Software Process and Process Models

Lecture # 05, 06
09,10 Sep

Rubab Jaffar
rubab.jaffar@nu.edu.pk

Software Design and Analysis
CS-324

Today's Outline

- Software Process
- Software Development approaches
- Process models
- Unified Process

Software Processes & Process Model

- **Software Process** is coherent sets of activities for specifying, designing, implementing and testing software systems.
- ***A (software/system) process model*** *is a description of the sequence of activities carried out in an SE project, and the relative order of these activities*

Plan-driven and Agile Processes(Development Approach)

- Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
- In agile processes, planning is incremental and it is easier to change the process to reflect changing customer requirements.
- In practice, most practical processes include elements of both plan-driven and agile approaches.
- There are no right or wrong software processes.

What is a Process Model ?

It is a description of

i) what tasks need to be performed in

ii) what sequence under

iii) what conditions by

iv) whom to

achieve the "desired results."

Why Have A Process Model?

- **Provide "guidance" for a systematic coordination and controlling of**
  - **a) the tasks and of**
  - **b) the personnel who perform the tasks**

Note the key words: coordination/control , tasks, people

Extending the "Simple" Process

As projects got *larger* and more *complex,* there is a need

- clarify and stabilize the requirements
- test more functionalities
- design more carefully
- use more existing software & tools
  - Database
  - Network
  - Code control
- more people to be involved

***Resulting in* more tasks *and* more people**

Effectiveness of using Correct Process Model

By changing the process model, we can improve:

- *Development speed (time to market)*
- *Product quality*
- *Project visibility*
- *Administrative overhead*
- *Risk exposure*
- *Customer relations, etc.*

Normally, a process model covers the entire lifetime of a product.

From birth of a commercial idea to final installation of last release.

