TAEval  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Requirements Analysis Document

**Team Romero’s Severed Head**  
Sean Benjamin  
Dylan Kristolaitis  
Justin Kung

Steven Wu

Submitted to:

Dr. Christine Laurendeau

COMP3004 Object-Oriented Software Engineering  
School of Computer Science  
Carleton University

Contents

1. Introduction………………………………………………………#
   1. Purpose of System………………………………………#
   2. Overview of Document………………………………..#
2. Proposed System……………………………………………….#

2.1. Overview…………………………………………………….#

2.2. Functional Requirements……………………………..#

2.3. Non-functional Requirements………………………#

2.4. System Models……………………………………………..#

2.4.1. Use Case Models…………………………………#

2.4.2. Object Model……………………………………...#

2.4.3. Dynamic Model………………………………….#

1. Glossary…………………………………………………………….#

Figures

tbd

Tables

tbd

1. Introduction

<Sean’s intro here, to have content appended by Steven & edited by Steven>  
1.1 Purpose of System  
1.2 Overview of Document

1. Proposed System
   1. Overview

In this section we outline the technical details of our proposed system, TAEval, by clearly defining functional requirements, non-functional requirements, and outlining unambiguous and complete system models.  
  
TAEval is a client-server application that is designed to optimize the line of communication between an instructor and his or her teaching assistants by automating the issuing and tracking of tasks, task evaluations, and metrics that can quantify the TA’s body of work.

* 1. Functional Requirements

Functional requirements are the concise, explicit details of what the system will be able to do with respect to functionality. For TAEval, for example, there is a distinct difference between the system allowing   
  
Table 1 – Functional Requirements

|  |  |
| --- | --- |
| Traceability Code | Functional Requirement |
| FR-00 | TAs must be able to view their assigned tasks assigned by the course instructor. |
| FR-01 | TAs must be able to view their tasks’ respective evaluation evaluated by the course instructor. |
| FR-02 | TAs must only be assigned to a maximum of one course at any given time. |
| FR-03 | Instructors must be able to create a task at the beginning of the term for each TA for each class they are instructing. |
| FR-04 | Instructors must be able to edit their existing delegated tasks. |
| FR-05 | Instructors must be able to delete their existing delegated tasks. |
| FR-06 | Instructors must be able to enter evaluation data for each existing delegated task. The evaluation scheme is 1-> ‘poor’, 2-> ‘fair’, 3-> ’good’, 4-> ’very good’, 5-> ’excellent’ |
| FR-07 | Instructors must be able to view a list of courses they are teaching in a specific term. |
| FR-08 | Instructors must be able to view the list of TAs that are assigned to a specific course they are instructing. |
| FR-09 | A course must have an existing instructor associated with it upon its creation. |
| FR-10 | Administrators must be able to run reports on TA evaluation data, such as: TA evaluation ratings for one TA spanning all terms, TA evaluation ratings for all TAs spanning one term, TA evaluation ratings for all TAs for a particular course offering |
| FR-11 | Administrators must be able to view a list of courses offered in a given term. |
| FR-12 | Administrators must be able to view a complete list of all instructors. |
| FR-13 | Administrators must be able to view a complete list of all TAs. |
| FR-14 | Administrators must be able to add course offerings. |
| FR-15 | Administrators must be able to edit course offerings. |
| FR-16 | Administrators must be able to delete course offerings. |
| FR-17 | Administrators must be able to add instructors. |
| FR-18 | Administrators must be able to edit instructors. |
| FR-19 | Administrators must be able to delete instructors. |
| FR-20 | Administrators must be able to add TAs. |
| FR-21 | Administrators must be able to edit TAs. |
| FR-22 | Administrators must be able to delete TAs. |
| FR-23 | Administrators must be able to assign existing TAs to any existing course at any time. |

* 1. Non-functional Requirements

|  |  |  |
| --- | --- | --- |
| Traceability Code | Type of NFR | Non-functional Requirement |
| NFR-01 | Usability | TAEval user interface must be graphical in nature. |
| NFR-02 | Usability | TAEval system must be easy to navigate via menu items and dialog boxes. |
| NFR-03 | Usability | TAEval user interface must have a professional look and feel that is consistent with other commercial UI. |
| NFR-04 | Usability | TAEval generated reports must be concise, consisting of summarized evaluation data, formatted as a single line per record. |
| NFR-05 | Usability | Each client process must execute on a different machine and support a single user. |
| NFR-06 | Usability | Data requested by user must be handled by the TAEval client which queries the central server, accessible at a configurable IP address, to populate the user’s client UI. |
| NFR-07 | Usability | All fields for user text input must have an upper limit that cannot be exceeded. |
| NFR-08 | Usability | All save operations must be confirmed by the user. |
| NFR-09 | Usability | All delete operations must be confirmed by the user. |
| NFR-10 | Usability | TAEval user interface must have the same color scheme that Carleton University uses. |
| NFR-11 | Usability | Explicit documentation on how to install and configure TAEval should be provided |
| NFR-12 | Reliability | All exceptions should be handled gracefully with appropriately detailed error messages |
| NFR-13 | Reliability | If TAEval crashes while an operation leading to a change in the database is occurring, the change must be halted and removed and the system should offer to restore itself to the last safe state. |
| NFR-14 | Performance | User must be able to view up to date information on the client UI instantly. |
| NFR-15 | Performance | There should be no duplication of data anywhere in the system. |
| NFR-16 | Supportability | TAEval must be built to run on a lightweight client such as a mobile device in a future phase. |
| NFR-17 | Supportability | TAEval must be able to support a minimum of four concurrent processes, each on a different host. |
| NFR-18 | Supportability | The system should be extensible to any GUI platform with minimal work required to port over to another. |
| NFR-19 | Implementation | All processes must work on the Linux Ubuntu 12.04 platform. |
| NFR-20 | Implementation | Source code must be written in C++. |
| NFR-21 | Implementation | Data storage organization must be designed for ease of retrieval and efficient use of storage space. |
| NFR-22 | Implementation | Data must be stored in SQLite. |
| NFR-23 | Implementation | Client processes must communicate with the central server using TCP/IP sockets. |
| NFR-24 | Interface | Every user must be running a separate client process which provides the TAEval UI. |
| NFR-25 | Operations | Client must be designed to use very little memory and must have no persistent storage. |
| NFR-26 | Operations | All data must be stored centrally on a single host. |
| NFR-27 | Operations | Server process must execute on central host and must manage updates and retrievals of the data. |
| NFR-28 | Operations | Queries to the server must return only the minimum amount of necessary data. |
| NFR-29 | Operations | Almost no data should be stored on the client when the user moves between UI screens. |
| NFR-30 | Operations | No client processes will run on the central server host. |
| NFR-31 | Packaging | The product must be delivered in a CD-ROM with everything necessary to install the program. |
| NFR-32 | Legal | All administrators must agree for all sensitive information to be kept confidential. |

* 1. System Models  
     2.4.1 Use Case Model  
     2.4.2 Object Model

2.4.3 Dynamic Model

1. Glossary