

# Software Engineering

## Software Reuse

Based on Software Engineering, 7<sup>th</sup> Edition by Ian Sommerville

Stan Kurkovsky

### Objectives

- To explain the benefits of software reuse and some reuse problems
- To discuss several different ways to implement software reuse
- To explain how reusable concepts can be represented as patterns or embedded in program generators
- To discuss COTS reuse
- To describe the development of software product lines

Stan Kurkovsky

## Software reuse

- In most engineering disciplines, systems are designed by composing existing components that have been used in other systems.
- Software engineering has been more focused on original development but it is now recognised that to achieve better software, more quickly and at lower cost, we need to adopt a design process that is based on *systematic software reuse*.
- Application system reuse
  - The whole of an application system may be reused either by incorporating it without change into other systems (COTS reuse) or by developing application families.
- Component reuse
  - Components of an application from sub-systems to single objects may be reused. Covered in Chapter 19.
- Object and function reuse
  - Software components that implement a single well-defined object or function may be reused.

Stan Kurkovsky

## Reuse benefits and problems

### Benefits

- Increased dependability
- Reduced process risk
- Effective use of specialists
- Standards compliance
- Accelerated development

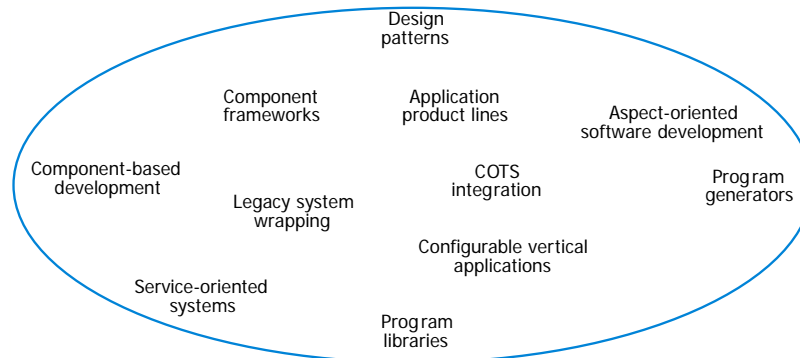
### Problems

- Increased maintenance costs
- Lack of tool support
- Not-invented-here syndrome
- Creating and maintaining a component library
- Finding, understanding and adapting reusable components

Stan Kurkovsky

## The reuse landscape

- Although reuse is often simply thought of as the reuse of system components, there are many different approaches to reuse that may be used.
- Reuse is possible at a range of levels from simple functions to complete application systems.
- The reuse landscape covers the range of possible reuse techniques.



Stan Kurkovsky

## Concept reuse

- When you reuse program or design components, you have to follow the design decisions made by the original developer of the component.
- This may limit the opportunities for reuse.
- However, a more abstract form of reuse is concept reuse when a particular approach is described in an implementation independent way and an implementation is then developed.
- The two main approaches to concept reuse are:
  - Design patterns
  - Generative programming

Stan Kurkovsky

## Design patterns

- A design pattern is a way of reusing abstract knowledge about a problem and its solution.
- A pattern is a description of the problem and the essence of its solution.
- It should be sufficiently abstract to be reused in different settings.
- Patterns often rely on object characteristics such as inheritance and polymorphism.

