UNIVERSITAT DE BARCELONA

Agile Methodologies

by Eloi Puertas

based on Oriol Pujol Software Engineering slides.

Default Model

The usual way of understanding the software development process is based on the sentence:

“Software development is a process based on code and fix“

Features:

- Little planning and design
- Design is a changing process

Suited for small projects.

Software Engineering

- Software engineering appears as a means to provide a more disciplined and structured process.

- This avoided many **bugs** and inefficiencies of the first stages of the project development.

- Based on strong emphasis in **planning**.

- The main criticism is that makes everything more **bureaucratic**.

Introduction to **agile**

Agile tries to simplify software engineering processes, making them less document oriented and more code-oriented.

Differences:

Agile methods are adaptive rather than predictive: Plans based on prediction are rigid in front of changes. Changes are a core mechanic in agile technologies.

Agile methods are people-oriented rather than process-oriented: Rather than figuring out a method good independent of the working team, agile methods try to support current teams.

Sofware **VS** Enginnering

CLASSIC ENGINEERING PROJECTS

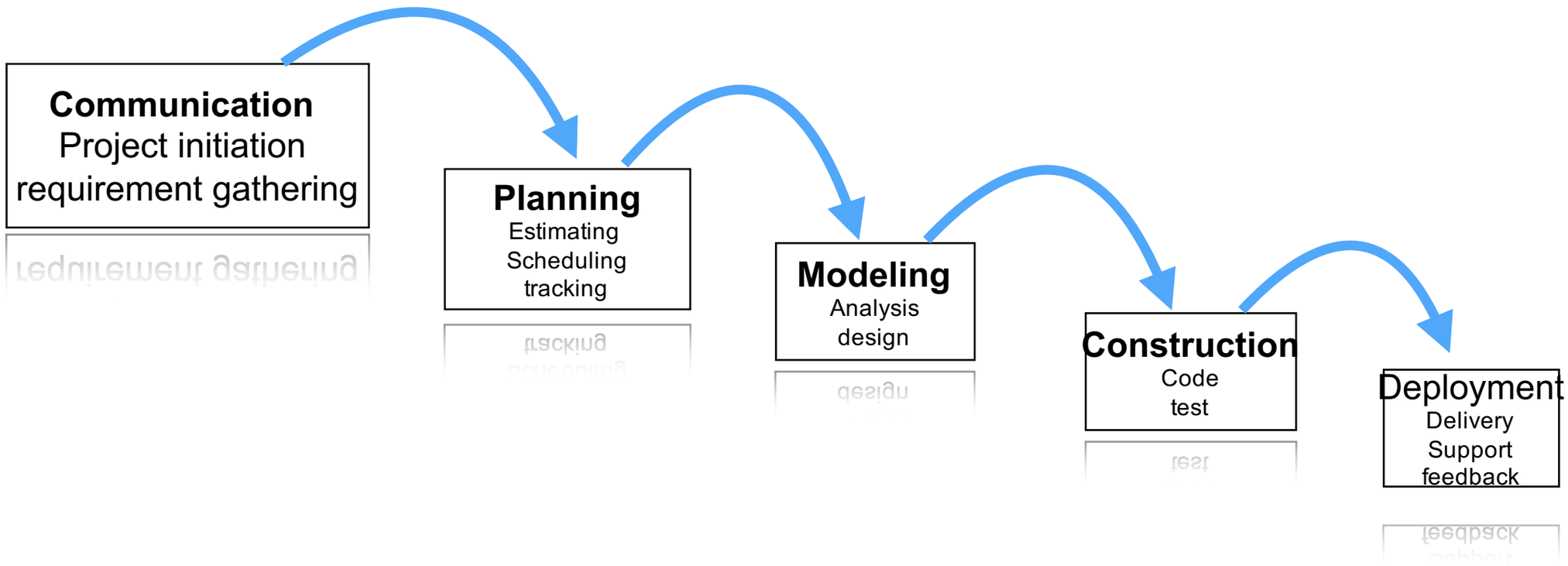
- Design is 15% of the process
- Design and construction are clearly differentiated
- While design is a creative process, construction is less intellectual and more manual.
- Design can be mathematically validated.

