SN^{2}ARK CO.

Arithmetic Parser
Software Development Plan
Version 1.2

Revision History

Date	Version	Description	Author
13/09/23	1.0	Team members completed detailed descriptions for Part 1: The Project Plan sections 1 and 2.	Sophia Jacob, Reeny Hunag, Navya Nittala, Kusuma Murthy, Anna Lin, Nimra Syed
15/09/23	1.1	Team members completed detailed descriptions for Part 1: The Project Plan sections 3-5.	Sophia Jacob, Reeny Hunag, Navya Nittala, Kusuma Murthy, Anna Lin, Nimra Syed
22/09/23	1.2	Edit out blue text, delete brackets, proof read and finalize document for submission.	Sophia Jacob, Reeny Hunag, Navya Nittala, Kusuma Murthy, Anna Lin, Nimra Syed

Table of Contents

1. Introduction....4

1.1 Purpose....4

1.2 Scope....4

1.3 Definitions, Acronyms, and Abbreviations....4

1.4 References....4

1.5 Overview....5

2. Project Overview....5

2.1 Project Purpose, Scope, and Objectives....5

2.2 Assumptions and Constraints....5

2.3 Project Deliverables....5

2.4 Evolution of the Software Development Plan....5

3. Project Organization....5

3.1 Organizational Structure....5

3.2 External Interfaces (N/A)....6

3.3 Roles and Responsibilities....6

4. Management Process....6

4.1 Project Estimates (N/A)....6

4.2 Project Plan....6

4.3 Project Monitoring and Control....7

4.4 Requirements Management (N/A)....7

4.5 Quality Control....7

4.6 Reporting and Measurement (N/A)....7

4.7 Risk Management....8

4.8 Configuration Management....8

5. Annexes....8

Software Development Plan

1. Introduction

1.1 Purpose

The ***Software Development Plan (SDP)*** provides an overview of the software project at a high level. It specifies the major elements, goals, resources, and procedures that are going to be used to successfully complete the project. The plan also acts as a reference for defining the project's scope, methodology, quality assurance documentation, and overall vision. This helps to align stakeholders and keeps everyone on the same page.

The Software Development Plan is used by the following persons in our company:

- It is used by the **Project Manager (PM)** to define scope, establish timelines, allocate resources, identify hazards, create methods, enable communication, and manage modifications. The PM uses the SDP as a road map to steer the project and team.
- It is used by the **Quality Assurance Engineer (QA)** to comprehend the quality processes, measurements, tests, verification factors, and audits that must be carried out. The QA engineer assures that the SDP's quality criteria are met.
- Based on the techniques given, the **Configuration Management Engineer (CM)** uses it to set up version control, build processes, release administration, and change control systems. In addition, the CM guarantees the reliability of code and configurations of the system.
- The **Scrum Master (SM)** consults the SDP to learn about the agile processes, services, and duties that must be fulfilled while using an agile methodology. Based on the SDP, the scrum master organizes stand-ups, retrospectives, and reviews.
- The **Software Developer (SD)** uses the SDP to adhere to the stated standards for coding, operations, documentation requirements, and testing processes. Developers create code in accordance with standards.
- The **User Experience Designer (UX)** employs the SDP to guarantee that the UI and client experience adhere to the requirements and usability guidelines specified. The UX follows the prescribed styles and standards.

1.2 Scope

This ***Software Development Plan*** is in association with the overarching plan of the Arithmetic Parser project. It encapsulates the management process, objectives, development, role specification, the iterations within the project, and detailed accounts of each iteration described in the Iteration Plan.

The plans as defined in the document are based upon the project requirements as outlined in the *Requirements Document*, mentioned on Canvas.

1.3 Definitions, Acronyms, and Abbreviations

See the Project Glossary.

1.4 References

For the ***Software Development Plan***, the list of referenced artifacts includes:

- Iteration Plans
- Project Management Plan

- Requirements Document
- Design Document
- Test Cases
- Glossary

1.5 Overview

This ***Software Development Plan*** contains the following information:

Project Overview – provides an overview of the project's purpose, scope, and objectives. It also defines the deliverables that the project is expected to deliver.

