**Red Team**

**Stargazer**

**Automatic Telescope Control System**

**Project Plan**

**Team Members:**

Jason Dunscombe

Rob Grmek

Robert Smith

**Instructor:**

**Youry Khmelevsky**

**Course:**

**COSC 470**

**Date:**

**November 29th, 2009**

# ****Revision History****

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 10/16/2009 | 1.0 | Document created. | Robert, Rob, Jason |
| 11/01/2009 | 1.1 | Added information for 2nd iteration. | Rob |
| 11/08/2009 | 1.2 | Added additional information for release #1 and #2. | Rob |
| 11/13/2009 | 1.3 | Minor updates. | Rob |
| 11/16/2009 | 1.4 | Added metrics for release #1 | Rob |
| 11/24/2009 | 1.5 | Added metrics for release #2 | Rob, Robert |
| 11/29/2009 | 1.6 | Added some changes. | Rob |

# ****Table of Contents****

**Page**

|  |  |
| --- | --- |
| Introduction | 1 |
| Risk Management Plan | 2 |
| Iteration 1: Inception, Partial Elaboration and Initial Design | 3 |
| Iteration 2: SPIKE Project | 5 |
| Iteration 3: Prototype (Release #1) | 6 |
| Iteration 4: Prototype (Release #2) | 7 |

# ****Project Plan****

This document covers the planning process for the design, development and deliverance of our Stargazer projectfor our customers Alexander and Nan.

Note:

Durations in our planning process is measured in units. 1 unit is equal to 15 minutes.

Task durations tracked with Time Tracker: <http://timetracker.wrconsulting.com>

Furthermore, the analysis of possible risks with corresponding risk management solutions which attempt to either minimize or avoid the risk from becoming an issue is included as well in this document.

# Risk Management Plan

|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Probability | Effect | Solution |
| Underestimate the time the project will take. | Medium to high. | Serious – we run out of time. | Remove non-essential features. |
| Illness to team members. | Low to medium. | Tolerable – We work without that team member until they are better. | Assign and distribute their tasks to other team members. |
| Not understanding requirements. | Medium to high. | Serious -The system behaves incorrectly. | Keep in close contact with the customer. |
| Scheduling conflicts between team members. | Low to medium. | Serious – Individual programming is not XP programming. | Schedule well before hand. |
| No available lab space. | Medium. | Tolerable – need to find other areas to work. | Use laptops in meeting rooms or other areas. |
| Server or technical issues. | Low to medium. | Catastrophic – May lose work. | Backup regularly; use a versioning control system. |
| Campus network problems | High. | Serious – Reduced productivity, difficulty accessing development server. | Work directly on development server or work locally on a laptop computer. |
| Lack of time at end of semester | High. | Tolerable – Reduced ability to work outside of class/lab time. | Use class/lab time as productively as possible. |
| Client unavailable or unresponsive. | Medium. | Catastrophic – No customer input. | Use alternative methods of communication (phone or email).  May have to select a different project. |

Probabilities can be: low, moderate, high.

Effects can be: tolerable, serious and catastrophic.

# Iteration 1: Inception, Partial Elaboration and Initial Design

We have selected our project and have begun drafting initial documentation related to the project and set up a development environment.

Also, we planned to run through several Ruby on Rails related tutorials to learn how to use the development, how to use versioning control with Git and how to use unit testing with Ruby’s built-in unit testing.

### Task Duration and Dependencies:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **#** | **Estimated Duration** | **Dependency** | **Actual Duration** |
| Create initial vision | 1 | 2 | - | 2 |
| Create project plan document | 2 | 2 | - | 3 |
| Add business case and product info to vision | 3 | 4 | 1 | 3 |
| Add architecture to vision | 4 | 1 | 1 | 1 |
| Add risks to project plan | 5 | 2 | 2 | 1 |
| Research image recognition technologies | 6 | 24 | - | - |
| Research hardware I/O specifications | 7 | 24 | - | - |
| Research development environment components | 8 | 24 | - | 12 |
| Set up development environment | 9 | 16 | 8 | 8 (inc.) |
| Create tutorial plan | 10 | 2 | - | 2 |
| Create initial developer's guide | 11 | 8 | - | 6 |
| Create use case diagram | 12 | 2 | - | 2 |
| Write fully dressed use cases | 13 | 6 | 12 | 4 |
| Complete tutorials using our framework | 14 | 32 | 8, 9 | - |
| Draft acceptance test | 15 | 4 | - | - |
| Add installation guide information | 16 | 4 | 11 | 4 |

### Tutorial Plan:

Setting up a development environment on Windows:

<http://www.akitaonrails.com/2009/1/13/the-best-environment-for-rails-on-windows>

Setting up a development environment on Fedora 11:

<http://www.technetra.com/2009/04/22/howto-setting-up-ruby-on-rails-for-fedora-10-and-11/>

Rails Guides:

1. <http://guides.rubyonrails.org>
2. <http://www.tutorialspoint.com/ruby-on-rails-2.1/index.htm>
3. <http://www.meshplex.org/wiki/Ruby/Ruby_on_Rails_programming_tutorials>

Rails Development Tutorials:

1. <http://guides.rubyonrails.org/getting_started.html>
2. <http://www.rails4days.pwp.blueyonder.co.uk/Rails4Days.pdf>
3. <http://www.tutorialized.com/view/tutorial/Image-uploads-and-resizing-for-Rails-models-with-mini-magick/19070>

Git Tutorial:

<http://harryseldon.thinkosphere.com/2009/01/14/git-and-rails-a-detailed-tutorial-including-plugins-submodules-development-and-production>

# Iteration 2: SPIKE Project

We have researched the technologies related to the project in iteration 1 and can now begin a short one week SPIKE project to learn the Ruby on Rails framework. The SPIKE project we created was that of a blogging site.

