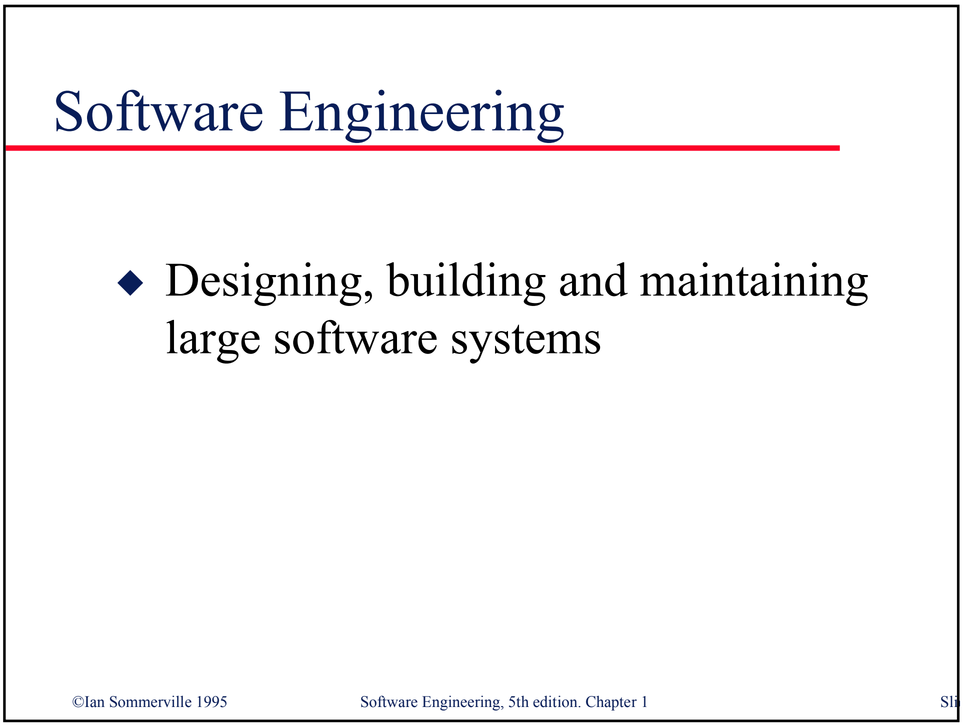
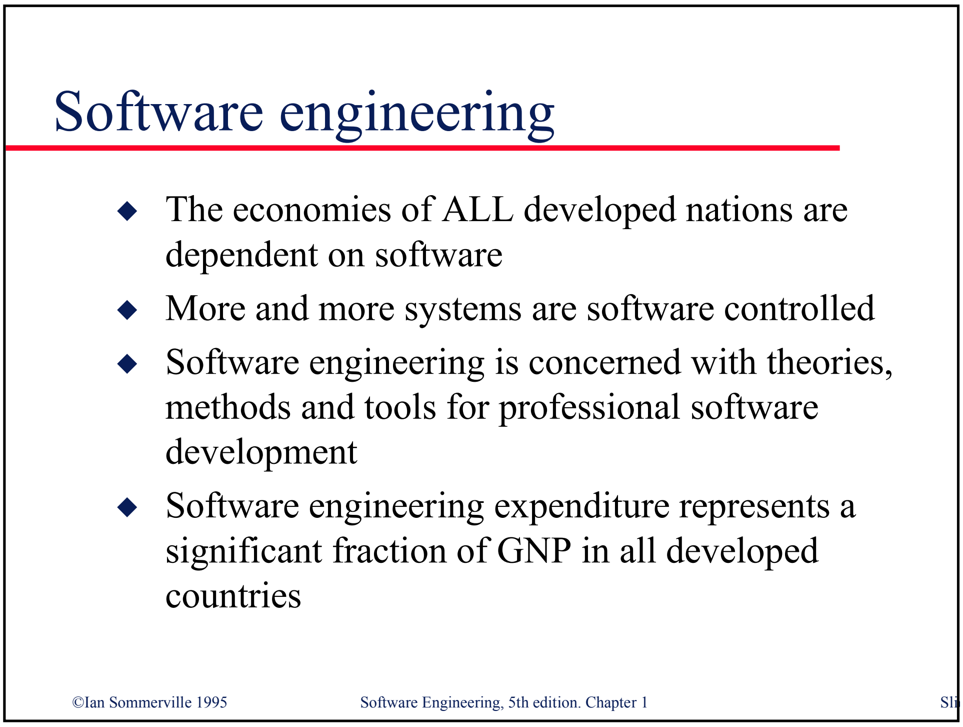


