|  |
| --- |
| RIT Senior Project |
| CoVal2 |
| Software Requirements Specification v1.0 |

|  |
| --- |
| Jonathon Leight, Trevin Maerten, Noel Nacion, John O’Connor  2/23/2012 |

Table of Contents

[Document Approval 3](#_Toc319590687)

[Problem 4](#_Toc319590688)

[Introduction 4](#_Toc319590689)

[Purpose 4](#_Toc319590690)

[Scope 4](#_Toc319590691)

[Definitions, Acronyms, and Abbreviations 4](#_Toc319590692)

[History 5](#_Toc319590693)

[Overview 5](#_Toc319590694)

[General Description 6](#_Toc319590695)

[Product Perspective 6](#_Toc319590696)

[Product Functionality 6](#_Toc319590697)

[Users and Characteristics 6](#_Toc319590698)

[Student 6](#_Toc319590699)

[Employer 7](#_Toc319590700)

[Evaluator 7](#_Toc319590701)

[Administrator 7](#_Toc319590702)

[Permissions 7](#_Toc319590703)

[Student 7](#_Toc319590704)

[Employer 8](#_Toc319590705)

[Evaluator 8](#_Toc319590706)

[Administrator 8](#_Toc319590707)

[Permission Matrix 8](#_Toc319590708)

[Evaluation Status States 12](#_Toc319590709)

[Operating Environment 12](#_Toc319590710)

[Design and Implementation Constraints 13](#_Toc319590711)

[Automated Notifications 13](#_Toc319590712)

[User Documentation 13](#_Toc319590713)

[External Interface Requirements 14](#_Toc319590714)

[User Interface 14](#_Toc319590715)

[Software Interfaces 14](#_Toc319590716)

[Communication Interfaces 14](#_Toc319590717)

[Nonfunctional Requirements 15](#_Toc319590718)

[Functional Requirements 15](#_Toc319590719)

[Form Administration 15](#_Toc319590720)

[Evaluations 15](#_Toc319590721)

[Submissions 16](#_Toc319590722)

[Reports 16](#_Toc319590723)

[Notifications 16](#_Toc319590724)

[Users 17](#_Toc319590725)

[Schedule 17](#_Toc319590726)

# Document Approval

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Name** | **Title** | **Date** |
|  |  |  |  |

# Problem

RIT’s current Co-op Evaluation System (CES)—an application used by the Office of Cooperative Education and Career Services (OCECS)—has a number of performance, reliability, usability, and maintainability issues. Among others, session timeouts and submission timeouts are inherent problems of the current CES. A new version started from scratch with up to date technologies needs to be developed.

# Introduction

## Purpose

This Software Requirements Specification (SRS) is intended to be the definitive list of requirements that will be used by the developers to develop the CoVal2 system. The document is intended to help the developers validate that they are building the right application, and verify that the features they have built are built correctly. This document is also intended for the project sponsors to sign off on as a definitive list of requirements.

## Scope

The application, whose requirements are described in this document, is the Co-op Evaluation System v2.0 (CoVal2). CoVal2 will replace the current CES, providing the same functionality as the current system, with fewer performance, reliability, and maintainability issues. CoVal2 will also contain a small number of enhancements to make the system more easily usable by its primary user base.

CoVal2 will have a few different user roles. Each role will have a different primary use for the system including creating and modifying forms, filling out and submitting forms, and generating statistics based on the form submissions. These permissions will be described in more detail in the Users and Characteristics section of this document.

The system will also handle features such as sending email reminders to users who need to submit a form. CoVal2 will be developed with the intention of using data from other OCECS databases rather than duplicating that data. CoVal2 will not be developed to be connected with any other OCECS applications, though it will be designed to use the RIT LDAP system for authentication.

CoVal2 will provide user-definable, parameter-driven reporting of the submitted and unsubmitted forms. CoVal2 will provide quick submission of forms, as well as success messages and error handling, something that the current system does not provide.

## Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition** |
| CES | Co-op Evaluation System |
| CoVal2 | Co-op Evaluation System v2.0 |
| DCE | Distributed Computing Environment |
| Evaluation | A form that has been sent out to students and/or employers to be filled out. The user interface will refer to these as “student work reports” and “work evaluations” for students and employers, respectively. |
| Form | A series of questions that can be answered. |
| IIS | Internet Information Services |
| LDAP | Lightweight Directory Access Protocol |
| Notification | An email message that will be generated and sent to students and/or employees. |
| OCECS | Office of Cooperative Education and Career Services |
| Report | An aggregation of submissions used to display statistics. |
| RIT | Rochester Institute of Technology |
| SRS | Software Requirements Specification |
| Submission | An evaluation that has been submitted and validated. |
| XML | Extensible Markup Language |

## History

The OCECS is in charge of making sure that co-op jobs are helping both RIT students, as well as the companies for which the students work. In order to do this, they require that the students, as well as their employers, fill out forms that ask questions about their co-op experiences. These forms were originally hand-written, mailed to, and then reviewed by, the OCECS. This process proved inefficient, and a Software Engineering senior project team was chosen to develop the CES.

The CES was developed by the senior project team using a Java backend with an Oracle database, and used the technology of the day: XML. A custom XML library was written by one of the students, although no documentation was ever provided for it. The system has lasted for a number of years, but it was not originally designed to handle the number of users that it currently sees. As such, the new CoVal2 system has been requested and assigned to a new senior project team.

## Overview

The rest of this SRS document contains all of the requirements information that is necessary to build the CoVal2 system. The document will continue with a general description of the system, as well as the factors that have affected the decisions made on the project. The document will then list the external interfaces, nonfunctional requirements, and the functional requirements that must be implemented. Finally, it will then cover the schedule of the project.

# General Description

## Product Perspective

The CoVal2 system will completely replace the current CES. All functionality from the CES will be implemented in the CoVal2 system with increased performance, reliability, usability, and maintainability.

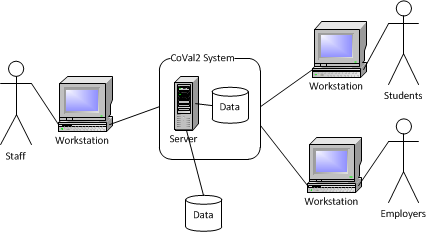


Figure : User/System Interaction

## Product Functionality

CoVal2 will provide the ability for evaluations to be submitted for approval. The evaluations must be approved by a user with the correct permissions. The specific evaluations given can be customized to suit the needs of the department requesting the evaluation. A notification system will be included with the system in order to notify users of any unsubmitted work evaluations. The system can also create reports based on the submitted work reports. Past submitted work evaluations can be searched for and reviewed as needed.

## Users and Characteristics

### Student

The student is one of the key users for this system. As part of the requirements for completing a co-op a student must also submit a student work report[[1]](#footnote-1) for approval. This evaluation will be submitted online through CoVal2. Once an evaluation has been submitted, it will be stored on the server and the student will be able to look at the evaluation at any time.

Since most students do not need to use the system more than once per co-op term, our main concerns for students are usability and performance. High usability is important to us because students do not spend a lot of time on the system, so they need to be able to quickly figure out what they are doing so that they do not need to spend any more time on the system than is necessary. Performance is also important for the same reason.

### Employer

The employer is another of the key users of the system. As with the student, the employer must also fill out and submit a work evaluation form for approval. While the employer isn’t directly given a grade as part of the evaluation, the evaluation form is still necessary in order for the associated student to receive a grade.

Like with the student, the employer’s main concerns with CoVal2 will be usability and performance. The employer will only need to use the system when completing evaluations and looking up previous evaluations. As such, our concerns for employers are the same as for students.

### Evaluator

The evaluator is the user in the system which will evaluate the submissions. After a student has filled in his evaluation, the student is eligible to receive a grade for their co-op. The evaluators will have access to both the student and employer evaluations relating to their department. The evaluators will also be able to send out notifications to both students and employers reminding them to complete the work evaluations. Evaluators also will have the ability to search for specific work evaluations, both ones that have been evaluated and ones that have not.

The evaluator’s needs for the system would also be centered on performance. They would be using more of the system’s functionality than regular users and thus would necessitate that the system is quick and responsive. Usability isn’t as high as a requirement since evaluators use the system more often than students and employers and will not be as likely to forget how to use the system in between uses, but it is still a necessity as they would be working with the system frequently.