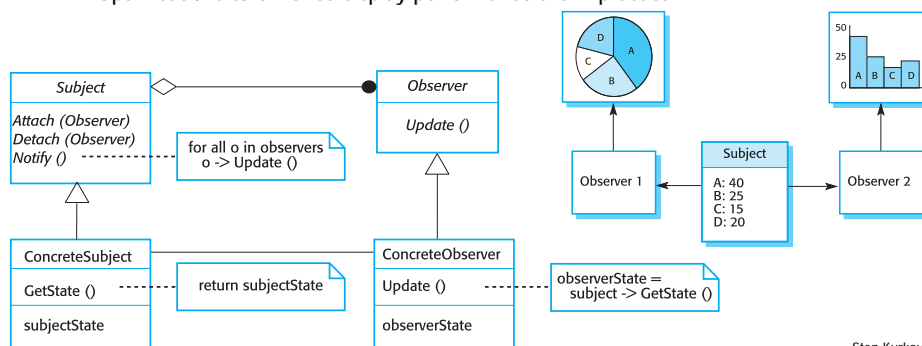
### Pattern elements

- Name
  - A meaningful pattern identifier.
- Problem description.
- Solution description.
  - Not a concrete design but a template for a design solution that can be instantiated in different ways.
- Consequences
  - The results and trade-offs of applying the pattern.

Stan Kurkovsky

## The Observer pattern

- Name
  - Observer.
- Description
  - Separates the display of object state from the object itself.
- Problem description
  - Used when multiple displays of state are needed.
- Solution description
  - See slide with UML description.
- Consequences
  - Optimisations to enhance display performance are impractical.



Stan Kurkovsky

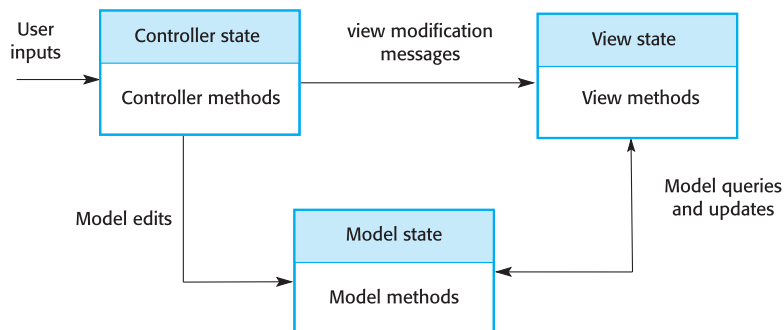
## Application frameworks

- Frameworks are a sub-system design made up of a collection of abstract and concrete classes and the interfaces between them.
- The sub-system is implemented by adding components to fill in parts of the design and by instantiating the abstract classes in the framework.
- Frameworks are moderately large entities that can be reused.
- System infrastructure frameworks
  - Support the development of system infrastructures such as communications, user interfaces and compilers.
- Middleware integration frameworks
  - Standards and classes that support component communication and information exchange.
- Enterprise application frameworks
  - Support the development of specific types of application such as telecommunications or financial systems.

Stan Kurkovsky

## Model-view controller

- System infrastructure framework for GUI design.
- Allows for multiple presentations of an object and separate interactions with these presentations.
- MVC framework involves the instantiation of a number of patterns (as discussed earlier under concept reuse).



Stan Kurkovsky

## Application system reuse

- Involves the reuse of entire application systems either by configuring a system for an environment or by integrating two or more systems to create a new application.
- COTS - Commercial Off-The-Shelf systems
  - COTS systems are usually complete application systems that offer an API (Application Programming Interface).
  - Building large systems by integrating COTS systems is now a viable development strategy for some types of system such as E-commerce systems.
  - The key benefit is faster application development and, usually, lower development costs.
- Software product lines
  - Software product lines or application families are applications with generic functionality that can be adapted and configured for use in a specific context.
  - Adaptation may involve:
    - Component and system configuration;
    - Adding new components to the system;
    - Selecting from a library of existing components;
    - Modifying components to meet new requirements.

Stan Kurkovsky

## Key points

- Advantages of reuse are lower costs, faster software development and lower risks.
- Design patterns are high-level abstractions that document successful design solutions.
- Program generators are also concerned with software reuse - the reusable concepts are embedded in a generator system.
- Application frameworks are collections of concrete and abstract objects that are designed for reuse through specialisation.
- COTS product reuse is concerned with the reuse of large, off-the-shelf systems.
- Problems with COTS reuse include lack of control over functionality, performance, and evolution and problems with inter-operation.
- ERP systems are created by configuring a generic system with information about a customer's business.
- Software product lines are related applications developed around a common core of shared functionality.

Stan Kurkovsky