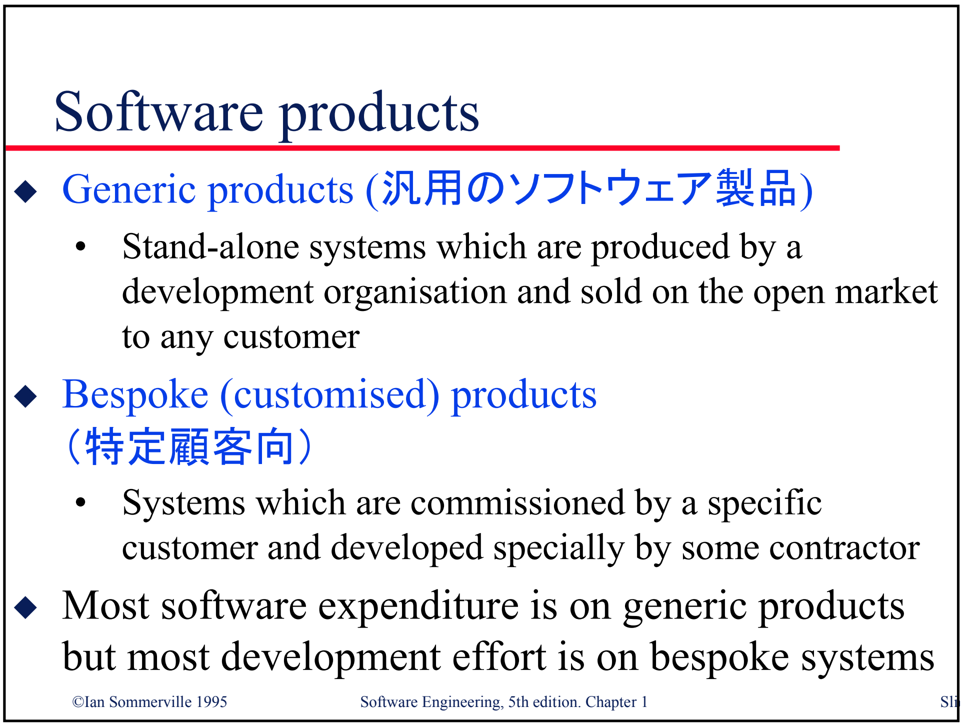
Software Engineering

- Designing, building and maintaining large software systems



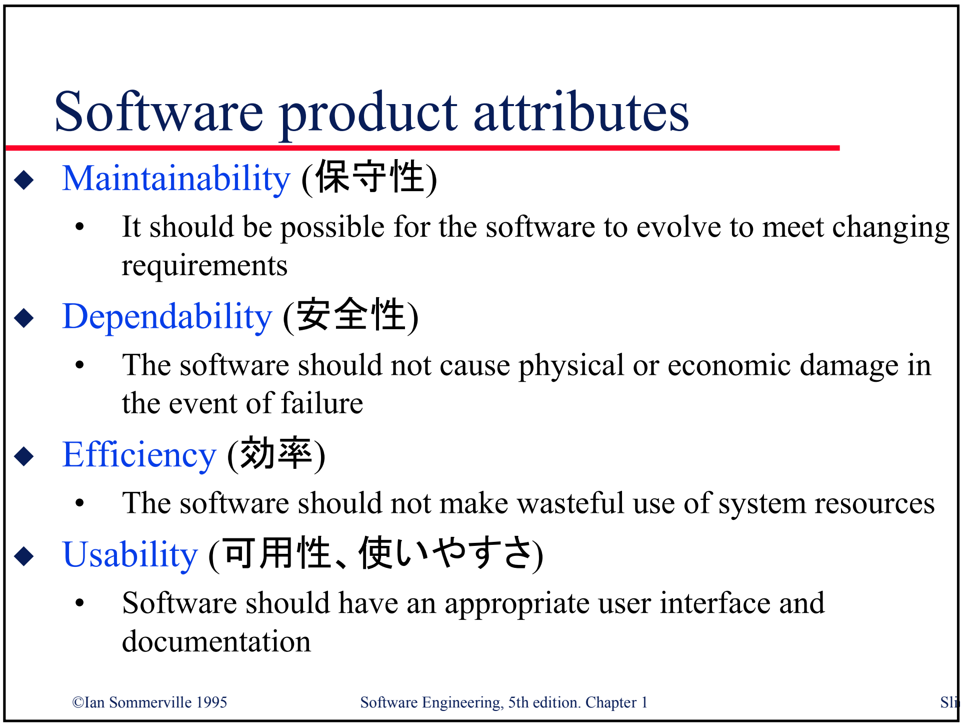
Software engineering

- The economies of ALL developed nations are dependent on software
- More and more systems are software controlled
- Software engineering is concerned with theories, methods and tools for professional software development
- Software engineering expenditure represents a significant fraction of GNP in all developed countries



Software products

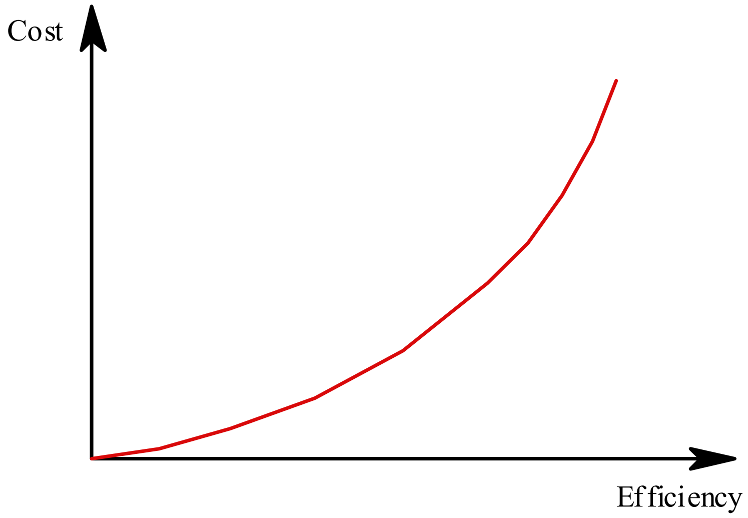
- Generic products (汎用のソフトウェア製品)
  - Stand-alone systems which are produced by a development organisation and sold on the open market to any customer
