# Architectural Styles and Tactics

**Group members:**

KrishnaSai Reddy Patil

Pranav Jaipurkar

Aman Geetey

Tigist Shibeshi

Yili Zhang

Vijayadhaarani Vijayaarunachalam

**Techno-Tidbits**

Contents

[**Architectural Styles and Tactics**](#_d3ai18ww9nwa) **1**

[**Architecture Styles and Tactics Driving Factor**](#_qugqei3mdms0) **3**

[Main Quality Attributes](#_9mufd1cx6o0s) 3

[High Priority Scenarios](#_gxzt85mfnlnq) 3

[Other Considerations](#_3e7fv8biq027) 3

[**Architecture Styles**](#_jfls49abbzad) **4**

[2.1 Rationale for selection](#_rys0wvc8p4sw) 4

[2.2 Advantages](#_2cmm4w99n42m) 4

[2.3 Disadvantages](#_t6v9nbejlu71) 5

[**Preliminary Architecture**](#_odnhqbuo32oe) **5**

[Architecture Elements and their relationship](#_r3jtrauoawpc) 5

[Description of responsibilities](#_m23gvnnihg1v) 6

[**Architecture Tactics**](#_p44nk224z5hr) **8**

[**Representation of selected Styles and Tactics**](#_6r9rdus5zvtj) **10**

[5.1 Architectural Style Representation:](#_ra0q1fse14kt) 10

[5.2 Architectural Tactics:](#_y2eb978uowkv) 11

# 

# Architecture Styles and Tactics Driving Factor

## Main Quality Attributes

a) Data Confidentiality- The grades and progress of the students and study material of the courses should be available only to students registered for that course and course’s instructor.

b) Security- The system should be able to stop unauthorized access.

## High Priority Scenarios

a) Course Details- Student can find Instructor’s name, syllabus, modules, assignments and quizzes for selecting courses.

b) Modify course content- Instructor should be able to update the course syllabus, modules and assignments.

c) Assist with issues- Admin should be able to assist instructors and students with issues they face.

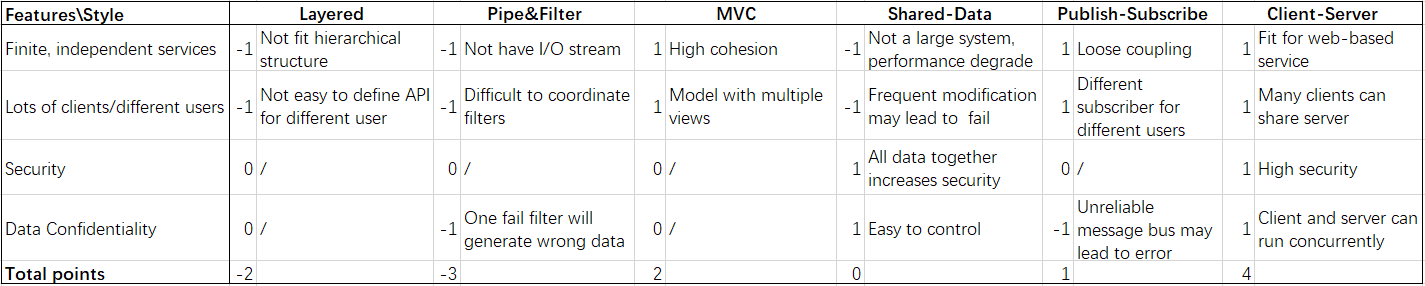
## Other Considerations

* The website / the platform provides free of charge service. Taking into consideration the limited financial resources, the cost of building and maintaining the website should be economical/ affordable so that the platform continues to serve the community who needs it most.

