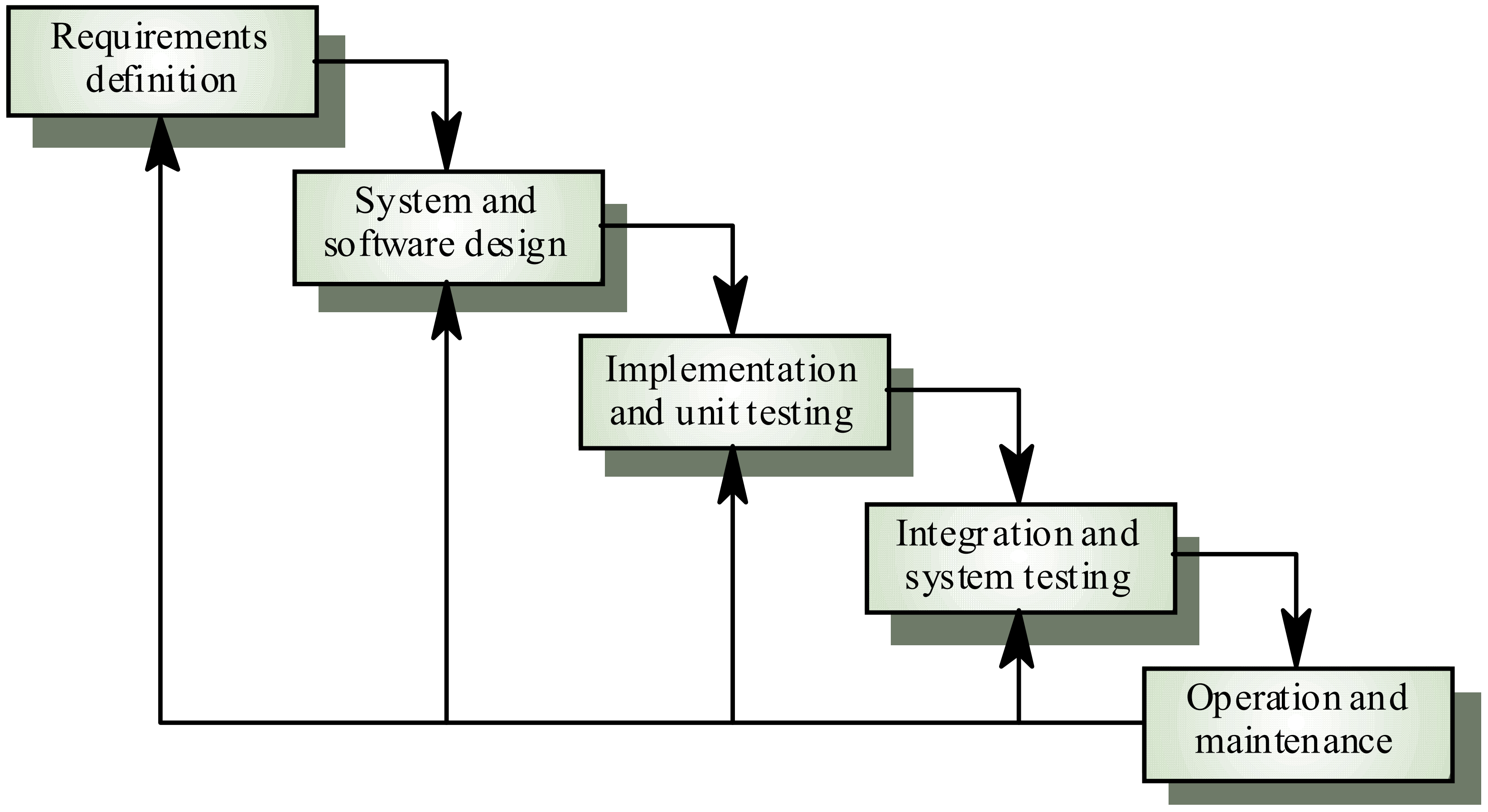
The Software Process

- **A structured set of activities required to develop a software system**
  - Specification
  - Analysis, design and implementation.
  - Validation
  - Evolution
- **A software process model is an abstract representation of a process**
  - It presents a description of a process from some particular perspective

Software Development Approaches

- **Generic Software Process Models**
  1. **The waterfall model** (Linear Sequential Model)
     - Separate and distinct phases of specification and development
  2. **Evolutionary development**
     - Specification, development and testing are interleaved
- **Formal systems development (example - ASML)**
  - A mathematical system model is formally transformed to an implementation

Waterfall Model



Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

Waterfall model phases

- **Requirements** analysis and definition
- System and software **design**
- **Implementation** and unit testing
- **Integration** and system testing
- **Operation** and maintenance

The drawback of the waterfall model is the difficulty of accommodating change after the process is underway

Waterfall model Requirement and Design

Artefacts produced in the requirements and Design phases

- **SRS** -**S**oftware **R**equirements **s**pecification document
- SRS might include:
  - User usage stories (scenarios) – **Use cases**.
  - Static analysis – **class diagrams**.
  - Behavioural analysis – **sequence diagrams**, **statecharts**.