- Bespoke (customised) products （特定顧客向）
  - Systems which are commissioned by a specific customer and developed specially by some contractor
- Most software expenditure is on generic products but most development effort is on bespoke systems



Software product attributes

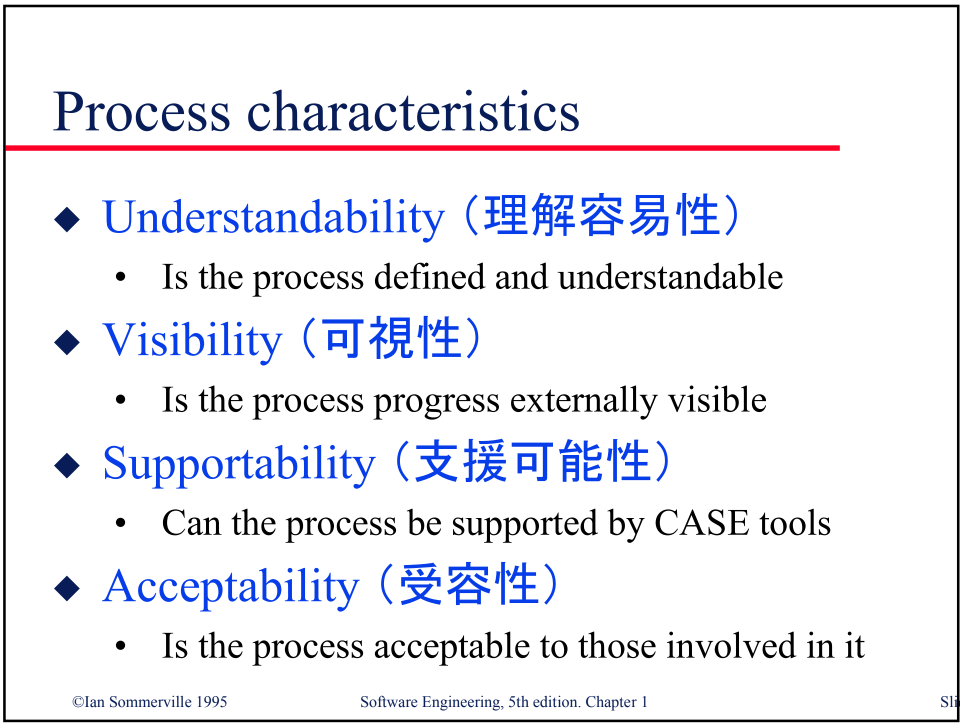
- Maintainability (保守性)
  - It should be possible for the software to evolve to meet changing requirements
- Dependability (安全性)
  - The software should not cause physical or economic damage in the event of failure
- Efficiency (効率)
  - The software should not make wasteful use of system resources
- Usability (可用性、使いやすさ)
  - Software should have an appropriate user interface and documentation