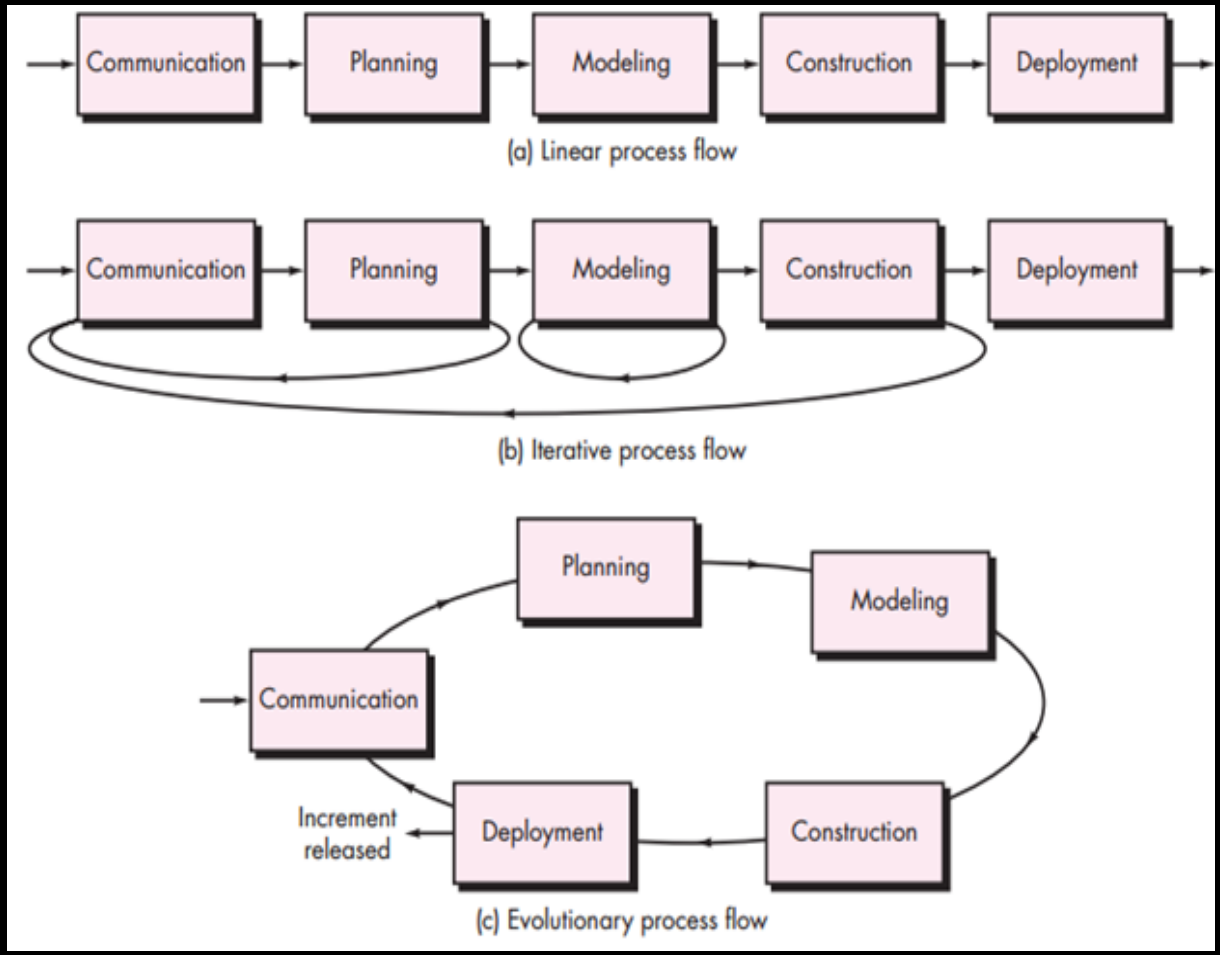
Common Activities of Process Model

Many different software processes but all involve:

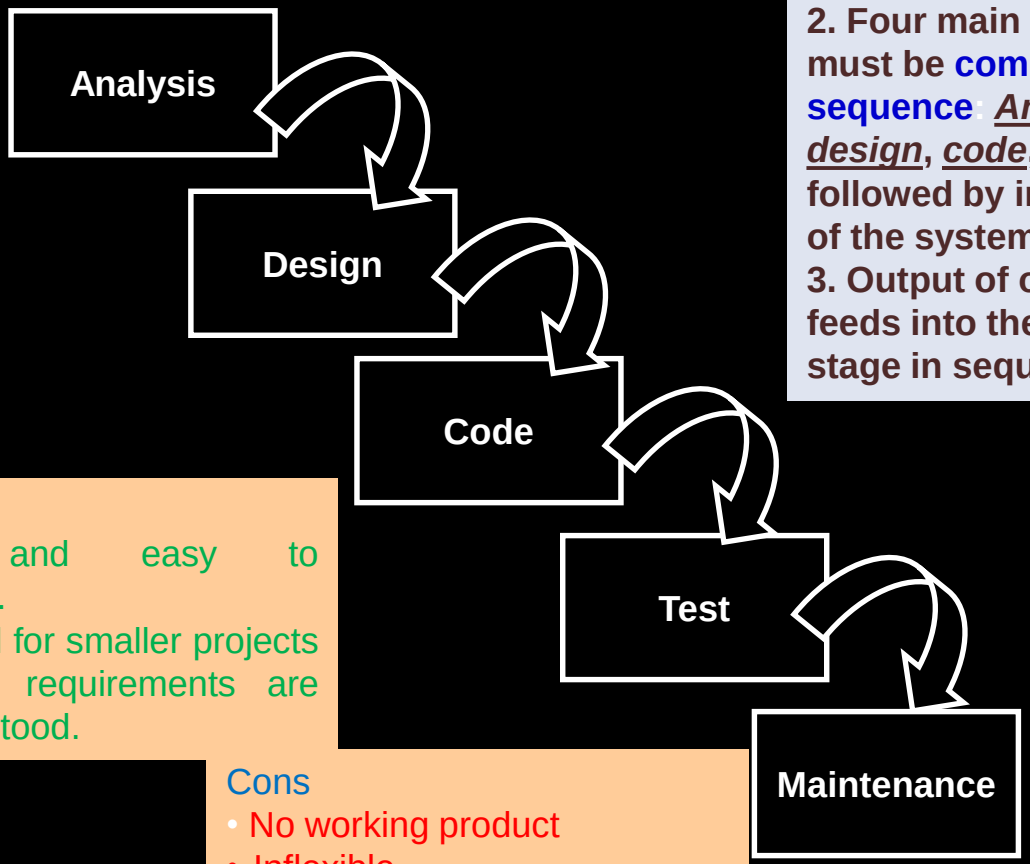
- Specification – defining what the system should do;
- Design and implementation – defining the organization of the system and implementing the system;
- Validation – checking that it does what the customer wants;
- Evolution – changing the system in response to changing customer needs.

