AMERICAN UNIVERSITY OF IRAQ SULAIMANI

Detailed Software Overview

Dr. Hoger Mahmud | 2022

Content

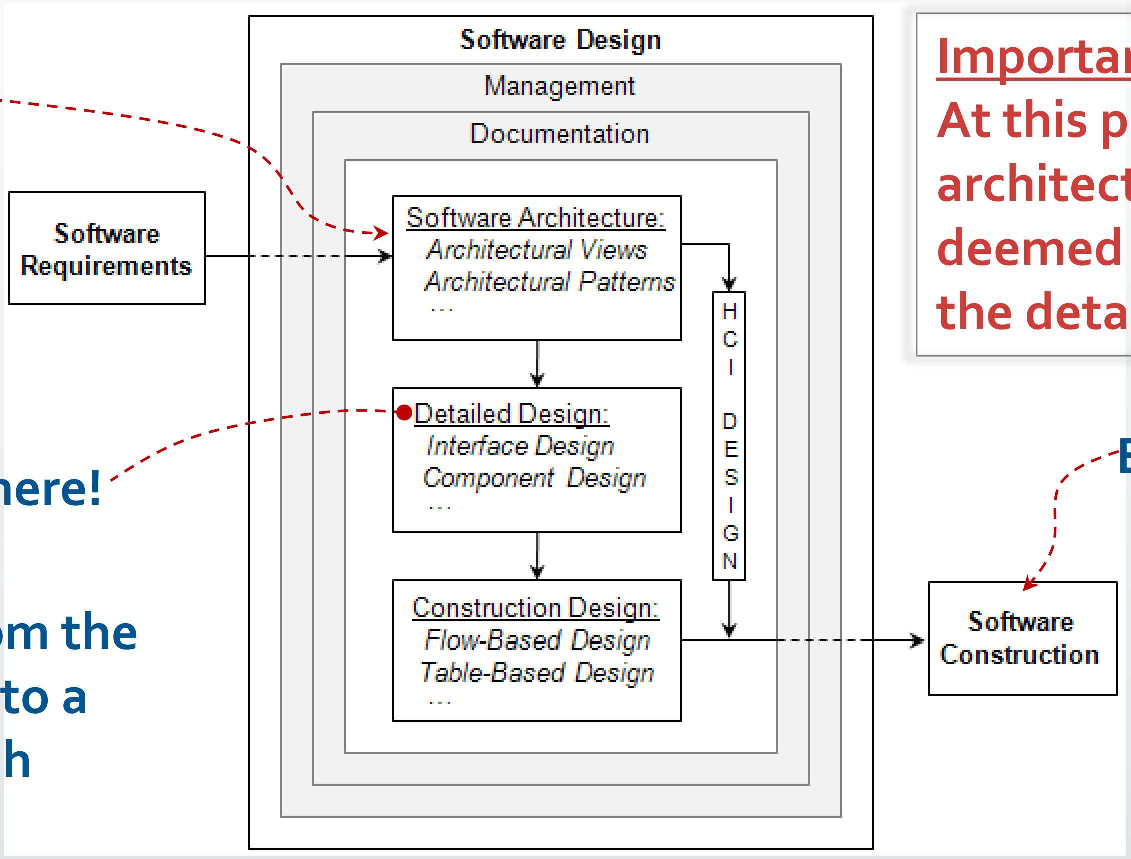
AUIS IT DEPARTMENT

First, Let's Think About Where We Are…

We started here

We are here!

Our design efforts shift from the macro-design approach to a micro-design approach



Important:
At this point, requirements and architecture are specified; they are deemed sufficiently complete to begin the detailed design of the system.

Eventually, we want to get here!

We now seek to further decompose and refine system components into one or more fine-grained elements, functions, and data variables.

