Process Specification for AI4Agile

Team Katara
Prepared by Nain Galvan
Submitted 12/13/20

The first iteration was working on the code for three different parts. For the epic decomposition, the main task for this piece is to get a text block from the user and decompose it into user stories dynamically based on similar clusters using clustering methods provided by the python library sklearn. For the user stories optimization this is making the user-stories more detailed and closer to actionable sentences. Lastly, for task generation after getting the user stories deliverables need to be generated for the developers which come in forms of tasks using the python library nltk. These tasks should be able to be completed by one developer in no longer than a day.

Later into development of code a couple of problems arose in each different part. For epic decomposition, the clustering was unable to happen dynamically without changing certain parameters. For user story optimization determining how to optimize them and check if they are already optimized was a challenge. For task generation we want to make sure that the sentence is a simple sentence therefore the problem was determining the structure of a simple sentence and classifying them based on their type. If the sentence is any other type then it would need to be separated or broken down until they are considered a simple sentence. Based on these problems, refinement in the code is needed to be completed.

Further into the development process, we had a change in priorities with the Graphic User Interface (GUI) taking the main priority over the code, since the main presentation would focus on showing the GUI rather than the code. We broke the GUI up into separate tasks for each team member, where Nain was in charge of the Main GUI. Aric was in charge of suggestions, which gave the user an option to keep what we provided for the epic decomposition, user story optimization, and tasks generated. Emily was in charge of the visual graph, which shows the relationship between each process and gives the user the options to select different views. Lastly, Phong was in charge of the word cluster.

Initially, it was thought that each code part would need to be developed in parallel, however, it was later determined that they can be worked on disjointly. Although the code parts somewhat rely on inputs and outputs from each other, it is not enough to necessitate parallel development. This was determined based on the different options the user had for entering the system. One such option is to skip the epic decomposition, write their own stories, then use the plugin only to generate tasks. Additionally, a user could write their own epic and simply generate tasks. Thus, it is because of the potential disjoint nature of user interactions that the components and their code can be developed and function disjointly.

To determine how the plugin will be visually implemented in Jira a storyboard was generated. The storyboard is maintained within a separate document found here: StoryBoard

To understand better on how the code and the Jira plugin will interact and what actions will be performed on the text being used for the code we used a block diagram.