# Architecture Styles

## 2.1 Rationale for selection

* The system provides lots of clients with finite, independent services, including video lectures, discussion board, modify/add/remove/take assignments/quizzes. Due to the independence of these services, the style structure has to be with high cohesion, flexible to modify, and easy to exchange data between modules. Therefore, we should not consider styles like layered style, which has a hierarchical structure, or pipe-and-filter style, which is hard to share data and mainly deal with input/output data streams. On the other hand, the client-server style perfectly fits the needs of "multiple services and lots of clients".
* We have considered using shared-data structure. However, considering that our system is not that large, so that we do not have the need to let that much modules to share some data. If using this style, it may lead to performance degrade. Hope that the system response fast to users, we decide not to use shared-data style. Besides, because of the fact that data in the system needs to be modified quite frequently, once some error happens to the shared-data, the system will crash, and we want to avoid that circumstance.
* The system may have different types of users. To show them different views of the same module, we decided to use MVC style. On the other hand, if using publish-subscribe style to meet this requirement, if there is unreliable message bus, error may be generated. Besides, MVC style has a high cohesion for independent modules.



## 2.2 Advantages

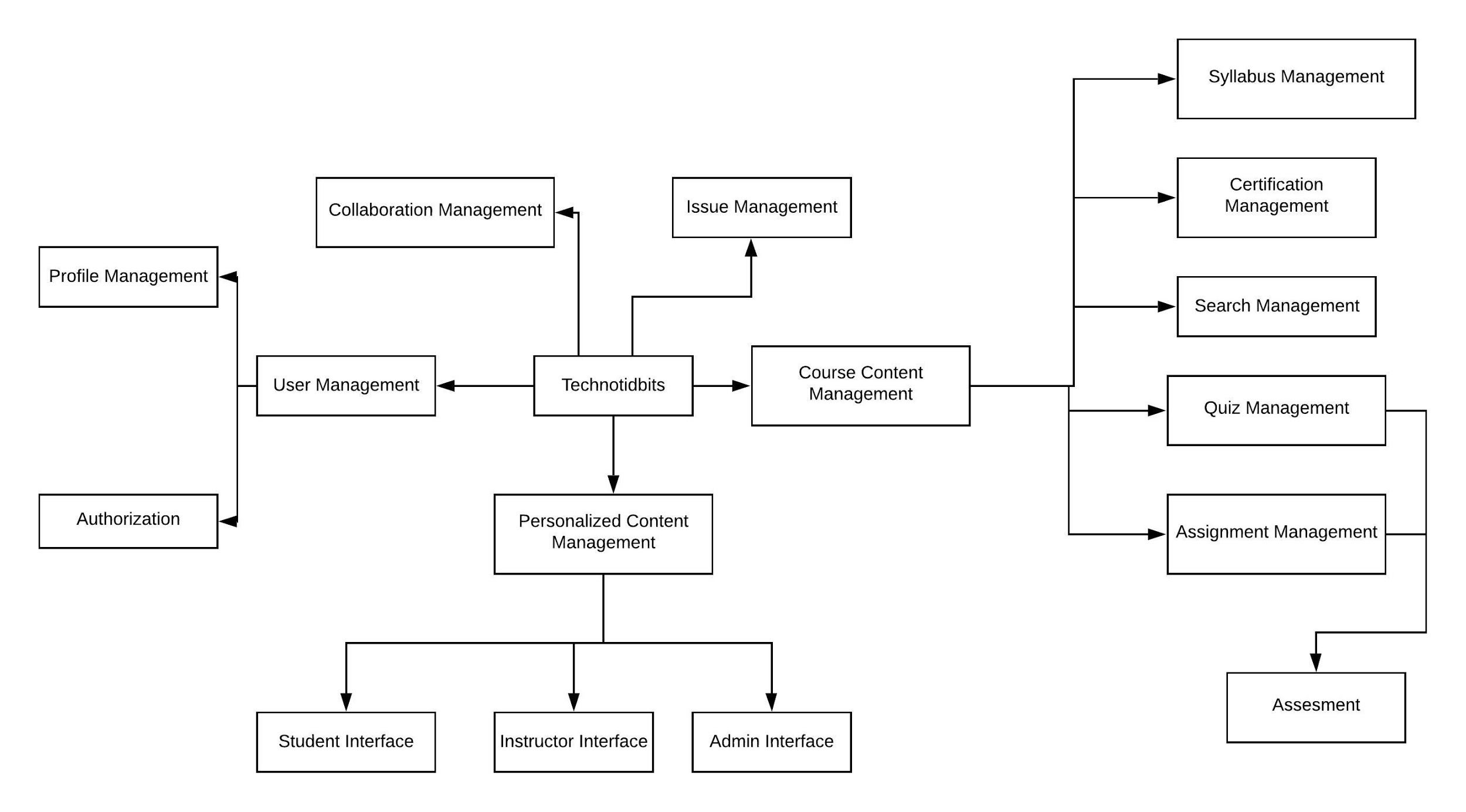
* Clients can share one server for different services.
* Responsibilities are clear separated among the client and server, modules can be with high cohesion and low coupling.
* Client and server can run concurrently.
* Different type of clients can have different views for the same module.
* Easy to maintain the system with high security.
* Our team can develop simultaneously.