The specification and design activities are heavily time consuming.

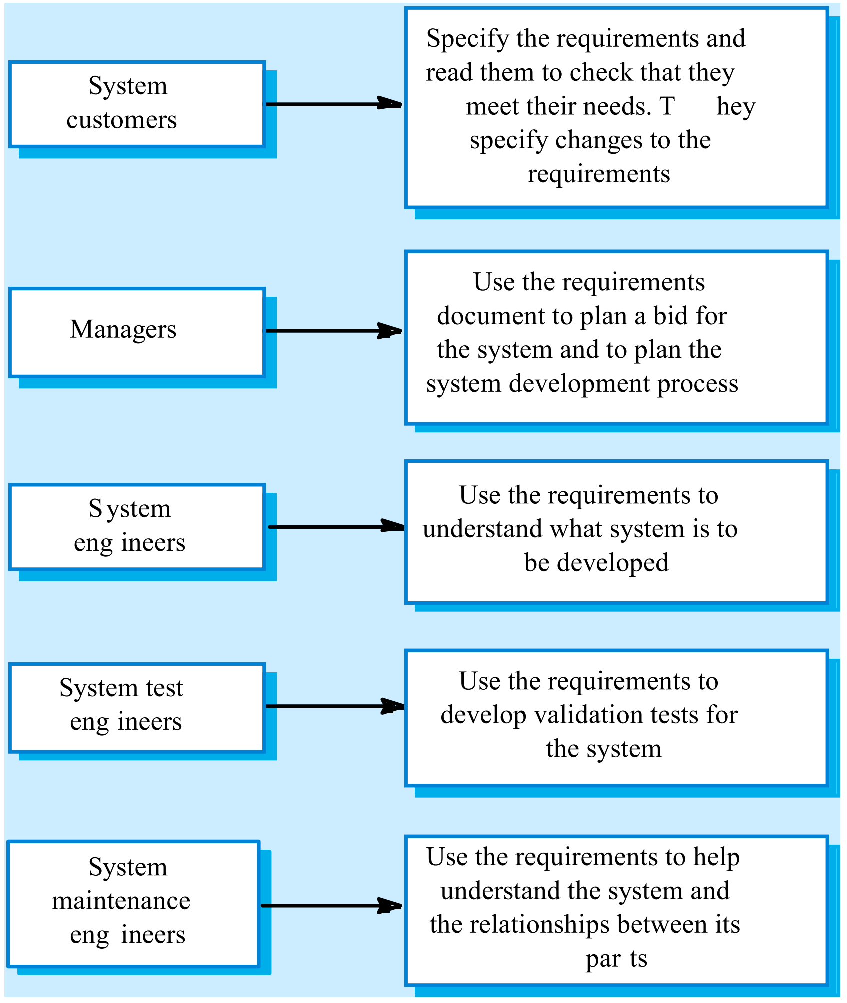
The requirements document

- The requirements document is the official statement of what is required of the system developers.

- Should include both a definition of user requirements and a specification of the system requirements.

- **It is NOT a design document**. As far as possible, it should set of WHAT the system should do rather than HOW it should do it

Users of a requirements document


System customers
Specify the requirements and read them to check that they meet their needs. T hey specify changes to the requirements
Managers
Use the requirements document to plan a bid for the system and to plan the system development process
S ystem eng ineers
Use the requirements to understand what system is to be developed
System test eng ineers
Use the requirements to develop validation tests for the system
System maintenance eng ineers
Use the requirements to help understand the system and the relationships between its par ts

The Waterfall process characterization:

- Figure out what
- Figure out how
- Then do it
- Verify was done right
- Hand product to customer
- Perfect requirement definition?