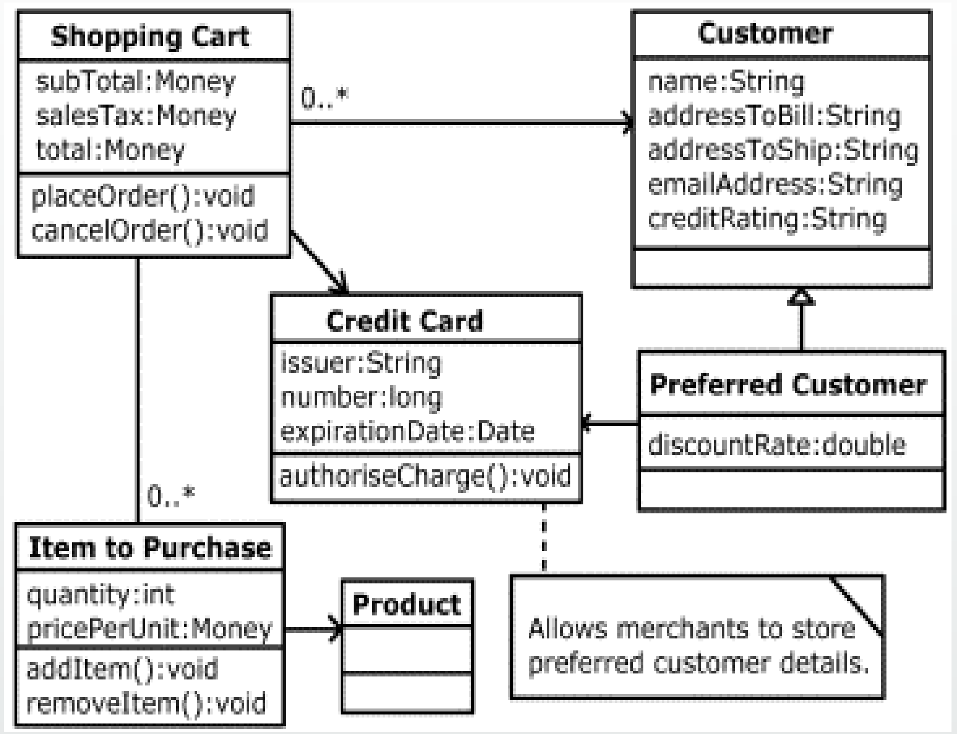
AUIS IT DEPARTMENT

What is Detailed Design?

- **According to the IEEE**

The process of refining and expanding the **preliminary design phase** of a system or component to the extent that the design is sufficiently complete to be implemented.

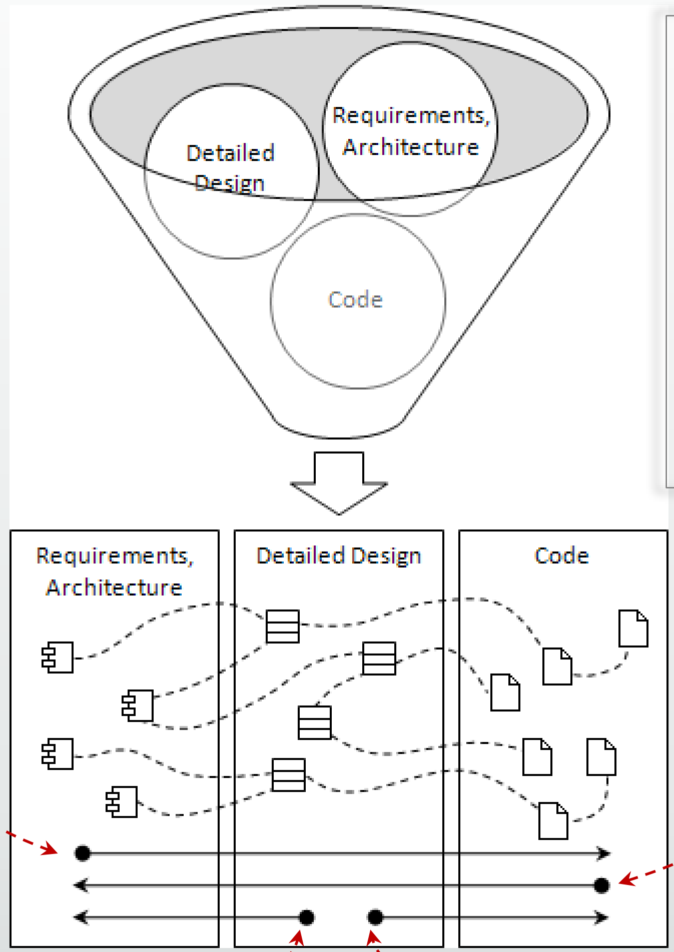
During **Detailed Design** designers go deep into each component to define its internal structure and behavioral capabilities, and the resulting design leads to natural and efficient construction of software.



AUIS IT DEPARTMENT

Detailed Design Overview

Designer's Mental Model During Detailed Design!



