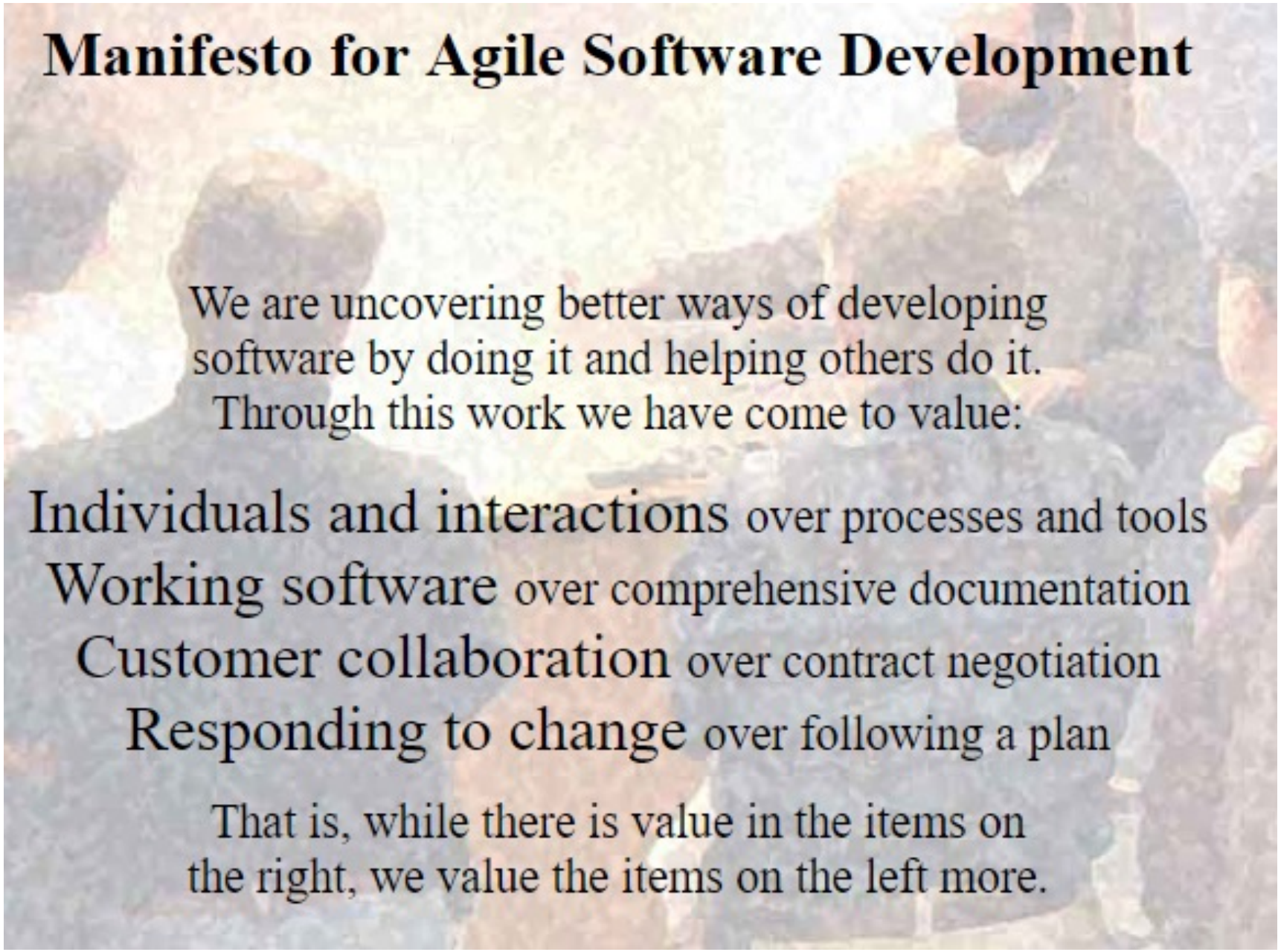
Agile methods



Customers reluctant to change product after investing on it
→ Getting product to customers **quickly** is critical
→ Need rapid software developments technique (**Agile** methods)

Agile SE
- To deliver functionalities quickly
- To respond quickly to changes
- To minimise development “overheads”

Plan-driven development (twentieth century)