### Administrator

The administrator has all the abilities of a regular evaluator as well as specific administrator abilities. The specific administrator abilities center on user management. The administrators would be able to authorize users as students, employers, evaluators or administrators as necessary.

The needs for the administrator would be similar as the evaluator but with a higher emphasis on accessibility. Since the administrators are in charge of maintaining the system it should be available as often as necessary in order to perform the necessary actions.

## Permissions

### Student

A student will be able to fill out, save, and submit, and view the status of evaluations. All previous evaluations that the student has filled out will be available for the student to view. The evaluations that employers fill out about the student will also be visible to the student. The student will also have the ability to verify that the employer evaluation is being sent to the correct person, and if it is not, they will be provided with a form that will allow them to re-submit their placement information with updated employer contact information.

### Employer

An employer will be able to fill out, save, and submit, and view the status of evaluations. Employers will be able to see the evaluations that they have previously submitted. Employers will be able to view current and previous evaluations for a student that they have filled out side-by-side.

Employers that have been verified but have not accessed the system before will receive an email that will provide them with a link that allows them to set up a password for their account. After setting up their password, they will be able to log in and see the evaluations that they need to fill out. Administrators will be able to change their passwords.

Employers, unlike students, will not have the ability to view the student’s evaluation, only their own.

#### Employer Representative

An employer representative will be one step above an employer. They will have the ability to view all of the evaluations for all of the employers that they represent. Unlike the employers, however, they will not be able to edit the evaluations, only view the answers and the status.

### Evaluator

An evaluator will be able to view all evaluations in the system. They will be able to search for specific evaluations based on a number of search criteria. The evaluators will be able to approve or reject an evaluation. Evaluators will be able to view both the student and employer evaluation, and view them side-by-side.

Evaluators will be able to generate reports based on the evaluations in the system. Reports will provide statistical information on the answers that have been submitted by the students and the employers.

Evaluators will be able to view the overall contents of email notifications, as well as change the department specific section of the notifications.

### Administrator

An administrator will be able to do everything that an evaluator can do, with added functionality. Administrators can not only view the overall contents of the email notifications, they can edit it as well. Administrators can also send manual email notifications as well as check the status of the emails that have been sent out by the system.

Administrators will also have the ability to create, edit, and archive forms. This includes the ability to modify and re-order questions on a form. Administrators can also associate which forms are associated with which departments.

