Software Requirements Specification

Group 7, System Black

April 19, 2015

1 Introduction

1.1 System Name

Score Management Subsystem of the Teaching Service System

1.2 Overview

Score management system is a critical component of teaching service system that provides support on score record, score query and score analysis. Up to now, we specified two major groups of actors as instructors and students and authorized them different accessibility to each function. We took use of the UML model include use case diagram, data flow diagram, state diagram, class diagram and CRC cards to elucidate our design. The details will be discussed in the following chapters.

1.3 Developers

Group-Black-7 is responsible for the Score Management Subsystem of the Teaching Service System. The members of the group are:

- Qi, ZHU
- Shijia, WEI
- Shiyi, ZHU
- Tao, LIN
- Ye, QI

(In Alphabetical Order)

1.4 Background

1.4.1 Customer

Yue Chen, faculty of CS Department, Zhejiang University

1.4.2 Latent User

Faculties, students, administrators and system maintainers of Zhejiang University

1.4.3 Running Environment

Local Area Network of a university

1.4.4 Purpose of this Document

This document provides a specific rule of the development of Score Management Subsystem of the Teaching Service System. This document declare the specific role of each part in the subsystem. The document is developed under the consensus of both customers and developers, which will eventually help two groups to build mutual understanding and better communication.

And the specific purpose are:

- To provide a detailed document of what is to be developed.
- To provide a detailed document of how the subsystem is developed.
- To provide a detailed document of how inspection is conducted.

1.4.5 Purpose of this Project

The system aims to provide a service that helps the faculties and school administrators with the score management. The system provides integrity, usability and security of the service. As the system developers are university undergraduate students, the project also aims to develop students' capability of planning, designing, implementing, testing, documenting and teamwork. Thus the system is divided into subsystems with several stages of measurements to enhance students' comprehensive competitiveness.

2 Description

2.1 User Scenarios

2.2 User Requirements

2.2.1 Instructors

Instructor needs to manage scores of all classes he/she takes. He/she should have access to upload new scores and look up/modify the committed scores. And instructor should be given proper right to access his/her own classes' scores and insulate them from other unrelated instructors.

2.2.2 Students

Student needs to view his/her scores on score manage system. Apart from that, student can not make any changes on scores committed by teachers. So, the differences between students and instructors should be considered into user authentication.

2.3 Scenarios

2.3.1 Upload

- Primary Actor: Instructor
- Goal in Context: Upload the scores of one class
- Precondition: The computer is online and the transcript is finished
- Trigger: Press on the UPLOAD button in system
- Scenario:
  1. Enter the page for managing scores;
  2. Select one class;
  3. Press the UPLOAD button;
  4. Download the transcript template;
  5. Fill the transcript;
  6. Upload the file.
- Exception:
  1. The transcript's format does not accord to the standard;
  2. The file is too large;
  3. There already exists one file.
- Priority: Obligatory
- Frequency: Often
- Usage Mode: Through browser

2.3.2 Commit

- Primary Actor: Instructor
- Goal in Context: Confirm the scores of one class and save them in the stable database
- Precondition: After uploading

- Trigger: Press on the COMMIT button in system
- Scenario:
  1. Enter the page for managing scores;
  2. Select one class;
  3. Press the COMMIT button;
  4. Click on OK.
- Exception:
  1. A parallel MODIFY method is open
  2. There is no file existing for the class.
- Priority: Obligatory
- Frequency: Often
- Usage Mode: Through browser

2.3.3 Modify

- Primary Actor: Instructor
- Goal in Context: Modify the scores of one particular class Precondition: After uploading
- Trigger: Press on the MODIFY button in system
- Scenario:

  Before commit

  1. Press on the MODIFY button;
  2. Modify on the online table;
  3. Save.

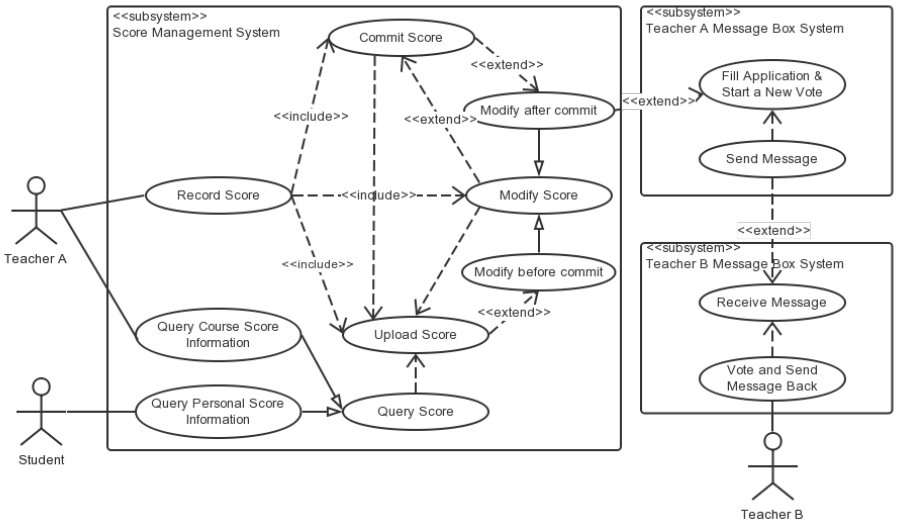
  After commit

  1. Press on the MODIFY button;
  2. Download an application form;
  3. Fill the application form;
  4. Upload the form;
  5. Broadcast the request for amendment;
  6. Wait.
- Exception:
  1. Fail to send the request;
  2. The format of the form doesnt accord to its standard.
  3. Accidentally quit. Priority:Obligatory Frequency:Often Usage Mode:Through browser

2.3.4 Query

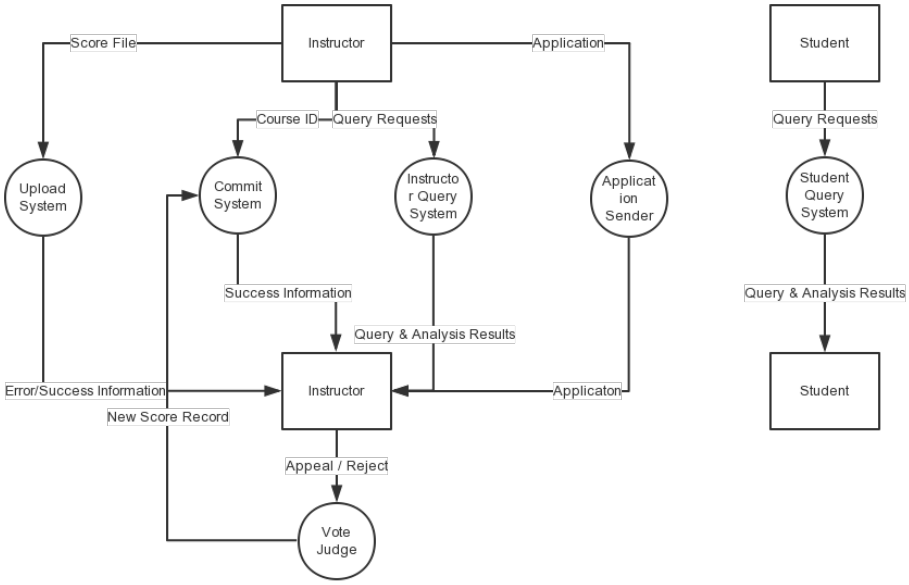
- Primary Actor: Instructor/Student
- Goal in Context: Query the statistical information of scores
- Precondition: After uploading / committing
- Trigger: Press VIEW SCORE button
- Scenario:
  1. Press VIEW SCORE button;
  2. Select view mode.
- Exception:
  1. The instructor hasnt uploaded scores;
  2. The instructor hasnt committed scores;
- Priority: Obligatory
- Frequency: Often
- Usage Mode:Through browser