## 2.3 Disadvantages

* The framework navigation may be complex.
* Thread management may be complex.
* It may not be easy to decompose every feature into three artifacts.
* The server may handle heavy traffic, so we will need load balancing.

# Preliminary Architecture

## Architecture Elements and their relationship



## Description of responsibilities

**3.2.1 Personalized content Management**

**Student Interface**

- For each student, the system should record the course he/she takes.

- For each student, the system should record the progress of each course he/she takes.

- For each student, the system should display up-coming events.

**Instructor Interface**

- For each instructor, the system should record the course he/she tutors.

- For each instructor, the system should allow him/her to modify the course content, quiz and assignment for the course he/she tutors.

- For each instructor, the system should display up-coming events.

**Admin Interface**

- The system should allow admin to modify the course content, quiz and assignment for courses.

- The system should display the help request raised by each user.

**3.2.2 Course content management:**

- This model handles the update, delete and view request from Syllabus, Quiz, Assignment and Authentication management.

- This model handles the course database

**3.2.3 Syllabus Management:**

- This module provides general information of a course.

- An instructor or an administrator should be able to modify it.

- All users should be able to read the information.

**3.2.4 Quiz Management:**

- An instructor should be able to add/remove/modify a quiz.

- A student should be able to take a quiz after he/she taking the corresponding course.

- A quiz should have corresponding gradebook.

**3.2.5 Assignment Management:**

- An instructor should be able to add/remove/modify an assignment.

- A student should be able to do an assignment after he/she taking the corresponding course.

- An assignment should have corresponding gradebook.

**3.2.6 Issue Management:**

- This module should show a user manual of the system.

- A user should be able to browse help for operations he/she is authorized to do in the system.

- This module allows user to seek help to the administrator to cope up with system issues.

**3.2.7 Authorization:**

- This module should record a user's identification as a student, an instructor, or an administrator.

- This module should only allow a user to do operations that he/she is authorized to do in the system.

**3.2.8 Collaboration Management:**

- This component should enable students and instructors to communicate through the system.

- A discussion should follow a specific course.

**3.2.9 Profile Management:**

- A student should be able to modify his/her profile.

- A student should be able to take/drop a course.

- An instructor should be able to modify his/her profile.

- An instructor should be able to apply to instruct a course or cancel to be instructor of a course.

- An administrator should be able to add/remove student from a course.

- An administrator should be able to approve/decline application of a user to be an instructor of a course.

**3.2.10 User Management:**

This module interacts with the user database and provides information for Authentication and Profile Management.

**3.2.11 Assessment:**

- This module grades the student based on his/her performance in the quiz and assignment.

**3.2.12 Certification Management:**

- After a student finishes a course, the system should provide standard to test if he/she should be certificated for this course.

- This feature should record if a student has passed a course.

**3.2.13 Search Management:**

- This Module allows users to search for any course, he/she desires.

# Architecture Tactics

**a) Usability tactics**

Design tactics:

There should be separate designated UI so that grades and progress of individual students should be confidential.

Runtime tactics :

System calls should be specific and well managed, uploading and submitting should have a reasonably defined time window of completion.

