UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Requirement Specification

CS Marking System (Mini-Project)

Pieter le Roux (10454862)
Chistopher Moodley (10457489)
Collen Mphabantshi (10404687)
Bernhard Müller (11037157)
Machocho Shika (12127877)
Gerhard Smit (12282945)
Handre Watkins (10115405)

V0.1

February 26, 2014

Change log/History

Date	Version	Description
20 February 2014	V0.1	Skeleton of document
20 February 2014	V0.2	Section 1
20 February 2014	V0.3	
20 February 2014	V0.4	

Contents

1 Introduction

This document serves to fully specify and outline the requirements of the marking-system in detail. The document also serves to give the client and developers a clear description and elaboration of the system to be implemented in its totality. Furthermore, this document formulates an agreement between the client, Mr Jan Kroeze from the Department of Computer Science at the University of Pretoria, and developers with regard to the system to be built.

2 Vision

The marking system aims to provide lecturers with the opportunity to centralize and digitalize the marking process for practical assessments carried out and evaluated. The system will also allow Teaching Assistants (TAs) and Tutors to access the system during a marking session for a module, to record the marks of students onto the system, through the use of their mobile smartphones or computers. The system will also allow the student to access his or her mark at a later stage to see how much they have acquired. The aim of this system is to remove tedious paperwork and to prevent the loss of marks.

3 Background

This project is due to the Computer Science department of the University of Pretoria currently facing an abundance of paper work and a potential loss of mark sheets and the opportunity of utilising the COS 301 course learning opportunity for the creation of an online marking system. The system is intended to improve the marking procedures currently in place within the Computer Science department and simplify the process of processing and distributing marks as they are released. The system also provides the COS 301 students with the opportunity to learn more with regards to the procedure used for creating, designing and developing projects for businesses while also providing the University with a potentially new system that may in future be used throughout the entire university.

4 Architecture requirements

4.1 Access channel requirements

4.2 Quality requirements

4.3 Integration requirements

4.4 Architecture requirements

5 Functional requirements

5.1 Introduction

5.2 Scope and Limitations/Exclusions

Scope The scope of the Marking System project can be encapsulated as a solution that allows the users of the system to:

1. Record the practical assignments of the students.
2. To view the marks of the practical assignments at a later stage.
3. Keep all the marks on a centralized database.

The system shall be designed primarily for use by heads of departments, lecturers, teaching assistants, tutors, and students for the core and sole purpose of viewing, inputting and modifying student marks in a centralized environment.

Exclusions
The following exclusions are made explicit:

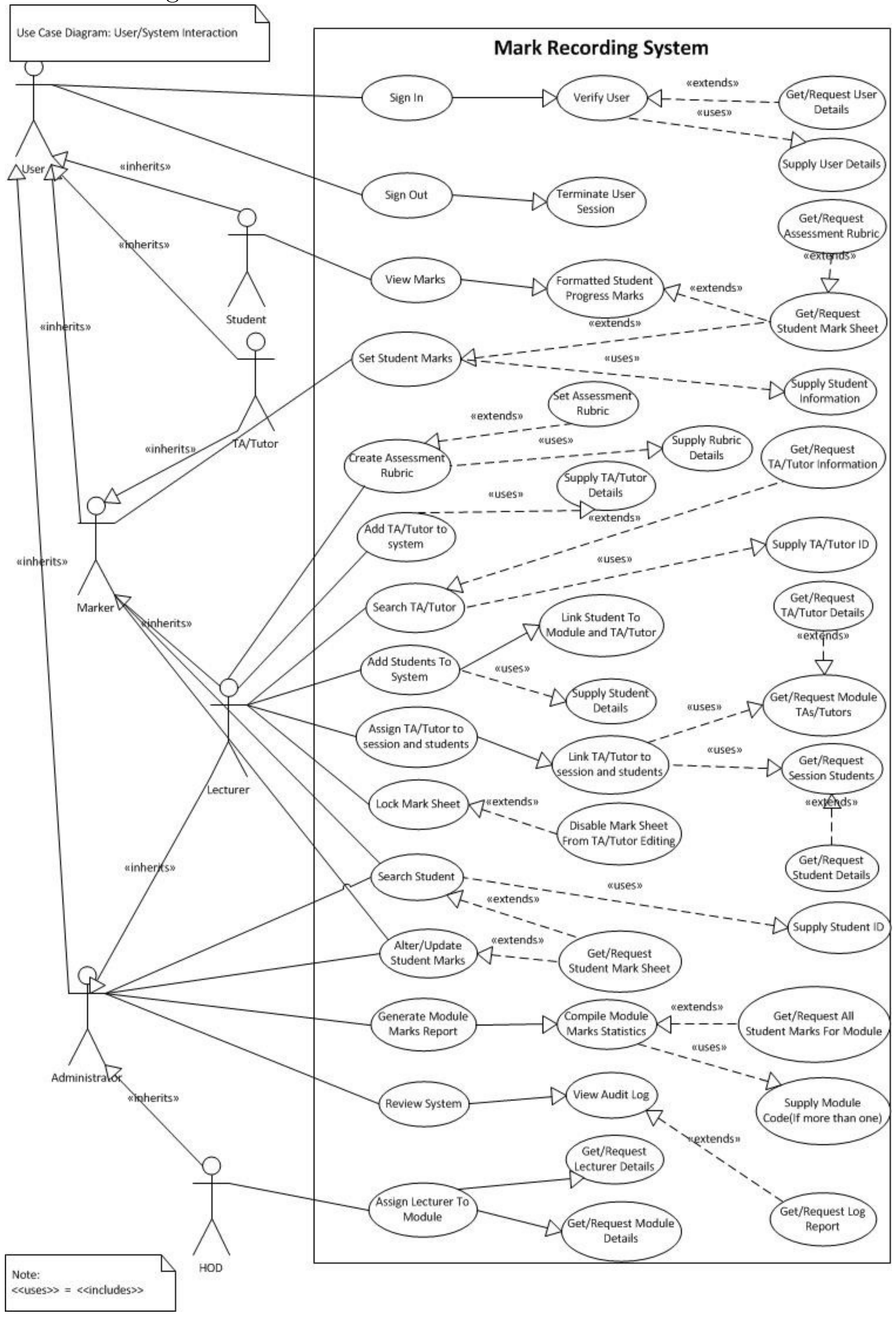
- [•] The system does not allow for assessments to be uploaded onto the system for auto marking.
- [•] The system strictly allows for mark-inputting by markers.

Limitations
The following limitations are made explicit:

- [•] The HOD cannot add lecturers not employed by the university to the system.
- [•] Lecturers cannot add individuals not registered or employed by the university on the system.

- [•] The lecturer cannot add students not registered with the university to the system.
- [•] Lecturers cannot view module reports for modules they do not lecture.

Use-Case Diagram



Use Case Diagram: User/System Interaction
Mark Recording System
User
Student
TA/Tutor
Marker
Lecturer
Administrator
HOD
«inherits»
«extends»
«uses»
Sign In
Verify User
Get/Request User Details
Supply User Details
Sign Out
Terminate User Session
Get/Request Assessment Rubric
View Marks
Formatted Student Progress Marks
Get/Request Student Mark Sheet
Set Student Marks
Supply Student Information
Set Assessment Rubric
Create Assessment Rubric
Supply Rubric Details
Get/Request TA/Tutor Information
Supply TA/Tutor Details
Add TA/Tutor to system
Supply TA/Tutor ID
Search TA/Tutor
Get/Request TA/Tutor Details
Link Student To Module and TA/Tutor
Add Students To System
Supply Student Details
Get/Request Module TAs/Tutors
Assign TA/Tutor to session and students
Link TA/Tutor to session and students
Get/Request Session Students
Lock Mark Sheet
Disable Mark Sheet From TA/Tutor Editing
Get/Request Student Details
Search Student
Supply Student ID
Alter/Update Student Marks
Get/Request Student Mark Sheet
Generate Module Marks Report
Compile Module Marks Statistics
Get/Request All Student Marks For Module
Supply Module Code(If more than one)
Review System
View Audit Log
Get/Request Lecturer Details
Assign Lecturer To Module
Get/Request Module Details
Get/Request Log Report
Note:
<<uses>> = <<includes>>

5.3 Required functionality

5.4 Use case prioritisation

Critical

[•] System runs on all web browser platforms. Application runs on android.

[•] All marks are stored on a centralized database; no data is stored on any other nodes.

[•] Audit log cannot be edited at all.

[•] SOAP interface.

Important

[•] TAs can edit and add marks during practical sessions.

[•] Lectures can set up a lock for mark sheets.

[•] Students can only view their marks after the practical.

[•] Should be an audit log for every change and action done.

[•] Students can only be marked by the TA assigned to them.

[•] If signal is lost while trying to update marks, should be saved by the browser temporarily till signal is restored.

[•] User must be logged in to access.

[•] Daily batch to the audit log to record what was changed.

