SE 4485: Software Engineering Projects

Spring 2025

Project Management Plan

| Group Number | 9 |
| --- | --- |
| Project Title | QNX and Python Implementation |
| Sponsoring Company | Communications and Power Industries |
| Sponsor(s) | Luke Carr |
| Students | 1. Muhammad Ali sxa220115@utdallas.edu  2. Nicholas Anderson nba220001@utdallas.edu  3. Tabark Abaid tya210000@utdallas.edu  4. Diego Ibarra dxi200004@utdallas.edu  5. Saghar Abdi sxa190148@utdallas.edu  6. Khaled Elkhaled kye210001@utdallas.edu |

QNX and Python Implementation Project Management Plan

**ABSTRACT**

This document outlines the project plan for integrating QNX and Python for Communications and Power Industries. It covers the team structure, development approach, risks, required resources, and project timeline. The goal is to ensure a smooth development process while managing challenges and meeting project deadlines. It talks about the project and scope of the project as well as what lifecycle we will be using for the project.

# 

# **TABLE OF CONTENTS**

# Abstract

# List of Figures

# List of Tables

# Introduction 4.1 Purpose and Scope 4.2 Product Overview 4.3 Document Structure

# Project Organization 5.1 Team Structure and Roles 5.2 Rationale for Organization

# Lifecycle Model Used 6.1 Chosen Model 6.2 Justification

# Risk Analysis 7.1 Identified Risks 7.2 Risk Likelihood and Impact 7.3 Risk Mitigation Strategies

# Software and Hardware Resource Requirements 8.1 Software Requirements 8.2 Hardware Requirements 8.3 Availability in Lab

# Deliverables and Schedule 9.1 Project Milestones 9.2 Task Dependencies 9.3 Resource Allocation

# Monitoring, Reporting, and Controlling Mechanisms 10.1 Reporting Structure 10.2 Monitoring Tools and Techniques 10.3 Control Mechanisms

# Professional Standards 11.1 Team Expectations 11.2 Scholastic Honesty and Conduct 11.3 Quality and Deadline Standards

# Evidence of Configuration Management

# Engineering Standards and Multiple Constraints 13.1 Relevant IEEE and ISO Standards 13.2 Compliance Considerations

# Additional References

# Appendix A: Professional Standards Guideline

**INTRODUCTION**

· This document is the Project Management Plan for the QNX and Python Implementation project, which will be developed in collaboration with Communications and Power Industries. It outlines the team structure, development process, risk management strategies, required resources, and project timeline to ensure that the development is well organized and successful.

· The purpose of this document is to define the structure and framework for the project, ensuring that all stakeholders understand the goals, expectations, and constraints. This plan provides a structured approach to development, tracking, and risk mitigation to make sure the deliverables are submitted on time.

The scope of this project includes:

Developing a QNX-based system with Python integration.

Ensuring compatibility with CPI’s hardware and software requirements.

Delivering a functional implementation, along with technical documentation and test results.

· The product aims to integrate Python-based functionalities within the QNX operating system, allowing efficient communication with CPI’s embedded systems. The key capabilities include: Real-time data processing using QNX. Python integration for scripting and automation. Interfacing with CPI hardware through secure protocols.

· The structure of the document is as follows:

Project Organization – Team structure and roles.

Lifecycle Model – Development approach and rationale.

Risk Analysis – Identification and mitigation of potential risks.

Software and Hardware Requirements – Necessary tools and resources.

Deliverables and Schedule – Project timeline and major milestones.

Monitoring and Reporting – How progress will be tracked and reported.

Professional Standards – Team responsibilities and ethical considerations.

Engineering Standards – Compliance with IEEE, ISO, and industry standards.

# 

# 

# 

# **PROJECT ORGANIZATION**

**Team Structure and Roles**

The development team for the **QNX and Python Implementation** project consists of the six members whose responsibilities will facilitate a structured functional workflow. Specific roles are assigned as follows:

1. **Project Manager: Muhammad Ali**

* Supervise the project with respect to its objectives.
* Manage communication with the sponsor, stakeholders, and team members.
* Facilitate meetings, track progress, and mitigate risks.
* Adhere to all deadlines and milestones of the project.

1. **Lead Developer: Nicholas Anderson**

* Oversees the implementation of Python integration in QNX environment.
* Gives technical guidance to team members.
* Code review and ensure that software quality standards are being met.
* Assist debugging and troubleshooting.

1. **Embedded Systems Engineer: Tabark Abaid**

* Ensures that QNX is compatible with CPI hardware.
* Deals with real-time processing, optimization of system performance, and designing mechanism for hardware-software interaction.