**b) Data Confidentiality- Security tactics.**

The software should detect, resist scenarios if unauthorized user is trying to get access of course materials and react to them accordingly.

**c) Security- Security tactics.**

The software should be able to detect, resist, react and recover from attacks from unauthorized users.

**Advantages** of using Usability, Runtime and Security tactics:

1) Robustness

2) Scalability

3) Reliable

4) User Friendly

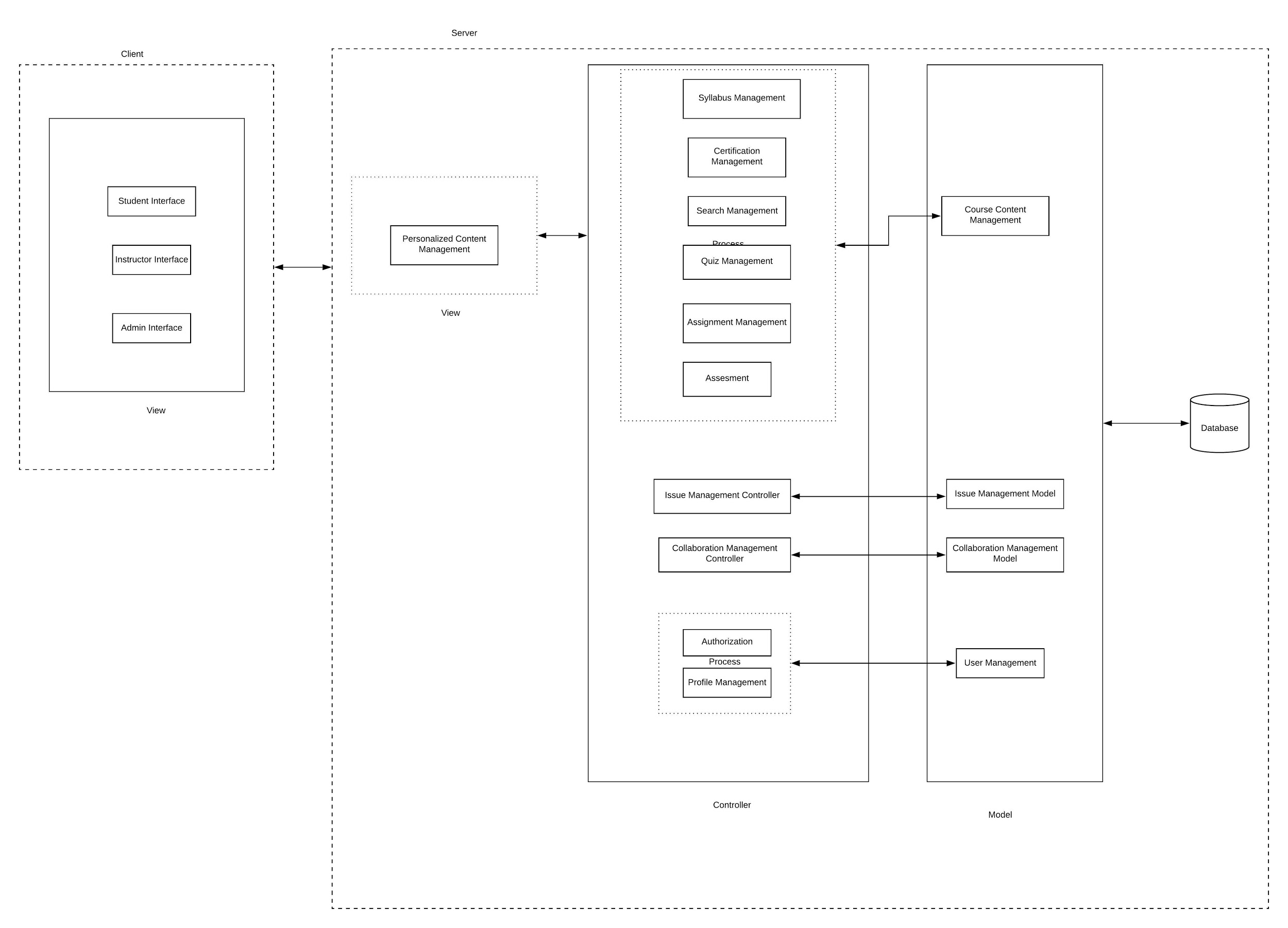
**Disadvantages** of using Usability, Runtime and Security tactics:

1) Slow (due to constraints of different tactics overlapping in each other’s performance).

