SFWRENG 4G06 - Development Process

Group: NextStep (Group 10)

Justin Rosner, rosnej1
Daniel Noorduyn, noorduyd
Mengxi Lei, leim5
Alexander Samaha, samahaa
Tishko Araz, arazt

Department of Computing and Software
McMaster University
October 22, 2021

Contents

List of Figures

List of Tables

1 Revisions

Revision Number	**Date**	**Reason for Change**
Revision 0	October 22, 2021	N/A

Table 1: Revision History

2 Overall Process Workflow

1. Decide on the component that needs to be implemented.
2. Create a plan for implementing the component by breaking it down into multiple modules.
3. Create a new branch based on the master branch.
4. Pull the new branch down locally and develop based on this branch.
5. Implement one of the modules.
6. Perform unit testing on the module.
7. Combine the changes into one commit and push these changes to the branch.
8. Repeat steps 5 to 7 until the component is fully implemented.
9. Perform testing on the component as a whole.
10. After approval from at least another team member, the branch will be merged to the master branch.

3 Details on Steps to be Taken

3.1 List of Development Tools

The following list outlines the development tools that will be used in the creation of NextStep:

1. Python (version 3.0.0 or newer): Will be used for the development of data sanitization, object detection algorithms, and sensor fusion algorithms. PEP 8 style formatting will be used to make sure that all code is consistent and easy to read.
2. OpenCV (version 4.5.0 or newer): Will be used to develop the object detection algorithm.
3. Maps Platform for Google Developers: Will be used to develop the navigation functionality.
4. Raspberry Pi: Will be the main processing unit that NextStep and its algorithms will run on. The Raspbian OS that the Raspberry Pi uses will be the development environment used.
5. Microsoft Visual Studio Code/Vim: Will be used as the IDE's for development.
6. Autodesk Inventor: Will be used for creation of wiring diagrams.
7. GitHub: Will be used for version control.
8. Jira: Will be used bug and feature requests in the development process.
9. PyTest (version 6.3.5 or newer): Will be used to write unit tests for NextStep.

3.2 List of Steps to be Taken

The following list outlines the steps that need to be completed for NextStep:

1. Create a wiring diagram to show the relative layout of how the sensors, speakers, and Raspberry Pi are to be connected using Autodesk Inventor.
2. Attach the electrical components mentioned above to the fashion garment of choice.
3. Develop data collection and sanitization algorithms using Python and the sensors mentioned above.
4. Develop object detection algorithms using Python and OpenCV.

5. Develop sensor fusion algorithms to combine the data from the different sensors using Python.
6. Develop the navigation functionality of NextStep using the Maps Platform for Google Developers and Python.
7. Write the text to speech functionality using Python and the speaker attached to the Raspberry Pi.
8. Write unit tests to ensure functional correctness using PyTest and Python.

4 Team Roles and Responsibilities

4.1 Justin Rosner - Project Manager

- Responsible for the development of object detection algorithms.
- Assisting in the design of how the electrical components are to be connected, including but not limited to the creation of wiring diagrams and the physical implementation.
- Assisting in the decision process of which of electrical components to utilize. These include but are not limited to a camera, sensors, Raspberry Pi and speakers.

4.2 Daniel Noorduyn - Scrum Master

- Responsible for the development of the text to speech functionality.
- Assisting in the development of the navigation functionality.
- Assisting in the design of how the electrical components are to be connected, including but not limited to the creation of wiring diagrams and the physical implementation.

4.3 Mengxi Lei - Scrum Member

- Responsible for the development of sensor fusion algorithms.
- Responsible for the development of data collection and sanitizing algorithms.
- Assisting in the development of text to speech functionality.

4.4 Alexander Samaha - Scrum Member

- Responsible for the development of sensor fusion algorithms.
- Responsible for the development of data collection and sanitizing algorithms.
- Responsible for error handling from all physical components.

4.5 Tishko Araz - Scrum Member

- Responsible for the development of object detection algorithms.
- Assisting in the development of the navigation functionality.
- Assisting in the decision process of which of electrical components to utilize. These include but are not limited to a camera, sensors, Raspberry Pi and speakers.

4.6 The Team

- Responsible for testing all software components utilizing PyTest.
- Ensuring quality of code through the use of various coding standards.

5 Version Control

For Version Control, GitHub will be used to store the files in a coherent server and git or GitHub desktop will be used for local files and local branches. These will be utilized in conjunction with one another to track the versions and fixes for the software components, hardware components, and documentation artifacts.

To use git you must first **git clone [repository link]** if it is your first time downloading the repository, else **git pull** the repository.

Secondly if you are making changes for a fix, you must create a new branch with **git checkout [branch name]** and then begin working.

If you are adding files or deleting files that have no conflicts with any other files you can **git add [files]** then **git commit -m "detailed message"**. Note that team members are expected to include Jira ticket ID's in their commit messages for traceability purposes.

Note team members must tag all commits for transparency and tracking purposes with **git tag [commit ID]**.

Once approved to merge a branch team members can merge the main branch with **git merge [branch name]** and commit changes (It is key to comment and tag all commits).

Finally when a team member wants to add their staging and local changes they committed on the master branch they can **git push** or **git push [branch name] master** (for local merges) their changes.

We will do all changes on local machines then push the changes for fixes to the remote repository.