Efficiency costs



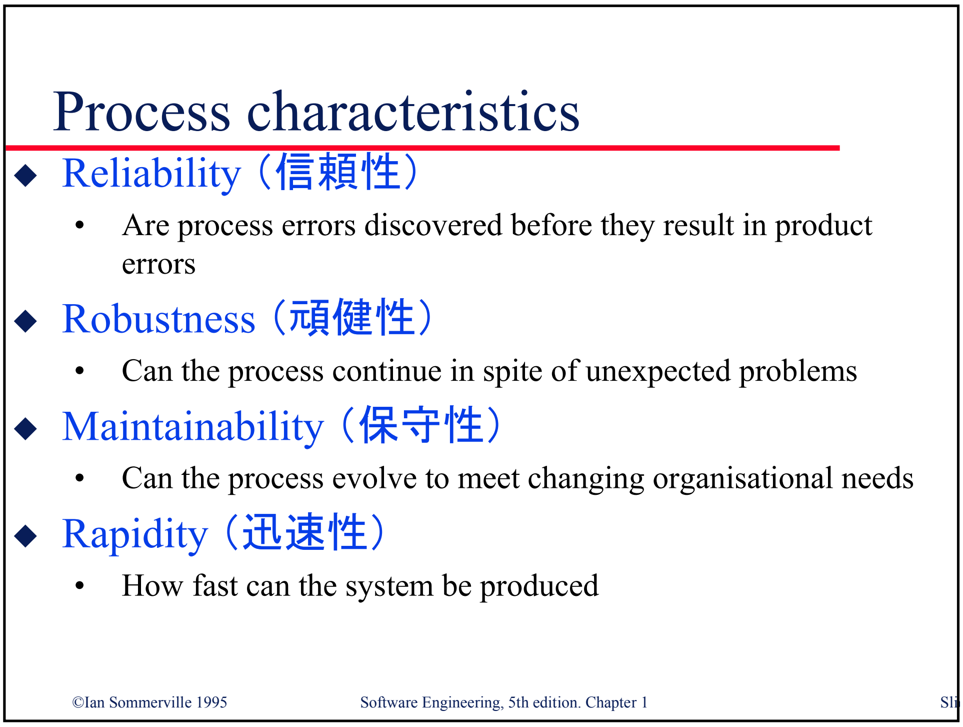
The software process

- Structured set of activities required to develop a software system
  - Specification（仕様化）
  - Design（設計）
  - Validation（検証）
  - Evolution（発展）
- Activities vary depending on the organisation and the type of system being developed
- Must be explicitly modelled if it is to be managed



Process characteristics

- Understandability（理解容易性）
    - Is the process defined and understandable
- Visibility（可視性）
    - Is the process progress externally visible
- Supportability（支援可能性）
    - Can the process be supported by CASE tools
- Acceptability（受容性）
    - Is the process acceptable to those involved in it



Process characteristics

- Reliability（信頼性）
    - Are process errors discovered before they result in product errors
- Robustness（頑健性）
    - Can the process continue in spite of unexpected problems
- Maintainability（保守性）
    - Can the process evolve to meet changing organisational needs
- Rapidity（迅速性）
    - How fast can the system be produced

Engineering process model

- Specification（仕様化）- set out the requirements and constraints on the system
- Design（設計）- Produce a paper model of the system
- Manufacture（製造） - build the system
- Test（テスト）- check the system meets the required specifications
- Install（インストール） - deliver the system to the customer and ensure it is operational
- Maintain（保守） - repair faults in the system as they are discovered