[•] Progression and individual practical session marks.

[•] Reason for editing marks.

[•] Report system, display of graphs.

[•] Fitchfork marks exported.

[•] Automatically decide the best set of marks.

[•] Search for individual students.

[•] Journal of audit log.

Nice to have

[•] Kick user out for inactivity.

[•] When marks are edited after practical lectures are emailed about changes.

[•] Specify weight of marks.

[•] Auto complete student number.

5.5 Use case/Services contracts

5.6 Process specification

Below are the specific process requirements needed for some of the use-cases:
Signing In

Summary:

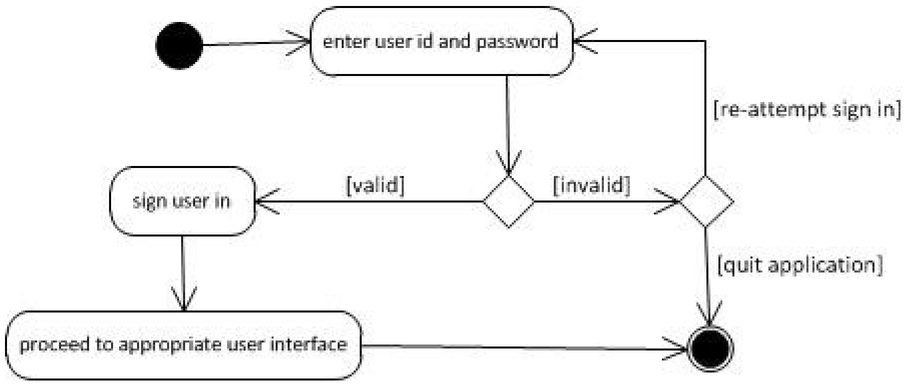
On signing in, users will present their user id and password and the system will validate them before signing them into the system and awarding them their privileges on the system.

Requirement needed for process:

User id and password should be valid.

Basic Sequence:

1. User enters user id and password
2. System validates user by checking user sign in details against the database.
3. If the user is valid, sign the user into the system and display the appropriate user interface. If the user is invalid, reject sign in attempt and request user to re-attempt sign in or exit the application



Adding Lecturer to the system

Summary:

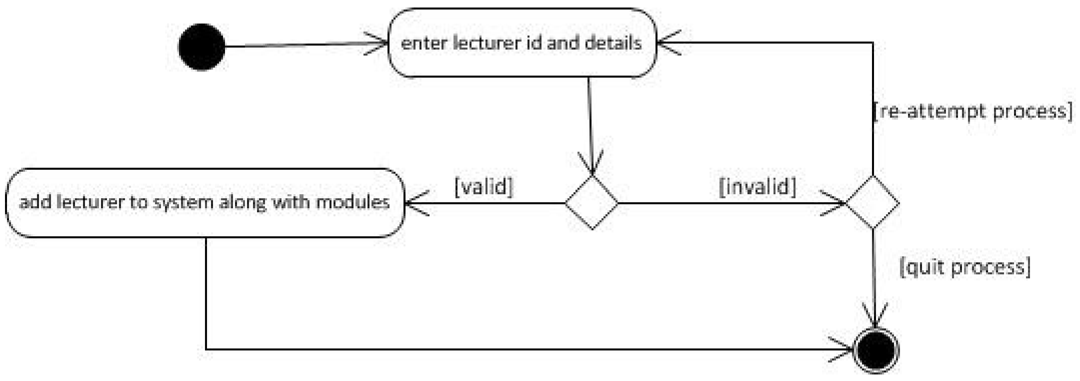
The HOD shall be able to add a lecturer to the system, however this lecturer has to be a valid lecturer at the university.

Requirement needed for process:

Lecturer should be valid lecture at the university.

Basic Sequence:

1. HOD will input lecturer id and details.
2. System will confirm lecturer against university database.
3. If lecturer is valid on university database, lecturer is added to the system along with his/her module. If the lecturer is invalid, the HOD can either re-attempt or quit the process.


enter lecturer id and details
[re-attempt process]
add lecturer to system along with modules
[valid]
[invalid]
[quit process]

Adding TA/Tutor to the system

Summary:

