Process Models: Specialized Process Models

LECTURE # 10

Lecture by Eng. Sidra

Advantages of Formal Methods Models

- The development of a formal specification provides insights and understanding of the software requirements and the software design
- Discovers ambiguity, incompleteness, and inconsistency in the software.
- Offers defect-free software.
- Incrementally grows in effective solution after each iteration.
- Formal specification may be automatically processed. Software tools can be built to assist with their development, understanding, and debugging.

Advantages of Formal Methods Models

- Formal Methods concentrate on consistent, correct models
- Depending on the formal specification language being used, it may be possible to animate a formal system specification to provide a prototype system.
- Formal specifications are mathematical entities and may be studied and analyzed using mathematical methods.
- Formal specifications may be used as a guide to the tester of a component in identifying appropriate test cases.

Disadvantages of Formal Methods Model

- Time consuming and expensive.
- Difficult to use this model as a communication mechanism for non technical personnel.
- Extensive training is required since only few developers have the essential knowledge to implement this model.
- They include too much detail
- Most of the time your models are inconsistent, incorrect, incomplete
- People get confused about which tools are appropriate:
- Formal methods require more effort ...and the payoff is deferred

Aspect-Oriented Software Development (AOSD)

- It is a software design solution that helps address the modularity issues that are not properly resolved by other software approaches, like procedural, structured and object-oriented programming (OOP).
- Aspect-oriented software development (AOSD), often referred to as aspect-oriented programming(AOP), is a relatively new software engineering paradigm that provides a process and methodological approach for defining, specifying, designing, and constructing aspects—"mechanisms beyond subroutines and inheritance for localizing the expression of a crosscutting concern".

Aspect-Oriented Software Development (AOSD)

- AOP doesn't replace existing programming paradigms and languages.
- Instead it works with them to improve their expressiveness and utility.
- AOP is designed to handle crosscutting concerns by providing a mechanism known as aspect.

Aspect-Oriented Software Development (AOSD)

- Computer systems are better programmed by separately specifying the various "concerns"
- **Concerns :**
  - Required properties or areas of technical interests,
  - behavior that we want to have in a particular module of an application.
  - may be defined as a functionality we want to implement.
- **High-level** – security, QoS
- **Low-level** – caching, buffering
- **Functional** – features, business rules
- **Non Functional (systemic**) – memory management, transaction management.

System = set of “concerns”

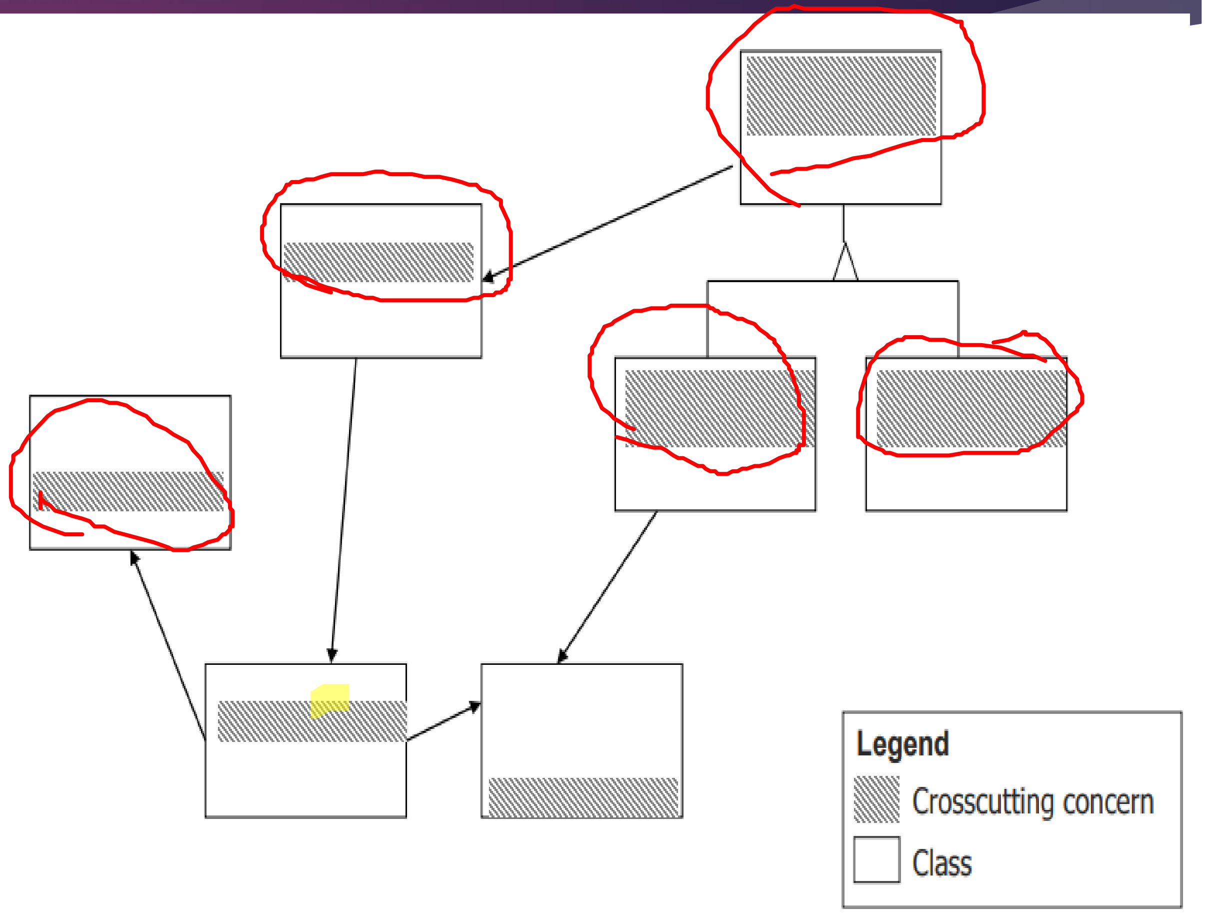
- A typical system may consist of several kind of concerns including
  - Business logic
  - Performance
  - Data persistence
  - Logging and Debugging
  - Authentication
  - Security
  - Multi-threaded safety
  - Error-checking