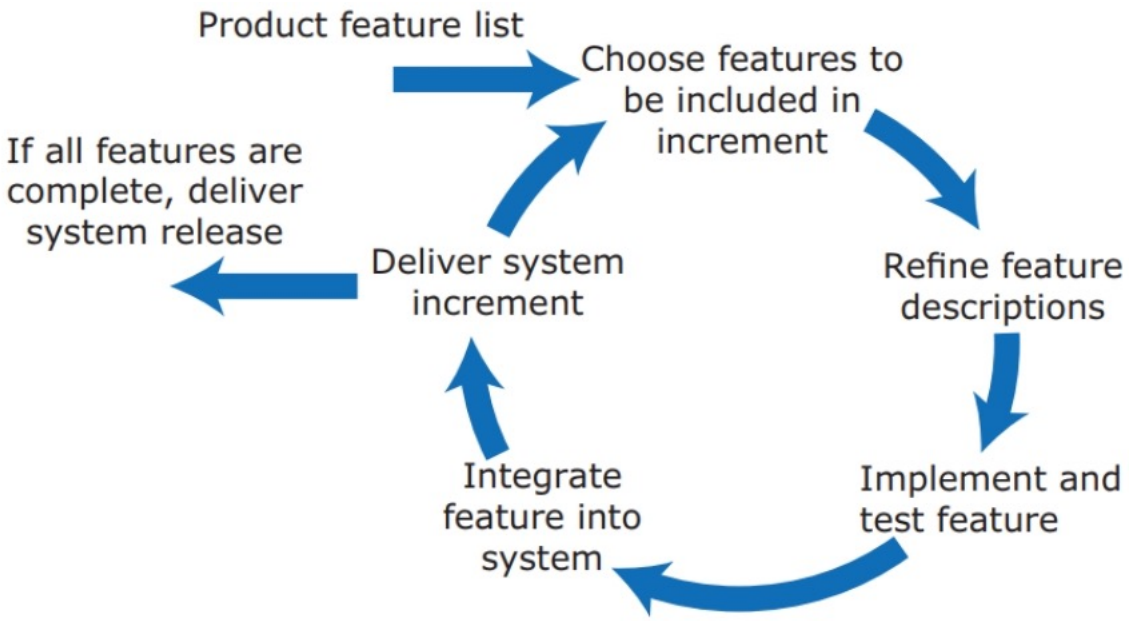
(2001)

https://agilemanifesto.org/

Incremental development

[2:20-3:04]

Incremental development

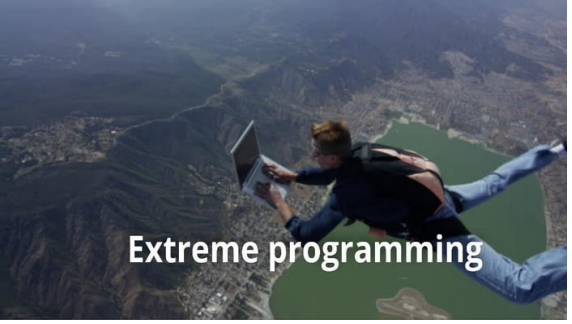


Activity	Description
Choose features to be included in an increment	Using the list of features in the planned product, select those features that can be implemented in the next product increment.
Refine feature descriptions	Add detail to the feature descriptions so that the team members have a common understanding of each feature and there is sufficient detail to begin implementation.
Implement and test	Implement the feature and develop automated tests for that feature that show that its behavior is consistent with its description. I explain automated testing in Chapter 9.
Integrate feature and test	Integrate the developed feature with the existing system and test it to check that it works in conjunction with other features.
Deliver system increment	Deliver the system increment to the customer or product manager for checking and comments. If enough features have been implemented, release a version of the system for customer use.