2.4 Scenario Graph

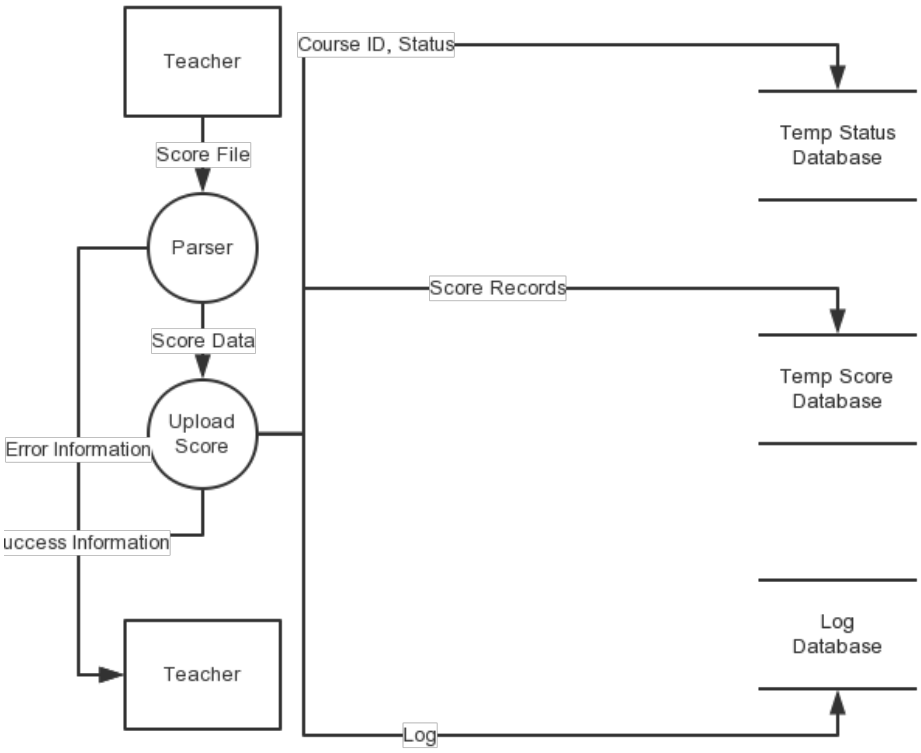


2.5 Data Flow Diagram

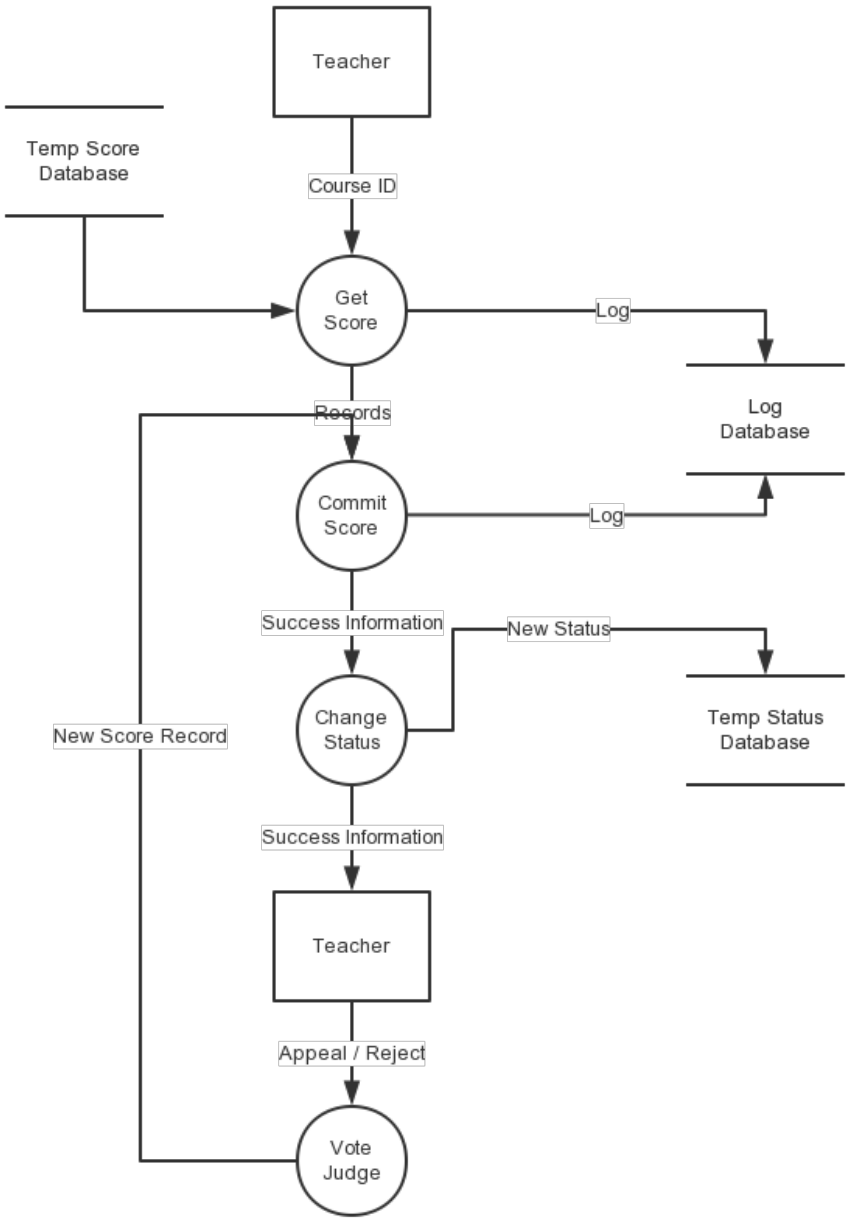
2.5.1 Overview



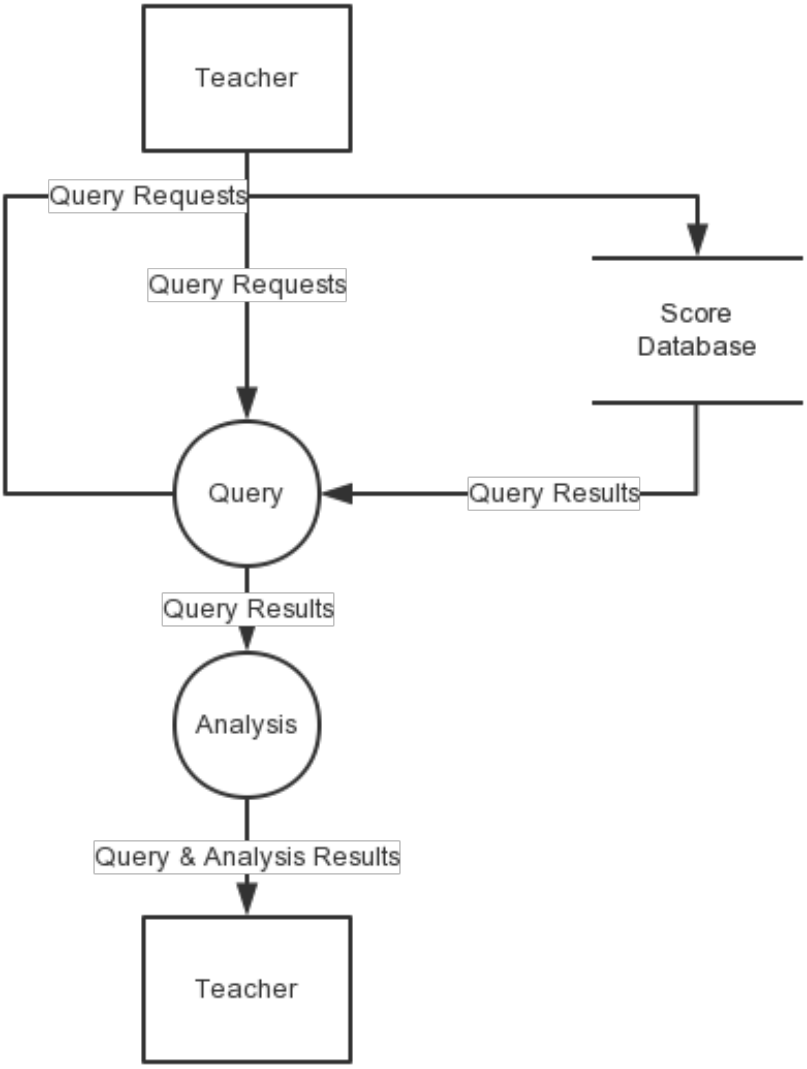
2.5.2 Upload System



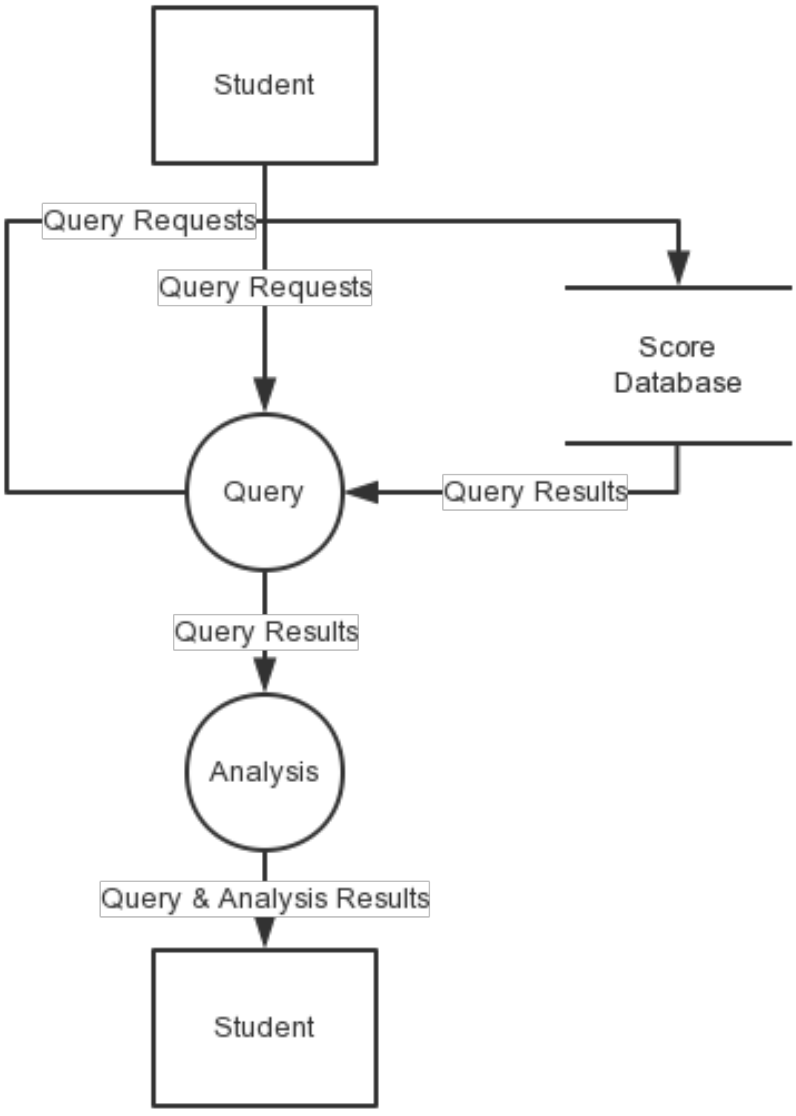
2.5.3 Commit System



2.5.4 Instructor Query System

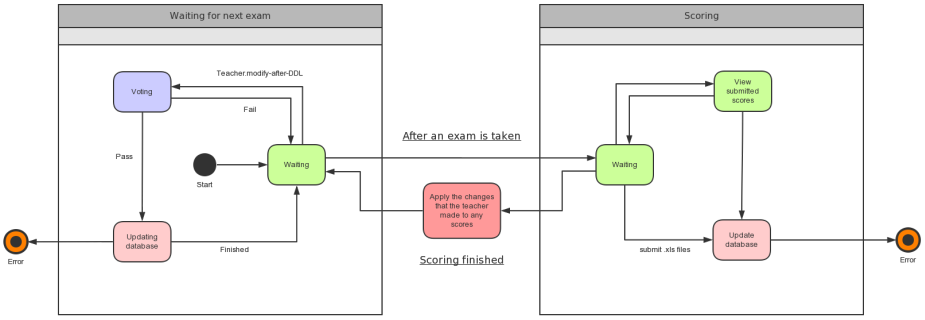


2.5.5 Student Query System

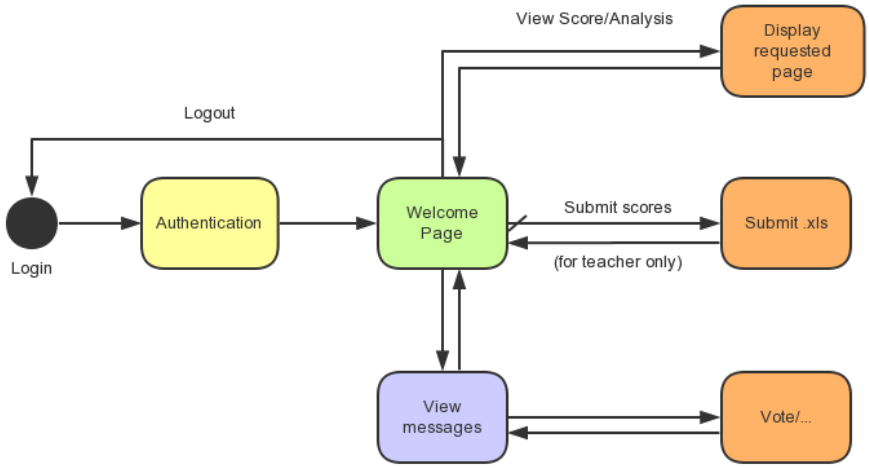


2.6 State Diagrams

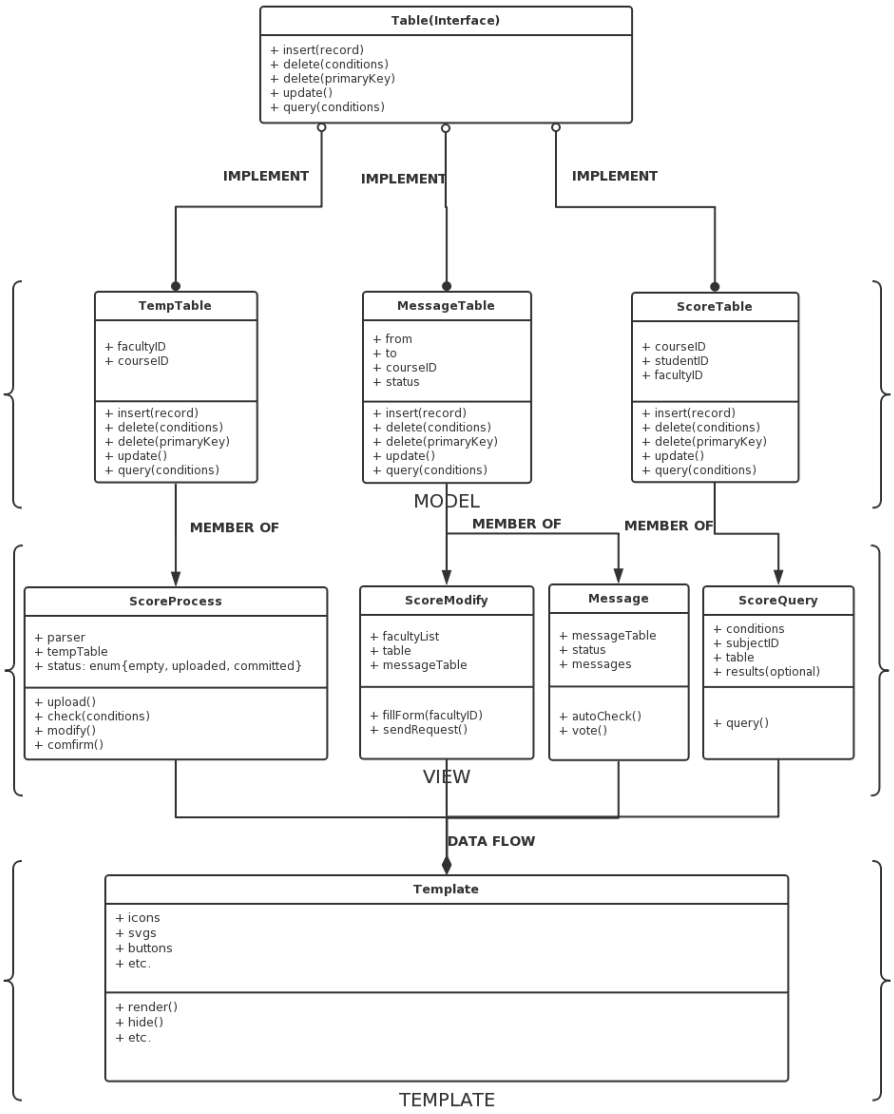
2.6.1 States Related to Courses



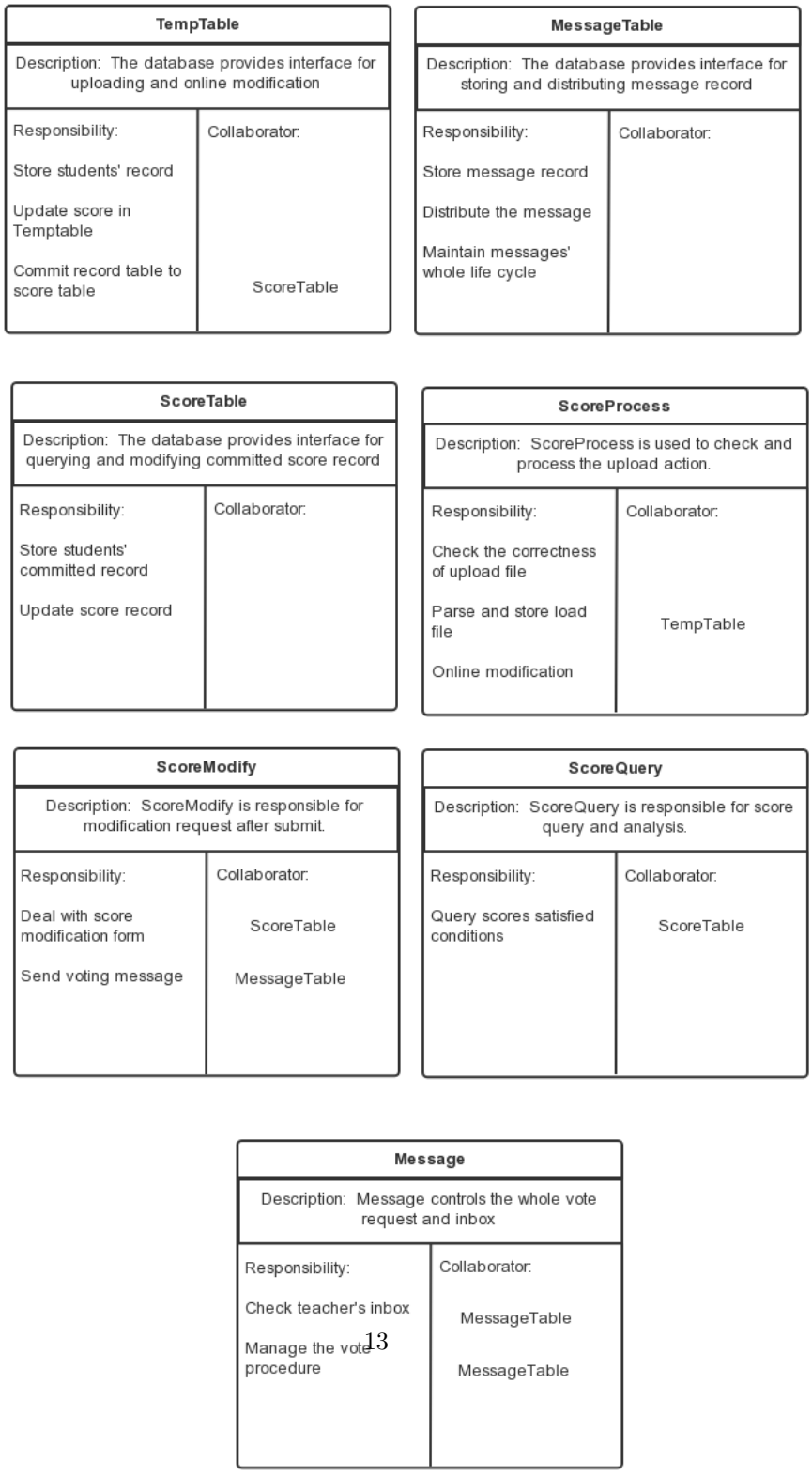
2.6.2 Web State Diagram



2.7 Class Diagrams



2.8 CRC Cards


TempTable
Description: The database provides interface for uploading and online modification
Responsibility:
Store students' record
Update score in Temptable
Commit record table to score table
Collaborator:
ScoreTable
MessageTable
Description: The database provides interface for storing and distributing message record
Responsibility:
Store message record
Distribute the message
Maintain messages' whole life cycle
Collaborator:
ScoreTable
Description: The database provides interface for querying and modifying committed score record
Responsibility:
Store students' committed record
Update score record
Collaborator:
ScoreProcess
Description: ScoreProcess is used to check and process the upload action.
Responsibility:
Check the correctness of upload file
Parse and store load file
Online modification
Collaborator:
TempTable
ScoreModify
Description: ScoreModify is responsible for modification request after submit.
Responsibility:
Deal with score modification form
Send voting message
Collaborator:
ScoreTable
MessageTable
ScoreQuery
Description: ScoreQuery is responsible for score query and analysis.
Responsibility:
Query scores satisfied conditions
Collaborator:
ScoreTable
Message
Description: Message controls the whole vote request and inbox
Responsibility:
Check teacher's inbox
Manage the vote procedure
Collaborator:
MessageTable
MessageTable

2.9 Data Description

2.9.1 Common

- facultyID: the teachers's registered ID
- courseID: the unique ID of the course lectured at a specific semester by a specific teacher
- studentID: the student's registered ID
- score: the number of a student's grade on a specific course.

2.9.2 Message

the message tool is used to communicate with other users of the system. e.g. After the teacher has uploaded the application to modify formally submitted score data, the voting requests will be sent in this message form.

- from: the sender of the message
- to: the receiver(s) of the message
- courseID: the unique ID of the course lectured at a specific semester by a specific teacher
- status: including several attributes:
  - isPassed: whether this request has passed voting.
  - votedInfo: a array shows that whether all the related teachers have passed, rejected, or haven't read it yet.
  - due date: when will the vote ends.
- send time: when is this message sent.

3 Validation Criteria

3.1 Defintion

- *Class:* A course instructed by a instructor in a semester.
- *Relevant Instructor:* The instructor who instructs the same course.

3.2 Functions for Instructors

