ClassAxis System Design

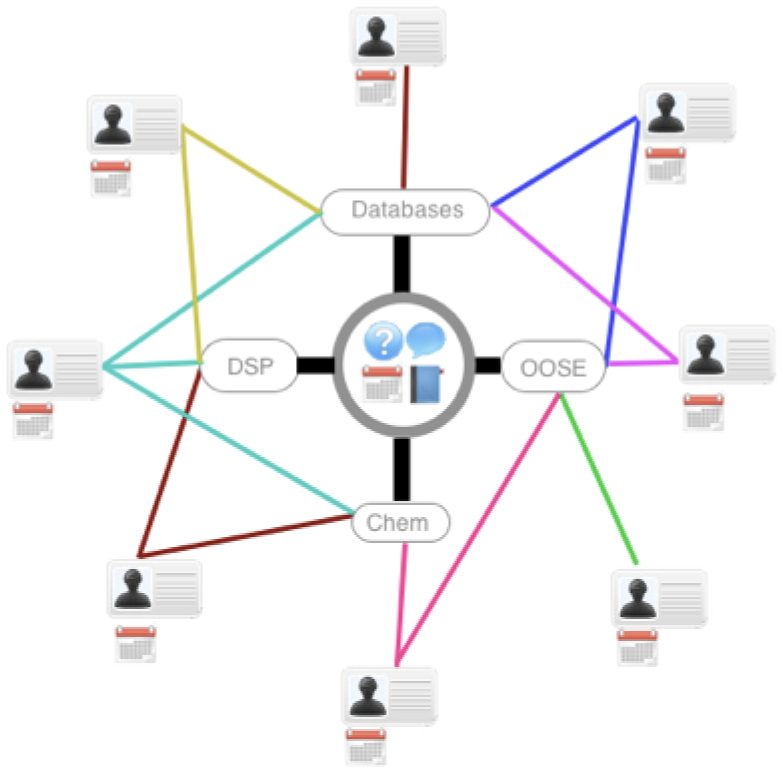
Marquette University, Department of Computer Engineering
COEN4610, Object-Oriented Software Engineering

Prepared for:
Dr. James Conigliaro

Prepared by:

Corey Bost	Johnny Fandel	Salome Marquez
Kellene Buchwald	Travis Huhn	Michael Santana
Dexter Del Frate	Andre Lancour	Sam Warmuth

December 10, 2010



Contents

1. Introduction

This document will outline the purpose of the system, the system's design goals, the current and proposed software architecture, going in detail on the several subsystems of the proposed system design.

1.1 Purpose of the system

ClassAxis allows for a way for users to interact in realtime with each other to discuss and collaborate about materials within a class. It will allow users to create an account which will then allow the user to interact with classmates through the use of forum like threads and messages. ClassAxis will also tell every member of a group when huge deadlines are upcoming. These deadlines must be inputted by a member of the group.

1.2. Design goals

Throughput: ClassAxis is meant to process many requests at a time and respond to many requests of users without much delay.
Robustness: ClassAxis is meant to not crash when invalid input was entered, also it should be prepared to inform the user if there was invalid input entered.
Reliability: ClassAxis should perform exactly as the functions are stated to the users.
Availability: ClassAxis should have 100% uptime, to allow the users to access the information on the website whenever they can.
Security: ClassAxis will not allow non-users to access any data on the website. Also, users who are not within a certain group or school will not be able to view what is within that school or group.
Safety: ClassAxis will not allow for a single human life to be in jeopardy if it went down.
Development Cost: The cost of creating the system will not exceed 100$.
Adaptability: ClassAxis should work on Firefox, Opera, Safari, Internet Explorer, and Chrome.
Utility: ClassAxis is meant to be created for the user, so it will be able to support anything the user wants to do that are within the functions of the system.
Usability: ClassAxis will be extremely simple to use for many students. If a student needs help there will be a help section on how to use a certain feature.

1.3. Definitions, acronyms, and abbreviations

ClassAxis - The production name of the system.

1.4. References

Bruegge, Bernd. *Object oriented software engineering using UML, patterns, and Java* . Upper Saddle River, NJ: Pretince Hall, 2009

Flanagan, David, and Yukihiro Matsumoto. *The Ruby programming language* . Beijing: O'Reilly, 2008

Why. Why’s (Poignant) Guide to Ruby

1.5. Overview

ClassAxis will provide a reliable and robust system that will allow for students to interact with each other. The system will have a high uptime to allow for users to access and input data whenever they can. It will be very user friendly, offering many help sections for features that are within the system. The level of security of ClassAxis should be pretty high. There will be admins viewing the information that is posted and if a user is not using the website for its purpose, the account will be locked. Also, a user will not be able to view any data within a class or group that they are not currently signed up for.