2) Difficult to implement.

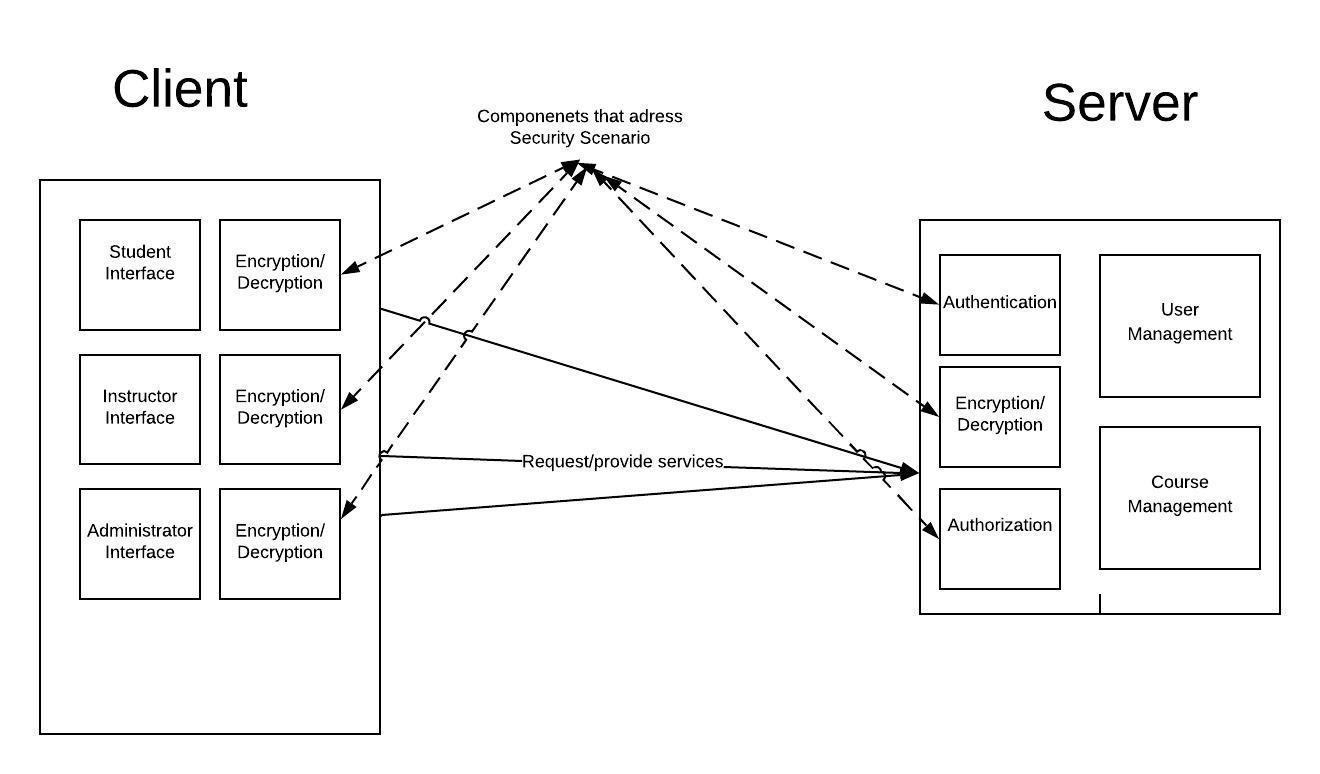
# Representation of selected Styles and Tactics

## 5.1 Architectural Style Representation:



## 5.2 Architectural Tactics:

Security Tactic Representation



Usability Tactic Representation