3.2.1 Record Scores

1. An instructor could check the information of the classes he/she is currently instructing.

2. An instructor could download a transcript template to be filled from the system.

3. If the format of the transcript meets the criterion, it will be immediately parsed and shown on the screen in tabular form, with class status altered to "UPLOADED"; otherwise, it will be rejected with detailed reason specified.

4. Any new uploading will directly cover previous files.

5. The scores of the class marked as "UPLOADED" could be browsed.

6. The scores of the class marked as "UPLOADED" could be directly modified online.

7. Having committed the scores of one class, an instructor could view the statistical information of it, with the class status altered to "COMMITTED".

8. The scores of the class marked as "COMMITTED" could only be browsed but not be directly modified.

3.2.2 Vote Modification

1. If an instructor wants to modify the scores of a class marked as "COMMITTED", he/she needs to fill an online application form including course name, names of students whose scores need modification, new scores and corresponding reasons. All the relevant instructor will be apprised of this request.

2. The relevant instructors can vote for the request for score modification: they could choose either to approve it or to reject it.

3. If all of the relevant instructors agree with this modification, this request will be adopted and the instructor who starts it will get a notification telling the result.

3.2.3 Query/Analysis Scores

This is valid after one specific class is marked as "SUBMITTED".

1. The instructor could view score distribution in graphic chart from.

2. The instructor could view ranking statistics in list chart from, including student names, raw scores and grade points.

3. The instructor could view average score in eye-catching place.

3.3 Functions for Students

3.3.1 Query/Analysis Scores

1. A student could view score information of all classes that he/she has registered and is marked as "SUBMITTED". The score information includes course names, raw scores and grade points.

2. A student could view his/her GPA, average scores, total credits of the current semester.

3. A student could view his/her GPA, average scores, total credits of the courses he/she major in.

3.4 Performance

For score management system, the human interaction inferface is very important. We propose following requirements:

1. Interface should be simple but specific to functions

2. System should be user friendly and provide help information

3. It should have short response time and provide correctness and efficiency.

3.5 Security

3.5.1 Privacy

Every different User should have own data separated from others. And only owner can access his/her data in System.

3.5.2 Authentication

Teachers and students should be detected correctly. Teacher has right to add and update on records, while students don't.

3.6 Maintenance

Score manage system can work correctly at most time. And even if it crushes, it can be recovered as soon as possible. Programmer should provide detailed documents and strategies towards every error.