SOFTWARE DEVELOPMENT PROJECTS

- All effort is in the design (*source code*), construction is extremely cheap (*compiler*)
- Software design, as any other creative process, is not easily planned and predicted
- Even if we use notation documentation for design, it can only be validated using code-review.

Waterfall Process

This Model suggests a systematic, sequential approach to SW development that begins at the system level and progresses through analysis, design, code and testing



Waterfall Process

- Detailed planning phase, end product is carefully designed, and documented in great detail.
- The work is organized using tools such as Gantt charts and applications such as MSProject.
- Stakeholders review the plan and provided their approvals, the team starts to work.
- Team members complete their specialized portion of the work.
- Once the work is complete, it is delivered to a testing organization (Quality Assurance).
- Throughout the process, strict controls are placed on deviations from the plan.

STRENGTH: It is logical, methodical, follows a plan, and keeps everything as organized as possible.

WEAKNESS: Humans are involved.

Waterfall Process

Real Life problems when using Waterfall:

- Real projects rarely follow the sequential flow since they are always iterative
- The model requires **requirements** to be explicitly write down in the beginning, which is often difficult
- A working model is not available until late in the project time plan.
- Great emphasis on **writing for communicating critical information** - in reality most of the time these highly detailed 50-page requirements documents are never read.
- The first time that you use the working product you immediately think of 20 ways you could have made it better.
- Humans are not able to predict the future and propose risk plans.

Predictive **VS** Adaptative

PROBLEM:

Software development depends on **requirements**. If you cannot get **stable** requirements you cannot get a predictable plan.

- If the situation is not predictable then most of the methods for managing projects are prone to failure.

Predictive **VS** Adaptative

HOW TO MANAGE UNPREDICTABLE SITUATIONS?

The most difficult point is to know **where the project is**.

❖ The solution is **evolutionary methodologies**: which can **adapt** to new changes:

- Produce working versions of the final system that have a subset of features at each iteration.

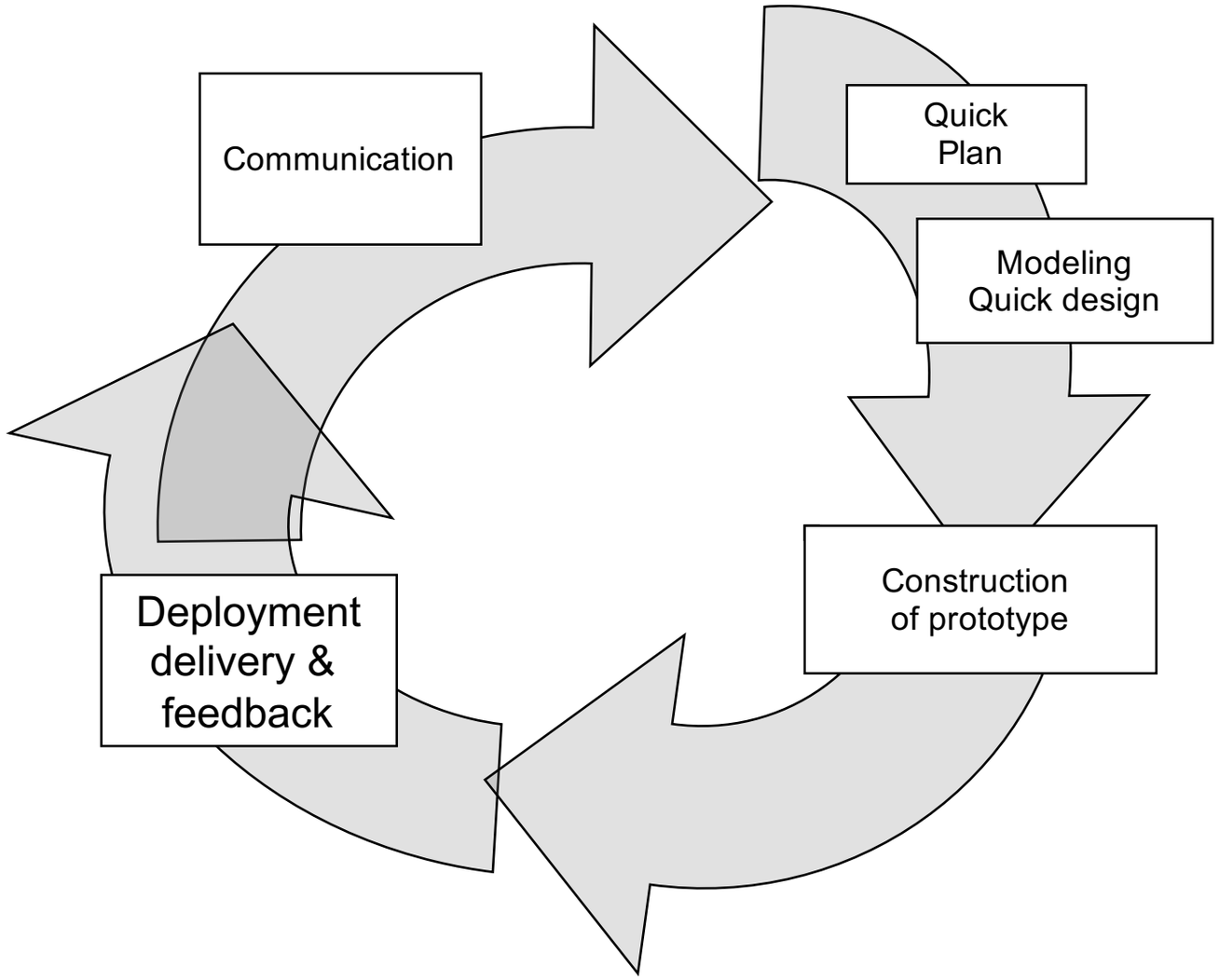
- These working versions should be fully integrated and as carefully tested as a final delivery.