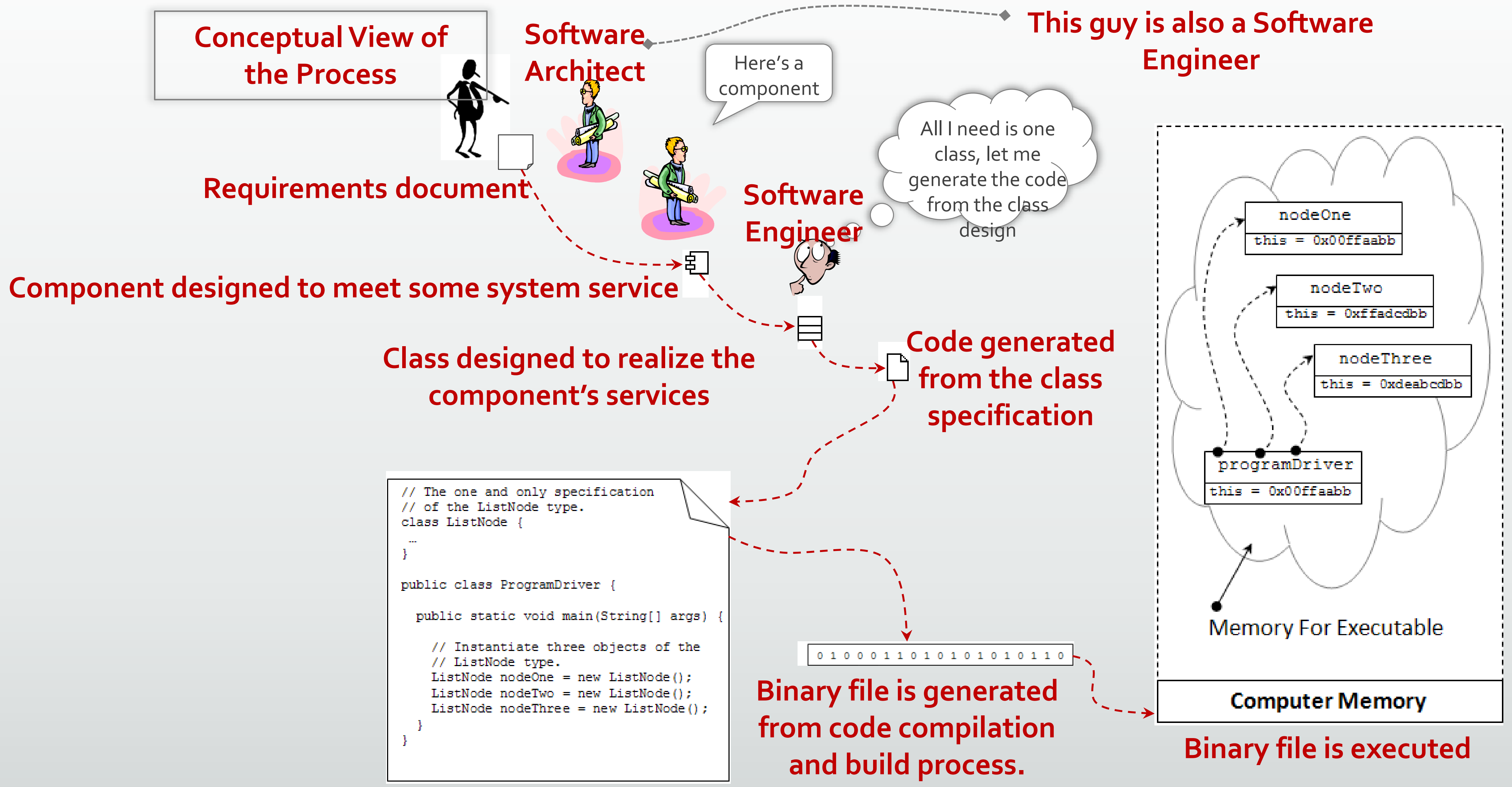
Important:
During detailed design, the use of industry-grade development tools are essential for modeling, code generation, compiling generated code, reverse engineering, software configuration management, etc.

If given requirements and architecture, detailed designers must move the project forward all the way to code

If given code, detailed designers must be able to reverse engineer the code to produce detailed and architectural designs.

When starting at detailed design, designers must be able to produce both code and architectural designs

Detail design process


Conceptual View of the Process
Software Architect
Here's a component
This guy is also a Software Engineer
All I need is one class, let me generate the code from the class design
Requirements document
Software Engineer
Component designed to meet some system service
Class designed to realize the component's services
Code generated from the class specification
nodeOne
this = 0x00ffaabb
nodeTwo
this = 0xffadcdbb
nodeThree
this = 0xdeabcdbb
programDriver
this = 0x00ffaabb
Memory For Executable
Computer Memory
0 1 0 0 0 1 1 0 1 0 1 0 1 0 1 0 1 1 0
Binary file is generated from code compilation and build process.
Binary file is executed

```
// The one and only specification
// of the ListNode type.
class ListNode {
 …
}

public class ProgramDriver {

  public static void main(String[] args) {

    // Instantiate three objects of the
    // ListNode type.
    ListNode nodeOne = new ListNode();
    ListNode nodeTwo = new ListNode();
    ListNode nodeThree = new ListNode();
  }
}
```

