Software Quality Assurance Plan

For Multiagent Control of Traffic Signals

Version 1.0

Submitted in partial fulfillment of the requirements of the degree of MSE

Bryan Nehl
CIS 895 – MSE Project
Kansas State University

Table of Contents

1 Purpose

This document serves as the Software Quality Assurance (SQA) plan for the Multiagent Control of Traffic Signals system. This project is the Master of Software Engineering final project for Bryan Nehl.

2 References

1. IEEE Std. 730-1998, IEEE Standard for Software Quality Assurance Plans, IEEE 1998.

2. IEEE Std. 730.1-1995 IEEE Guide for Software Quality Assurance Planning, IEEE, 1995.

3. Python Software Foundation, "PEP 8 -- Style Guide for Python Code", Python, 24 Sep. 2011; http://www.python.org/dev/peps/pep-0008/.

4. Python Software Foundation, "PEP 257 – Docstring Conventions", Python, 24 Sep. 2011; http://www.python.org/dev/peps/pep-0257/.

5. K. Hill, "GMoDS-based Runtime Agent Role Interpreter SQA Plan 1.0", People, 15 Sep. 2011; http://people.cis.ksu.edu/~kylhill/phase_1/sqa_plan.pdf.

6. B. Nehl, "Multiagent Control of Traffic Signals Project Plan 1.0", People, 26 Sep. 2011; http://people.cis.ksu.edu/~bnehl/repos/macts.git/.

7. W. Royce, Software Project Management; Addison-Wesley, 1998, pp. 290-291.

3 Management

3.1 Management Organization

3.1.1 Supervisory Committee

- Dr. Scott DeLoach
- Dr. Gurdip Singh
- Dr. David Gustafson

3.1.2 Major Professor

- Dr. Scott DeLoach

3.1.3 Developer

- Bryan Nehl

3.1.4 Technical Inspectors

- Inspector 1 (To be determined)
- Inspector 2 (To be determined)

3.2 Tasks

The project plan [6] details the major tasks that are to be completed.

3.3 Roles and Responsibilities

3.3.1 Supervisory Committee

The supervisory committee will review the materials that are provided for the presentations. Committee members will attend the presentations and will approve the materials or provide a list of specific issues that need to be addressed to gain acceptance. The developer may call on the committee as subject matter experts.

3.3.2 Major Professor

The major professor is responsible for providing guidance and feedback to the developer. The major professor will review materials before presentations to the supervisory committee for compliance to project requirements and standards. Then they will suggest areas that may need additional refinement.

3.3.3 Developer

The developer is responsible for implementing the project, creating and providing all materials for the presentations. It is also the responsibility of the developer to identify and remediate risks. The developer is also responsible for addressing any of the concerns or issues brought by the supervisory committee and major professor.

3.3.4 Technical Inspectors

Technical Inspectors are responsible for using the provided inspection check list to complete a formal technical inspection. Once completed, the inspector should provide the completed check list along with any issues identified in a letter to the major professor and the developer.

4 Documentation

All project documentation will be published at http://people.cis.ksu.edu/~bnehl/.

4.1 Purpose

The purpose of the documentation is to provide insight into the status of progress on the project. In addition it should guide other researchers in understanding the project, the work involved in the project and the associated costs.

4.2 Minimum Documentation Requirement

The minimum documentation necessary for showing project progress is identified in the following subsections.

4.2.1 Phase 1

- Time Log
- Risk Log
- Vision Document 1.0
- Project Plan 1.0
- Software Quality Assurance Plan
- Presentation 1

4.2.2 Phase 2

- Time Log

- Risk Log
- Action Items from Phase 1
- Vision Document 2.0
- Project Plan 2.0
- Formal Requirements Specification
- Architectural Design 1.0
- Test Plan
- Formal Technical Inspection Checklist
- Executable Architectural Prototype
- Presentation 2

4.2.3 Phase 3

- Time Log
- Risk Log
- Action Items from Phase 2
- User Manual
- Component Design
- Source Code
- Assessment Evaluation
- Project Evaluation
- References
- Formal Technical Inspection Letters
- Presentation 3

5 Standards, Practices, Conventions and Metrics

The IEEE standards for documents in [1] and [2] will be followed. I will follow the code conventions known as PEP 8 [3] and PEP 257 [4]. To aid in this the pep8 tool will be used to examine the code.