1. **Software Developer: Diego Ibarra**

* Writes and implements the modules required for interaction between Python and QNX.
* Testing and debugging software.
* Testing that Python scripts are working properly within the QNX system.

1. **Quality Assurance & Testing Manager: Saghar Abdi**

* Creating test cases and frameworks for testing.
* Conducting performance testing and stability of the system.
* Defect tracking and ensuring fixes are completed before delivery.

1. **Configuration & Documentation Manager: Khaled Elkhaled**

* Manage version control through GitHub in conjunction with Google Docs.
* Document pre-development, system design, and technical specification.
* Compliance to IEEE and ISO standard.

**Project Organization Rationale:**

The QNX and Python Implementation project structured project organization ensures that the project will be completed smoothly and in a timely manner upholding standards of quality and maintainability.

# **LIFECYCLE MODEL USED**

· For this project, our team will be using a combination of Waterfall and Agile Lifecycle Models to ensure better results. The waterfall Lifecycle Model will be used in the beginning stages to complete tasks such as planning, requirements gathering, and design. The Agile Lifecycle Model will then be used during the project's development phase. The flexibility of this Model will allow the team to build the system in smaller sessions, test it, and proceed to the next stages.

· We chose to use a combination of Waterfall-Agile models because it allows us to have a clear and structured start. The Waterfall Model allows us to establish objectives and design the system before starting the development phase. As we move on to development, the Agile approach provides us flexibility, allowing us to make changes if we run into unexpected errors when integrating QNX and Python. Additionally, Agile provides a possibility for our team to receive continuous feedback, which will help us stay aligned with the project requirements. Combining the two models allows us to plan efficiently from the beginning and remain flexible as the project evolves, which will help us minimize risks as we move forward.

# **RISK ANALYSIS**

# **Identified Risks**

| Risk | Description | Likelihood  (L/M/H) | Impact  (L/M/H) |
| --- | --- | --- | --- |
| Software Hardware Issues | Python may not function smoothly within QNX | M | H |
| Real-Time Performance Limitations | Python may introduce performance bottlenecks, impacting QNX's real-time capabilities. | M | H |
| Resource Availability | Lack of access to required CPI hardware for testing may delay development. | H | H |
| Scope Creep | Additional feature requests from stakeholders may extend project scope beyond initial estimates. | M | M |
| Hardware Malfunctions | CPI hardware may fail during development, causing delays. | L | H |

# **Risk Likelihood and Impact**

**High Likelihood, High Impact (HH)** – Needs an immediate plan to fix.

**Medium Likelihood, High Impact (MH)** – Take steps to prevent it from happening.

**High Likelihood, Medium Impact (HM)** – Keep a close watch on it.

**Low Likelihood, High Impact (LH)** – Have a backup plan in case it happens.

**Low Likelihood, Low Impact (LL)** – Keep an eye on it, but no action is needed right away.

# **Risk Mitigation Strategies**

| Risk | Strategy |
| --- | --- |
| Software Hardware Issues | Conduct early testing to detect issues before full implementation. |
| Real-Time Performance Limitations | Optimize Python scripts to minimize latency |
| Resource Availability | Secure access to CPI hardware early in the project |
| Scope Creep | Establish strict project scope and deliverables. Review and approve additional feature requests only if time and resources allow. |
| Hardware Malfunctions | Have backup hardware available. Maintain plan for switching to alternative platform |

# 

# **SOFTWARE AND HARDWARE RESOURCE REQUIREMENTS**

**Software Requirements**

| **Software** | **Purpose** | **Lab Availability** |
| --- | --- | --- |
| Python 3.11+ | Primary language for scripting and automation | Yes |
| QNX Software Development Platform | Primary operating system | Yes |
| Git/Github | Version control | Yes |
| QNX IDE | IDE for QNX app development | Yes |
| PyQNX Library | Python binding with QNX processes | Yes |
| QNX Qnet | Network file system for communication | Yes |
| VMware | Virtualization | Yes |

**Hardware Requirements**

| **Hardware** | **Purpose** | **Lab Availability** |
| --- | --- | --- |
| CPI Embedded Hardware | For QNX-Python integration devices | Yes |
| Ethernet Switch & Cables | Device testing network communication | Yes |
| Serial Debugger | Access QNX devices for debugging | Yes |
| Multimeter | Voltage and current in hardware | Yes |

**Availability in Lab**

For the time being all the tools required now and later are available in the lab.