AUIS IT DEPARTMENT

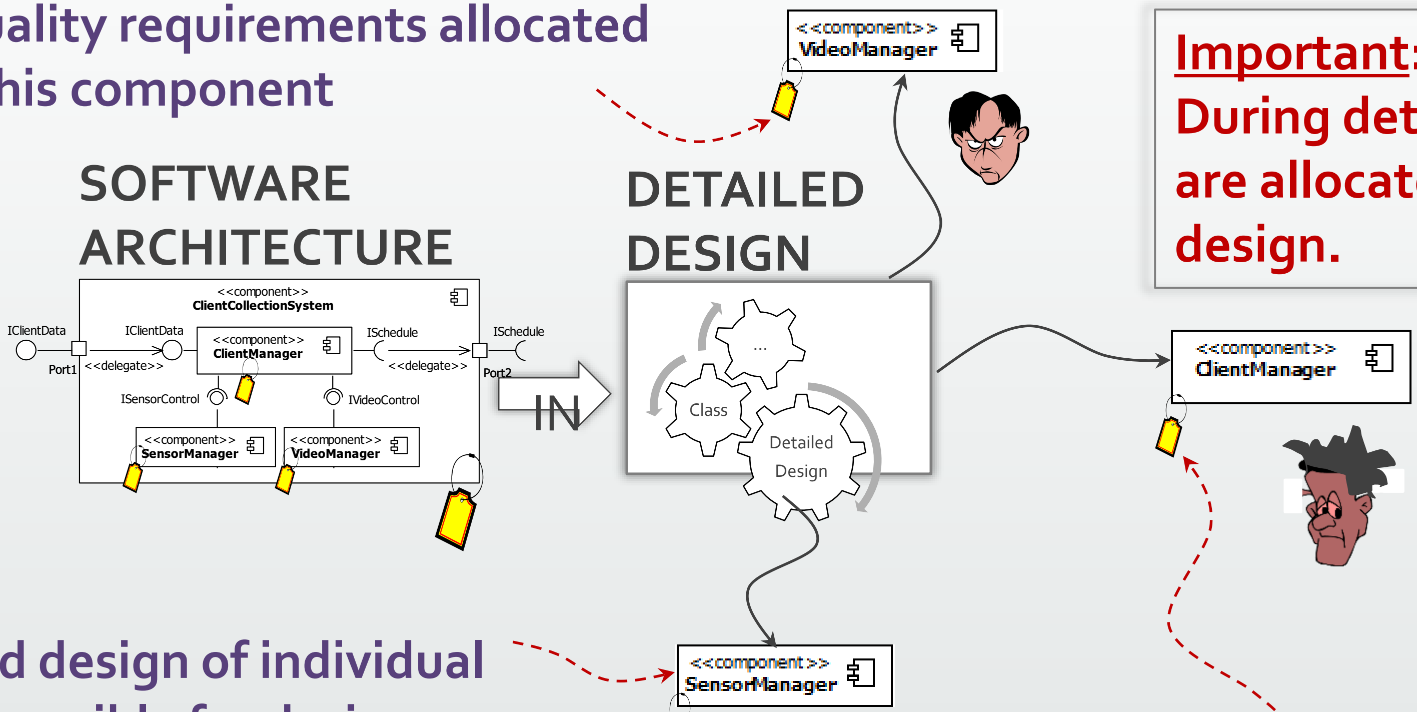
Key Tasks in Detailed Design

The major tasks identified for carrying out the detailed design activity include:

- Understanding the architecture and requirements
- Creating detailed designs
- Evaluating detailed designs
- Documenting software design
- Monitoring and controlling implementation

Understanding the Architecture and Requirements

Functional and quality requirements allocated to this component



<<component>> VideoManager
SOFTWARE ARCHITECTURE
DETAILED DESIGN
<<component>> ClientCollectionSystem
IClientData
Port1
IClientData
<<delegate>>
<<component>> ClientManager
ISchedule
<<delegate>>
ISchedule
Port2
ISensorControl
IVideoControl
<<component>> SensorManager
<<component>> VideoManager
IN
...
Class
Detailed Design
<<component>> ClientManager
<<component>> SensorManager

Important:
During detailed design, components are allocated to teams for further design.

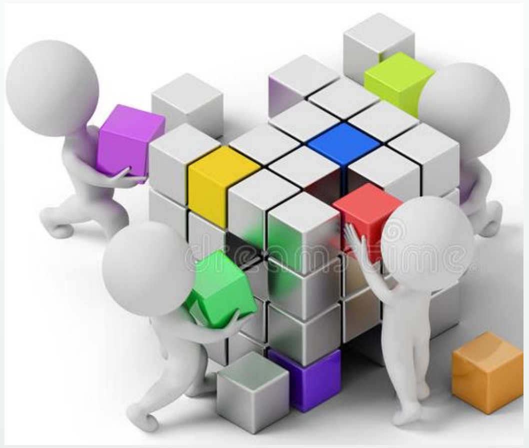
During the detailed design of individual components, it is possible for designers to derive requirements and impose them on the implementation of the component.

Functional and quality requirements allocated to this component

AUIS IT DEPARTMENT

Creating detailed designs

When creating detailed designs, focus is placed on the following:



- **Interface Design**

  - Internal interface design

  - External interface design

- **Graphical User Interface Design**

  - This may be a continuation of designs originated during architecture.