Waterfall model problems

- **Inflexible partitioning** of the project into distinct stages
- Difficult to respond to changing customer requirements

⇩

This model is only appropriate when the requirements are well-understood

Waterfall model describes a staged development process

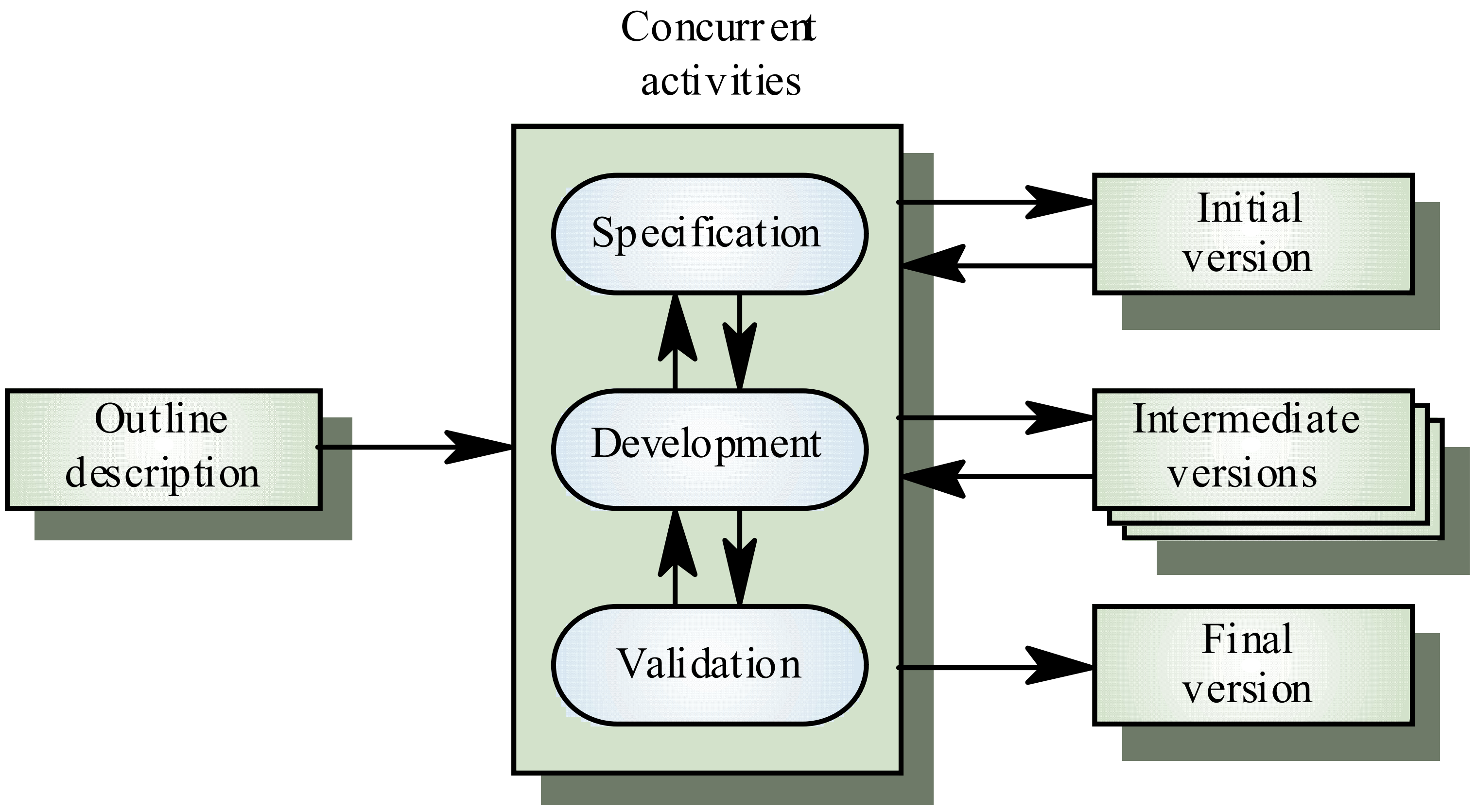
- **Based on hardware engineering models**
- **Change during development is unlikely**
- **Widely used in large systems: military and aerospace industries**

Why Not a Waterfall

Because software is different :

- **No fabrication step**
  - Program code is another design level
  - Hence, no "commit" step – software can always be changed…!
- **No body of experience for design analysis (yet)**
  - Most analysis (testing) is done on program code
  - Hence, problems not detected until late in the process
- **Waterfall model takes a static view of requirements**
  - Ignore changing needs
  - Lack of user involvement once specification is written
- **Unrealistic separation of specification from the design**
- **Doesn't accommodate prototyping, reuse, etc**