# **DELIVERABLES AND SCHEDULE**

Project Management Plan -> February 7

Requirements Documentation -> February 21

Architecture Documentation -> March 21

Detailed Design Documentation -> April 4

Test Plan -> April 18

Final Presentation and Demo -> May 3

Final Report -> May 9

These deliverables follow a waterfall method but we will be allowed to make changes to the deliverables.

# **MONITORING, REPORTING, AND CONTROLLING MECHANISMS**

**Reporting Structure**

1. Status Report
   * Frequency: Weekly
   * Content: Progress summary, completed tasks, schedule updates
   * Rationale: Keeps sponsor and team-members informed of any potential issues or delays with deliverables
2. Quality Assurance Report
   * Frequency: At milestone completion
   * Content: Testing results, defect tracking, possible improvements
   * Rationale: Ensures deliverables meet expectations and align with requirements
3. Risk Report
   * Frequency: At milestone completion
   * Content: New risks, risk updates, mitigation strategies
   * Rationale: Ensures proactive risk management

**Monitoring Tools and Techniques**

1. Github
   * Version Control and Code Tracking
   * Bug Monitoring
2. Google Docs

**Control Mechanisms**

1. Schedule Control
   * Gantt Chart
   * Critical Path
   * Rationale: Ensures timely completion of tasks and helps prevent project delays
2. Quality Control
   * Defect Tracking
   * Testing processes
   * Rationale: Ensures deliverables meet predefined quality standards
3. Risk Control
   * Risk assessments
   * Risk meetings
   * Rationale: Identifies, analyzes and mitigates potential risks

# 

# 

# 

# 

# **PROFESSIONAL STANDARDS**

# 

# **Team Expectations**

# Maintain Professional Communication

# Use respectful and clear communication in meetings, emails, and reports.

# Document discussions and decisions in meeting minutes.

# Provide constructive feedback and accept criticism professionally.

Collaborate Effectively

* Attend all scheduled meetings unless prior notice is given.
* Actively participate in discussions etc
* Assist teammates when possible and share knowledge openly.

Commit to Responsibilities

* Complete assigned tasks on time and meet quality expectations.
* Communicate about any delays or challenges.

Adhere to Technical and Ethical Standards

* Follow industry best practices and IEEE/ISO standards (detailed in Section 13).

# 

# **Scholastic Honesty and Conduct**

To maintain academic integrity, all team members must:

* Submit original work
* Avoid plagiarism
* Do not engage in cheating, falsification of results, or misrepresentation of contributions.
* Report any unethical behavior

First occurrence → The issue will be discussed and documented in meeting minutes.

Second occurrence → The instructor will be notified, and a formal warning will be issued.

Third occurrence → The violating member may be removed from the team and receive a prorated grade based on participation.

# **Quality and Deadline Standards**

**Code Quality:**

* Code must be well-structured, documented, and tested before submission.
* Use version control (Github) to track changes and maintain clean commits.

**Documentation Standards:**

* Follow a consistent format for reports, including clear explanations and references.
* All technical documents must be peer-reviewed before final submission.

**Deadlines and Milestones:**

* Each milestone must be completed at least 24 hours before the deadline for review.
* Any delays must be communicated immediately to the team lead and instructor.

# **EVIDENCE THE DOCUMENT HAS BEEN PLACED UNDER CONFIGURATION MANAGEMENT**

# 

# We will be using Google Docs because it allows real-time collaboration, automatic version history tracking, and document review.

Each time a document is updated and reviewed, it is checked in with an incremented version number and a brief description of the update.

Before making modifications, the document version will be recorded. The previous version number will be maintained until the changes are reviewed and approved.

Each update must document what changes were made compared to the previous version. This way we will be tell the difference between two consecutive versions.   
Each document version undergoes the following steps: A team member checks out the document from the previous version and makes sure the version number is same as when it was checked in. Modifications are made, including additions, edits, or refinements. A summary of changes is documented, detailing what was altered from the previous version. Two reviewers provide feedback via Google Docs comments. Approval is recorded, and the document is checked back in with an incremented version number. Final changes are logged, along with timestamps, authorship, and review details.

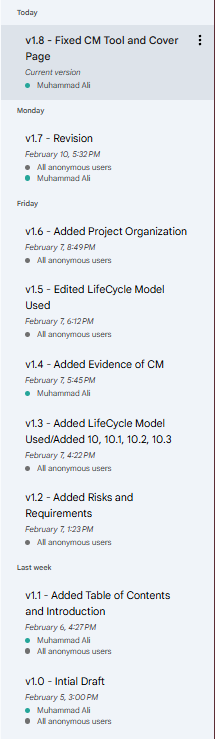
Each reviewer will leave a comment in Google Docs stating approval.