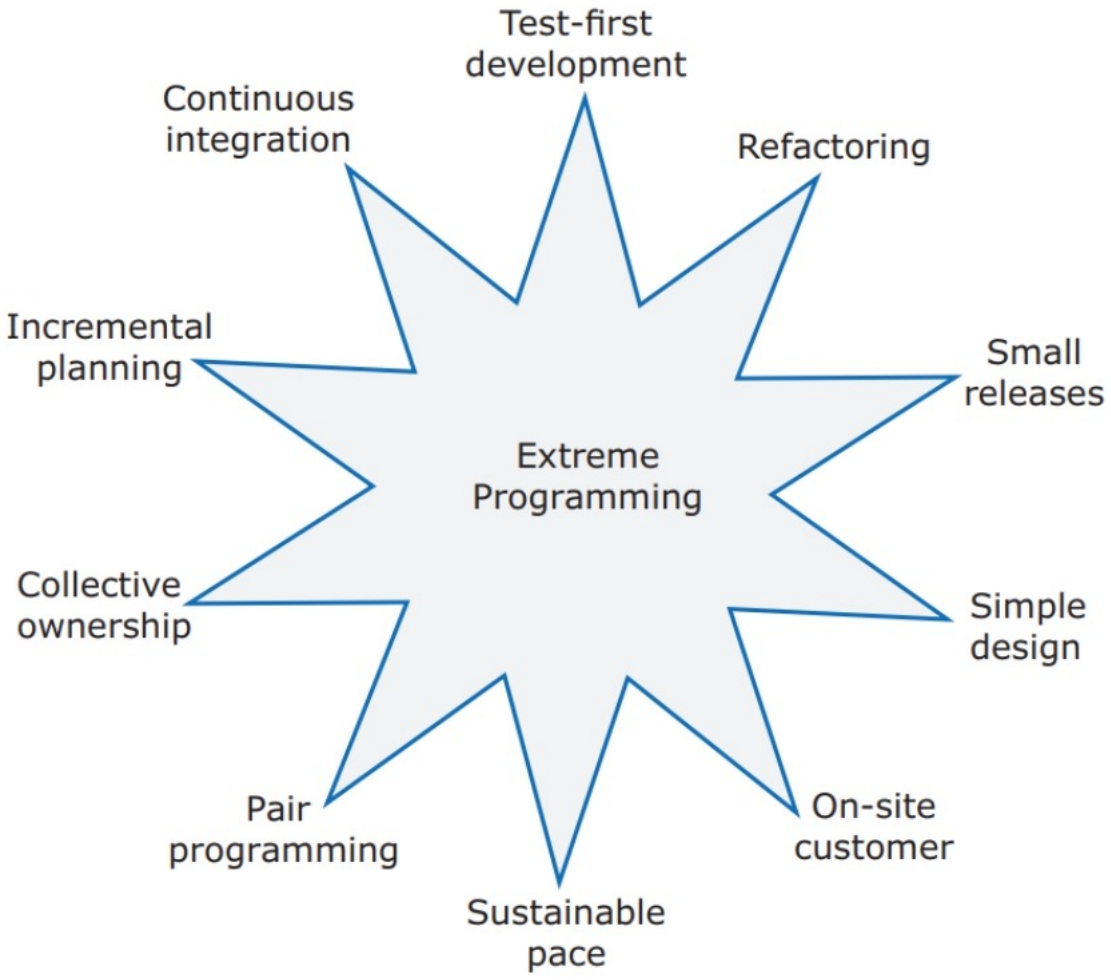
Twelve principles of Agile

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.
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.
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.


Agile methods
Extreme programming



Extreme Programming (XP)



Practice	Description
Incremental planning/ user stories	There is no “grand plan” for the system. Instead, what needs to be implemented (the requirements) in each increment are established in discussions with a customer representative. The requirements are written as user stories. The stories to be included in a release are determined by the time available and their relative priority.
Small releases	The minimal useful set of functionality that provides business value is developed first. Releases of the system are frequent and incrementally add functionality to the previous release.
Test-driven development	Instead of writing code and then tests for that code, developers write the tests first. This helps clarify what the code should actually do and that there is always a “tested” version of the code available. An automated unit test framework is used to run the tests after every change. New code should not “break” code that has already been implemented.
Continuous integration	As soon as the work on a task is complete, it is integrated into the whole system and a new version of the system is created. All unit tests from all developers are run automatically and must be successful before the new version of the system is accepted.
Refactoring	Refactoring means improving the structure, readability, efficiency, and security of a program. All developers are expected to refactor the code as soon as potential code improvements are found. This keeps the code simple and maintainable.


Agile methods
Extreme programming
Scrum

Please see the separate set of slides on Scrum

Reference

Chapter 2 – Agile Software Engineering