Rampage Organizational Document

Sprint 4
12/1/2023

Name	**Email Address**
Andrea Florence	andrea.florence180@topper.wku.edu
Tameka Ferguson	tameka.ferguson183@topper.wku.edu
Shikha Sawant	shikha.sawant059@topper.wku.edu
Aaron Beasley	aaron.beasley908@topper.wku.edu

CS 360
Fall 2023
Dr. Michael Galloway
Project Organization Documentation

Contents

List of Figures

1 Project Team's Organizational Approach

Sprint 1: When starting the project, the first thing we did was get all our resources needed for the project like getting an understanding which game engine to use when recreating the Rampage game. Most of our meetings were held during class time and sometimes through Discord. Even though most of our meeting times conflicted with other times like work, activities, etc., we are still working on trying to meet whenever possible. At the moment, we have mostly decided on meeting through Discord and at some point, we will also try to make a video to show that we are doing our part. Overall, we are still looking at ways to stay in full contact with each other.

We meet kind of periodically since we all have conflicting schedules with each other. So, we compromised that instead of just finding the right time, we could videotape ourselves doing each aspect of the project, so it doesn't look like we did it on a whim. Really, there is no straightforward way to make meetings easy for us without contradicting each other's times. But we do discuss what we are doing so that everyone is aware that we are still working on the project. Overall, like I said above, we are still in the process of figuring stuff out, but the actual project is running smoothly.

We didn't really decide on a leader for the project, but I think that Andrea was the main contributor to the entire project as she was the main help for everyone involved.

Sprint 2: Our team has developed an organized strategy on how we will communicate to each other and effectively complete this project. From the beginning we had realized that our schedules would clash from various things (such as work and other classes) and we wouldn't be able to meet up physically. We had researched what types of communication platforms would best aid us for this small setback, and we had ultimately decided that Discord would be the best communication platform since we can not only text and communicate that way within our communication channels, but we can also easily distribute documents and files within our documents channel. It would take pressure off of us having to squeeze in time and accommodate to the other members' schedules since we can work on aspects of the project at our own pace. It has made such a great impact on the production of our recreation of Rampage. We have our weekly meetings as well with Dr. Galloway to discuss our progress. This is essentially the only time we are all free to discuss our project, and it helps talking to Dr. Galloway about our progress because we can not only see where we are excelling at but where our faults are at as well. The management has been taken upon various members of the group. Initially it was mainly Andrea who has been managing the project by writing out the tasks and letting us know which tasks still need to be completed and asking who could complete them. During this sprint, Tameka has also been managing alongside Andrea with organizing the tasks as well as delegating them to us evenly.

Sprint 3: We have many tasks to ensure that we can complete our goal of recreating one level of Rampage successfully. Our main method of communication is through Discord. In the very first sprint we had acknowledged our future issues because many of our team members work and have clashing class schedules. Discord is our preferred method of communication since it is internet based and we can communicate on it at any location at any time. Additionally, having a platform we can communicate in and send documents in is extremely useful for us, so we all prefer to use Discord. This platform makes communication much more efficient without having to alter our personal schedules too much to meet up. We also take the group discussions conducted in class to communicate in person. Thirty minutes before our weekly client meeting, we as a team meet up and discuss our progress along with the project. This is essentially the only time we are all free to discuss our project, and it helps talking to Dr. Galloway about our progress because we can not only see where we are excelling at but where our faults are at as well. The task management and task organization have been delegated to various members. We had taken the aspects documented in the previous sprint and converted them into tasks and assigned these tasks to each member of the group. We had approached this sprint in a "divide and conquer" method, dividing the tasks evenly between us. We typically document all the tasks and then color code the tasks to organize which member is assigned what task.

Sprint 4: For this sprint, the team developed a strategy on how tasks would be handled and how communication would be done. We also started with coding into the beginning of this sprint. We successfully completed making a level of Rampage though it was not the replicated game. We completed all of our testing and approved of the

results that we got from each test.
As said from the other three sprints, we communicated mainly through discord. We utilised discord for our task distribution, progress updates, help with tasks and documentation reviews. It was hard for us to have in person meetings as a whole group because of our conflicting schedules. We were able to meet twice for testing during this sprint and one being during our client meeting on Wednesday. As a team, we all spend roughly 20 hours a week working on the project including testing with it being a total of 320 hours total.
We experienced a lot of difficulties overall. One being navigating Unity up until the final second. There were a lot of struggles with making things function as they should because for some reason Unity hated a lot of the 2D aspects of our game. The approach for this was to simplify what we were doing in terms of coding and UI and hope that Unity accepted the simplification. Another difficulty was the communication and scheduling meetings, but we resolved that discord was the best way for us to keep in contact with each other and share needed information. Tameka and Andrea both managed the tasks delegation so that everyone knew what they needed to and if they needed help understanding their tasks, they both assisted with that.