Evolutionary development



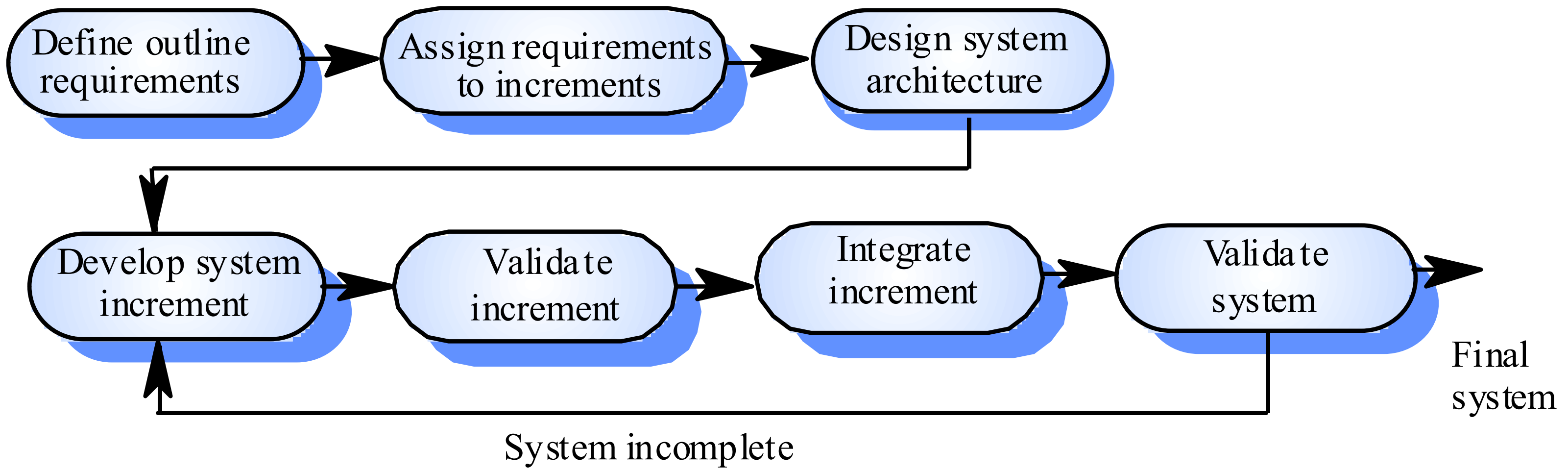
Characteristics of Evolutionary Development

- Modern development processes take ***evolution*** as fundamental, and try to provide ways of managing, rather than ignoring, the risk.
- System requirements always ***evolve*** in the course of a project.
- Specification is ***evolved*** in conjunction with the software – No complete specification in the system development contract. Difficult for large customers.
- Two (related) process models:
  - **Incremental development**
  - **Spiral development**

Incremental Development

- Rather than deliver the system as a single delivery, **the development and delivery is broken down into increments** with each increment delivering part of the required functionality.
- **User requirements are prioritised** and **the highest priority requirements are included in early increments.**
- **Once the development of an increment is started, the requirements are frozen** though requirements for later increments can continue to evolve.

Incremental Development – Version I



Incremental Development – Advantages

- **Customer value** can be delivered with each increment so system functionality is available earlier.
- **Early increments** act as a prototype to help elicit requirements for later increments.
- Lower risk of overall project failure.
- **The highest priority system services**

  tend to receive the most testing.

Incremental Development – Problems

- Lack of process visibility.
- Systems are often poorly structured.

- **Applicability claims in the literature:**
  - For small or medium-size interactive systems.
  - For parts of large systems (e.g. the user interface).
  - For short-lifetime systems.

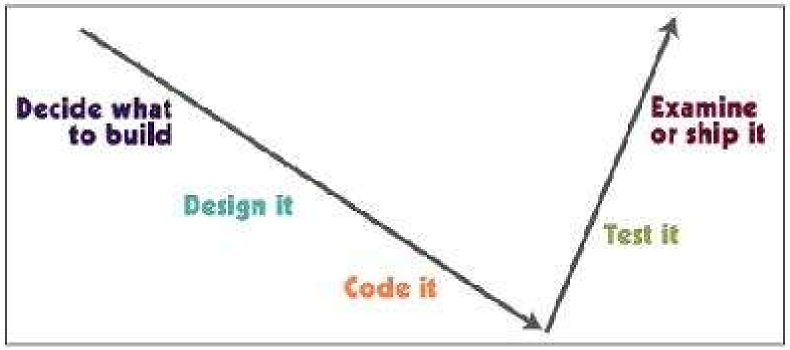
Incremental means adding, iterative means reworking (by Alistair Cockburn)

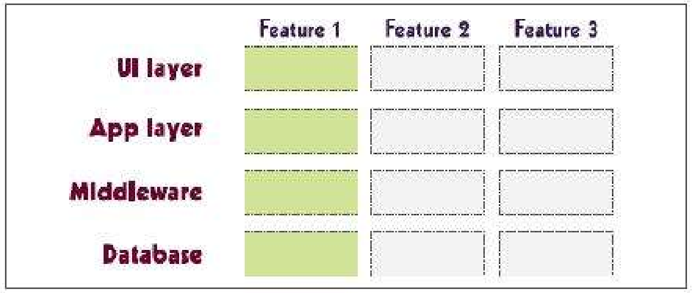
- **Incremental development** is a staging and scheduling strategy in which the various parts of the system are developed at different times or rates and integrated as they are completed. The alternative is to develop the entire system with a big bang integration at the end.
- **Iterative development** is a rework scheduling strategy in which time is set aside to revise and improve parts of the system. The alternative development is to get it right the first time (or at least declare that it is right!).

