AI-Assisted Agile Project Management

Final Project Plan

Phong Bach

Emily Cawlfield

Nain Galvan

Aric Monary

Submitted for:

CptS 484

Phase 2

12/11/20

1. Introduction

1.1 Project overview

This project aims to create an application for the Jira platform, in order to assist Agile teams in user story streamlining. The project consists of four main components: epic decomposition, story optimization, task generation, and dependency visualization
Overall, these will work in combination to help teams break epics into progressively smaller user stories and tasks, down to a level where they are ready to be put into sprints.

Individually the three main decomposition components will work in a linear flow but will be usable individually. The user can start with an epic and convert it into stories followed by tasks. On the other hand a user could solely decompose an epic into stories or convert manually entered stories into tasks. Dependency visualization will be derived solely using information that is not dependent on the decomposition components. The dependency visualization can be used

Project Requirements Link: AI4Agile: Developing AI-enabled tool support for user story refinement in JIRA

1.2 Project deliverables

The documentation deliverables for this project are:

1. Project Phase I: Preliminary Project Plan
2. Project Phase I: Final Submission/Presentation
3. Technical Specifications and Requirements document
4. Solution Approach document
5. Alpha Prototype Results report
6. Project Phase II: Final Submission
7. Summary Report Submission

The programmatic documentation deliverables are:

1. Initial AI Model for Epic Decomposition, Story Optimization, and Task Generation
2. UI Components
    a. Main GUI
    b. Suggestions
    c. Graph for dependency visualization
3. Refined AI Model for Epic Decomposition, Story Optimization, and Task Generation

1.3 Evolution of this document

This final project documentation covers our revised expectations for this project and its layout. It will be changed to reflect later schedule changes, which are likely to occur as the project progresses. During the actual development time, changes may be required to approaches given technical viability. The finished version of this document will be uploaded to the team GitHub repository.

1.4 References

Hoa Khanh Dam, Truyen Tran, John Grundy, Aditya Ghose and Yasutaka Kamei, *Towards effective AI-powered agile project management*, Proceedings of the 41st International Conference on Software Engineering (ICSE 2019), New Ideas and Emerging Results.

M. Choetkiertikul, H. K. Dam, T. Tran, T. T. M. Pham, A. Ghose, and T. Menzies, *A deep learning model for estimating story points*, IEEE Trans. Softw. Eng., vol. PP, no. 99, p. 1, 2018.

M. Choetkiertikul, H. K. Dam, T. Tran, A. Ghose, and J. Grundy, *Predicting Delivery Capability in Iterative Software Development*, IEEE Trans. Softw. Eng., vol. 14, no. 8, pp. 1–1, 2017.

M. Choetkiertikul, H. K. Dam, T. Tran, and A. Ghose, *Predicting the delay of issues with due dates in software projects*, Empir. Softw. Eng., vol. 22, no. 3, pp. 1223–1263, 2017.

M. Choetkiertikul, H. K. Dam, T. Tran, and A. Ghose, *Characterization and prediction of issue-related risks in software projects*, Proceedings of the 12th Working Conference on Mining Software Repositories (MSR), 2015, pp. 280–291.

M. Choetkiertikul, H. K. Dam, T. Tran, and A. Ghose, *Predicting delays in software projects using networked classification*, Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2015, pp. 353–364.

1.5 Definitions, acronyms, and abbreviations

- AI: Artificial Intelligence
- Agile: A software process in which a project is broken up into bite-sized portions and tasks are grouped together to be completed in relatively short time periods.
- Backlog: A list of tasks to be completed over an Agile project’s span.
- Deep learning: An AI technique that mimics the workings of the human brain in the processing of information.
- Epic: A topic that encompasses a series of user stories that fulfill requirements.
- Jira: A product made by Atlassian for issue and bug tracking in Agile development.
- NLP: Natural Language Processing; a field of AI in which human interactions with computers are dealt with via natural language (any human language that was not artificially created).
- Sprint: A short period in which a team completes a set amount of tasks.
- Story: An end goal from the perspective of an end-user, written in non-technical language.
- Task: The smallest unit of work that a team will undertake, from a developer’s point of view.