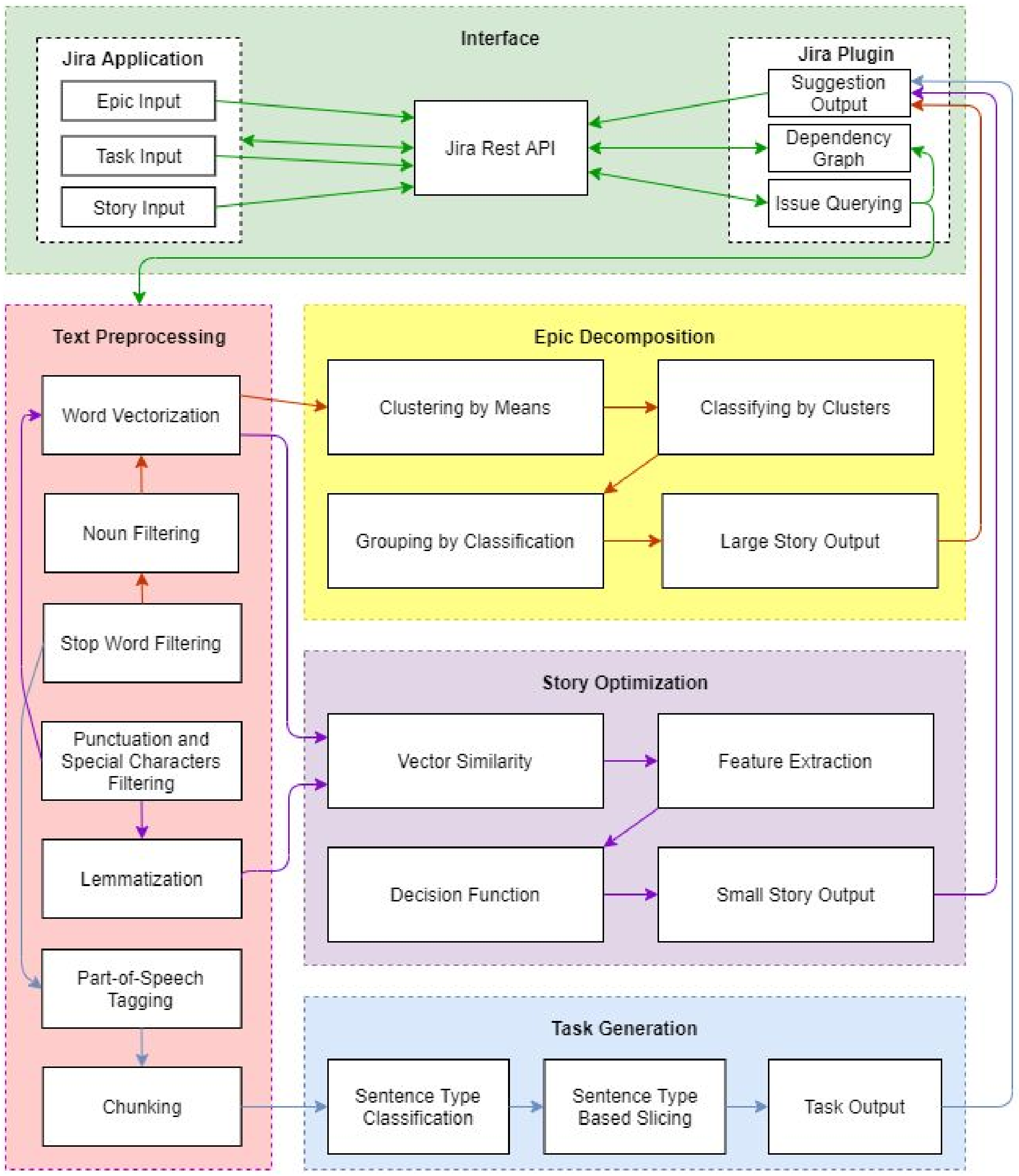

Figure 1: Block diagram, wherein arrow colors correspond to the component they belong to. Inputs to the text preprocessing portion are sent to their respective starting point task depending on where the original request originated from, i.e. which component process is being called.

The data flow diagram shows how the flow in Jira works. At first glance the ai components seem to be done in a linear manner but in reality they could be done independently. The user can put input in any stage for example they could skip epic decomposition and start at story optimization then task generation. This gives the user many options in where in the ai process they want to start.

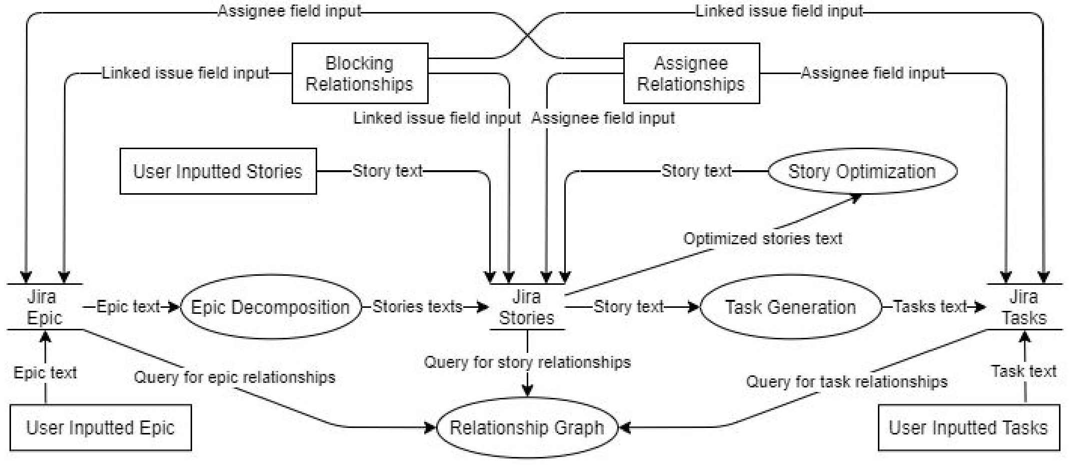

Figure 2: Data flow diagram. Users can input epics, stories, and tasks, and for the purposes of this plugin they may use any of the processes (shown in ovals) as starting points for interaction.