Project Organization – describes the organizational structure of the project team.

Management Process – explains the estimated cost and schedule, defines the major phases and milestones for the project, and describes how the project will be monitored.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The purpose of this project is for SN^2ARK to develop a multifunctional arithmetic expression evaluator utilizing C++. The objective is to create a simulation of a calculator, through a program, that is able to handle expressions as input and calculate the result using mathematical operations like (PEMDAS). Furthermore, SN^2ARK will fully integrate the Software Development Process through timely deliverables within the semester. These deliverables include a detailed project plan, a requirements document, a design document that is in alignment with the requirements, a set of test cases, and a fully-developed product. As part of the scope of this project, each iteration should be timely and in-line with the functionality of the requirements.

2.2 Assumptions and Constraints

Equipment: Laptops and KU Lab Linux Machines

Software Tools: Google Docs, Visual Studio Code, Github, Git, UML Design Software

Schedule: Weekly Scrum Meetings based on availability (Wednesday at 4:30 PM and Friday at 5:00 PM Based on When2Meet)

Staff: Teaching Assistance and Professor Hossein Saiedian

2.3 Project Deliverables

Deliverables include software engineering artifacts, e.g., a project management plan, a requirements document, a design document, and test cases. These will be described in separate announcements, including a well-documented C++ program that can evaluate arithmetic expressions with the specified operators and features. Lastly, a user manual or README file demonstrating how to use the program, including examples. This is specified in section *4.2 Project Plan.*

2.4 Evolution of the Software Development Plan

The ***Software Development Plan*** will be revised prior to the start of each Iteration phase.

The iteration phases are as follows:

1. Integer Operands and the + Operators
2. Introduce the Minus Operator
3. Introduce Parentheses for Grouping
4. Introduce the * Operator
5. Introduce the / Operator
6. Introduce the % Operator
7. Introduce the ^ Operator

The criteria for the unscheduled revision and reissue of this plan is to only create improvements and modifications to the ***Software Development Plan*** if a deliverable affects the plan.

3. Project Organization

3.1 Organizational Structure

1. The team's organizational structure includes the **SD**, who will create the base code for the project. Other team members will collaboratively develop the Arithmetic Parser.
2. Then, the project will be reviewed and tested by the **CM**, who will test base cases and unique cases for the Arithmetic Parser. This ensures that the code is working sufficiently.
3. The **SM** will guarantee that the project is submitted in a timely manner, and initiate note-taking for the meetings, to know the improvements that must be made.
4. The **UX** will ensure that the code is readable, adequately commentented, and aesthetic on user-based scenarios.
5. The **QA** will ensure that the requirements from the initial RE gathering are met, and all objectives are aligned with the deliverables and artifacts. The project then goes to the **PA** who will review the final product and submit deliverables to the TA, who is one of the review authorities.
6. The project will be submitted to the Professor in the end.

3.2 External Interfaces - NA

3.3 Roles and Responsibilities

Person	**Unified Process for EDUcation Role**
Sophia Jacob	Project Administrator (PA)
Kusuma Murthy	Quality Assurance Engineer (QA)
Nimra Syed	Configuration Management Engineer (CM)
Navya Nittala	Scrum Master (SM)
Reeny Huang	Software Developer (SD)
Anna Lin	User Experience Designer/Tester (UX)

Note: All parts of the project will be a team effort; deliverables will not be on a single person's responsibility.

