**Software Requirements Specification**

**for**

**Project Codename Olympia**

**Version 3.0 approved**

**Prepared by Landen Marchand, Josh King, Andrew Unger**

**JAL Group**

**04/01/2018**

**Table of Contents**

**Table of Contents** 2

**Revision History** 2

**1. Introduction** 3

1.1 Purpose 3

1.2 Document Conventions 3

1.3 Intended Audience and Reading Suggestions 3

1.4 Product Scope 3

1.5 References 3

**2. Overall Description** 4

2.1 Product Perspective 4

2.2 Product Functions 4

2.3 User Classes and Characteristics 4

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 5

2.7 Assumptions and Dependencies 5

**3. External Interface Requirements** 5

3.1 User Interfaces 5

3.2 Hardware Interfaces 5

3.3 Software Interfaces 5

3.4 Communications Interfaces 6

**4. System Features** 6

4.1 System Feature 1 6

4.2 System Feature 2 (and so on) 7

**5. Other Nonfunctional Requirements** 7

5.1 Performance Requirements 7

5.2 Safety Requirements 7

5.3 Security Requirements 7

5.4 Software Quality Attributes 7

5.5 Business Rules 7

**6. Other Requirements** 8

**Appendix A: Glossary** 8

**Appendix B: Analysis Models** 8

**Appendix C: To Be Determined List** 8

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| **Landen Marchand** | 02/28/2018 | Added sections [1.1-1.4 , 2.2] | 1.0 |
| **Landen Marchand** | 03/14/2018 | Added sections [2.1, Appendix A] | 2.0 |
| **Landen Marchand** | 03/30/2018 | Added Header Title. Edited 2.1 and 2.2. Added Appendix A and B, Section 1.5 | 3.0 |

# Introduction

## Purpose

The purpose of *Project Codename Olympia* (**PCO**) is to have a fully automated system for the various skating events in the *Winter Olympics* (**WO**) . This automated system takes care of registration for each team, also known as country, scoring for each team and individual, and general record keeping of the events, times, and teams.

## Document Conventions

Major titles, such as the project name and the event itself are first written out completely and in italics. From there, they will be denoted by their respective acronym, which is provided immediately after the full name in bold.

## Intended Audience and Reading Suggestions

The intended audience for this documentation comprises the following entities: the *Olympic League* (**OL**) which oversees and organizes all of the **WO** skating events, and the software development team constructing **PCO**. There are no special reading suggestions as of version 1.0 and should be read in a straightforward manner.

## Product Scope

**PCO** manages the registration of individuals and teams, scheduling of events, allocation of qualified judges, and manages a running database of scores and standings for the **WO** skating events which, in turn, provides a fully automated system for the **OL**.

## References

**Appendix A:** Documentation for Data Dictionary / Class Specifications

**Appendix B:** Visual of Class Diagrams / Use Cases and Use Case Diagram

# Overall Description

## Product Perspective

**PCO** is a replacement of humans to manually coordinate the **WO** skating events. This system will be self-contained automating the registration, scheduling, allocation of judges, and statistics of the **WO** skating events. **PCO** will contain subsystems consisting of registration, scheduling of/and events, scoring, and a database for an aggregation of information. The definitions of these subsystems can be found in the Data Dictionary file.

## Product Functions

* Register countries as teams and individuals as athletes
* Allocate each event to a specific rink
* Allocate athletes to each event
* Assign judges to each event
* Record scores/times from each event
* Display event information, scores, and current standings

## User Classes and Characteristics

**Officials of the WO**

* Official Registrant - Registers teams and athletes
* Official Scheduler - Handles scheduling of events, judges, and athletes
* Official Judge - Verifies and processes scores
* General Official - Display current information of the WO

## Operating Environment

*<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>*

## Design and Implementation Constraints

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>*

## User Documentation

*<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>*

## Assumptions and Dependencies

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

# External Interface Requirements

## User Interfaces

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

## Hardware Interfaces

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>*

## Software Interfaces

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

## Communications Interfaces

*<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>*

# System Features

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

## System Feature 1

*<Don’t really say “System Feature 1.” State the feature name in just a few words.>*

4.1.1 Description and Priority

*<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>*

4.1.2 Stimulus/Response Sequences

*<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>*

4.1.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>*

## Safety Requirements

*<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>*

## Security Requirements

*<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>*

## Software Quality Attributes

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

## Business Rules

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

# Other Requirements

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

**Appendix A: Glossary**

**PCO** - Project Codename Olympia

**WO** - Winter Olympics

Data\_DictionaryV3.docx

ClassSpecifications.docx

**Appendix B: Analysis Models**

ClassDiagramV5Functions.png

ClassDiagramV6Uses.png

ClassDiagramV7Attributes.png

UseCases2.docx

UseCaseDiagramV4.png

**Appendix C: To Be Determined List**