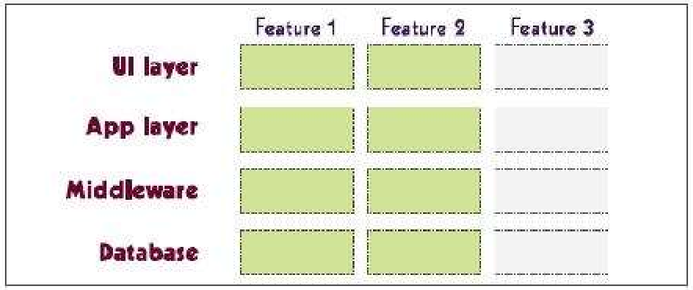
Iterate	Increment
fundamentally means "*change*".	fundamentally means "*add onto*".
repeating the process on the *same* section of work	repeating the process on a *new* section of work.
repeat the process (, design, implement, evaluate),	repeat the process (, design, implement, evaluate),

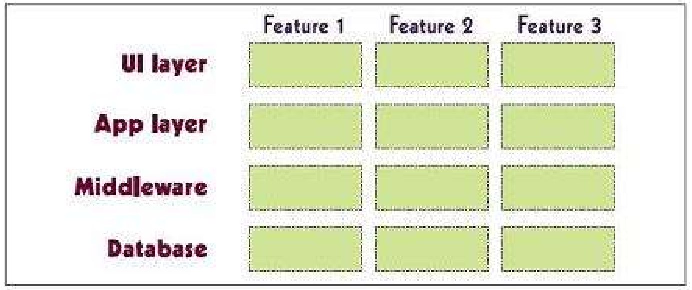
- http://www.stickyminds.com/BetterSoftware/magazine.asp?fn=cifea&id=108

Incremental Development



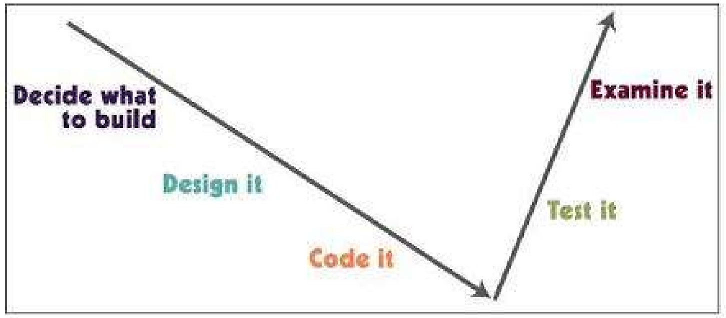


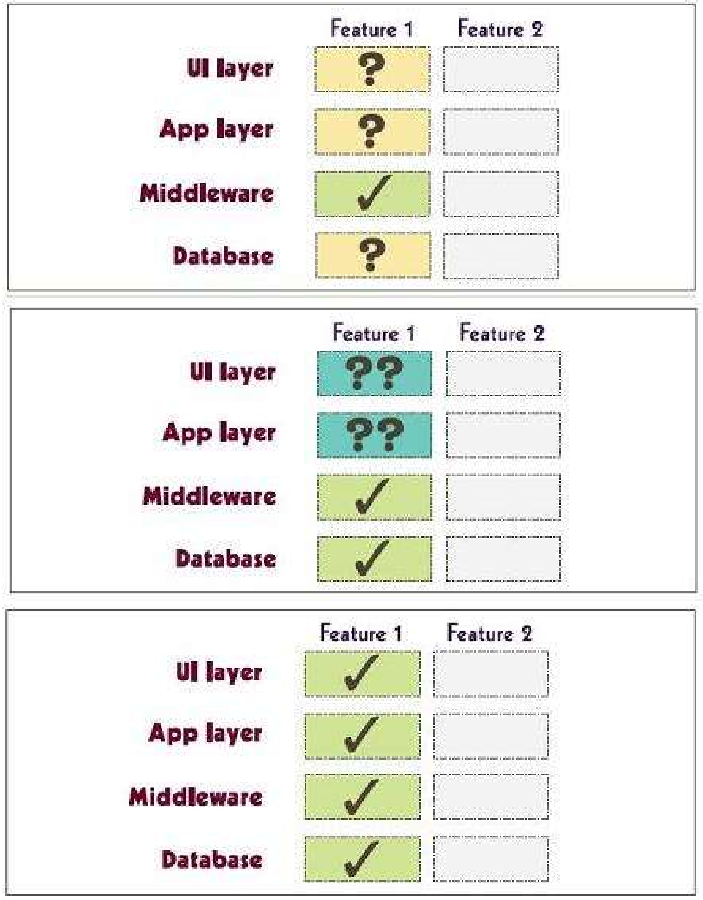




- The first increment delivers one slice of functionality through the whole system.
- The second increment delivers another slice of functionality through the whole system.
- The third increment delivers the rest of the system