ClassAxis will be created to run on many of popular websites such as Firefox, Opera, Chrome, Safari, and Internet Explorer. The cost of operating the system originally should not exceed $20 a month and the cost of creating the system will not exceed $100. ClassAxis will be able to accept wrong input without it affect the system at all. If wrong input was to be entered, then ClassAxis will realize this and inform the user of the error that has occurred.

2. Current software architecture

TODO: Add section

3. Proposed software architecture

The proposed software architecture follows a three-tier architecture consisting of a presentation layer, a logic layer, and a data layer.

3.1. Overview

The following diagram shows the proposed software architecture:

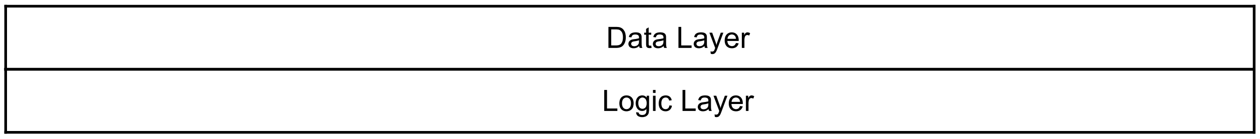


Figure 1: Proposed software architecture

3.2. Subsystem decomposition

The three main layers are composed of several sub-systems:

- Data Layer: Only one system here; a “ClassAxis Database” system
- Logic Layer: This layer is composed of several systems, each mapping to a particular functionality that users need:
    - Broadcast Subsystem
    - Calendar & Event Subsystem
    - Group Subsystem
    - Tag Subsystem
    - Topic, Post, and Message Subsystem
    - User Subsystem
- Presentation Layer: This layer provides the user access to the several subsystems that are part of the logic layer.

3.3. Hardware/software mapping

The three main layers of this architecture map elegantly to software modules of their own. They map in the following way:

- The Data Layer’s “ClassAxis Database” subsystem is a single database that is accessed that stores all of the necessary user, events, groups, and message information
- The Broadcast subsystem is a single class that handles ClassAxis’s broadcasting functionality
- Calendar & Event Subsystem are two classes, Event and Calendar, that handle ClassAxis’s Calendar and Events functionality
- Group Subsystem is a single class that handles ClassAxis’s Groups functionality
- Tag Subsystem is a single class that handles ClassAxis’s tagging functionality
- Topic, Post, and Message Subsystem is composed of three separate classes, Topic, Post, and Message that all have to do with assisting Users with communication
- User Subsystem is a single class that handles ClassAxis’s User functionality

3.4. Persistent data management

The proposed system design addresses data management by having a subsystem whose sole responsibility is to manage and handle data: the ClassAxis Database Subsystem.

The ClassAxis Database stores all of the data necessary to run ClassAxis. The data stored in the ClassAxis Database System includes the following:

- Calendar and Event Data

- Group Data and Tags
- Messages
- User Data

3.5. Access control and security

There are several security issues within the system. One that was brought up was the potential of hackers getting into the system and affecting it in undesirable ways. There is no way to solve this issue, however, measures will be used to decrease the chances of hackers affecting the system and increasing the complexity of getting into the system. The use of encryption will be used for data being transferred from the user to the system.

Another security issue that is a part of the system is other users viewing and manipulating information to a group or class they are not part of. This issue will be combated by setting permissions to each individual user, and if the user does not have a certain permission then he would not be allowed access to viewing the information from the class.

4. Subsystem services

- ClassAxis Database system
  - Services: (Return various data items to systems in the Logic and Presentation layers)
- Broadcast Subsystem
  - Services: Announce, Retract
- Calendar & Event Subsystem
  - Services: Events, Calendar, getAttendees
- Group Subsystem
  - Services: Discussions, Members, Events
- Tag Subsystem
  - Services: Return_Tag
- Topic, Post, and Message Subsystem
  - Services: Topic, Creator, Children, Time_Since, Newest
- User Subsystem
  - Services: Set_Password, Valid_Password
- Presentation System
  - Services (none)