Team Availability: When2Meet

Project Administrator: Sophia Jacob

Contact: sophiajacob@ku.edu, 913-708-5764

Availability:

Wednesday and Friday: 2:00 PM or later

Tuesday and Thursday: 4:30 PM or later

Computer Science - Sophomore (2026)

Relevant Coursework: EECS 168, EECS 268, EECS 141, EECS 348, EECS 210

Programming Languages: Python, C++, HTML, CSS, Java (Basic)

Responsibility:

- Sets up and manages team meetings.
- Ensures that project deliverables are submitted in a timely manner and deadlines are met.
- Creates a planned agenda for meetings that adhere to the guidelines of the project.
- Collaborates with other team members for completing project artifacts/deliverables, such as the development of the Arithmetic Parser code.
- Streamlines communication between the TA, Professor, and team members.
- Main point of contact.

Quality Assurance Engineer: Kusuma Murthy

Contact: kusuma.s.murthy@ku.edu, 913-703-8378

Availability:

Wednesday and Friday: 2:00 PM or later

Computer Science - Sophomore (2026)

Relevant Coursework: EECS 168, EECS 268, EECS 141, EECS 348, EECS 210

Programming Languages: Python, Java, C++, HTML, CSS

Responsibility:

- Identifies and sets requirements according to the requirements document.
- Establishes project features and attributes.
- Assures system requirements are met.
- Works with the SM to ensure that the team is within the scope of the established requirements.
- Collaborate with other team members to help meet project artifact/deliverable deadlines.

Configuration Management Engineer: Nimra Syed

Contact: nimrasyed@ku.edu, 913-749-7148

Availability:

Wednesday and Friday: 2:00 PM or later

Computer Science - Sophomore (2026)

Relevant Coursework: EECS 168, EECS 268, EECS 141, EECS 348, EECS 210

Programming Languages: Python, Java, and C++

Responsibility:

- Tests base cases and unique cases of the project.
- Unit Testing.
- Compiles all project deliverables and reviews them.
- Reports technical issues and identifies bugs in code.
- Decides which software application/tools are necessary for the team's development process.
- Works with the QA to ensure software decisions align with project requirements.

Scrum Master: Navya Nittala

Contact: navya@ku.edu, 913-401-8360

Availability:

Wednesday and Friday: 2:00 PM or later

Computer Science - Sophomore (2026)

Relevant Coursework: EECS 168, EECS 268, EECS 141, EECS 348, EECS 210

Programming Languages: Python, Java, and C++

Responsibility:

- Takes and posts notes for each meeting.
- Organizes team member availability to ensure attendance at meetings.
- Manages the GitHub page.
- Identifies and reports possible improvements on the project.
- Assists the SD to produce and structure the program.
- Resolves any team conflicts.

Software Developer: Reeny Huang

Contact: reeny.huang@ku.edu, 913-563-8358

Availability:

Wednesday and Friday after 4:30

Majors: IC-Economics and Mathematics - Sophomore (2026)

Relevant Coursework: EECS 168, EECS 268, EECS 140, EECS 348, EECS 210

Programming Languages: Python, C++, HTML, CSS

Responsibility:

- Reviews requirements and structures the program.
- Produces clean, efficient code based on specifications.
- Integrates software components and third-party programs.
- Verifies and deploys programs and systems.
- Troubleshoots, debugs and upgrades existing software.

User Experience Designer: Anna Lin

Contact: anna.lin@ku.edu, 316-250-1568

Availability:

Wednesday and Friday: 3PM or later

Computer Science - Sophomore (2026)

Relevant Coursework: EECS 168, EECS 268, EECS 140, EECS 348, EECS 210

Programming Language: Python and C++

Responsibility:

- Guarantees that the program output is well-formatted and readable.
- Looks for and notes all base cases and unique cases to the project.
- Tests the user software and ensuring there is no vulnerabilities present in the software
- Works with the CM to create a solution for the vulnerabilities.
- Ensures overall project deliverables have organized and well-formatted content.

4. Management Process

4.1 Project Estimates - NA

4.2 Project Plan

The team will have weekly meetings, with two possible days that all team members are available.

Meeting Times: Wednesdays at 4:30 PM and Fridays at 5:00 PM.

Meeting logs and itinerary are on the GitHub page.