- **Internal Component Design**

  - Structural

  - Behavioral

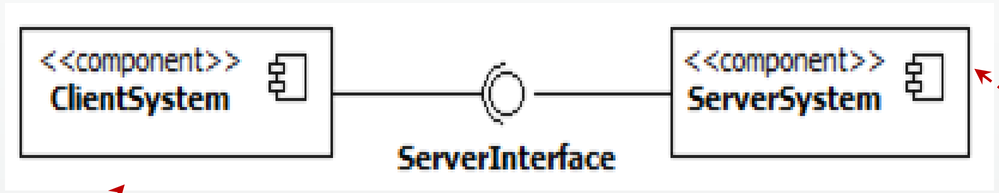
- **Data Design**

AUIS IT DEPARTMENT

From Component to Detail Design

Consider the following components and interface identified during architecture

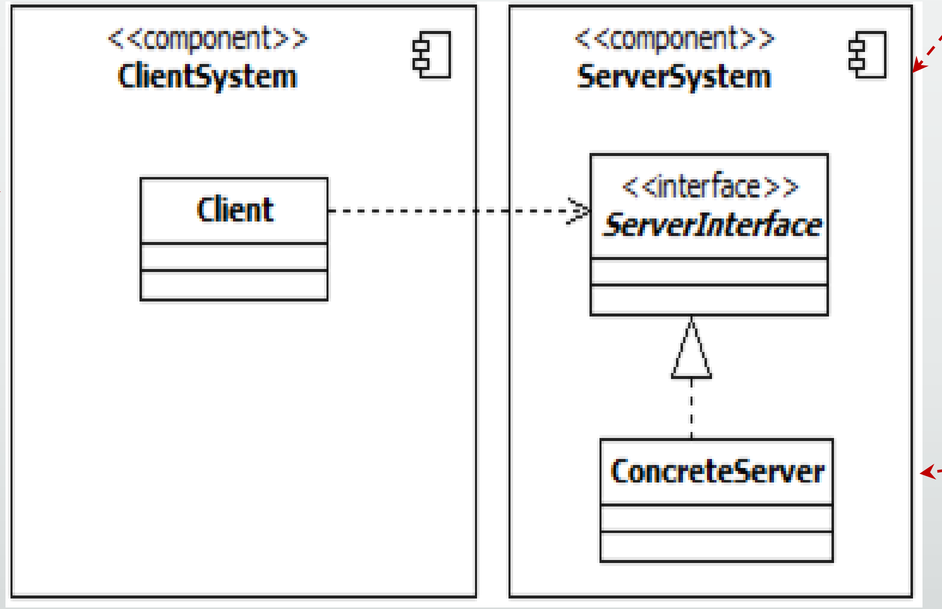
During Architecture



This is the same as that

During the component design task of the detailed design activity, these components are refined to fully define how they realize the component's services