Iterative Development





- The first iteration delivers enough of feature 1 to evaluate what is correct and what needs revision.
- After the second iteration, some revised parts still need improvement.
- The third iteration produces the final and stable feature

Incremental & iterative - summary

- Iterative model - This model iterates requirements, design, build and test phases again and again for each requirement and builds up a system iteratively till the system is completely build.
- Incremental model - It is non integrated development model. This model divides work in chunks and one team can work on many chunks. It is more flexible.
- Spiral model - This model uses series of prototypes in stages, the development ends when the prototypes are developed into functional system. It is flexible model and used for large and complicated projects.
- Evolutionary model - It is more customer focused model. In this model the software is divided in small units which is delivered earlier to the customers.
- V-Model - It is more focused on testing. For every phase some testing activity are done.

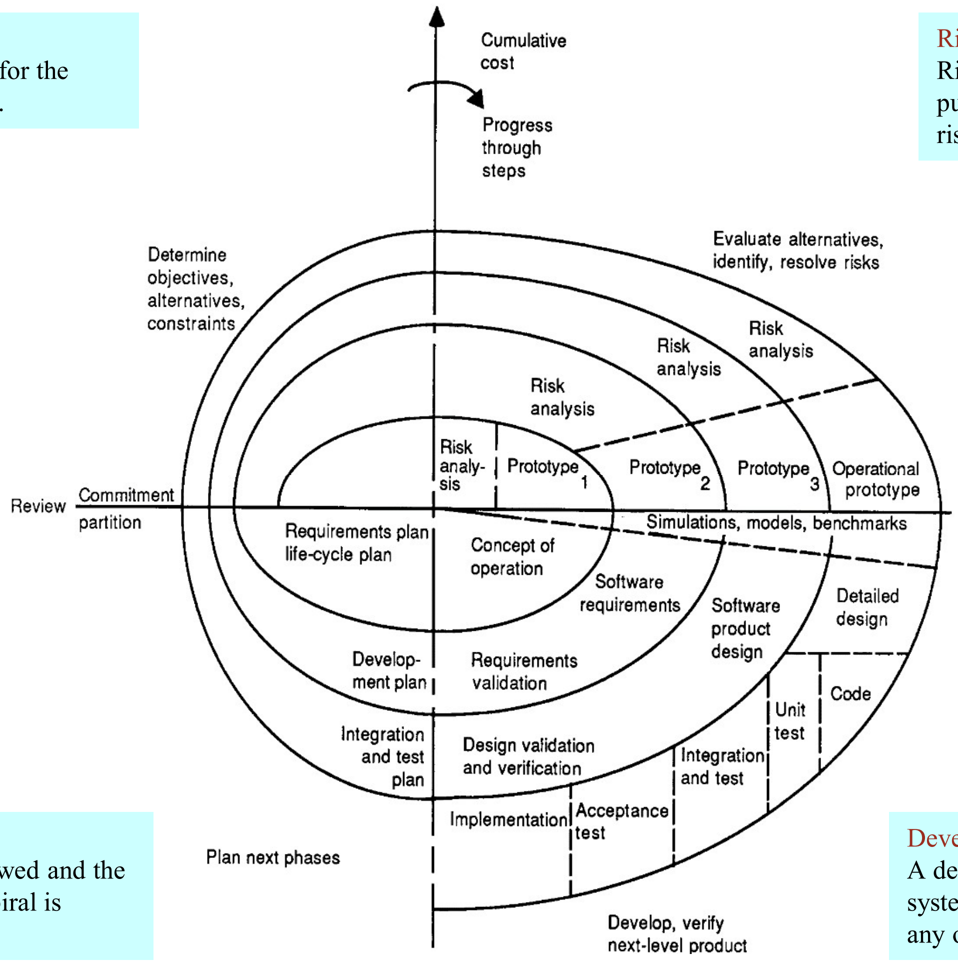
Spiral Development

- **Process is conceived as a spiral** rather than as a sequence of activities with backtracking.
- **Each loop in the spiral represents a phase** in the process.
- **No fixed phases such as specification or design** - loops in the spiral are chosen depending on what is required.
- **Risks are explicitly assessed and resolved** throughout the process.

Spiral model (Boehm 87)

Objective setting
Specific objectives for the phase are identified.

Risk assessment and reduction
Risks are assessed and activities put in place to reduce the key risks.