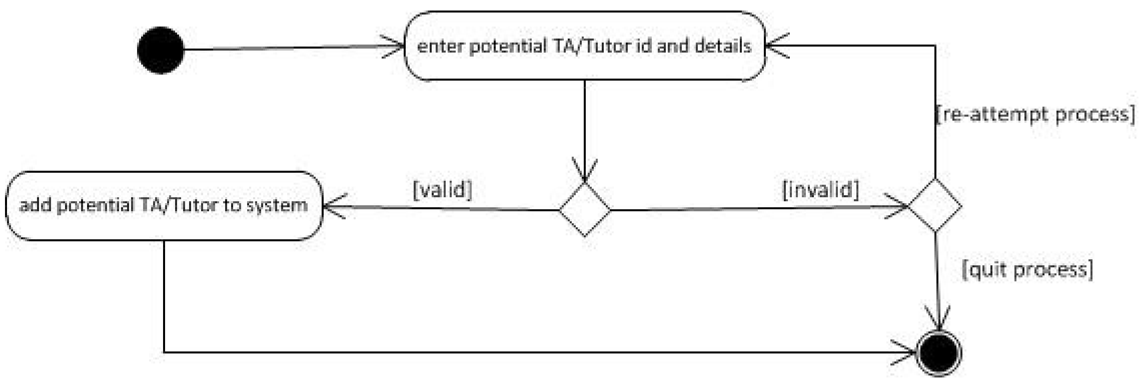
Lecturers shall be able to add TAs and Tutors to the system. However these TAs and Tutors need to be either valid students or lecturers (or employees) at the university

Requirement needed for process:

TA/Tutor is a valid student or lecturer (or employee) at the university.

Basic Sequence:

1. Lecturer attempts to add potential TA/Tutor to the system.
2. System validates potential TA/Tutor against university student and employee database.
3. If potential TA/Tutor exists on the university database, TA/Tutor is added to the system. If TA/Tutor does not exist on the database, the Lecturer may either re-attempt or quit the process.


enter potential TA/Tutor id and details
[re-attempt process]
add potential TA/Tutor to system
[valid]
[invalid]
[quit process]

Locking and finalizing mark-sheet

Summary:

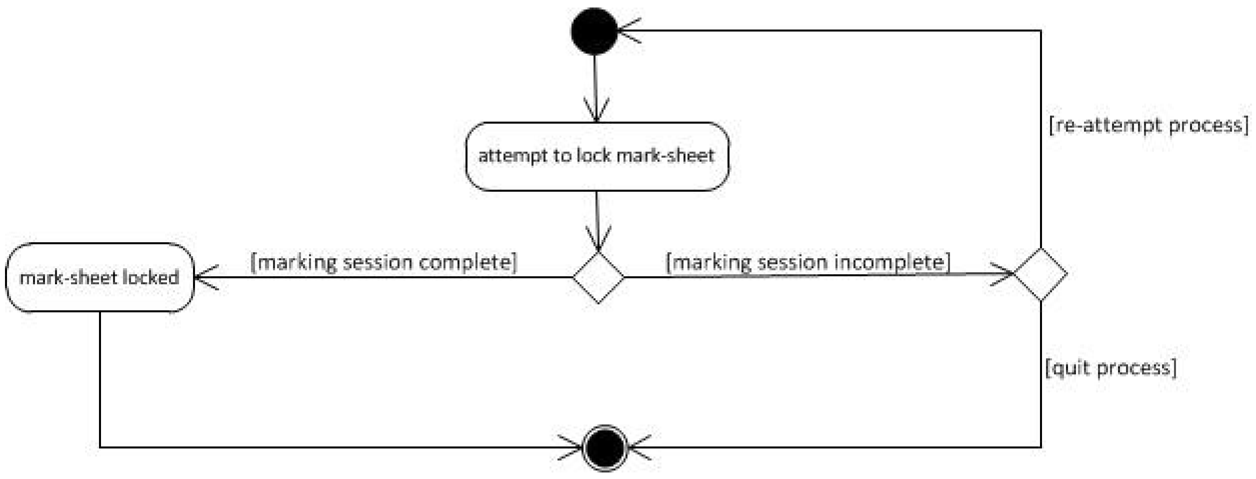
Lecturers shall be able to lock and finalize mark sheets. However, this process is dependent on the actual marking session being completed. Hence the lecturer should not be able to lock a mark sheet while a marking session is in progress.

Requirement needed for process:

Marking session should be complete.

Basic Sequence:

1. Lecturer attempts to lock mark-sheet.
2. If marking session is complete, mark-sheet is locked. If marking session in not complete, mark-sheet lock attempt fails, and lecturer may re-attempt until marking session is complete.


attempt to lock mark-sheet
[re-attempt process]
mark-sheet locked
[marking session complete]
[marking session incomplete]
[quit process]

5.7 Domain Objects

6 Open Issues

7 Glossary