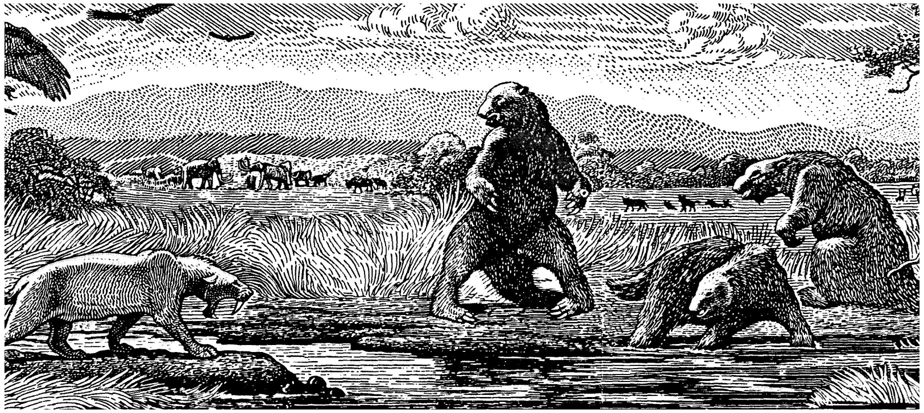
CS3500 Software Engineering

2017/2018



Dept. Computer Science
Dr. Klaas-Jan Stol

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Welcome to CS3500

Software and its Development

After studying this material and associated papers, you should be able to:

- Describe the different types of software and how software can be deployed.
- Explain the difference between bespoke and market-driven software.
- Describe who creates software.
- Describe the key activities in software engineering.

Contents

Types of Software Projects

1. Bespoke solutions

2. Market-driven solutions

Bespoke solutions

The term solution is just a fancy word for software system

Bespoke software is developed for a specific customer, e.g.:

- New website for AIB
- New payroll system for Chrysler Motor Co.

Market-targeted solutions

- Software solutions for a market of customers, not a specific customer in mind
  - Operating systems, e.g. Microsoft Windows
  - Database management systems, e.g. Oracle database
  - Web servers, e.g. Apache HTTP server
  - ERP (enterprise resource planning) systems, e.g. SAP
- Also known as Commercial Off-The-Shelf (COTS)
  - Most open source software (OSS)
  - Includes development frameworks e.g. Spring, Struts, Node.js – not usable by an end-user directly

