Chapter 3

System Development Life Cycle (SDLC)

Evolution of Methodologies

- Methodologies have not sprung to life spontaneously
- Here we provide a brief historical perspective up to SDLC
- The SDLC (or waterfall model) has had a great impact on other methodologies
- Therefore we are going to have a closer look

Early Years (1940s-1950s)

- Shift from scientific/mathematical to business applications
- Programmer is not always user anymore
- Shift from technical viewpoint to more general view

Early Techniques (1960s)

- Need for analysis/design phase in addition to coding phase
- Examples of early analysis techniques
  - Accurately Defined Systems (ADS) by NCR
  - Time Automated Grid (TAG) by IBM
- Not widely used; complex and not well supported
- Many principles incorporated into later approaches:
  - Starting with output, deriving necessary input
  - Graphical representation of system structures
  - Capture requirements in formal specification

Early Techniques(2)

- SDLC makes first appearance
- Known earlier as approach to problem solving in other disciplines
- Now applied to systems development
- Top-down approach (going from overall view into technical details)
- Helped eliminate premature coding (first analysis, then design, then coding)

Structured Approach (1970s)

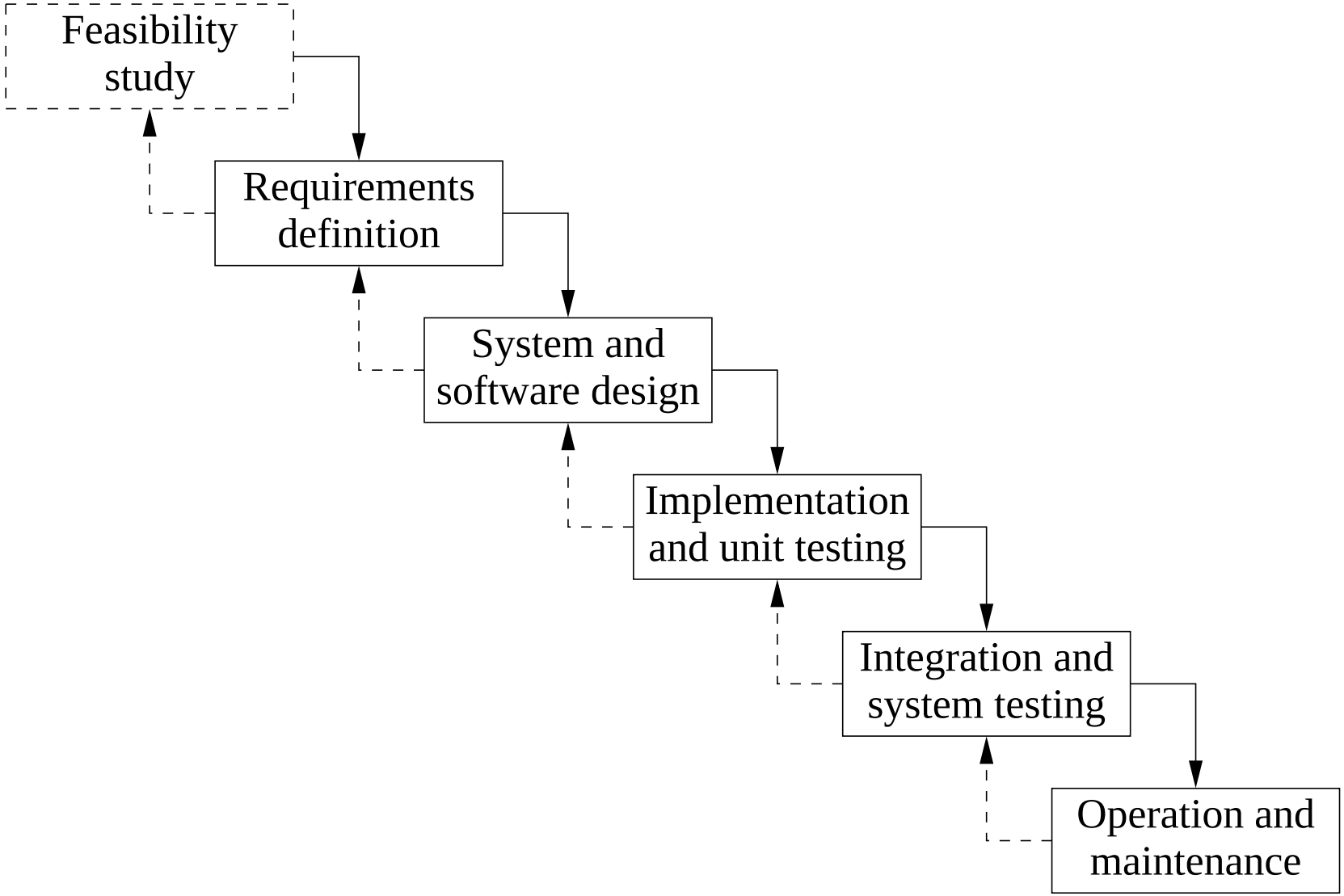
- Based on waterfall life cycle
- Specification, design and coding map well to structured analysis, structured design, structured programming
- Top-down approach
- Ironically, evolved bottom-up:
  - Structured programming
  - Structured design
  - Structured analysis

SDLC

- System Development Life Cycle (SDLC) is the most well-known of the so-called phase models
- Still in use today, although rarely found in pure form
- Process divided into five to eight distinctive phases*
- Each phase has as output documentation/programs

* Your mileage may vary

Phases


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

Feasibility Study

- Feasibility study tries to find out
  - if it's possible to build a certain system
  - and if it's possible to do so at a reasonable price
- Output: decision to go ahead or stop project

Requirements

- Requirements definition:
  - System’s services, constraints, and goals are established
  - Heavy consultation of end users/customers
- Output: system specification

Design

- System and software design:
  - Partitions requirements into hardware or software systems
  - Establishes overall system architecture
- Output: design documentation

Implementation

- Implementation and unit testing:
  - Actual code gets written or generated
  - Verifying that each part meets specification
- Output: set of programs or program units

Integration

- Integration and system testing
  - Individual program units are put together
  - System is tested as a whole (interaction between units)
- Output: system that is ready to be delivered

Operation

- Operation and maintenance
  - Should be longest life cycle phase
  - System is installed and put into use
  - Correcting errors that were not caught before
  - Improving system in the light of changes
- Output: improved or enhanced system

Techniques

- Various techniques for documentation:
  - Flowcharts
  - Organizational charts
  - Manuals
  - Grid charts
  - Discussion records
  - File/record specifications

Tools

- There is also a great number of tools available
  - Project management tools
  - Report generating tools
  - Tools for producing documentation
  - Tools for generating code

Strengths of SDLC

- Methodologies incorporating this approach have been well tried and tested
- Divides development into distinct phases:
  - Makes tasks more manageable
  - Offers opportunity for more control over development process
- Provides standards for documentation
- Much better than trial and error

Weaknesses of SDLC

- Fails to see the “big picture” of strategic management
- Too inflexible to cope with changing requirements
- Emphasis on “hard” thinking (which is often reflected in documentation that is too technical)
- Unable to capture true needs of users

Chapter Summary

- Traditional phase model like SDLC better than no methodology
- Has been used successfully in many projects
- SDLC may be outdated*, but used as basis of numerous other methodologies
- Developers have tried to fix some of its weaknesses
  - while retaining basic SDLC
  - coming up with totally different approaches

* Outdated $\neq$ bad: ancient Roman buildings, medieval cathedrals