2. Project organization

2.1 Process model

We will be using the agile process model to develop this project, with short sprints to complete the various coding phases. Documentation will be produced in accordance with the formatting required for associated coursework.

2.2 Organizational structure

The members involved in developing this project are: Phong Bach, Emily Cawlfield, Nain Galvan, Aric Monary, and Isaac Schultz. Our Communication Liaison will be Aric Monary. All members will be Developers, but Documentation Authors will consist of only Phong Bach, Emily Cawlfield, Nain Galvan, and Aric Monary.

2.3 Organizational boundaries and interfaces

Communication interfaces between mentors and sponsors will be by the Communication Liaison. Communications will be conducted through Microsoft Teams and email, with all members included for transparency. Intercommunication between team members will be conducted via Microsoft Teams (textual and resource communication) and Discord (voice and real time collaboration). Google Drive and GitHub will serve as the primary sources of file sharing.

2.4 Project responsibilities

All the team members will be involved in development at all phases of the project life cycle. Documentation associated with coursework will be completed by Phong Bach, Emily Cawlfield, Nain Galvan, and Aric Monary. Isaac Schultz will be serving solely in an advisor role, on an as-needed basis.

The project will be broken into four primary epics derived from the key features of the application. Each of those epics is assigned to one of the primary developers to take the lead on.

Project Requirement	**Owner**
Decomposing epics to user stories	Aric Monary
Breaking down user stories down to smaller user stories	Phong Bach
Creating tasks from user stories	Nain Galvan
Visualization of epics, user stories, tasks, and their relationships.	Emily Cawlfield

In addition, the UI was separated into three sub-components with the responsibilities distributed due to the technical limitations to the AI development.

UI Delegation	**Owner**
Main GUI	Nain Galvan
Suggestions	Aric Monary
Graph	Emily Cawlfield/Phong Bach

3. Managerial process

3.1 Management objectives and priorities

The management philosophy for this project is that it should be broken into tasks with clear, achievable deliverables. Tasks will be clearly assigned and posted with various communication forms used by the team, including Microsoft Teams, Discord, and email. The accountability for these tasks will be maintained with a mutual understanding that the project's underlying success is dependent on the commitment of its members and completion of tasks.

Progress towards the end goal will be reevaluated constantly, due to the uncertainty pertaining to the final product. The skill sets, experience, and knowledge of the team is lacking to provide reasonable long term deadlines and deliverables. As the project progresses through the research phase, long term deliverables can be drafted.

Due to circumstances pertaining to Covid-19, an emphasis on communication in regards to availability will be crucial and achieved through various check-ins at meetings scheduled multiple times per week.

Communication between sponsors and mentors will be conducted by the communication liaison. Such communications will be forwarded with needed information to its relevant team members.

3.2 Assumptions, dependencies, and constraints

The functionality of our application will depend upon the version of Jira Software supported by Atlassian. The largest dependencies of the application are the Atlassian Connect Express (ACE) framework and the Atlassian User Interface (AUI) library.

3.3 Risk management

No.	Risk	Type	Likelihood	Description
1	Inappropriate versions of the tools and components.	Technical	Likely – High potential impact	The versions of tools and components used to develop the system do not have compatibility with Atlassian’s frameworks and libraries.
2	Failure to meet deadlines for deliverable.	Managerial	Unlikely – High potential impact	There is a failure to come up with a deliverable on schedule.
3	Unavailability of the resources	Managerial	Unlikely – High potential impact	The resources (including the team members) required to complete the project on time are not available or are scarce.
4	Requirements change	Technical	Unlikely – High potential impact	The requirements for the project are unlikely to be subject to change unless by the suggestions of a mentor.
5	Accidental loss of valuable information	Technical	Unlikely – High potential impact	A piece of project work is lost due to a system crash, damage to storage devices, incorrect git usage, etc.
6	Lack of team member’s commitment	Managerial	Likely – High potential impact	A team member does not perform the desired work on schedule and shows irresponsible behavior.