Bespoke vs Market-driven software

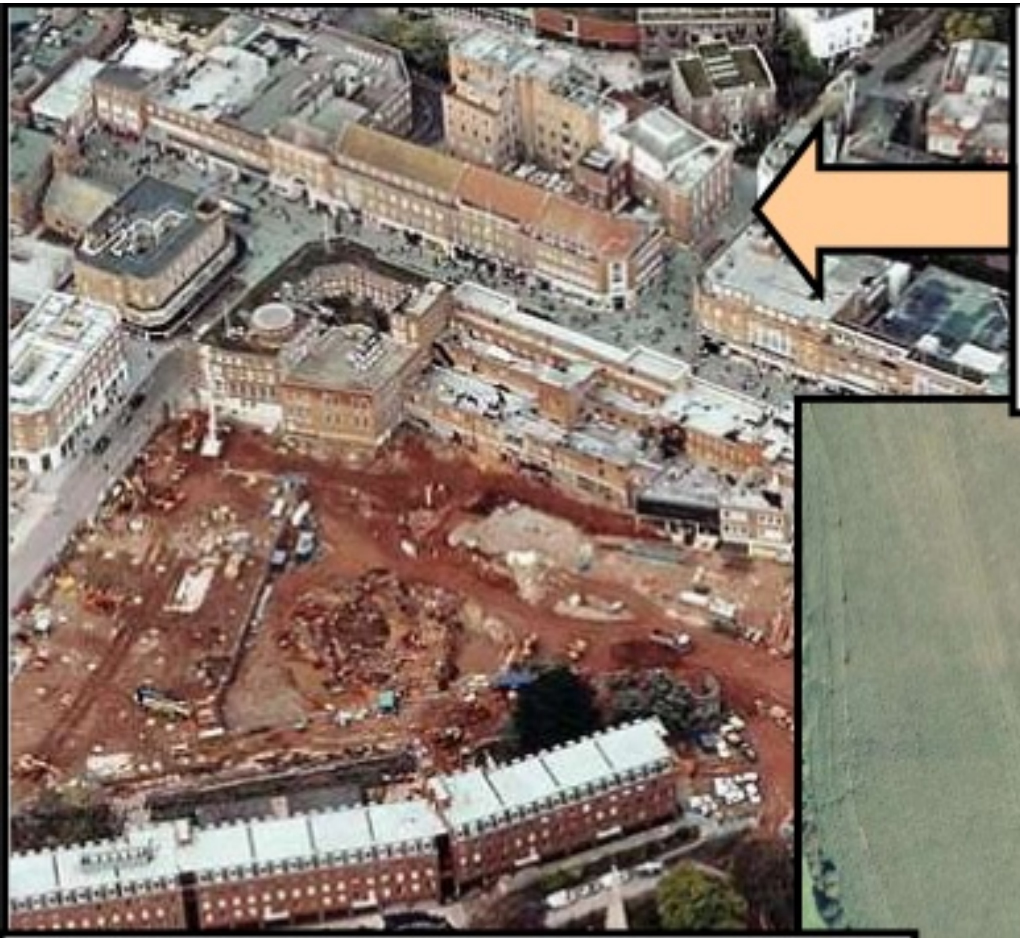
	Bespoke	Market-driven
Main stakeholder	Customer organization	Developing organization
Users	Known, identifiable	Unknown, may not exist (yet)
Distance to users	Typically small	Typically large
Requirements conception	Elicited, analyzed, validated	Invented (market pull, technology push)
Typical lifecycle	One release, then maintenance	Continuous / regular releases
Requirements issues	Elicitation, modeling, validation, conflict resolution	Stream of requirements, prioritization, cost estimation, release planning
Primary goal	Compliance to specification (legal), customer satisfaction (practical)	Time-to-market
Measure of success	Customer satisfaction, system acceptance	Sales, market share, imitation by competitors

Adapted from: Carlshamre, “A usability perspective on requirements engineering”, 2001.

Greenfield vs Brownfield

- Greenfield development:
  - Completely new project
- Brownfield development:
  - Modification or extension of a legacy system
- Most software development projects are brownfield projects

Brownfield vs greenfield terminology



Brownfield: a site that has been built on before and is ready for development. Normally associated with urban inner city areas

Greenfield: a site that has not been built on before. Often rural/ countryside areas. This includes rural-urban fringe.

Source: https://www.slideshare.net/geodebs/land-use-change-brownfield-vs-greenfield

How is Software Deployed?

1. Embedded software

2. Stand-alone software

3. Distributed systems

This classification is somewhat arbitrary, and other classifications exist. The key point is to understand the varying runtime environments of software.

Embedded software

- Software that runs on special-purpose devices
  - Home appliances e.g. TV, microwave, coffee makers
  - Cars
  - Industrial machinery
  - … many, many other examples

- Cannot separate software from the hardware
  —software usually not portable.

- Software development is dependent on hardware development
  —changing hardware design will affect software design

Stand-alone software

- Software that runs and depends on standard platforms
  —e.g. Windows, Linux, OSX
- Can run independently—function does not rely on other systems
- Examples:
  - Microsoft Office
  - IBM SPSS (statistics package)

Distributed systems

- Systems that runs on different nodes
- Physically separated
- Function depends on interaction

- Examples:
  - Web-based systems
  - Air Traffic Control systems
  - Infrastructural systems (e.g. electric grid)

Who is making all this software?

1. Software product houses

2. Non-traditional SW companies

3. Consultancy & services companies

4. Individual contractors

5. Open source developers

Dedicated software houses

Companies with software as a core activity.