The SPIKE project must:

* Have login system and authentication; since this is important to learn how to do for our real project.
* Be committed regularly to our GitHub repository and use source control management (SCM).
* Have unit testing; we’ll need to use test-driven development (TDD) in our real project, so it is key to learn the built-in ruby testing tool.

### List of Tasks:

|  |  |  |
| --- | --- | --- |
| **Task** | **Estimated Duration** | **Actual Duration** |
| SPIKE: Unit testing tutorials | 4 | 3 |
| SPIKE: Use Git to push/pull project to/from GitHub | 2 | 2 |
| SPIKE: Post scaffolding tutorial | 6 | 5 |
| SPIKE: Comment and tag functionality tutorial | 6 | 6 |
| SPIKE: Collect SPIKE code for submission | 2 | 3 |
| Design: Create domain model | 4 | 4 |
| Design: Create architecture diagram | 2 | 2 |
| Documentation: Add using Git information to developer's guide | 4 | 3 |
| Documentation: Update project plan for release #1 and #2 | 4 | 6 |
| Documentation: Add architecture diagram to developer's guide | 2 | 2 |
| Documentation: Get extra user stories from customer | 3 | 4 |
| Documentation: Prioritize the user stories | 1 | 1 |
| Documentation: Update installation information | 4 | 4 |
| Documentation: Update vision | 2 | 2 |
| Set up development environment | 8 | 6 |
| Test telescope interface/drivers | 12 | 26 |
| Test telescope programs (Autostar, ASCOM) | 8 | 10 |
| Total: |  | 89 |

### Milestones:

* SPIKE completion: Nov. 6, 2009

# Iteration 2: Prototype (Release #1)

From our knowledge gained from the SPIKE project, we can begin development of the web application portion of our project.

In this one week iteration, we plan to implement the following functionality:

* Authentication; force the user to login in order to access the system.
* Scheduling telescope position; show a form to allow the user to input a time, date, coordinates, etc which contains all the information necessary to properly position the telescope.
* Review scheduling logs; show entries for all schedules that have been created in the past, present and future.
* Online help documentation; gives users information on using the system.

### List of Tasks:

|  |  |  |
| --- | --- | --- |
| **Task** | **Estimated Duration** | **Actual Duration** |
| Documentation: Update testing plan | 3 | 2 |
| Create web system: Add authentication | 12 | 11 |
| Create web system: Add scheduling form (with validation/tests) | 16 | 18 |
| Create web system: Create scheduling review logs | 2 | 1 |
| Create web system: Add online help documentation | 12 | 4 |
| Create web system: Add authorization | 16 | 23 |
| Documentation: Create change requests document | 1 | 1 |
| Documentation: Update all other documents. | 24 | 18 |
| Design: Create class diagram | 2 | 8 |
| Design: Update use case diagram | 1 | 1 |
| Refactoring | 4 | 4 |
| Total: | 93 | 89 |

### Milestones:

* Release #1: Nov. 18, 2009

#### Metrics:

Unit Testing:

* Unit tests: 2
* Test cases: 18
* Assertions: 31

Implementation:

* Views: 15 (including partials)
* Models: 3
* Controllers: 4

# Iteration 4: Prototype (Release #2)

In this iteration, we are to primarily build onto the web systems functionality. We plan to create:

* Stylesheets; create separate CSS for both mobile device users and PC users for easier visibility and usability.
* An image gallery; show latest composite images from the telescope.
* HTTPS/HTTP for admin/user
* Set up production server (use Apache as production web server)
* Changes to schedule input forms and model

Also, extensive refactoring must be done in this iteration.

### List of Tasks:

|  |  |  |
| --- | --- | --- |
| **Task** | **Estimated Duration** | **Actual Duration** |
| Add separate stylesheets for PC users and mobile device users | 8 | 8 (inc.) |
| Add gallery section - display latest images from telescope | 16 | 22 |
| HTTPS/HTTP for admin/user | 8 | 6 |
| Set up production server | 8 | 10 |
| Update all documentation and create design diagrams | 40 | 6 (inc.) |
| Refactoring | 20 | 6 |
| Change schedule input forms | 12 | 16 |
| Fix unit tests | 4 | 4 |
| Fix CSS issues | 4 | 5 |
| Create PowerPoint Presentation | 6 | 10 |
| Total: | 100 | 63 |

### Milestones:

* Release #2: Nov. 25, 2009
* Refactored Release #2: Dec. 4, 2009
* Final Presentation: Dec. 4, 2009

#### Metrics:

Unit Testing:

* Unit tests: 3
* Test cases: 17
* Assertions: 29

Implementation:

* Views: 26 (including partials)
* Models: 4
* Controllers: 6