Generic Process Flows

- Linear process flow
- Iterative process flow
- Incremental process flow



Waterfall Model

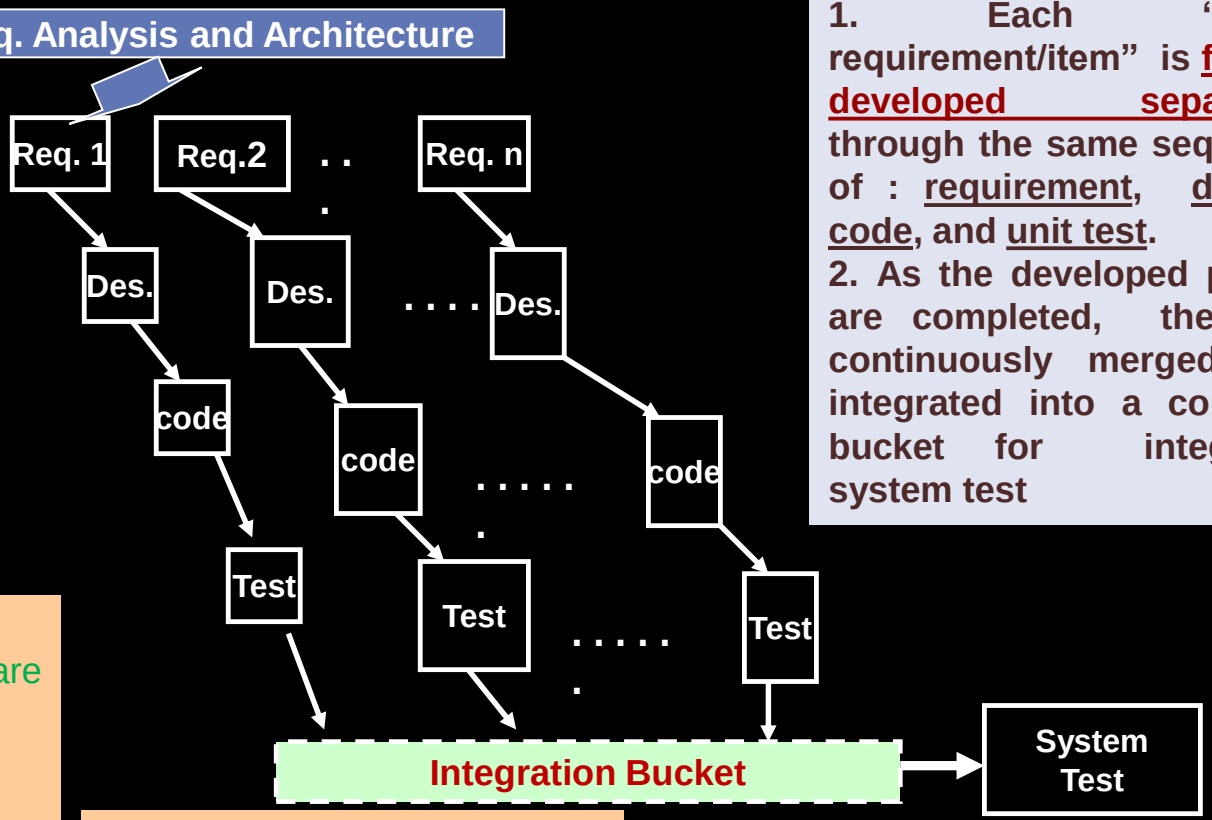


1. Requirements must be specified.
2. Four main tasks must be **completed in sequence**: ***Analysis***, ***design***, ***code***, and ***test***, followed by integration of the system.
3. Output of one stage feeds into the next stage in sequence

Pros
- Simple and easy to understand.
- Works well for smaller projects where the requirements are well-understood.