Small sample of typical household names:

Non-traditional software co.’s with in-house teams

- Companies that don’t have software as core business but create software
- You wouldn’t think of these companies as typical software companies
  - Automotive sector (all car brands)
  - Financial sector (e.g. AIB)
  - Telecoms (e.g. Eir)
  - Home & professional appliances (e.g. Husqvarna)

Examples of non-traditional software co.'s with in-house teams

Rolls-Royce®

GE

neopost

Vitalograph® 1963-2013 50th ANNIVERSARY
Your respiratory partner

PHILIPS

BOSCH

BMW

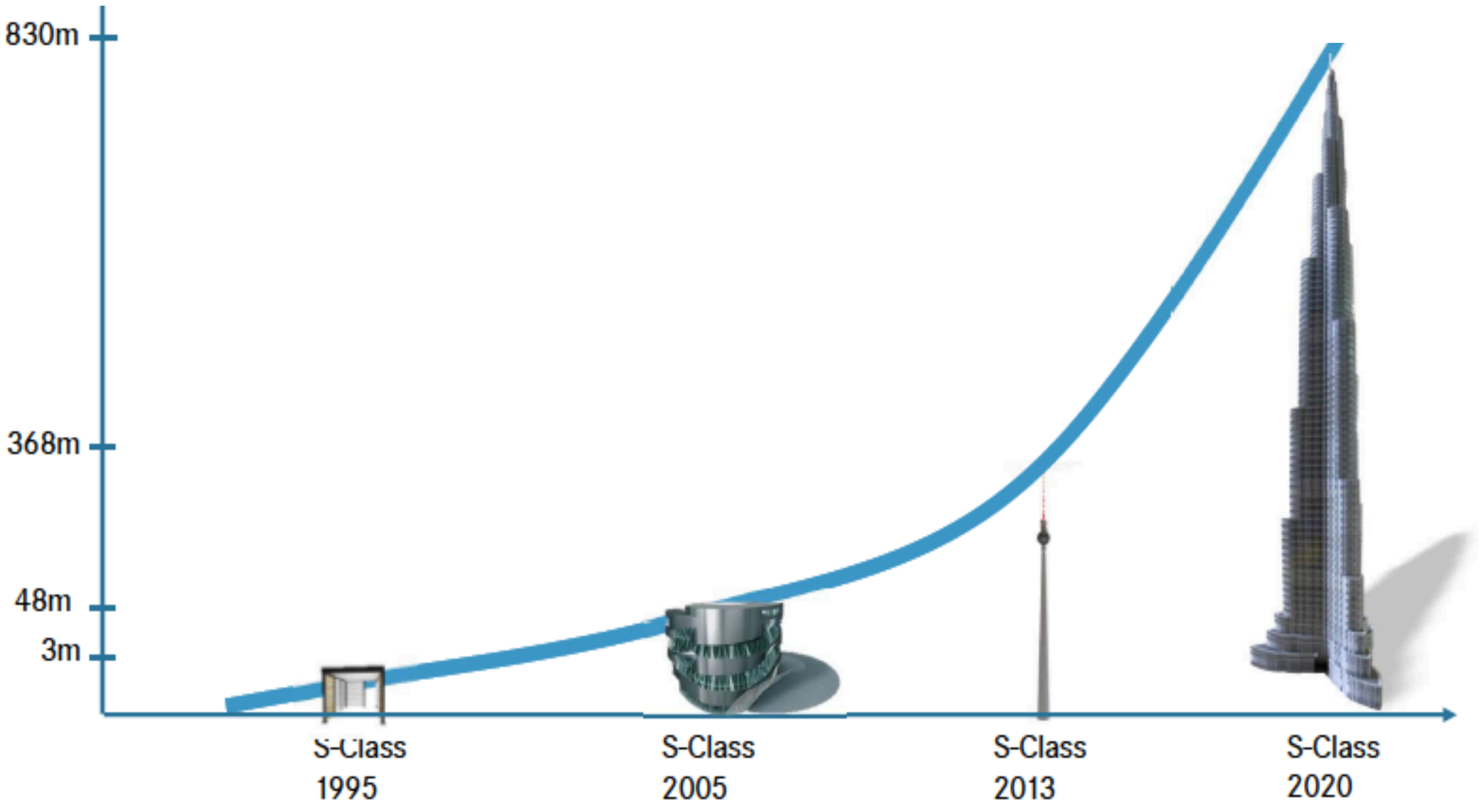
AEROSPACE
Assuring Space Mission Success