Once both reviewers approve, the change is marked as “Ship-It” and checked in.

Google Docs allows for changes that will be tracked using comments for easy reference.

Each saved version will be named with version number and description for clarity.

For proper tracking, we maintain a version history table that logs all document updates. This includes date, authors, changes made, and reviewer approvals. Each reviewer must leave a comment in Google Docs indicating their approval before the version is checked in as final.



| Version | Date | Author | Changes Made | Previous Versions | Reviewers | Ship-it |
| --- | --- | --- | --- | --- | --- | --- |
| v1.8 | 2/13/2025 | Muhammad Ali, Nicholas Anderson, Saghar Abdi | Fixed CM Tool and Cover Page | v1.7 | Khaled Elkhaled, Diego Ibarra | Approved |
| v1.7 | 2/10/2025 | Khaled Elkhaled, Tabark Abaid, Diego Ibarra | Revision | v1.6 | Nicholas Anderson, Muhammad Ali | Approved |
| v1.6 | 2/7/2025 | Khaled Elkhaled | Added Project Organization | v1.5 | Saghar Abdi, Diego Ibarra | Approved |
| v1.5 | 2/7/2025 | Saghar Abdi, Tabark Abaid | Edited LifeCycle Model Used | v1.4 | Nicholas Anderson, Diego Ibarra | Approved |
| v1.4 | 2/7/2025 | Muhammad Ali | Added Evidence of CM | v1.3 | Nicholas Anderson, Diego Ibarra | Approved |
| v1.3 | 2/6/2025 | Saghar Abdi, Nicholas Anderson | Added LifeCycle Model Used and Sections 10, 10.1, 10.2, 10.3 | v1.2 | Diego Ibarra, Khaled Elkhaled | Approved |
| v1.2 | 2/5/2025 | Diego Ibarra | Added Risks and Requirements | v1.1 | Nicholas Anderson, Saghar Abdi | Approved |
| v1.1 | 2/5/2025 | Muhammad Ali | Added Table of Contents and Introduction | v1.0 | Diego Ibarra, Khaled Elkhaled | Approved |
| v1.0 | 2/4/2025 | Muhammad Ali | Initial Draft | - | - | - |

ENGINEERING STANDARDS AND MULTIPLE CONSTRAINTS

o IEEE Std 1058-1998: Software Project Management Plans [[pdf](https://course.techconf.org/se4485/IEEE/IEEE-Std-1058-1998-Software-Project-Management-Plans.pdf)]

o PMBOK® Guide: Project Management Body of Knowledge [[pdf](https://course.techconf.org/se4485/IEEE/PMBOKR.pdf)]

o IEEE Std 12207: Software Life Cycle Processes [[pdf](https://course.techconf.org/se4485/IEEE/IEEE%2012207%20(2017)%20-%20Software%20Life%20Cycle%20Processes.pdf)]

o IEEE Std 15939: Measurement Process [[pdf](https://course.techconf.org/se4485/IEEE/IEEE%2015939%20(2017)%20-%20Measurement%20Process.pdf)]

o ISO/IEC/IEEE Std 29148-2018: Systems and Software Engineering

§ Life Cycle Processes

§ Requirements Engineering [[pdf](https://course.techconf.org/se4485/IEEE/ISO-IEC-IEEE-29148-2018.pdf)]

ADDITIONAL REFERENCES

o Larson, E. and Gray, C., 2014. *Project Management: The Managerial Process.* McGraw Hill

o Humphrey, W.S. and Thomas, W.R., 2010. *Reflections on Management: How to Manage Your Software Projects, Your Teams, Your Boss, and Yourself.* Pearson Education

**Appendix A.**

The following provides a professional standards guideline for the teams. This guideline may be tailored.

Guideline:

On the first occurrence of unacceptable behavior, determine the circumstances involved, resolve the problem, and document the event in the meeting minutes.

On a second occurrence, notify the instructor of the problem. A meeting will be set up to evaluate the situation and resolve the problem.

On a third occurrence, again notify the instructor of the problem. A meeting will be set up to evaluate the situation and resolve the problem. At this point, the team will have the *option* of removing the team member. If removed, then the team member receives a prorated grade based on the number of weeks they have participated in the group.

Examples of unacceptable behavior may include not delivering on time, delivering poor quality work, missing team meetings, being unprepared for team meetings, disrespectful or rude behavior, etc. Reasons such as “too busy” or “I forgot”, or “my dog ate my design model” are unacceptable.

Valid reasons that must be considered include those listed for obtaining an incomplete standing in a course (illness, death in the family, travel for business or academic reasons, etc.)