The three main parts for the IDEF0 are Requirement Elicitation, Design GUI, and Design AI. Each of these parts can be expanded upon, as shown in figures 3 through 6. Requirement Elicitation consisted of determining the GUI and AI requirements necessary to fulfill the SCORE 2021 project prompt. Next, specific types of AI needed to be chosen, depending on which would best meet the requirement, along with determining what GUI implementations were needed. The Design GUI activity consisted of sub-activities related to the design of the GUI, such as verification, development, and testing processes.

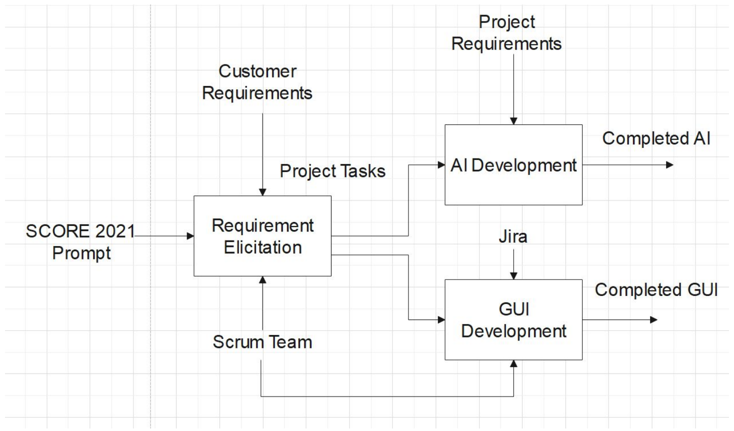

Figure 3: IDEF0. The three main processes performed for the project.

Requirement Elicitation was composed of three processes. In the beginning contact with the research sponsor was done to clarify score 2021 prompts requirements. Next, multiple team meetings were done weekly. The team meetings feed on each other to further refine the requirements. The team meeting feeds into the meeting with the mentor and the feedback from the mentor feeds into the next team meeting.

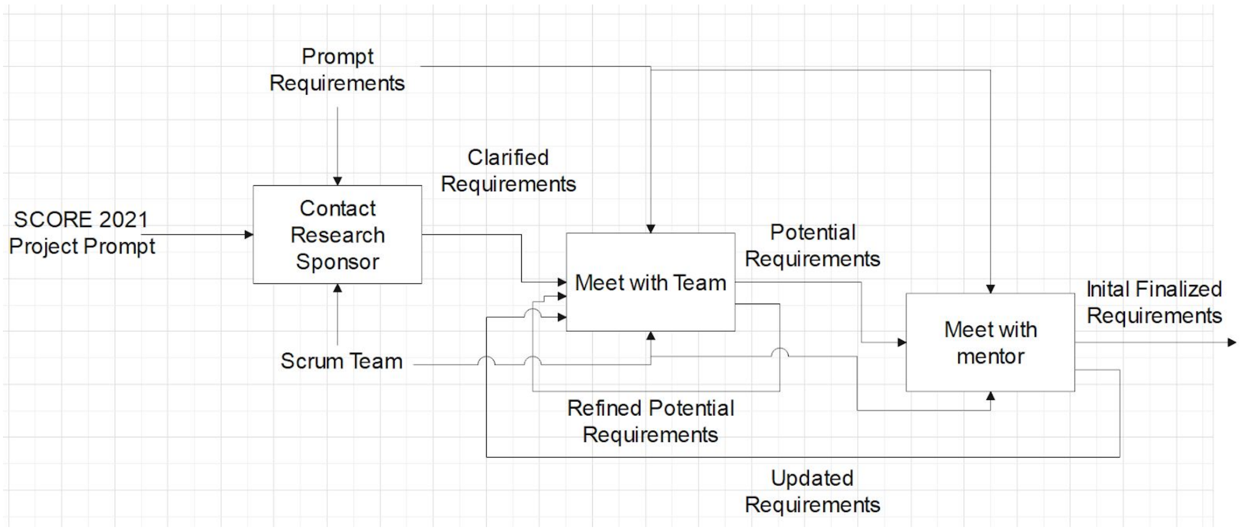

Figure 4: IDEF0. Detailed process of Requirement elicitation process.