“ Every company is becoming a software company

EXAMPLE

Growth of software in Mercedes S-Class

Height of software printout stack in Mercedes S-Class (Schneider 2015)



EXAMPLE

Trends in automotive and aviation

- 1.7m LOC in F-22 Raptor US jet fighter
- 5.7m LOC in F-35 Joint Strike Fighter (JSF)
- 6.5m LOC in Boeing 787 Dreamliner
- 30-50 Electronic Control Units (ECU) in mid-range car
- 70-100 ECUs in high-end car

Software consultancy & services

- Companies that sell their expertise and time to customers that need software-based solutions
  - Outsourcing
- Typical examples include:
  - Dell Services (now NTT)
  - CapGemini
  - Kugler Maag (German automotive consultancy)
- Companies can be domain specific
  e.g. within telecoms, medical, automotive

Individual contractors

- Independent, self-employed consultants who sell their time and expertise.
- Often just blend-in with staff developers.
  - Sometimes perceived to be “outsiders” by company-employed in-house team
- Gives companies flexibility to scale up/down workforce, at a price.

Open source developers

- Open source developers are traditionally volunteers
  —but increasingly employed by companies

- Why?
  - “Itch to scratch”
  - Enjoy hacking
  - Future career prospects (showing off skill)

- More later on open source.

Key Activities of Software Engineering

Recognition in 1960s that software was not simply an afterthought led to a focus on systematic approaches.

Activities of Software Engineering

1. Specification

2. Design

3. Code

4. Quality Assurance

5. Delivery & Deployment

6. Maintenance & Evolution

Specification

Created by Creative Stall
from Noun Project

Define what the system must do.

Typical output:

- Software requirements specification (SRS)

Common typology of requirements:

- Functional: what should the system do?
- Non-functional: constraints on the type of solutions, e.g. performance, security, usability
    - So-called -ilities a.k.a. Quality Attributes

EXAMPLE

Neopost Inserter machine

EXAMPLE

Example: Inserter machine

Functional requirements:

- Select paper sheets
  - How many?
  - Which order?
  - Different sets for different recipients?
- Fold them
  - Zero, once or twice?
- Insert into envelope
  - Which order?
- Close the envelope
- Frank the envelope
  - Which value? Depends on destination!