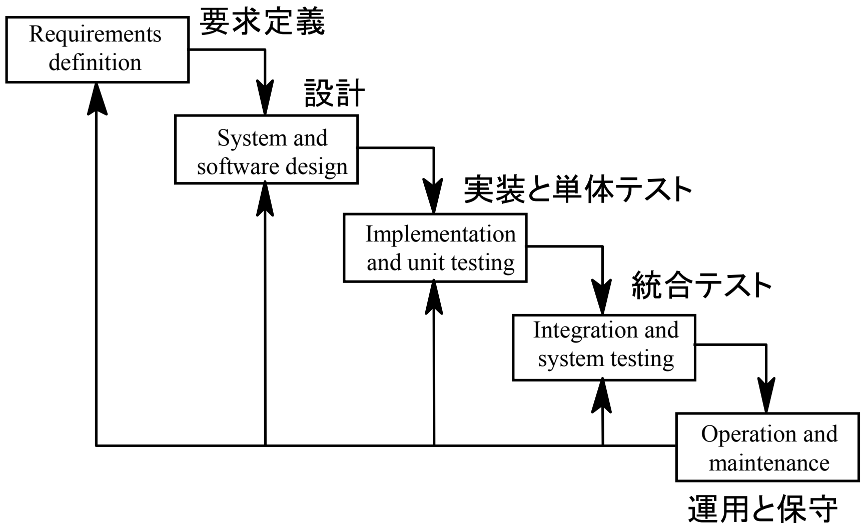
Software process models

- Normally, specifications are incomplete/anomalous
- Very blurred distinction between specification, design and manufacture
- No physical realisation of the system for testing
- Software does not wear out - maintenance does not mean component replacement

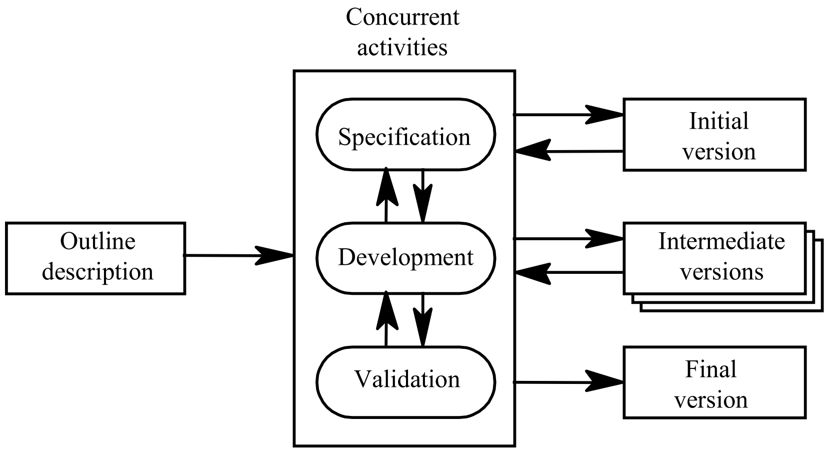
Generic software process models

- The waterfall model（ウォーターフォールモデル）
  - Separate and distinct phases of specification and development
- Evolutionary development（発展型の開発）
  - Specification and development are interleaved
- Formal transformation（形式的変換）
  - A mathematical system model is formally transformed to an implementation
- Reuse-based development（再利用に基づく開発）
  - The system is assembled from existing components

Waterfall model


要求定義
Requirements definition
設計
System and software design
実装と単体テスト
Implementation and unit testing
統合テスト
Integration and system testing
Operation and maintenance
運用と保守