During Detailed Design

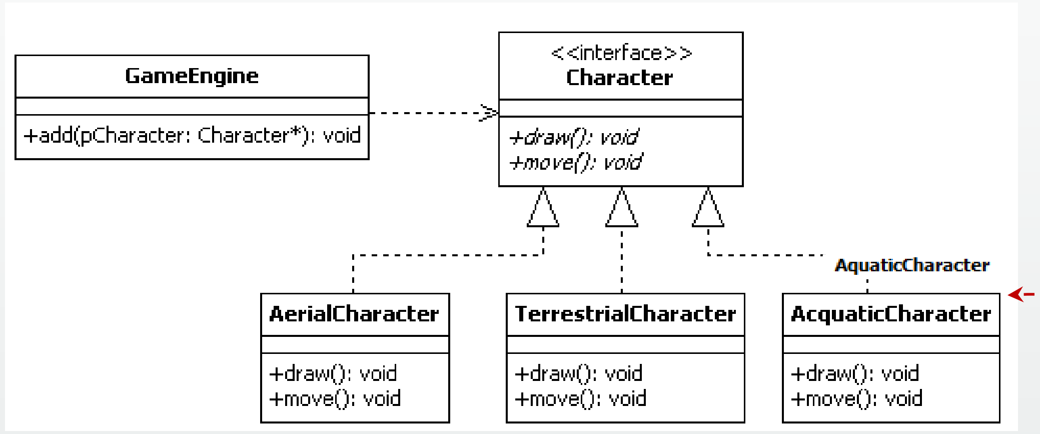


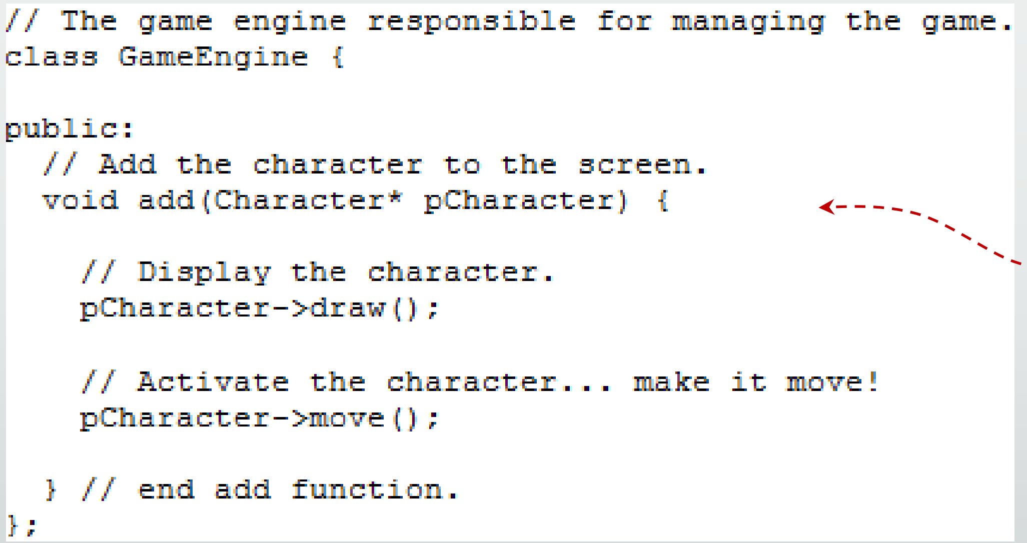
Not quite enough details, but you get the idea, right?

Important:
In OO, during detailed design, we shift away from the more abstract UML component and begin to think in terms of classes, interfaces, types, etc.

Design Principles for Internal Component Design
The Open-Closed Principle (OCP)

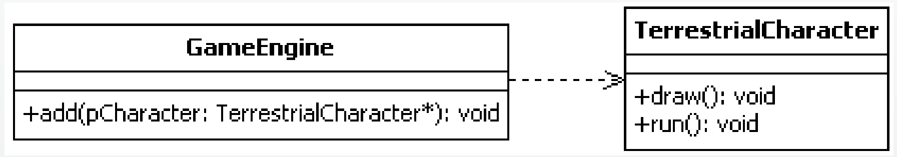
New redesign! Adheres to OCP!





```
// The game engine responsible for managing the game.
class GameEngine {

public:
  // Add the character to the screen.
  void add(Character* pCharacter) {

    // Display the character.
    pCharacter->draw();

    // Activate the character... make it move!
    pCharacter->move();

  } // end add function.
};
```

Old design! Violates OCP!



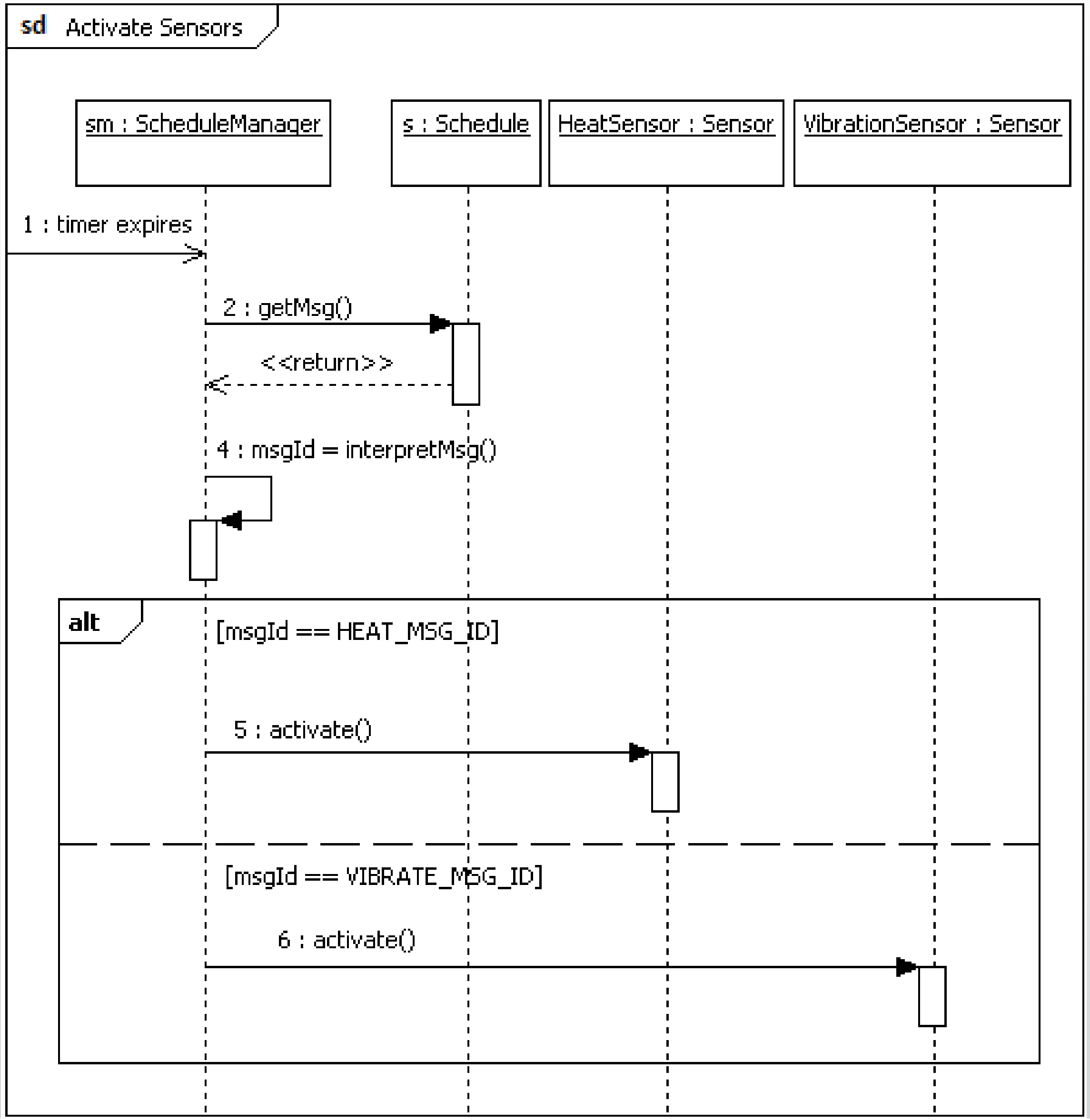
New Aquatic Character added by extension and not by modifying existing working code!