Evolutive Process

It is based on the idea of developing an initial implementation and refine over a period of time

- A thorough knowledge of the requirements is not necessary
- It is based on feedback from the client
- The project is divided into "builds"
- The product is shown to the clients very soon.

Evolutive Process: Prototype



- Begins with requirement known until this moment
- A quick design occur
- Quick design leads to the construction of prototype
- Prototype is evaluated by the customer
- Requirements are refined
- Prototype is tuned to satisfy the needs of customer

Pros & Cons of Prototyping

PROS

- The prototype is used as machinery to identify requirements .
- It develops very fast.

CONS

- In a rush to get it working, overall software quality or long term maintainability are generally overlooked
- Use of inefficient algorithm

Putting the **people** first

TRADITIONAL METHODOLOGIES

- People are replaceable parts
- Only roles exist

In a predictable process you need predictable units, but people are not predictable and we forget the creativity of each Individual.

AGILE METHODOLOGIES

- Developers are responsible professionals
- They make ALL technical decisions.
- They estimate time.
- Managers and developers play an equal role (Technological changes are so fast managers can barely keep with it and must rely on developers.)

Agile **Methodologies**

- **SCRUM.** Methodology centered in collaborative team-work. Work is organized in a feature-centered process to be developed in small sprints with the aim of building functional software using cross-experienced teams.

- e**Xtreme Programming (XP)** Work is organized in a feature-centered process to be developed in teams organized in pairs. One of the key features of XP is that testing is a project driving mechanism. This gives rise to TDD (Test Driven Development)

Agile **Methodologies**

- **Kanban.** The kanban methodology relies upon full transparency of work and real-time communication of capacity. Therefore the team work revolves around a kanban board, a tool used to visualize work and optimize the flow of the work among the team, which should be seen as the single source of truth for the team's work.

- **Lean.** Lean comes from Lean Manufacturing and is a set of principles for achieving quality, speed & customer alignment. Lean says to eliminate anything that isn’t adding value and work on what we absolutely need to be doing at this moment in time.

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.

http://agilemanifesto.org/principles.html

Conclusions

AGILE PRINCIPLES EMPHASIZE:

- Building working software that people can get hands on quickly – Less initial documents.

- Agile development focuses on cross-functional teams empowered to make decisions.

- No hierarchies and compartmentalization by function.

- Focuses on rapid iteration, with continuous customer input along the way.