### Permission Matrix

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Admin** | **Evaluator** | **Employer** | **Student** |
| **Form Administration** |  |  |  |  |  |
| F1 | The system shall be able to create forms. | r/w | n | n | n |
| F2 | The system shall be able to display forms. | r | r | r | r |
| F3 | The system shall be able to update forms. | r/w | n | n | n |
| F4 | The system shall be able to archive forms. | r/w | n | n | n |
| F5 | The system shall be able to delete forms that have no submissions. | r/w | n | n | n |
| F6 | The system shall allow for the following student progress statuses: Submitted, In Progress – Saved, Open, Pending, Manually Completed, and Archived. | r/w | r | r | r |
| F7 | The system shall allow for the following evaluation approval statuses: Approved, Pending Approval, and Rejected. | r/w | r/w | n | n |
| F8 | The system shall contain an evaluation approval status change trail. | r/w | r | n | n |
| F9 | The system shall allow for the following employer evaluation statuses: Submitted, In Progress – Saved, Open, Pending, Manually Completed, and Archived. | r/w | r | r | r |
| F10 | The system shall automatically change the progress status of all evaluations from Pending to Open three weeks before the end of a term. | r/w | r | r | r |
| F11 | The system shall serialize the answers for the questions for storage in the database. | r/w | r/w | n | n |
| F12 | The answers stored in the database will be queryable in the database with the use of a stored procedure. | r/w | r | n | n |
| **Evaluations** |  |  |  |  |  |
| E1 | The system shall be able to display an evaluation form. | r | r | r | r |
| E2 | The system shall pre-populate and allow for copying of general information from previous evaluations. | r | r | r/w | r/w |
| E3 | The system shall be able to display the status of available evaluation forms. | r | r | r | r |
| E4 | The system shall be able to save non-submitted evaluation forms. | r/w | r | r/w | r/w |
| E5 | The system shall be able to retrieve previously saved non-submitted evaluation forms. | r | r | r | r |
| E6 | The system shall be able to display previously submitted evaluation forms. | r | r | r | r |
| E7 | The system shall be able to validate an evaluation form for completeness. | r | r | r | r |
| E8 | The system shall be able to submit an evaluation form when prompted to do so. | r/w | r | r/w | r/w |
| **Submissions** |  |  |  |  |  |
| S1 | The system shall be able to assign forms to departments. | r/w | r | n | n |
| S2 | The system shall be able to display evaluations. | r | r | n | n |
| S3 | The system shall display the submissions in a format that is printable. | r | r | n | n |
| S4 | The system shall be able to display student and employer submissions side-by-side. | r | r | n | n |
| S5 | The system shall be able to save non-submitted evaluations | r/w | r | n | n |
| S6 | The system shall be able to validate evaluations for completeness. | r | r | n | n |
| S7 | The system shall be able to submit completed evaluations. | r/w | r | n | n |
| S8 | The system shall be able to search forms based on student last name, student first name, student id, company name, term, department, advisor’s DCE username, student progress status, evaluation progress status, and employer evaluation status. | r | r | n | n |
| **Reports** |  |  |  |  |  |
| R1 | The system shall generate reports based on a user-defined selection of submissions and statistics. | r | r | n | n |
| R2 | The system shall produce statistics on all questions that have numeric answers. | r | r | n | n |
| **Notifications** |  |  |  |  |  |
| N1 | The system shall be able to send generated email notifications to students and faculty both manually and automatically. | r/w | r/w | n | n |
| N2 | The system shall be able to configure the department, source address, source name, body and signature of the evaluation notification. | r/w | r | n | n |
| N3 | The system shall be able to generate evaluation notifications to all employers and students with the option of specifying a send date. | r/w | r/w | n | n |
| N4 | The system shall be able to generate a work rejection notification. | r/w | r/w | n | n |
| N5 | The system shall be able to configure the department, source address, source name, body and signature of the work rejection notification. | r/w | r | n | n |
| N6 | The system shall be able to generate a student confirmation email to students and employers. | r/w | r/w | n | n |
| N7 | The system shall be able to configure the department, source address, source name, body and signature of the confirmation notification. | r/w | r | n | n |
| N8 | The system shall be able to display notification statuses for student and employer notifications. | r | r | n | n |
| N9 | The system shall be able to display failed emails and sent emails in the notifications statuses. | r | r | n | n |
| N10 | The system shall send a notification to a user when their evaluation has been rejected. | r/w | r | n | n |
| **Users** |  |  |  |  |  |
| U1 | The system shall use the user information from the already existing OCECS | r | r | n | n |
| U2 | The system shall be able to initialize the next quarter for students and employers. | r | r | n | n |
| U3 | The system shall be able to update submissions’ statuses when supplied with the current status, new status, terms, departments, and students’ submissions or employers’ submissions. | r/w | r/w | n | n |
| U4 | The system shall be able to undo a status update. | r | r | n | n |

## Evaluation Status States

Evaluations, both employer and student, will start out as Pending. In the pending state, the evaluation appears in the system, but cannot be filled out yet.

Three weeks from the end of a term, all evaluations will be changed from Pending to Open. An evaluation in the Open state can be filled out by the employer or student that it is assigned to. An open evaluation can be changed to the Saved, Submitted, Manually Completed, or Archived state.

If the user saves the evaluation, the evaluation is changed to the Saved state. In this state, the user can open the evaluation and continue filling out answers, and can continue saving the evaluation until they are finished with the evaluation. The Saved state will be referred to as “In Progress – Saved” in the user interface. The evaluation change to the Submitted, Manually Completed, or Archived state from the Saved state.

The Submitted state is reached by submitting the form from either the Open state or the Saved state. In this state, the evaluation cannot be changed by any class of user. At this point, the only change that can be made to the evaluation is either an evaluation approval change. If the evaluation is approved, nothing else will happen. If the evaluation is rejected, the evaluation will go back to the Saved state.