Project Due Dates and Deliverables:

Name of Project Part	Assigned	Due Date
Part 1: Project Management Plan	September 7th	September 24th
Part 2: Project Requirements	September 19th	October 21st
Part 3: Project Architecture and Design	October 24th	October 28th
Part 4: Project Implementation	October 31st	November 11th
Part 5: Project Test Cases	November 14th	November 25th
Part 6: Project User Manual	November 28th	December 5th

Resources include: Sample project plan (Canvas), provided project description (Canvas), lecture slides (Canvas), Professor, and TA.

4.2.1 Phase Plan - NA

4.2.2 Iteration Objectives

The following is a brief outline for each iterations of the project:

- Integer Operands and the + Operators
  - Create a functioning program that can parse through integer and addition operands.
- Introduce the Minus Operator
  - Add to the functioning program that can parse through minus operands.

- Introduce Parentheses for Grouping
    - Add to the functioning program that can parse through parentheses for grouping.
- Introduce the * Operator
    - Add to the functioning program that can parse through the multiplication operator.
- Introduce the / Operator
    - Add to the functioning program that can parse through the division operator.
- Introduce the % Operator
    - Add to the functioning program that can parse through the modulus operator.
- Introduce the ^ Operator
    - Add to the functioning program that can parse through the exponential operator.

Refer to ***Iteration Plan Documents*** for detail.

4.2.3 Releases

The following is the tentative schedule for each release:

- Beta release – November 1st
- Initial product release – November 11th
- Final product release – November 25th

4.2.4 Project Schedule - either end of semester or NA

4.2.5 Project Resourcing - NA

4.3 Project Monitoring and Control

The following is a checklist of items to consider:

- Quality Control: Describe the timing and methods to be used to control the quality of the project deliverables and how to take corrective action when required. Include techniques, metrics, criteria, and procedures used for evaluation— this will include walkthroughs, inspections, and reviews. Note that this is in addition to the Test Plan, which is not enclosed in the Software Development Plan.
- Risk Management: Describe the approach that will be used to identify, analyze, prioritize, monitor and mitigate risks.
- Configuration Management: Describe the process by which problems and changes are submitted, reviewed, and dispositioned. Describe how project or product artifacts are to be named, marked, and numbered, including system software, plans, models, components, test software, results and data, executables, and so on. Describe retention policies, and the back-up, disaster, and recovery plans.

4.4 Requirements Management - NA

4.5 Quality Control

Defects will be recorded and tracked as ***Change Requests***, and defect metrics will be gathered (see Reporting and Measurement below). This includes errors in code and unhandled exceptions.

All deliverables are required to go through the appropriate review process, as described in the Organizational Structure. The review is required to ensure that each deliverable is of acceptable quality, using guidelines and checklists.

Any defects found during review which are not corrected prior to releasing for integration must be captured as Change Requests so that they are not forgotten.

4.6 Reporting and Measurement - NA

4.7 Risk Management

Risks will be identified in the Inception Phase using the steps identified in the RUP for Small Projects activity “Identify and Assess Risks”. Project risk is evaluated at least once per iteration and documented in this table.

Since every team member is part of each deliverable process, there is a safety mechanism for this project if there is a key team member lost. Each individual still in the project will be delegated certain responsibilities associated with the lost key team member, to make up for any gaps in the project. The delegated responsibilities will be closely tied to the roles that the existing team member already has. For example, if the SM was lost in the project, then the PM can take on the managerial roles, while the SD could take on the roles of the coding aspect.

4.8 Configuration Management

Appropriate tools will be selected which provide a document of ***Change Requests*** and a controlled versioned repository of project artifacts, to track changes made on Git.

All source code, test scripts, and data files are included in baselines. Documentation related to the source code is also included in the baseline, such as design documentation. All customer deliverable artifacts are included in the final baseline of the iteration, including executables.

The ***Change Requests*** are reviewed and approved by one member of the project.

5. Annexes

The project will follow the UPEDU process.

Other applicable process plans are listed in the references section, including Programming Guidelines.