Evolutionary development



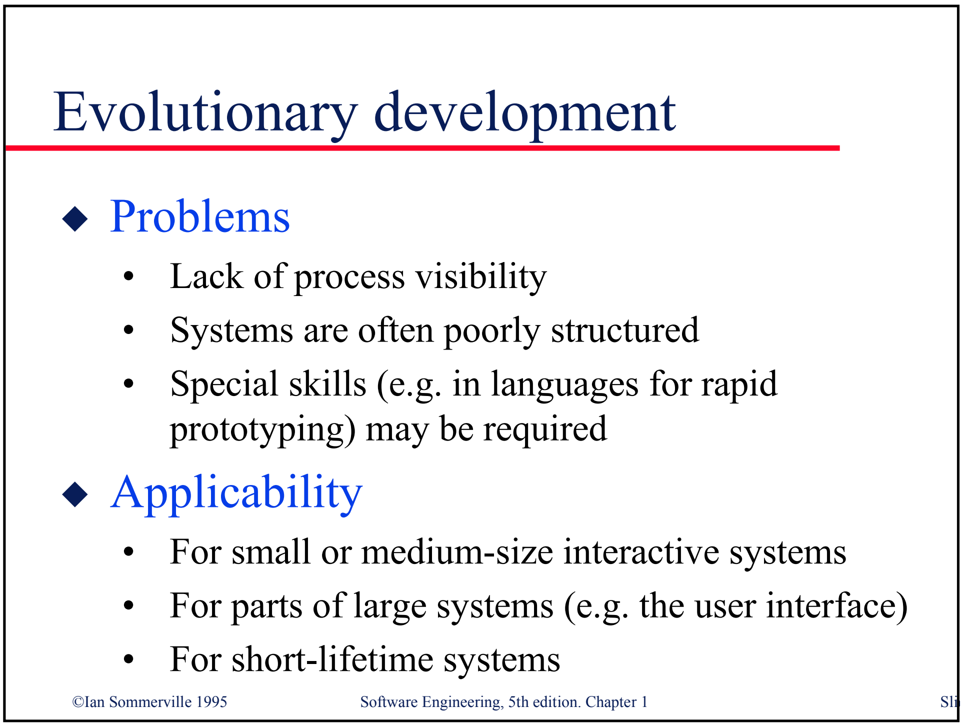
Evolutionary development

◆ Exploratory prototyping （探索型プロトタイピング？）

- Objective is to work with customers and to evolve a final system from an initial outline specification. Should start with well-understood requirements

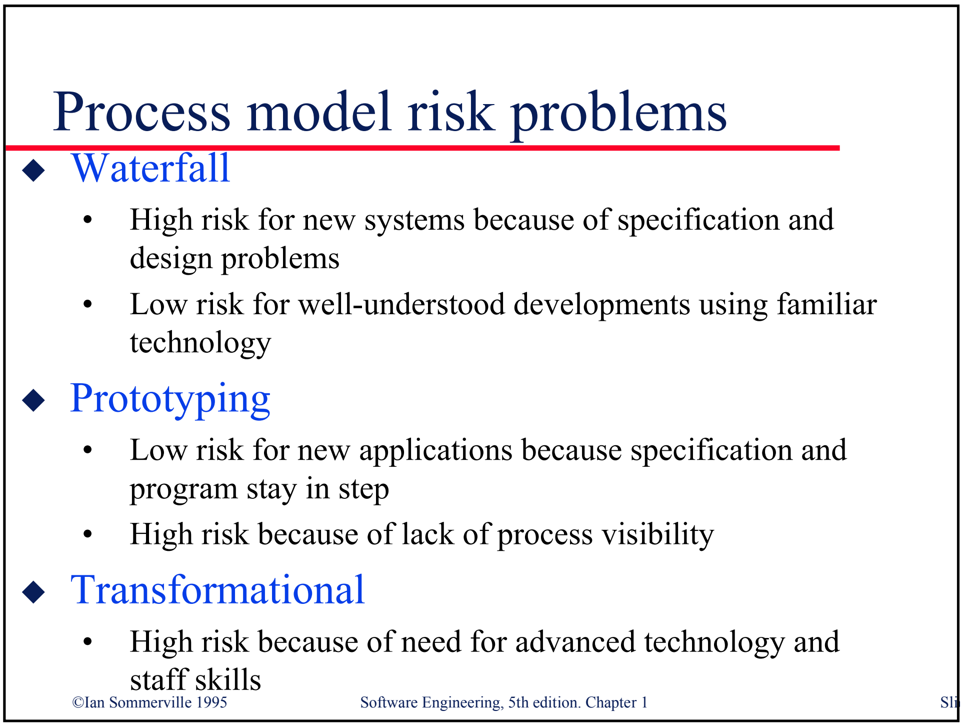
◆ Throw-away prototyping （使い捨て型プロトタイピング）

- Objective is to understand the system requirements. Should start with poorly understood requirements