There are two other states that an evaluation can end up in: Manually Completed and Archived. If the evaluation is completed in some way other than the normal process, an evaluation can be changed to the Manually Completed state. If the evaluation will never be completed, and the user does not want to receive messages telling them to fill out the evaluation, the evaluation can be marked as Archived. The Archived state was known as the Past Pending state in the previous version of the CES.

## Operating Environment

CoVal2 will be designed to work on the system that the OCECS currently uses for both the CES and their other current applications. This includes the following components:

* Windows Server 2008 R2
* Microsoft SQL Server 2008
* C# .NET
* ASP .NET MVC3 Framework

Other libraries on the NuGet[[2]](#footnote-2) gallery, as well as open source libraries may be used, so long as the libraries do not require any of the following:

* Participation in a community forum
* Registration of company information
* Licensing fees
* Sharing source code and/or binaries

The following libraries are already in use in other OCECS applications, and should be considered requirements if their functionality is desired in CoVal2:

* [Data Annotation Extensions](http://dataannotationsextensions.org/) for more data annotations
* [ELMAH](http://code.google.com/p/elmah/) for error handling and logging
* [Foolproof Validation](http://foolproof.codeplex.com/) for more data annotations and validators
* [jQuery](http://www.jquery.com/) for client side JavaScript
* [Ninject](http://ninject.org/) for dependency injection
* [NUnit](http://www.nunit.org/) for unit testing

Finally, the source code and any other artifacts that need to be version controlled will be stored in the OCECS Mercurial repository.

## Design and Implementation Constraints

CoVal2 will be required to run on a Windows Server 2008 R2 virtual machine. The virtual machine will have a 2.0 GHz processor with 2 GB of RAM. The virtual machine will have .NET 4 installed. The virtual machine will also have IIS 7 installed and configured to serve the application.

## Automated Notifications

Notification emails will be automatically sent by the system at specific times.

The first notification email will be sent to students and employers exactly three weeks before the end of a quarter—this is the same time that the evaluations will change from pending to open.

Notifications will be sent out automatically exactly one week after the previous, until the first week of the next quarter. Notifications will not be sent to students and employers who have submitted their evaluations.

## User Documentation

There will not be any formal user documentation in a document form. The user interface will be intuitive and help messages will be placed in locations on the screen in order to make the application more usable. Helpful error and success messages will also be provided to make the application more usable.

# External Interface Requirements

## User Interface

The system shall adhere to all user interface standards for similar systems. The interface shall be optimized for use within a standard web browser.

## Software Interfaces

The system will interface with RIT’s LDAP in order to authenticate users.

## Communication Interfaces

The system will communicate over the standard internet protocols of HTTP and TCP/IP. For accessing secure sections of the system HTTPS will be used. LDAP will be used to authenticate users.

# Nonfunctional Requirements

|  |  |  |
| --- | --- | --- |
| **Number** | **Priority** | **Requirement** |
| NF1 |  | The system shall be available 95% of the time. |
| NF2 |  | The system shall be modifiable for future updates. |
| NF3 |  | The system shall be able to handle up to 6,000 users at a time. |
| NF4 |  | The system shall run on the OCECS provided VM at final release. |
| NF5 |  | The system shall include a test plan, including up to 95% code coverage. |
| NF6 |  | The system shall be usable from all major up-to-date web browsers. |
| NF7 |  | The system shall not time out during evaluation submission more than 5% of the time. |
| NF8 |  | The system shall respond to requests within three seconds. |

# Functional Requirements

## Form Administration

|  |  |  |
| --- | --- | --- |
| **Number** | **Priority** | **Requirement** |
| F1 |  | The system shall be able to create forms. |
| F2 |  | The system shall be able to display forms. |
| F3 |  | The system shall be able to update forms. |
| F4 |  | The system shall be able to archive forms. |
| F5 |  | The system shall be able to delete forms that have no submissions. |
| F6 |  | The system shall allow for the following student progress statuses: Submitted, In Progress – Saved, Open, Pending, Manually Completed, and Archived. |
| F7 |  | The system shall allow for the following evaluation approval statuses: Approved, Pending Approval, and Rejected. |
| F8 |  | The system shall contain an evaluation approval status change trail. |
| F9 |  | The system shall allow for the following employer evaluation statuses: Submitted, In Progress – Saved, Open, Pending, Manually Completed, and Archived. |
| F10 |  | The system shall automatically change the progress status of all evaluations from Pending to Open three weeks before the end of a term. |
| F11 |  | The system shall serialize the answers for the questions for storage in the database. |
| F12 |  | The answers stored in the database will be queryable in the database with the use of a stored procedure. |