With this design, GameEngine can draw and activate current and future Characters in the game without modification!

AUIS IT DEPARTMENT

Modeling Internal Behavior of Components

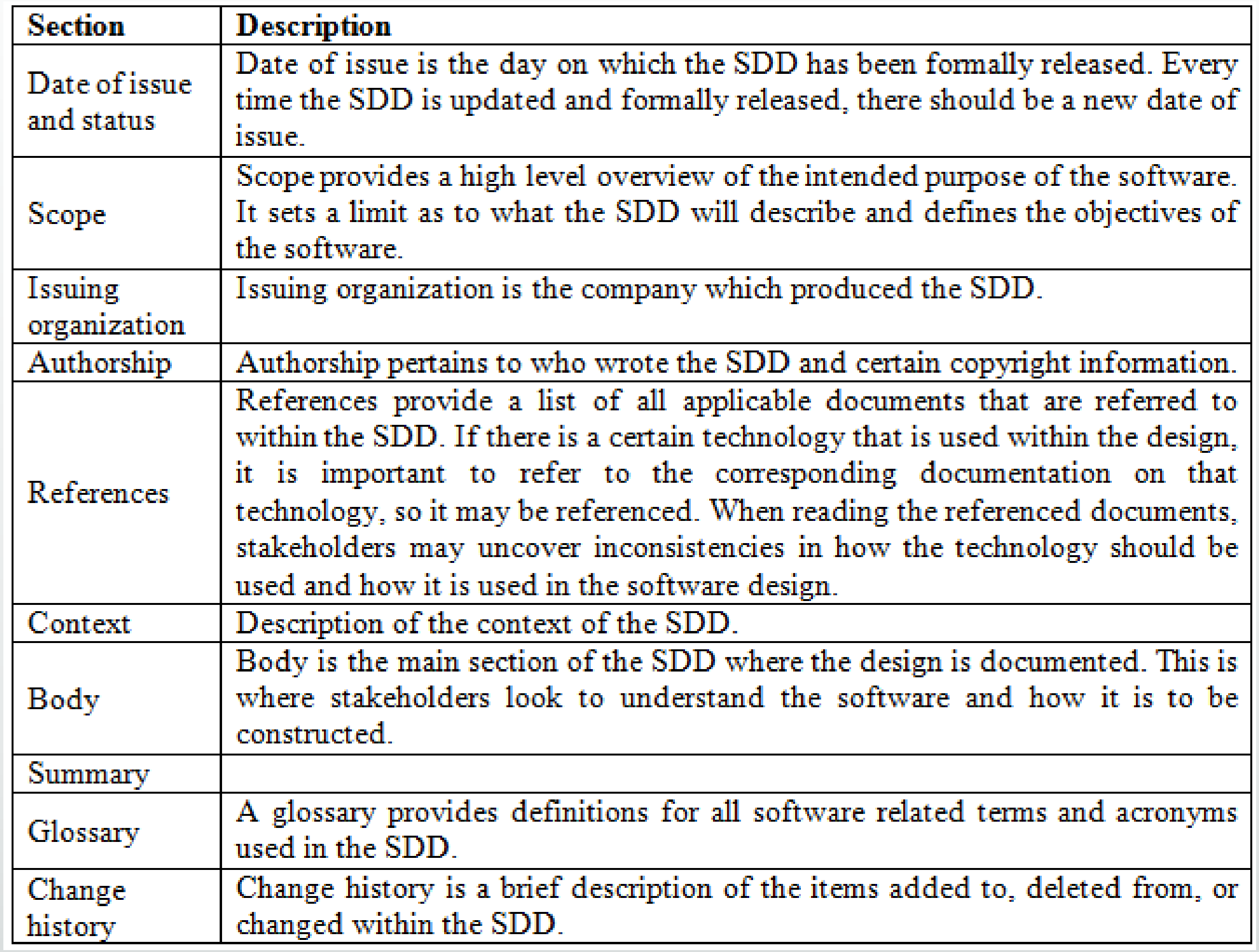
Operator	Description
seq	Default operator that specifies a weak sequencing between the behaviors of the operands.
alt	Specifies a choice of behavior where at most one of the operands will be chosen.
opt	Specifies a choice of behavior where either the (sole) operand happens or nothing happens.
loop	Specifies a repetition structure within the combined fragment.
par	Specifies parallel operations inside the combined fragment.
critical	Specifies a critical section within the combined fragment.