2 Schedule Organization

2.1 Gantt Chart v1:

The focus of Sprint 1 is to get started on our first technical and organizational documents as well as evaluations and presentations. Sprint 1's main focus falls under feasibility study and planning. As a team, we have analyzed the relevant factors of the project, such as the project scope, technical requirements, risk analysis, software process model, software product development, hardware/software requirements, schedule and timeline, costs/time costs, project visibility, deliverables, and team and client communications. With these factors we can get a better understanding of what the overall project requires and start planning ahead on what and how we are going to execute the requirements.

The Gantt chart was created by Tameka in Excel. It is updated a few times a week, and lists both group tasks and individual tasks. It is divided up by the four sprints, and color coded for each sprint as well. The four tasks we had listed for sprint 1 were the technical document, the organizational document, presentation, and evaluation. Each of those tasks were for everyone in the group, minus the creation of the slideshow for our presentation which Aaron did not assist with. Shikha created the slide layout and found the game style template, and Tameka and Andrea helped her add content to the slides. As for the documentation and evaluation – each person was assigned sections of documentation to do, and each person had to complete an individual evaluation of the team. Moving forward the entire group will be able to access and edit the Gantt chart as needed throughout the project. There were little to no individual tasks in this sprint of the project, whereas there will be many more individual tasks in the sprints to follow.

The Gantt Chart can be found in our Rampage zip file.

2.2 Gantt Chart v2:

For Sprint 2, our main focus was to create the UML diagrams for our project. This includes the three object-oriented design patterns, such as the prototype, observer and façade design pattern diagrams, component and deployment diagrams, the use case diagrams, use case scenarios, and the sequence and state diagrams. Aaron did the prototype and façade diagrams as well as the use case diagram(s) and scenarios. Shikha did the observer diagram. Andrea did the component and deployment diagrams. Tameka did the sequence and state diagrams. Along with that we worked on our Sprint 2 presentation, technical and organizational documentations. Tameka, Andrea and Shikha pitched in to complete the set up and design of the presentation slides.

For the technical and organizational documents, the physical system boundary was worked on by Tameka and the logical system boundary was done by Shikha. Shikha also completed our traceability table and updated our functional and non-functional requirements. Andrea worked on this sprint's progress visibility and risk analysis. Tameka also updated our Software Process Model and created the Gantt Chart. There were a lot of individual

tasks in this sprint with the presentation being a group task and the documentations as a whole, a group effort.

The Gantt Chart can be found in our Rampage zip file that includes both Sprint 1 and Sprint Gantt Charts. Directory "Rampage/Rampage Gantt Chart".

2.3 Gantt Chart v3:

For this Sprint, the main focus was implementing some functionality of our game as well as the deliverables. We had a lot of issues during the sprint, with our building sprites which caused us not do advance as much as we would have liked to. We are hoping in the next and final sprint we would be able to complete the functionality of a complete level without any errors if we are lucky. The design of the login and register screens were completed as well as the heads-up display. The functionality isn't completely there as yet but it shouldn't be too difficult to implement.

For the technical and organizational documentations, the Gantt chart, software process model, product security, and synthetic performance benchmarks' graphs were worked on by Tameka. The data dictionary, test case definitions and results, and organizational approach were worked on by Shikha. The performance requirements, performance objectives, application workload, bottleneck, synthetic performance benchmarks were worked on by Aaron. The user experience/game design, progress visibility and updated risk analysis was worked on by Andrea. The presentation was a group effort.

The Gantt Chart can be found in our Rampage zip file that includes the completed Sprint 1, 2 and 3 Gantt Charts. Directory "Rampage/Rampage Gantt Chart".

2.4 Final Gantt Chart:

For the final sprint, the main focus was testing our game. We were still coding during this sprint and waited until the last week to focus on testing. There were a lot of issues when trying to completely replicate the game, some being that the character movement was slow, and we could not figure out the npc and building functionality. We had still managed to complete the game in a way where the user can login or register an account as well as view the leaderboard and play the improvised gameplay of Rampage. We did not have enough time to implement saving the user's score so that was not a part of our game.