Cons
- No working product
- Inflexible

Incremental Model

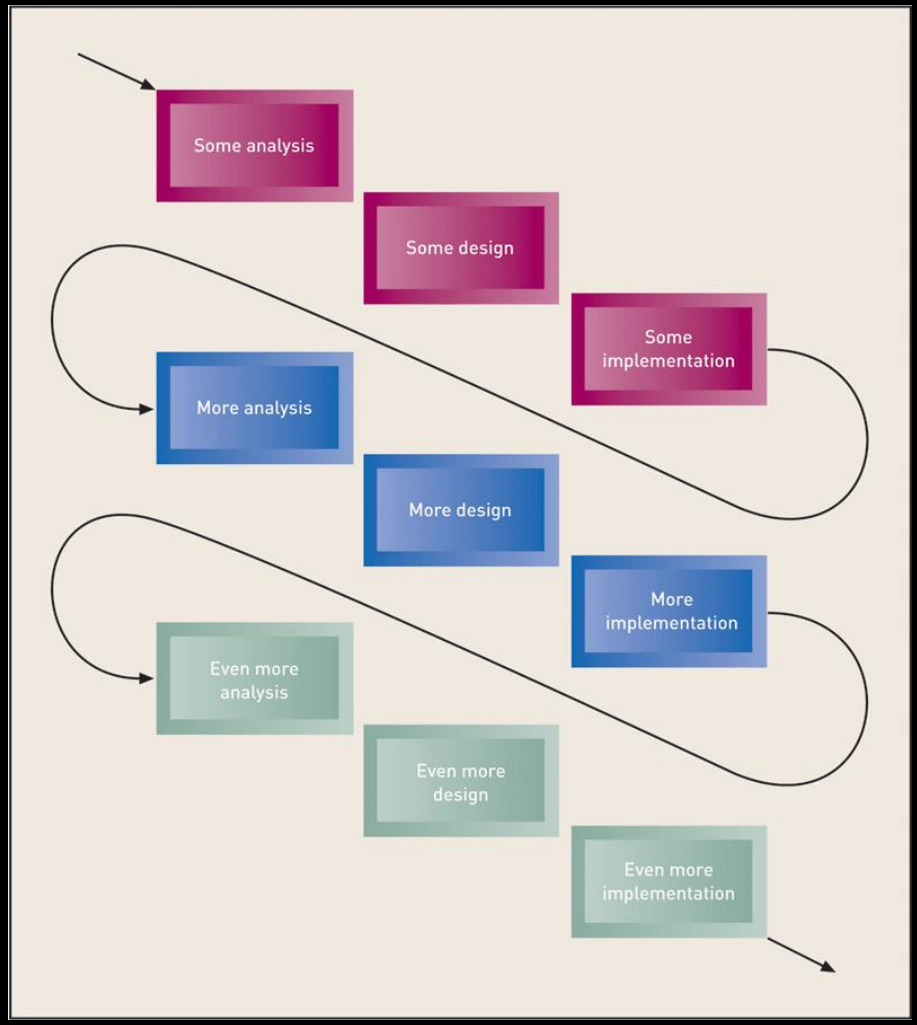


1. Each "major requirement/item" is further developed separately through the same sequence of : requirement, design, code, and unit test.
2. As the developed pieces are completed, they are continuously merged and integrated into a common bucket for integrated system test

Pros
- Generates working software quickly.
- More flexible
- Less costly
- easier to test and debug.
- customer can respond to each built.

Cons
- Needs good planning and design.
- Needs a clear and complete definition of the whole system