Concerning “Concerns”

- They are concerns that are implemented by several modules of the system and mix up with other concerns.
- When concerns cut across multiple system functions, features, and information, they are often referred to as crosscutting concerns.
- It is applicable throughout the application and it affects the entire application.
- It is hard to change, remove or evolve a crosscutting concern of the system
- Two typical problems
  - Scattering
  - Tangling

Two problems with Crosscutting Concerns

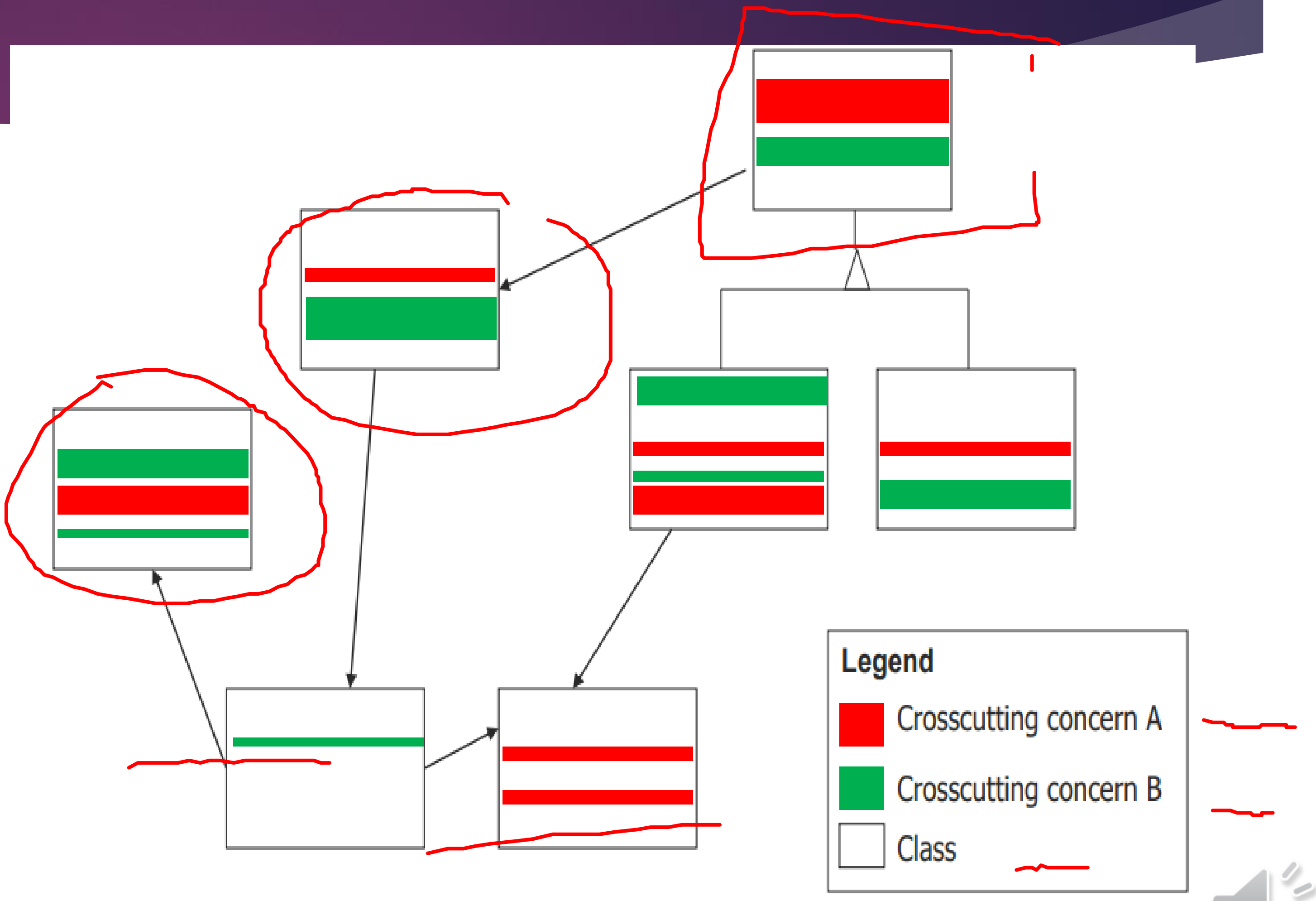
1. **Code Scattering**
   - Implementation for authentication, contract checking and logging is not localized.
   - Code spans over many methods of potentially many classes and packages.