## Evaluations

|  |  |  |
| --- | --- | --- |
| **Number** | **Priority** | **Requirement** |
| E1 |  | The system shall be able to display an evaluation form. |
| E2 |  | The system shall pre-populate and allow for copying of general information from previous evaluations. |
| E3 |  | The system shall be able to display the status of available evaluation forms. |
| E4 |  | The system shall be able to save non-submitted evaluation forms. |
| E5 |  | The system shall be able to retrieve previously saved non-submitted evaluation forms. |
| E6 |  | The system shall be able to display previously submitted evaluation forms. |
| E7 |  | The system shall be able to validate an evaluation form for completeness. |
| E8 |  | The system shall be able to submit an evaluation form when prompted to do so. |

## Submissions

|  |  |  |
| --- | --- | --- |
| **Number** | **Priority** | **Requirement** |
| S1 |  | The system shall be able to assign forms to departments. |
| S2 |  | The system shall be able to display evaluations. |
| S3 |  | The system shall display the submissions in a format that is printable. |
| S4 |  | The system shall be able to display student and employer submissions side-by-side. |
| S5 |  | The system shall be able to save non-submitted evaluations |
| S6 |  | The system shall be able to validate evaluations for completeness. |
| S7 |  | The system shall be able to submit completed evaluations. |
| S8 |  | The system shall be able to search forms based on student last name, student first name, student id, company name, term, department, advisor’s DCE username, student progress status, evaluation progress status, and employer evaluation status. |

## Reports

|  |  |  |
| --- | --- | --- |
| **Number** | **Priority** | **Requirement** |
| R1 |  | The system shall generate reports based on a user-defined selection of submissions and statistics. |
| R2 |  | The system shall produce statistics on all questions that have numeric answers. |

## Notifications

|  |  |  |
| --- | --- | --- |
| **Number** | **Priority** | **Requirement** |
| N1 |  | The system shall be able to send generated email notifications to students and faculty both manually and automatically. |
| N2 |  | The system shall be able to configure the department, source address, source name, body and signature of the evaluation notification. |
| N3 |  | The system shall be able to generate evaluation notifications to all employers and students with the option of specifying a send date. |
| N4 |  | The system shall be able to generate a work rejection notification. |
| N5 |  | The system shall be able to configure the department, source address, source name, body and signature of the work rejection notification. |
| N6 |  | The system shall be able to generate a student confirmation email to students and employers. |
| N7 |  | The system shall be able to configure the department, source address, source name, body and signature of the confirmation notification. |
| N8 |  | The system shall be able to display notification statuses for student and employer notifications. |
| N9 |  | The system shall be able to display failed emails and sent emails in the notifications statuses. |
| N10 |  | The system shall send a notification to a user when their evaluation has been rejected. |

## Users

|  |  |  |
| --- | --- | --- |
| **Number** | **Priority** | **Requirement** |
| U1 |  | The system shall use the user information from the already existing OCECS database to provide authorization. |
| U2 |  | The system shall be able to initialize the next quarter for students and employers. |
| U3 |  | The system shall be able to update submissions’ statuses when supplied with the current status, new status, terms, departments, and students’ submissions or employers’ submissions. |
| U4 |  | The system shall be able to undo a status update. |

# Schedule

|  |  |
| --- | --- |
| **Date** | **Deliverables** |
| 2012-02-16 | Sign off on the requirements document. |
| 2012-02-16 | Be familiarized with development tools. |
| 2012-02-23 | Sign off on the architecture document, staff interviews complete. |
| 2012-04-19 | Finish agreed upon functional requirements of the system. |
| 2012-05-03 | Finish usability and accessibility improvements, deliver final product. |
| 2012-05-17 | Finish poster for Senior Project Day. |

1. For the purposes of this document, a “student work report” will be synonymous with an “evaluation.” [↑](#footnote-ref-1)
2. <http://www.nuget.org/> [↑](#footnote-ref-2)