The design of the GUI was based on the previously gathered requirements. After determining the design, the next step was to verify that design to ensure it would meet the stakeholders' needs. After verification, the GUI went through development, and finally testing.

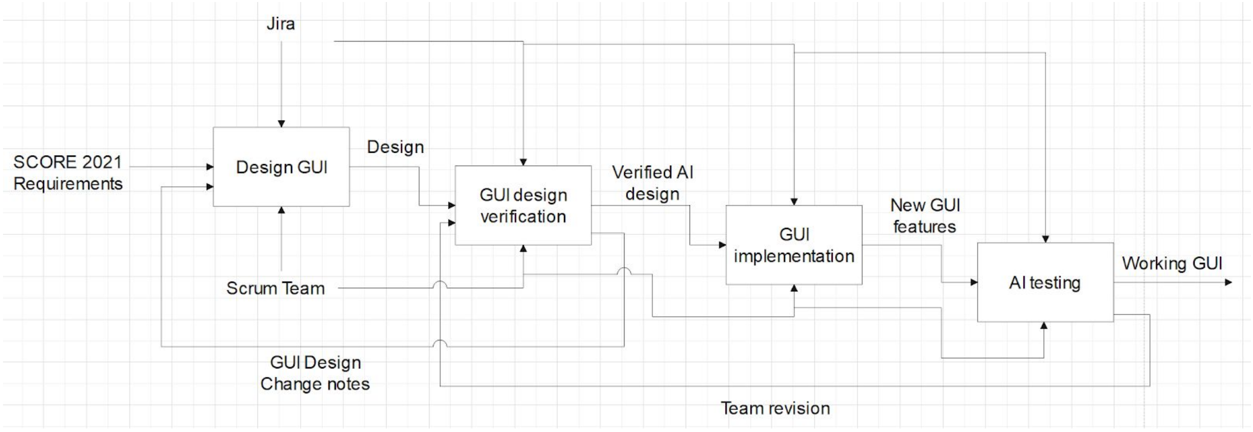

Figure 5: IDEF0. Detailed process of Designing the GUI.

The Design AI activity consists of all sub-activities related to the AI component development. The designs for each of the AI components--epic decomposition, story optimization, and task generation--were based on the AI formats. Once the designs were made they needed to get verified. After the designs were verified, development could begin. Lastly, the AI were tested to make sure they yielded the correct results.

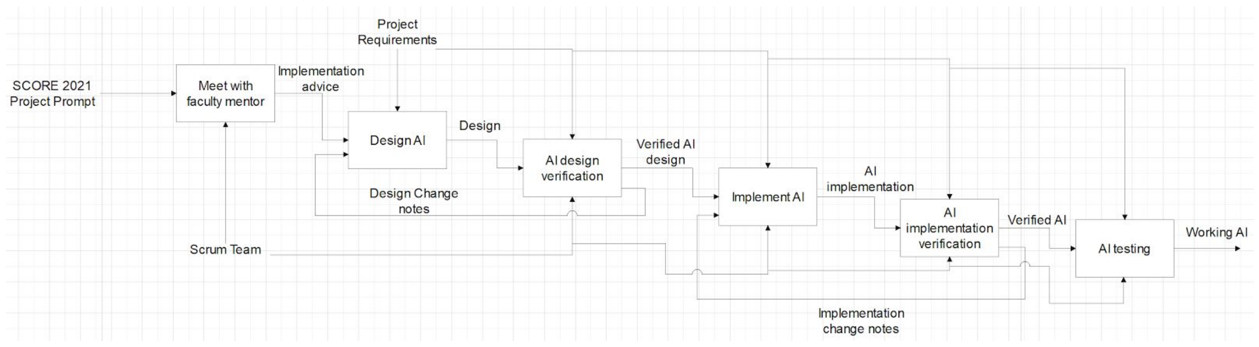

Figure 6: IDEF0. Detailed process of Designing the AI