The developer will use PyCharm integrated code inspector to check code for common errors. The code will be test driven and mocks utilized where practical. Where test driving is not practical because of the use of external components, I will utilize solution spikes. These spikes will serve to explore the intended architecture/solution without simultaneously trying to incorporate it into the core application. I will strive to create code that honors the object-oriented principles and exhibits low cyclomatic complexity.

I will use the rework metric as a measure of code quality. The metric as presented in [7] is:

$$RW = \frac{E_{Rework}}{E_{Development}}$$

E_{Rework} is the amount of effort spent in rework. $E_{Development}$ is the overall amount of development effort. I will be able to provide this metric by reviewing my engineering notebook and viewing effort as a unit of time. Rework will then be a decimal between 0 and 1. For this project, my goal is to keep the rework below 0.20.

6 Reviews and audits

The supervisory committee will review all materials at the three presentations. The major professor will review materials as they are provided by the developer. The technical inspectors will perform a formal technical inspection. The developer will use tools to that generate metrics to watch code quality.

7 Test

During Phase 2 a comprehensive test plan will be created which will include unit and integration tests.

8 Problem reporting and corrective action

When the developer becomes aware of a problem they will notify the major professor of the issue, the impact or severity of the issue and the mitigation or remediation plan. The developer may discover the problem themselves or have it communicated to them by a reviewer. The developer will maintain a risk/defect log. The risk log will include: identification number, type (risk or defect) who identified the risk, short title, description of the risk, mitigation or remediation used, responsible reviewer, date when identified, date developer completed and date when accepted/approved.

The major professor will advise the developer of any discernible issues and suggest potential approaches to mitigation or remediation. Communication will primarily occur between presentations and via email.

The supervisory committee will identify issues for the developer to address as well. They will primarily communicate during the presentations and with follow-up email. The supervisory committee will review and approve mitigation and remediation actions as subsequent presentations.

Technical inspectors will review the system and provide detailed feedback of potential problem areas. Reviewers will provide feedback via email or written response.

9 Tools, techniques, and methodologies

- Python 2.7 will be used for implementing the agents and tests.
- RabbitMQ is a message queue platform which is how the agents communicate.
- MongoDB is a document store database where agents store settings and knowledge base information.
- PyMongo is a Python interface library for MongoDB.
- pika is a Python interface library for RabbitMQ.
- Unittest is a Python module for doing unit testing.
- PEP8 is a tool that will be used for compliance with Python coding standards.
- Coverage is a Python module for evaluating test coverage.
- Simulation for Urban Mobility is an environment for running traffic simulations.
- Microsoft Word 2010 and Excel 2010 will be used for creating the documentation.
- PyCharm is an integrated development environment for Python. It will be used for creating the unit tests and source code as well as static code analysis.

- PyMetrics is a package that will be used to check the cyclomatic complexity of the code as well as gather other metrics.
- Visual Paradigm will be used for creating UML models.

10 Code control, media control and supplier control

The developer will also make use of a repository that is on the development machine. The source code and other project documents will be kept in a git repository at http://people.cis.ksu.edu/~bnehl/repos/macts.git/. The local repository will be pushed to the web based repository at least once a week. All milestones will be tagged in the repository.

Supplier control is not applicable to this project.

11 Record collection, maintenance and retention

Technical reviewer documentation will be transformed into a PDF format and included in a feedback folder.

PDF versions of all documents will be created prior to each presentation. The PDF versions will be inside the portfolio folder. The developer will also create a zip file for each presentation which will include all project materials (documents, source code and diagrams). The zip file will be made available for download on the http://people.cis.ksu.edu/~bnehl/ web page.

At a minimum, the git repository and the website will be maintained for the duration of the project. At the completion of the project, the repository and website files will stay in place for an indefinite period of time. At the request of the Major Professor or the supervisory committee the materials may be moved to a department website.

12 Risk management

The developer will be responsible for identifying, and mitigating risk. When a new risk item is identified it along with the remediation strategy will be communicated to the major professor.

The major professor will review and advise the developer regarding the risks and mitigation strategies. The major professor will also let the developer know if they identify additional potential risks.