Evolutionary development

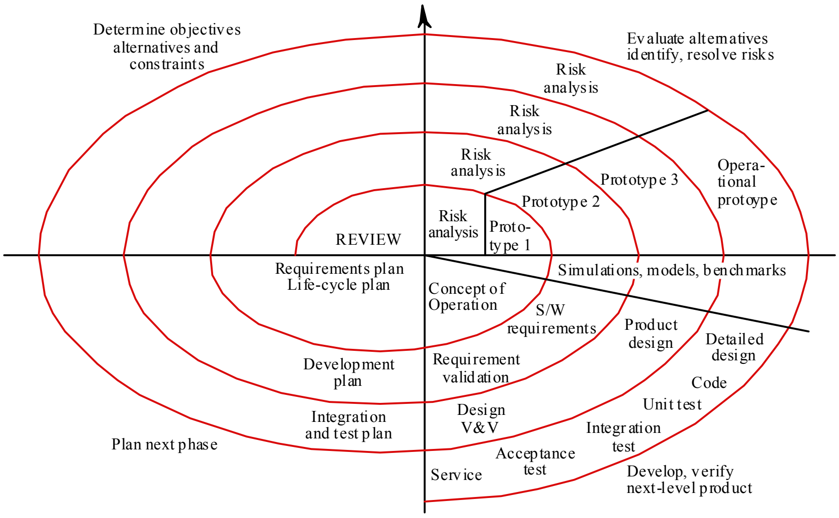
- Problems
  - Lack of process visibility
  - Systems are often poorly structured
  - Special skills (e.g. in languages for rapid prototyping) may be required
- Applicability
  - For small or medium-size interactive systems
  - For parts of large systems (e.g. the user interface)
  - For short-lifetime systems



Process model risk problems

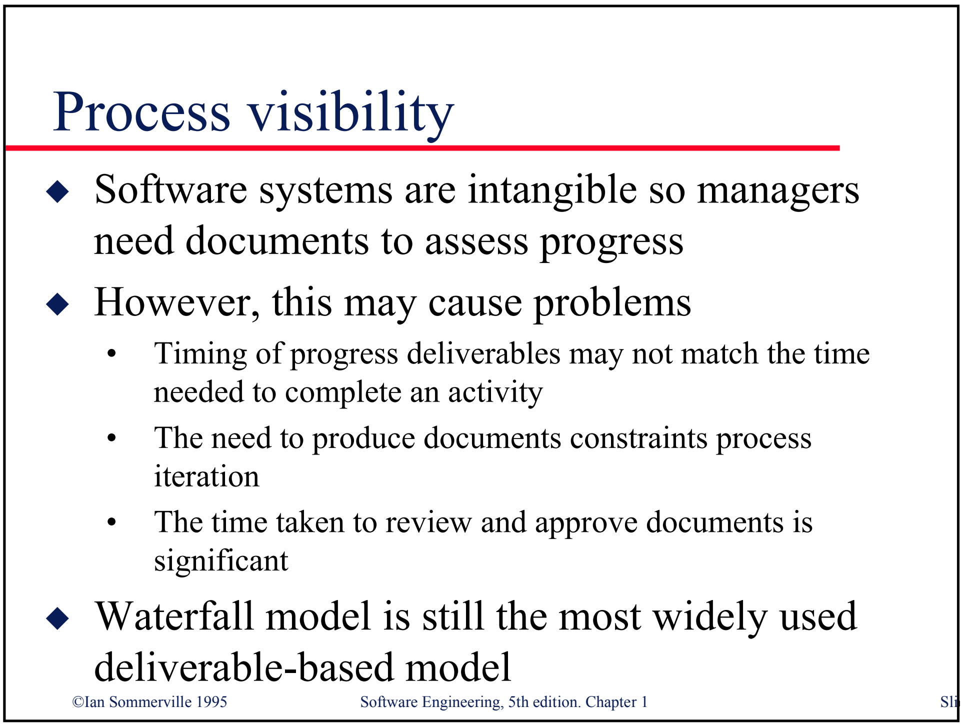
- Waterfall
  - High risk for new systems because of specification and design problems
  - Low risk for well-understood developments using familiar technology
- Prototyping
  - Low risk for new applications because specification and program stay in step
  - High risk because of lack of process visibility
- Transformational
  - High risk because of need for advanced technology and staff skills