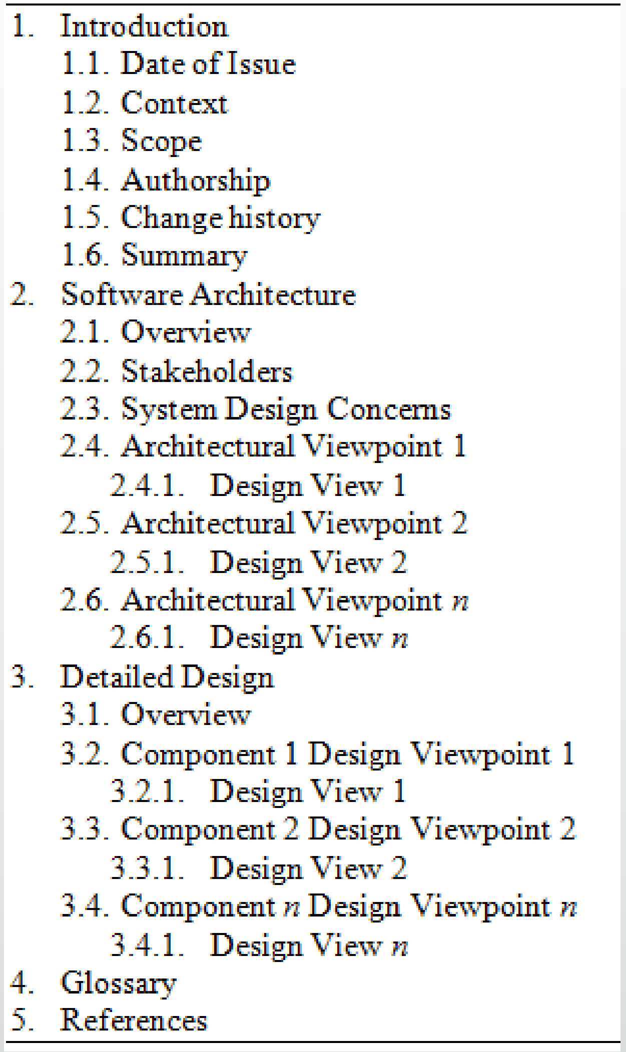
AUIS IT DEPARTMENT

Documenting Detailed Designs

- Documentation of a project's software design is mostly captured in the software design document (SDD), also known as software design description.



Section	Description
Date of issue and status	Date of issue is the day on which the SDD has been formally released. Every time the SDD is updated and formally released, there should be a new date of issue.
Scope	Scope provides a high level overview of the intended purpose of the software. It sets a limit as to what the SDD will describe and defines the objectives of the software.
Issuing organization	Issuing organization is the company which produced the SDD.
Authorship	Authorship pertains to who wrote the SDD and certain copyright information.
References	References provide a list of all applicable documents that are referred to within the SDD. If there is a certain technology that is used within the design, it is important to refer to the corresponding documentation on that technology, so it may be referenced. When reading the referenced documents, stakeholders may uncover inconsistencies in how the technology should be used and how it is used in the software design.
Context	Description of the context of the SDD.
Body	Body is the main section of the SDD where the design is documented. This is where stakeholders look to understand the software and how it is to be constructed.
Summary	
Glossary	A glossary provides definitions for all software related terms and acronyms used in the SDD.
Change history	Change history is a brief description of the items added to, deleted from, or changed within the SDD.



1. Introduction
   1.1. Date of Issue
   1.2. Context
   1.3. Scope
   1.4. Authorship
   1.5. Change history
   1.6. Summary
2. Software Architecture
   2.1. Overview
   2.2. Stakeholders
   2.3. System Design Concerns
   2.4. Architectural Viewpoint 1
      2.4.1. Design View 1
   2.5. Architectural Viewpoint 2
      2.5.1. Design View 2
   2.6. Architectural Viewpoint *n*
      2.6.1. Design View *n*
3. Detailed Design
   3.1. Overview
   3.2. Component 1 Design Viewpoint 1
      3.2.1. Design View 1
   3.3. Component 2 Design Viewpoint 2
      3.3.1. Design View 2
   3.4. Component *n* Design Viewpoint *n*
      3.4.1. Design View *n*
4. Glossary
5. References

- **References**

- As specified in the syllabus