Iteration of System Development Activities



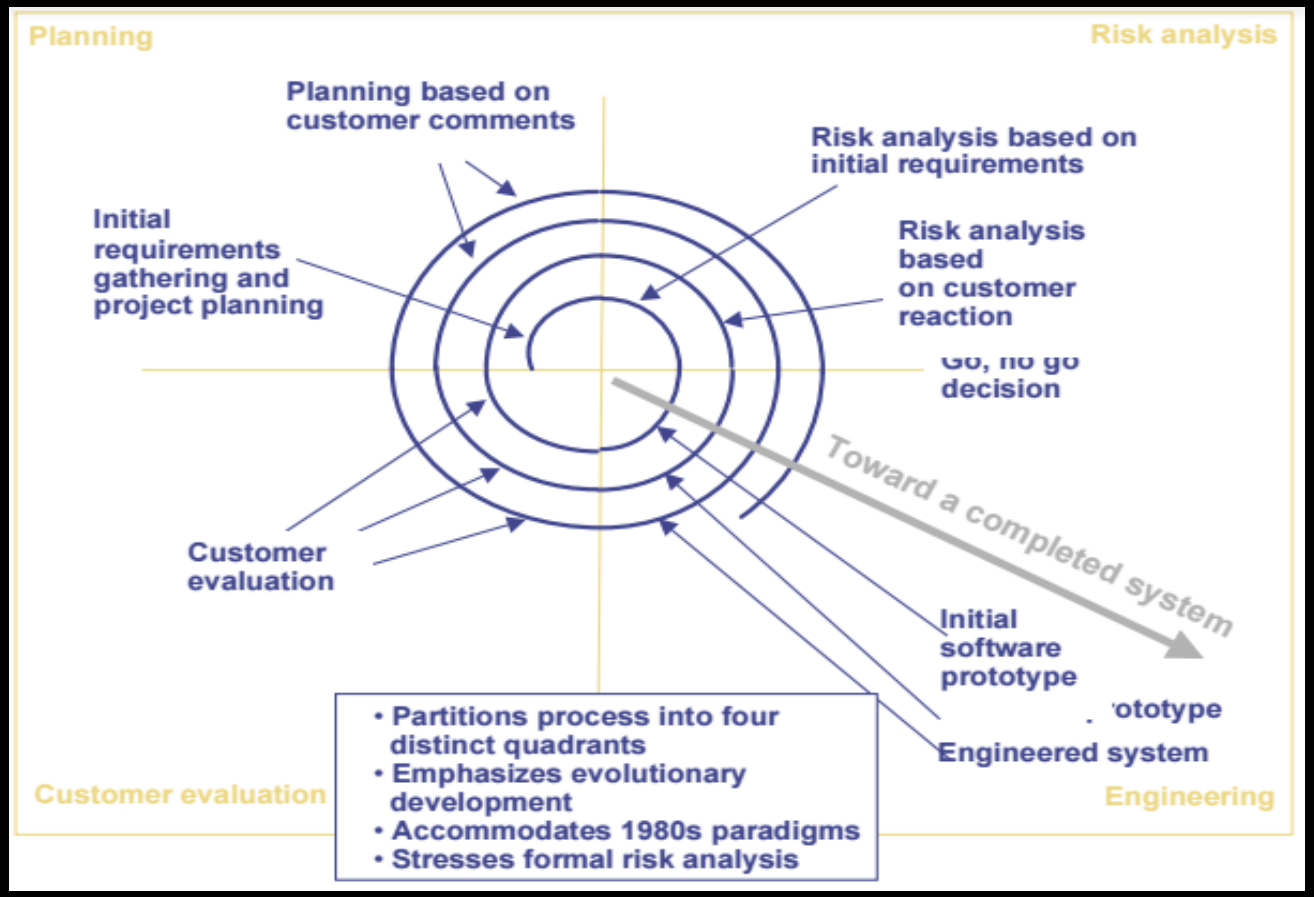
Problems

- Lack of process visibility
- Systems are often poorly structured

Applicability

- For small or medium-size interactive systems
- For parts of large systems
- For short-lifetime systems

Iteration Model



Rapid Software Development

- Rapid development and delivery is now often the most important requirement for software systems
  - Businesses operate in a fast –changing requirement and it is practically impossible to produce a set of stable software requirements
  - Software has to evolve quickly to reflect changing business needs.
- Plan-driven development is essential for some types of system but does not meet these business needs.
- Agile development methods emerged in the late 1990s whose aim was to radically reduce the delivery time for working software systems