Spiral model of the software process



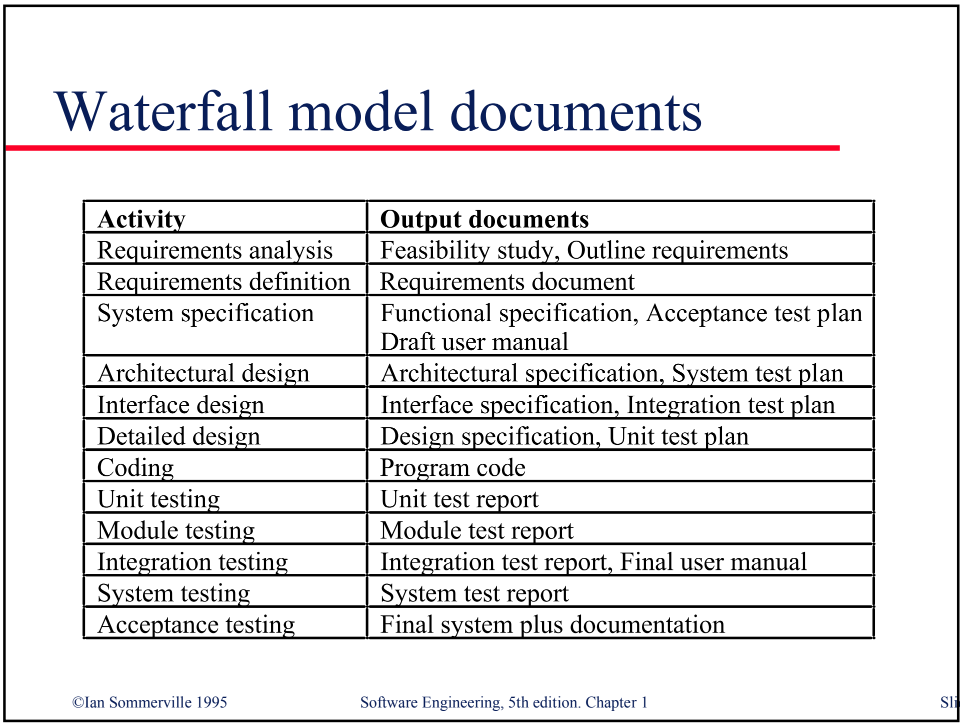
Phases of the spiral model

- Objective setting
  - Specific objectives for the project phase are identified
- Risk assessment and reduction
  - Key risks are identified, analysed and information is sought to reduce these risks
- Development and validation
  - An appropriate model is chosen for the next phase of development.
- Planning
  - The project is reviewed and plans drawn up for the next round of the spiral



Process visibility

- Software systems are intangible so managers need documents to assess progress
- However, this may cause problems
  - Timing of progress deliverables may not match the time needed to complete an activity
  - The need to produce documents constraints process iteration
  - The time taken to review and approve documents is significant
- Waterfall model is still the most widely used deliverable-based model



Waterfall model documents

Activity	Output documents
Requirements analysis	Feasibility study, Outline requirements
Requirements definition	Requirements document
System specification	Functional specification, Acceptance test plan Draft user manual
Architectural design	Architectural specification, System test plan
Interface design	Interface specification, Integration test plan
Detailed design	Design specification, Unit test plan
Coding	Program code
Unit testing	Unit test report
Module testing	Module test report
Integration testing	Integration test report, Final user manual
System testing	System test report
Acceptance testing	Final system plus documentation

Process model visibility

Process model	Process visibility
Waterfall model	Good visibility, each activity produces some deliverable
Evolutionary development	Poor visibility, uneconomic to produce documents during rapid iteration
Formal transformations	Good visibility, documents must be produced from each phase for the process to continue
Reuse-oriented development	Moderate visibility, it may be artificial to produce documents describing reuse and reusable components.
Spiral model	Good visibility, each segment and each ring of the spiral should produce some document.

Ethical issues

- Confidentiality
- Competence
- Intellectual property rights
- Computer misuse

Key points

- Software engineering is concerned with the theories, methods and tools for developing, managing and evolving software products
- Software products consist of programs and documentation. Product attributes are maintainability, dependability, efficiency and usability
- The software process consists of those activities involved in software development

Key points

- The waterfall model considers each process activity as a discrete phase
- Evolutionary development considers process activities as concurrent
- The spiral process model is risk-driven
- Process visibility involves the creation of deliverables from activities
- Software engineers have ethical, social and professional responsibilities