Non-functional requirements

- E.g. 2 sheets per second

Design

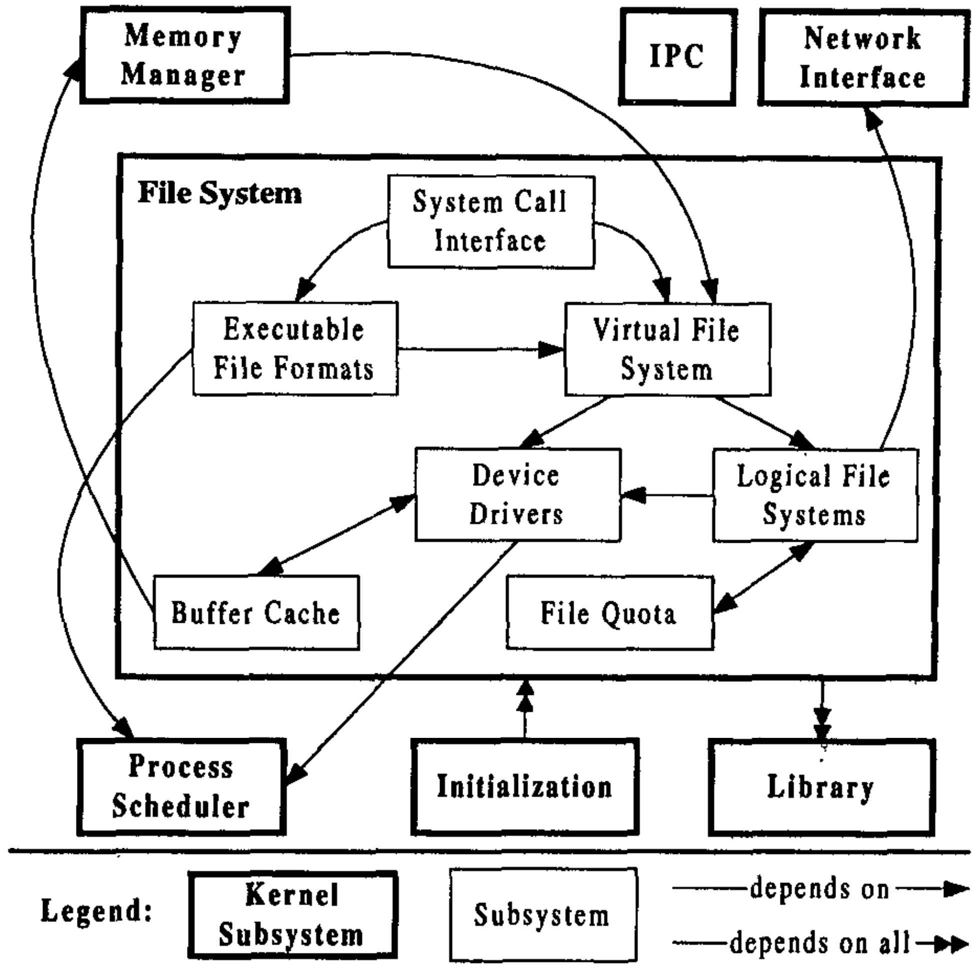
Create a design for the software that satisfies the requirements.

Typical outputs:

- High-Level or Architecture design
- Detailed design documents

EXAMPLE

Architecture of Linux



Source:

Bowman et al.: "Linux as a Case Study: Its Extracted Software Architecture."

In Proceedings of the International Conference on Software Engineering, 1999.

Coding

Created by Creative Stall
from Noun Project

Implement the software.

Expected outputs:

- Software
- Test suite (potentially—more on this later)
- Documentation (ideally!)

Business Information Systems (BIS) people use the verb to implement to mean installation of a system at an organisation.

Quality Assurance

Test the software.

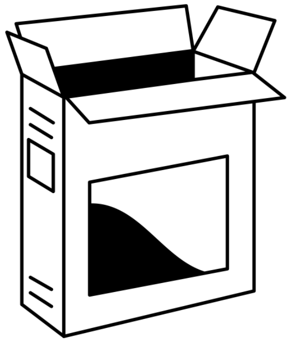
Typical outputs:

- Unit tests / test suite
- Integration tests
- System tests
- Test results or report

Synonyms:

- Verification & Validation (V&V)
- Quality Assurance (QA)
- Testing

Delivery and deployment


Created by Wing from Noun Project

Deliver or deploy the software.

Typical outputs:

- Running system on customer's site
  (in case of bespoke development)
- Running system on a server
  (e.g. Facebook and Google)
- Installable software on some media
  (in case of market-driven development,
  e.g. launch of Windows 95)

Maintenance & Evolution

Created by BomSymbols
from Noun Project

Adapt and maintain the software.