For the technical and organizational documentations, the Gantt chart, software testing and software process model were worked on by Tameka. The integration testing was worked on by Aaron. The unit testing, acceptance testing and organisational approach were worked on by Shikha. Part of the acceptance testing, the conclusion, progress visibility and risk analysis were worked on by Andrea.

The Gantt Chart can be found in our Rampage zip file that includes all of the completed sprints and their Gantt charts. Directory "Rampage/Rampage Gantt Chart".

3 Progress Visibility

3.1 Sprint 1 Progress Visibility

The group uses a shared Discord server to communicate. Within the server, we have a channel specifically for task related subjects such as assigning them to people, tracking progress and completion of them, and asking for any help related to them when needed. This ensures that each member knows what tasks are assigned to them, and if they were to forget it makes it easy to go back into the channel and check what was assigned to them. When assigning tasks, the group makes of list of each thing that needs to be completed. We then allow each person to volunteer for tasks they feel confident in. If there are still tasks that need to be completed after that, we consider each member's strengths and delegate them from there. We also take into consideration how many tasks each person has at this point to ensure that no one member is getting overworked. If another person has to complete a task before someone can move on, the person waiting could tag them in the channel to give them reminders or ask

for updates, and that pushes a notification straight to the other person. When that task is done, the same could be done in reverse to notify that person that they can begin their task. The group meets with the client once a week on Wednesdays at three-thirty to update him on the progress of the project.

3.2 Sprint 2 Progress Visibility

In this sprint, we continued to divvy up tasks in the same way that we did previously. Each of us could volunteer for tasks we felt confident in our ability to complete first. There were hardly any issues with people wanting to do the same tasks, and on the rare occasion that it did happen a compromise was reached quickly. For any remaining tasks that no-one wanted to claim we simply evaluated who should take it on based on how much work they currently had on their plate, as well as if their capabilities and skills were suited for the task. We update each other on our progress and completion of tasks through our 'tasks' Discord channel. We also make sure to send a message with the tasks we claimed if we did so in person so that there is a record to reflect back on and there is no question of who is doing what. If a member needs someone else to finish their task before they can complete theirs, they either direct message that person, or tag them in the chat to get their attention and let them know that they need it completed by a certain time. Everyone is doing well completing their tasks on time. The development team still meets with the client every Wednesday at three-thrity pm to update him and ask any neccessary qustions. The group holds an in-person meeting for twenty-thirty minutes before the meeting, as well.

3.3 Sprint 3 Progress Visibility

This sprint required a lot more collaboration between users which therefore means that we needed to hold more team meetings. The work was still divvied up between members to what we hoped would be fair amounts. We also tried to keep people's strong-suits and skill sets in mind when dividing up the work to each person - for example, Tameka did well in our database class last year so she is handling the login and registration which will rely on a database that she puts together. We still have our weekly client meetings at 3:30pm each Wednesday, and we use a half hour before each meeting to get together as a team and work out any issues we may be having our talk about a plan moving forward. We have also been holding weekly emergency meetings to handle bugs in Unity and/or with our code, as well as make sure that we are all operating on the correct updated branch of the project off of GitHub. We have also utilized time given in class to meet as a team and work on certain portions of our documentation and/or project (these total up to roughly 15-20 minutes per week). We update each other on our progress and completion of tasks through our 'tasks' Discord channel. This has kept us as organized and on-task as possible in this sprint.

3.4 Sprint 4 Progress Visibility

This sprint really showed how burned out our team was. We found it much harder to collaborate on the scale that we had been, however it was imperative that we keep at the pace we had been going at. Things ramped up more the closer it got to the final due date. We held holding weekly emergency meetings to handle bugs in Unity and/or with our code, as well as make sure that we are all operating on the correct updated branch of the project off of GitHub. We also utilized time given in class to meet as a team and work on certain portions of our documentation and/or project (these total up to roughly 15-20 minutes per week). We still met with our client, Dr. Galloway, once a week at 3:30 pm in his office in College High Hall on Wednesdays. We discussed any questions we had come up with, as well as the client's requirements and ensuring that we were meeting them. We divided up different test types for each member to carry up and prepare the documentation for. This enabled us to have an easier time completing the documentation for this sprint. Each team member had to become more flexible with their time and commitments during this sprint to ensure that the project was completed to the best of our abilities by the due date.