Two problems with Crosscutting Concerns

2. Code tangling

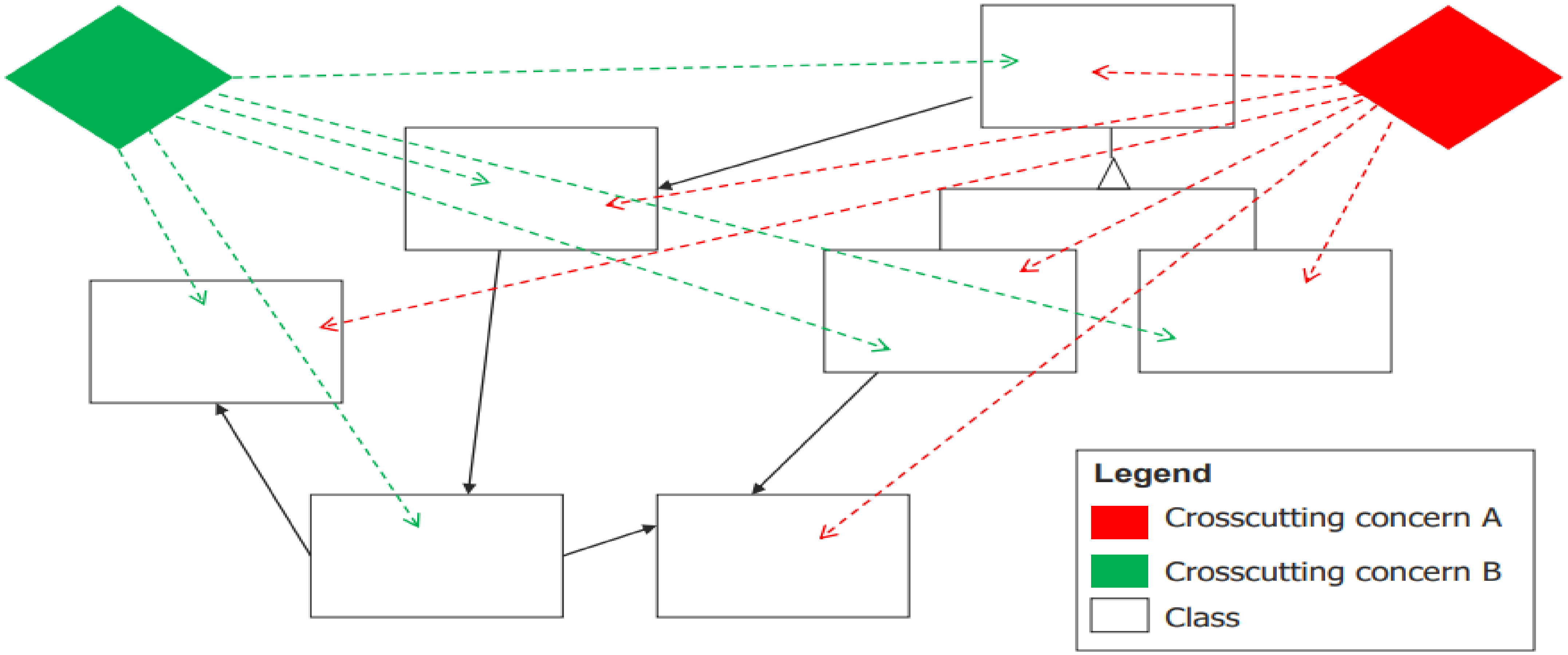
- Implementation of some Operation() does much more than performing some core functionality. It contains code for more than one concerns.



Aspect-Oriented Software Development (AOSD)

- AOSD, in brief, focuses first on identification, specification and also on the representation of concerns which are cross-cutting.
- It also involves the modularization of these concerns into distinct functional units known as aspects,
- Aspects
  - Express these concerns and automatically incorporates them into a system.
  - Enhances the ability to express separation of these concerns.
  - Leads to well-designed, maintainable software system.
  - localization can be promoted.
- and finally their automation to work together as a complete system
- This results in better support for modularization hence reducing development, maintenance and evolution costs.

Solution with Aspects



Advantages of AOSD

- It gives complementary benefits and can be used along with other coding standards.
- It is a better way to handle localization of cross-cutting concerns as can be they compressed into various modules.
- It ensures improved and smaller size of code as a result of addressing the cross-cutting issues
- It encourages the reuse of code created by the modularization technique.
- It improves the maintenance and understanding of the code as it modularizes cross-cutting concerns

Disadvantages of AOSD

- A new development technique
  - It requires learning and training (a new way of thinking)
- Code fragmentation
  - Several small classes and aspects
- Lack of tool support
  - There are a few tools and IDE extensions
- Code bloating
  - As small source can tip to much bigger object code.