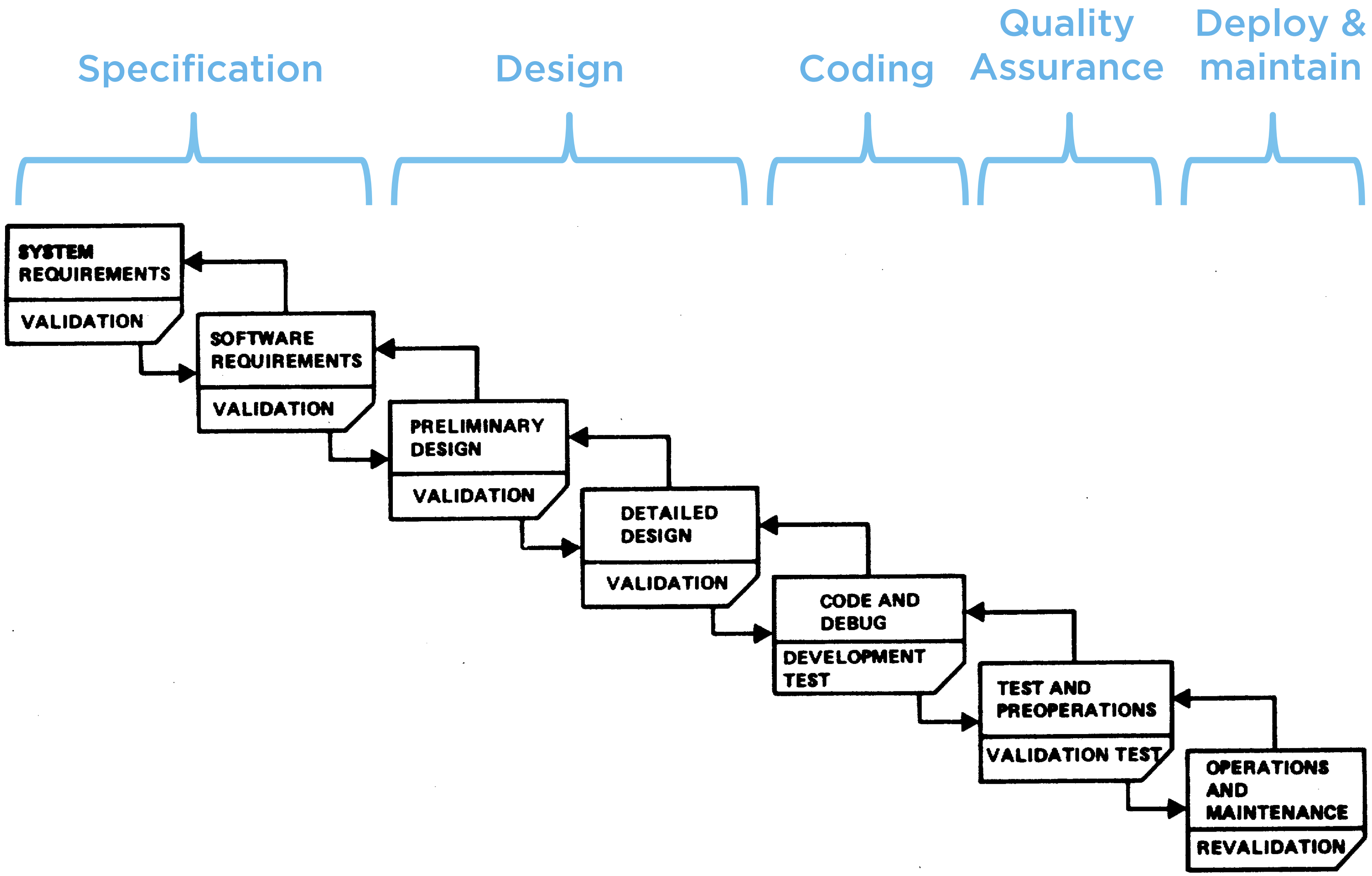
Typical outputs:

- Updated specification (hopefully!)
- Test results
- New version of the software

A Rational Process for Software Development?

Now we know the key activities, but, we need a process to coordinate all people involved, including business managers, architects, developers, testers, and operations staff.

Waterfall model



Waterfall model

- When people speak of traditional approaches, they usually mean waterfall (or a variant)
- Also called Plan-driven
- Rational logic, not empirical logic
- Very sensitive to changes downstream in the process
- There are better ways to do it—discussed later!

Summary

- Software projects are bespoke or market-driven.
- Software projects are ‘brownfield’ or ‘greenfield’
- Software can be deployed as embedded system, stand-alone system, or as a distributed system.
- Software is made by software houses, non-software companies, consultancy & service companies, individual contractors, and OSS devs.
- Activities of software engineering comprise specification, design, coding, testing, deployment, and maintenance & evolution.

Thank you for your attention

Questions & suggestions can be sent to:
k.stol@cs.ucc.ie