4 Software Process Model

The software process model we used throughout our project was the Waterfall model. We chose this one because as we are going through one sprint at a time, before we move onto the next, we have to ensure that the previous requirements in the previous sprint are already fulfilled. We cannot move onto the next sprint unless we

are done, so it is like the Waterfall model.

It increased our overall quality for the final deliverables because instead of having to try to do a rough draft of everything all at once and then going back to improve it, we can give our best for each step at a time. This also helped us to focus on one step rather than all at once. The quality control steps of the Waterfall model would be feasibility study and planning (Sprint 1), requirements, system/ program design and modeling (Sprint 2), implementation (Sprint 3), and testing/maintenance (Sprint 4).

The feasibility study and planning was the analysis of the project's relevant factors, such as the overview, scope, technical requirements, production techniques, risk analysis, execution and development environments, schedule and timeline, costs, and project visibility and communication. The requirements define function of the system from the client's viewpoint. This establishes, system functionality, constraints, dependencies, goals, and the development process. The system/program design and modeling, describes the system from the software developer's viewpoint. The implementation is the coding of the software. The testing was to ensure that the code for the software works as intended.

During Sprint 2, we completed all of the tasks related to requirements. This entailed the diagrams that help the client understand our plans for the project. We took our time to focus on the tasks at hand to ensure we gave our best outcomes.

During Sprint 3, we were not able to implement all of our UML diagrams. We did not complete much coding because of an issue we encountered with Unity and our building sprites which caused us to lose time to work on it for this sprint. With the little time we had left we had to focus on our deliverables but will be diving into the coding for sprint 4 and hopefully Unity isn't moody.

During the 4th and final sprint, we changed the requirements for the product because we ran out of time to replicate Rampage to the best of our abilities. Our is still somewhat functional and fulfils the new requirements.

5 Risk Management

5.1 Risk Identification

One risk was that the group could gain or lose a member at any time. Another was that the client could change their requirements and/or deadlines.There was also a chance that we could have had issues with meshing our code together. As we were moving through sprint two we identified a few more risks to bear in mind moving forward. For instance, there was a strong possibility that our project in Unity does not share correctly - a drive may not upload the files correctly, and if we used GitHub there was a chance we may use a local file for something and not realize, which would mean that it would not work for the other members. That also introduces a stronger likelyhood of our code not meshing together. Furthermore, any one of these risks could have caused our current plan of action to fall through. Moving into sprint three we identified a few more risks to add to the plate. Our security may encounter some issues since we used a third-party service for our database for user account information. We cannot control data breaches on their end, and that puts our users at risk. Throughout the development of the game, we had issues with Unity. Whether it was not syncing, or glitching sprites out of existence, or any other issues, it delayed us putting out our finished product. Communication between our database and unity could potentially go wrong as well at any point of use after development.

5.2 Risk Planning

We kept records of what each member of the group was supposed to do. If we lost a member, we would have divvied out the tasks they had left to the remaining team members. If we had gained one. we would each have given some tasks to that member to balance out the workload between all members. If the client changed requirements we would update tasks as needed and update our time esitmation as well. If the deadlines change, we would consider task priorities and fullfill all completely necessary tasks first; tackling minor bugs and design with any time that may remain. If our code was not meshing, we would hold an emergency team meeting to work

out the issues and rework things as needed. If we had issues sharing the project, we would go straight to Dr. Galloway for assistance in getting things shared correctly. If we were unable to get it shared, we would hold as many emergency meetings as needed to work off of one member's computer for all remaining issues. If our planned course of action fell through because of these things, we would have been able to pivot at any point. We could not get overly invested in any one aspect that is not basic functionality. As for issues that arise in sprints three and four, we would hold emergency meetings more often to put our heads together and work through any issues that got thrown at us. Three-four heads is better than one, and that allowed us to resolve issues faster and move forward with the project.

5.3 Risk Monitoring

We ensured that deadlines and requirements did not change at each client meeting. We asked for prior notice of group membership changes, however we were prepared for little to no notice of this risk coming to fruition. Wetested code as a team in person, so we could simply monitor code meshing issues as they arose in real time. When we first shared the project, we would do so while we are all physically together. That way we were be able to pinpoint any sharing issues straight away and work together to resolve them right then and there. We would then doublecheck that all of our updates synch correctly in our shared files moving forward. GitHub played a big role in sprints three and four because we could not commit any changes that will not work with files that were already present. This helped prevent us from pushing code that wouldn't mesh into the project. It's history funtion also allowed us to go back to a previous working version of the project after a commit has been made when it was needed.