Planning
The project is reviewed and the next phase of the spiral is planned.

Development and validation
A development model for the system is chosen which can be any of the generic models.

Spiral model sectors

- **Objective setting**
  - Specific objectives for the phase are identified.
- **Risk assessment and reduction**
  - Risks are assessed and activities put in place to reduce the key risks.
- **Development and validation**
  - A development model for the system is chosen which can be any of the generic models.
- **Planning**
  - The project is reviewed and the next phase of the spiral is planned.

Agile Methods

Result from dissatisfaction with the overheads involved in design methods.

Software Development History:

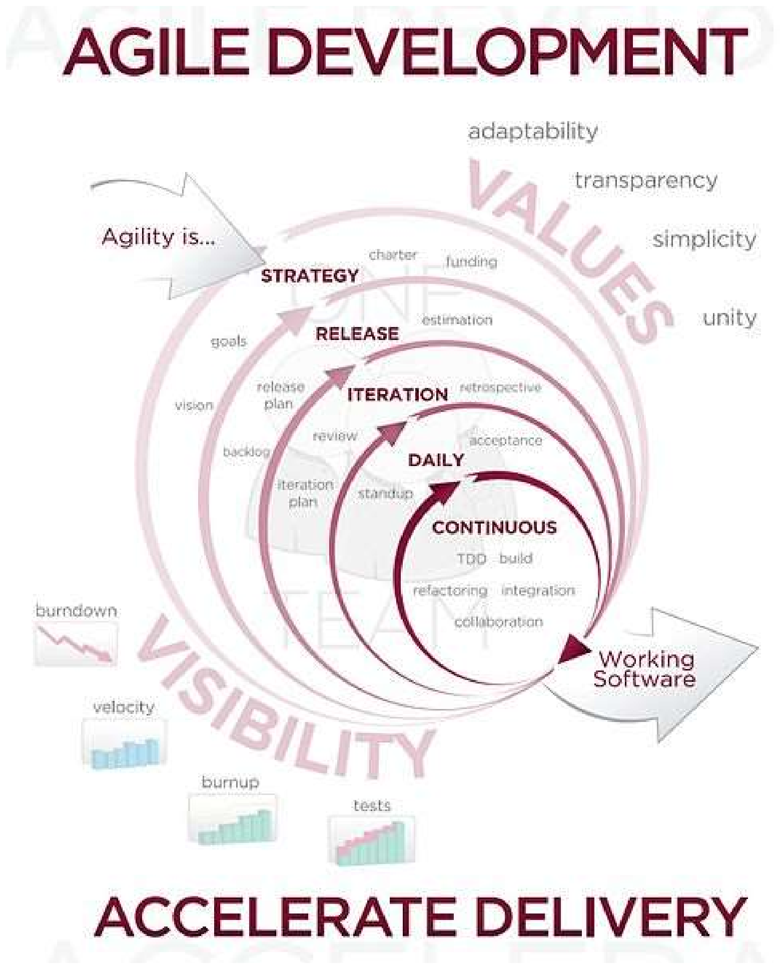
- **During the 1970s**, it was discovered that most large software development projects failed.
- **During the 1980s**, many of the reasons for those failures began to be recognized.
- **In the 1990s**, experiments and measurements were used to validate individual methods to prevent failure.
- **The current decade** is characterized by complete processes to improve success.

Project Failure – the trigger for Agility

- One of the primary causes of project failure was the **extended period of time it took to develop a system**.
- Costs escalated and requirements changed.

- Agile methods intend to **develop systems more quickly with limited time spent on analysis and design.**

Agile Development



What is an Agile method? (1)

- **Focus on the code rather than the design.**
- **Based on an iterative approach to software development.**
- **Intended to deliver working software quickly.**
- Evolve quickly to **meet changing requirements**.
- There are claims that agile methods are probably best suited to small/medium-sized business systems or PC products.