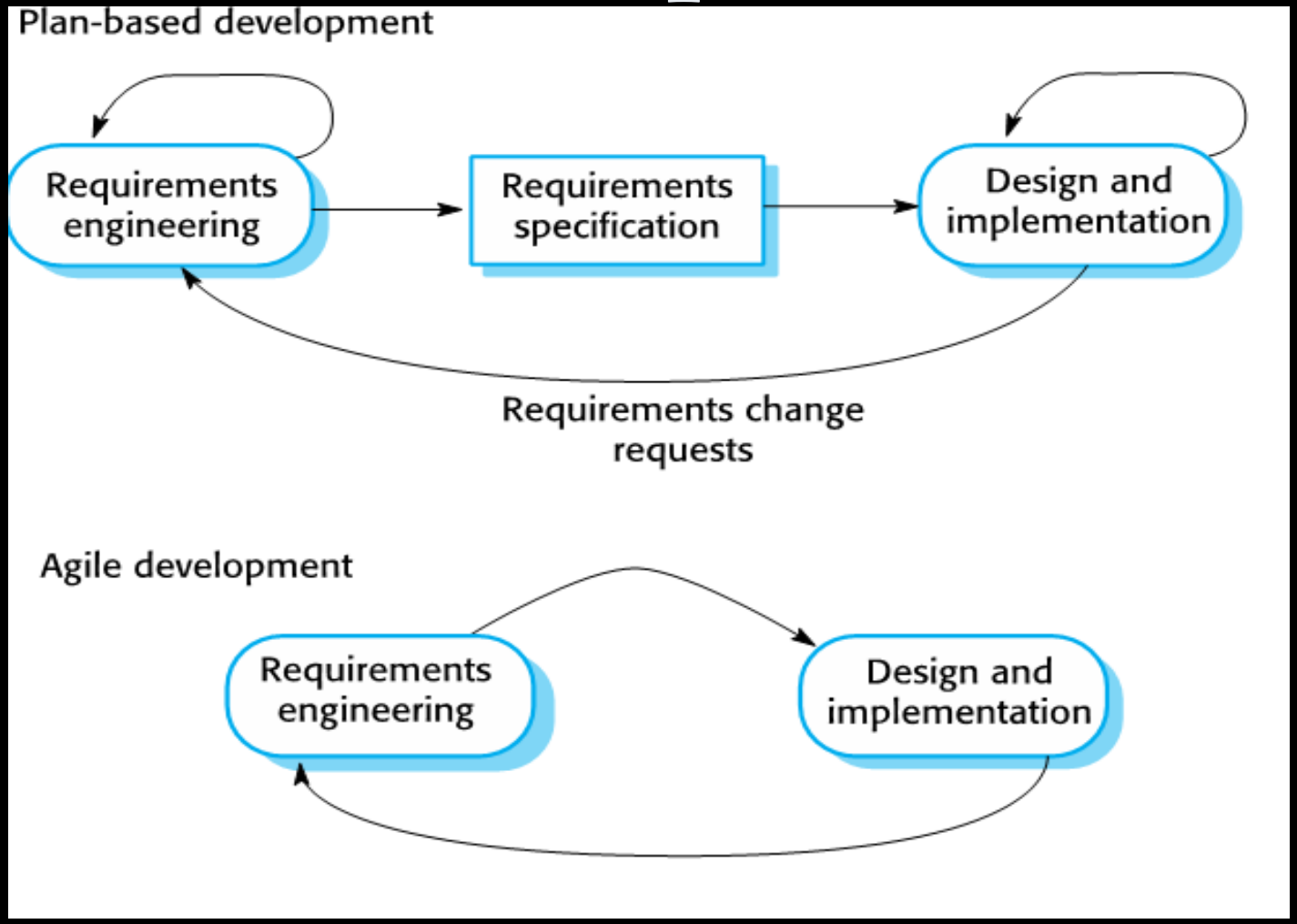
Agile Development Characteristics

- Program specification, design and implementation are inter-leaved
- The system is developed as a series of versions or increments with stakeholders involved in version specification and evaluation
- Frequent delivery of new versions for evaluation
- Extensive tool support (e.g. automated testing tools) used to support development.
- Minimal documentation – focus on working code

Plan-driven and Agile Development

- Plan-driven development
  - A plan-driven approach to software engineering is based around separate development stages with the outputs to be produced at each of these stages planned in advance.
  - Not necessarily waterfall model – plan-driven, incremental development is possible
  - Iteration occurs within activities.
- Agile development
  - Specification, design, implementation and testing are inter-leaved and the outputs from the development process are decided through a process of negotiation during the software development process.

Plan-driven and Agile Development



Agile models

- XP
- Scrum

That is all