Table 1: Potential risks and their descriptions.

3.4 Monitoring and controlling mechanisms

No.	Risk	Monitoring and Controlling
1	Inappropriate version of the tools and components.	● Clearly understand the requirement specifications. ● Verify the version of tools and components used at developer's/team member's site.
2	Failure to meet deadlines for deliverable.	● Set up milestones before the scheduled date for each deliverable. ● If 2 or more milestones are not met, then seek the help of the faculty mentor.
3	Unavailability of the resources	● Check availability during the twice a week meetings. ● Reassign resources to fill gaps. ● Negotiate and reevaluate deadlines.
4	Requirements change	● Assess the amount of work to be changed and negotiate. ● Assign more resources to meet additional requirements.
5	Accidental loss of valuable information	● Maintain back-ups for every piece of project work. ● Keep storage devices in a safe place.
6	Lack of team member's commitment	● Set and strictly follow the Professional Standards.

Table 2: Mechanisms for controlling and monitoring of the risks predicted in Table 1.

4. Technical process

4.1 Methods, tools, and techniques

Python is the programming language of choice in developing this Jira application, since there are many AI libraries. This is where the AI will be used to provide an automated support for all three levels of epic breakdown.

The UI will rely on the ACE framework provided by atlassian which is Javascript and React based.

For this project, the team is using Microsoft Teams for general communications. GitHub is being used to keep all team information together, and Google Drive to collaborate on documentation.

4.2 Software documentation

This project will mainly focus on using the WRS template along with Technical Specifications and Requirements for documentation. At an increased level of technical detail, Component Vision Documents will be created to describe more specifically the implementation of each piece.

5. Work elements and schedule

This project is scheduled to be completed by January 10th, 2021 for the final submission to the competition. For academic purposes the schedule concludes on December 11th, 2021 (the Friday before Finals Week). The roadmap of the project is as follows:

- Planning/Proof of Concept (3 weeks)
    - Component Research (8/31 - 9/7)
    - Prototype Development (9/7 - 9/20)
    - Prototype Demo (9/20)
    - Component Document Draft (8/31 - 9/20)
- Development (9 weeks)
    - Infrastructure (2 weeks)
        - Component Vision Document and IO (9/18 - 9/25)
        - Database Creation and Schema (9/25 - 9/29)
        - Testing Text Preprocessing (9/25 - 10/2)
        - Compile Training/Test Data (9/25 - 10/2)
    - AI
        - AI Model (10/02 - 10/16)
        - AI Validation Testing Round 1 (10/16 - 10/23)
        - Component Vision Revision (10/16 - 10/23)
        - Requirement Validation (10/16 - 10/23)
        - AI Refinement (10/23 - 11/06)
        - AI Validation Testing Round 2 (11/06 - 11/13)
        - Final AI Refinement/Testing (11/13 - 11/20)
    - Visualization
        - UI/UX Setup (9/18-9/25)
        - Validation Component (9/25-10/02)
        - UI Validation Walkthrough (10/06)
        - UI Refinement (10/06-10/13)
        - Issue Relationship Clustering (10/09-10/30)
        - Issue Clustering Demo (11/03)
        - UI/UX Validation Round 2 (11/03)
        - Final UI/UX Refinement and Testing (12/04 - 12/08)
- Component Testing (1 week)
    - Unit Testing (11/20 - 11/27)
    - Requirement Validation (11/20 - 11/27)
- Deployment, Integration, and Systems Testing (1 week)
    - Test AI flow (11/27 - 12/04)
    - Test AI output into Jira (11/27 - 12/04)
    - AI Validity (11/27 - 12/04)
    - Accessibility (11/27 - 12/04)
    - UI/UX Refinement (11/27 - 12/04)
- Alpha Prototype Presentation (12/11)