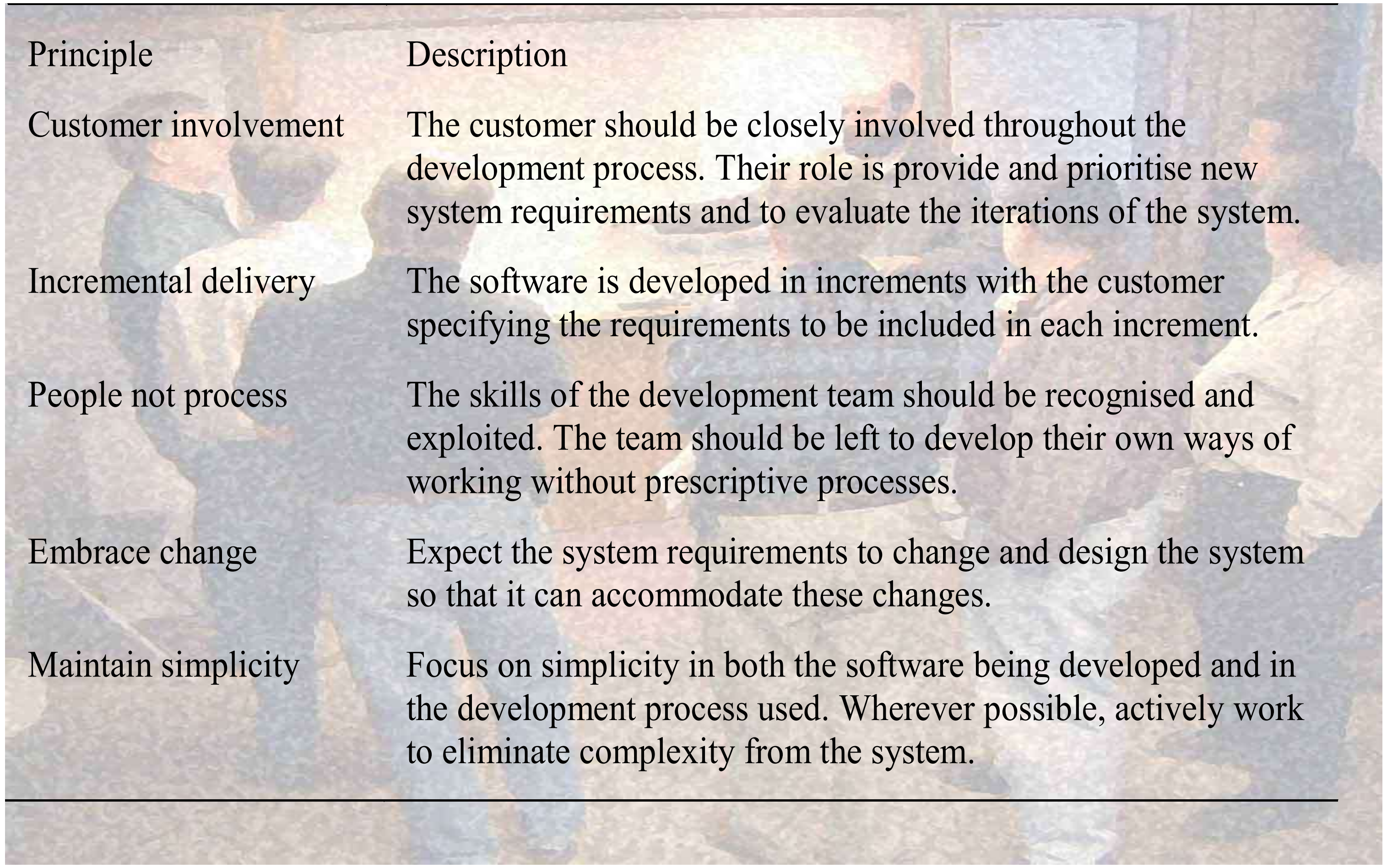
What is an agile method? (2)

- Agile proponents believe:
  - **Current software development processes are too heavyweight or cumbersome**
    - Too many things are done that are not directly related to software product being produced
  - **Current software development is too rigid**
    - Difficulty with incomplete or changing requirements
    - Short development cycles (Internet applications)
  - More **active customer** involvement needed

What is an agile method? (3)

- Agile methods are considered
  - **Lightweight**
  - **People-based** rather than Plan-based
- Several agile methods
  - No single agile method
  - Extreme Programming (XP) most popular
- No single definition
- Agile Manifesto closest to a definition
  - Set of principles
  - Developed by Agile Alliance

Summary of Principles of agile methods



Principle	Description
Customer involvement	The customer should be closely involved throughout the development process. Their role is provide and prioritise new system requirements and to evaluate the iterations of the system.
Incremental delivery	The software is developed in increments with the customer specifying the requirements to be included in each increment.
People not process	The skills of the development team should be recognised and exploited. The team should be left to develop their own ways of working without prescriptive processes.
Embrace change	Expect the system requirements to change and design the system so that it can accommodate these changes.
Maintain simplicity	Focus on simplicity in both the software being developed and in the development process used. Wherever possible, actively work to eliminate complexity from the system.

Problems with incremental development (from a waterfall eye...)

- **Management problems**
  - Progress can be hard to judge and problems hard to find because there is no documentation to demonstrate what has been done.
- **Contractual problems**
  - The normal contract may include a specification; without a specification, different forms of contract have to be used.
- **Validation problems**
  - Without a specification, what is the system being tested against?
- **Maintenance problems**
  - Continual change tends to corrupt software structure making it more expensive to change and evolve to meet new requirements.