6 Handling Changes to Development Artifacts

6.1 Using Jira

The NextStep development team will be using Jira to track changes related to development artifacts. Using Jira will allow us to organize our major features and split them into smaller issues or stories that can then be split among the development team. These issues can then be prioritized into sprints, which will split up work based on their importance to the project as a whole. This will ensure that everyone will know what is being worked on throughout the project and progress can be easily tracked and visually displayed through a dashboard.

Changes will be classified into the following Jira items;

Epic A major feature which can take place over many weeks and can be split into smaller issues and stories.

Story An issue assigned to a developer that can be split into smaller tasks, these are generally attached to a requirement.

Task The smallest amount of work that can be split up and assigned to a developer; generally a *story* is made up of tasks.

Bug An issue assigned to a developer documenting a problem with an existing system that may be of high or low priority and will get scheduled accordingly.

Before every sprint, issues will be prioritized into the sprint from the product backlog page and will be assigned to developers throughout the sprint. These items can be viewed in the sprint dashboard for easy tracking by developers.

6.2 Jira Item Workflow

The workflow for items in the project are as follows:

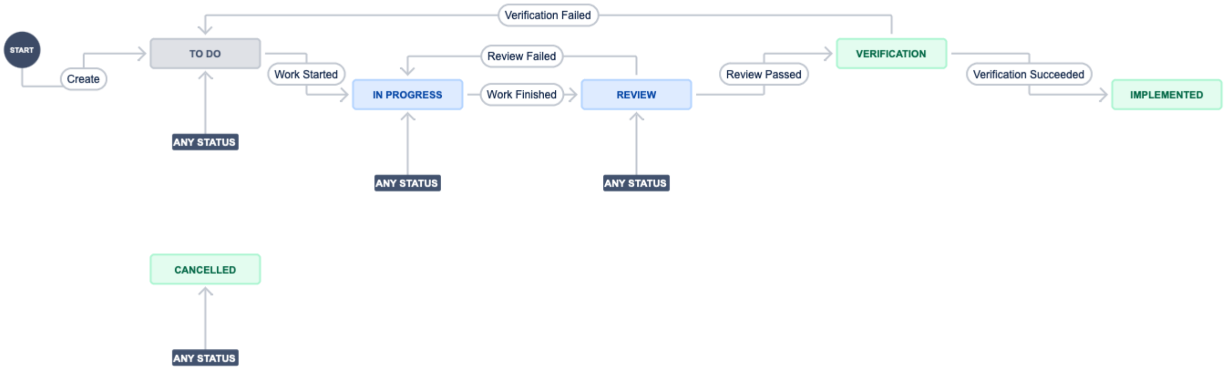

Figure 1: Item workflow for Task and Bug items.

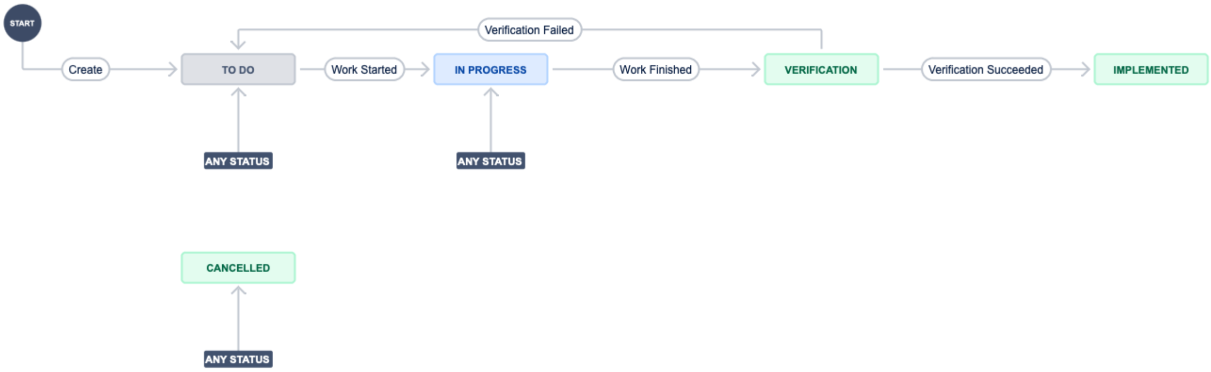

Figure 2: Item workflow for Story and Epic items.

After each Task or Bug item is finished development work, there will be a Review stage where the code developer will organize a code review and assign a different developer to carry it out. If a code review is failed, the code developer will revert the item back to In Progress and work to implement the feedback received during the code review. After a code review is complete and passed, the item will enter a verification stage where the feature will be verified against the requirements of the project through testing. If verification is failed, the item will either be placed in To Do state, where the item deliverables may need to be re-conceived to better fit the requirement the issue is meant to fulfill, or sent back to In Progress to address issues that failed testing. When verification is passed, the work is finished and the item is complete.

Each Epic and Story will follow a similar workflow to a Task or a bug, however, in this case we do not have a review stage since Epic and Story items should not have any code changes attached to them. Code changes related to an Epic or Story item should have been split into smaller Task issues which would have already passed a code review stage.

Each Jira item has a canceled state, this will be used to document features that are no longer going to be present in the final product due to changes in requirements or pruning of stretch goals.