Project 2: Technical Document / Risk Assessment

EGD220: Project Team 3

(Jak Tiano, Steven Baptiste, Tim Yasi, Evan Schipellite)

Prepared by: Evan Schipellite

Technical Synopsis

The goal of this document will be to introduce the general game concept in development, the coding and language structure in the technical areas of the project, and the various requirements that will be set out by the designer during the future milestones. The overall document will reflect the beginning and end of each milestone, and therefore each portion will be written as tasks are given out and after the tasks reach their conclusion. The purpose of the document will be to divide the sections so that the programmer will be able to break up each week and determine the requirements and potential risk pieces that the team may encounter. Each milestone the technical risk assessment will discuss the objectives of the milestone, the requirements, the optional pieces, and the potential risk areas. At the end of each milestone the review section will briefly discuss what went well, what wasn't completed, and the overall perspective of the project at the current time.

This document will assist, not only the programmer, but the entire development team by tracking the progress of the project and predicting areas that may either be challenging or time-consuming. The programmer will largely be in charge of noting the possible technical difficulties ahead in each milestone, but when possible, the programmer could break down various art and design aspects to bring up during group meetings. Since the programmer will be largely aware of the game mechanics and required art assets, there will be situations where the programmer will be able to assist the team by generating lists of required tasks and breaking down the complexities for each tasks. While this may be utilized in this document, the overall focus will still be the evaluation of the required programming components that will discern the requirements for each week and describe the risks that may go along with the various coding tasks involves in each milestone.

Programming Requirements / Further Introduction

The given amount of time for this project is a total of four weeks, allowing for one to two weeks for initial concept design, and two more weeks for gameplay and art implementation. One of the most pressing challenges of this project will not necessarily be the gameplay, but rather the overall design of the project to meet the project expectations. The project is based around creating an educational game for children between the ages of six and nine, and therefore the members of the design team will have to take on the process of structuring a game for the target audience, the six to nine year olds, and the target clients, teachers in the education system. This doesn't directly impact the technical and programming components of the project, although it is likely that the programmers of the team will also be involved in the brainstorming, research, and design of the eventual product. Rather, the indirect impact of the challenge is the fact that the game concept will not be completely founded after the first week, and it may not be until the middle of the second milestone that the team reaches a conclusive decision on the gameplay structure.

Still, there are various aspects of the game that are currently unlikely to radically change, and therefore it will be worthwhile to at least begin breaking them down and understanding the potential risks in those areas, if any. Once again, much of the error space surrounding this project will be focused on the visuals and design, as such roles will be required to craft a cultural and educational experience for children, and due to this, the actual process for conveying the meaning and message to the children may require reworking throughout the project. With regards to the programming portion of the project, once the basic outline for the game is set, the building of the project should be relatively simple as the gameplay should be very simple in order to be accessible and enjoyable for the target audience.

The game will be developed in AS3, utilizing FlashDevelop. FlashDevelop seems like a good pick, as both the programmer and designer should be able to work together and understand the coding structure. In comparison to FlashProfessional, FlashDevelop will allow the programmer much more freedom and control over the various aspects incorporated in the game's core, where FlashProfessional would limit the control over aspects including the stage, graphics, and camera. As a result, the entirety of the programming will be done via code without support from the features found in FlashProfessional. This should not have an impact on the artists or designers, as the programmer will be able to create sections the designer can edit and all art will simply need to be provided in proper format and, or, with included sprite sheets. As noted beforehand, the expectations for the programming portions should be relatively simple, but most of it will be based around how soon the concept is finalized and the major risks may occur where design is required to alter and further change the game structure.

Milestone 1: Risk Expectations

For the first milestone, the programmer is only required to set-up the utilization of the Technical Risk Assessment and develop the mechanic for the game. This mechanic will likely incorporate the Japanese “Maneki-Neko” icon into a game where the player will be able to experience aspects of Japanese culture and learn about the impact of the “Maneki-Neko” on everyday life in Japan. The game mechanic currently will be taken from a top-down perspective and allow the player to navigate through a traditional Japanese setting while collecting items and bringing them back to certain locations. Items will be color coded, explain aspects of the culture and item in simple text, and will be the main objectives in the game. The player will control a cat that will navigate through the area, and the player will take note of the various quest objectives that appear in the form of thought bubbles above houses. The player will then have to find the desired quest item and bring it back to the location. During this first milestone, while the design is further founded, the TRA should be completed and the basic movement should be created. It is wise to avoid extensive programming development until the design is further grounded.

Required Components

- The Technical Risk Assessment Document
  - The TRA should be created and the initial synopsis and expectations should be written down to track the progress of the project.
- The Mechanic
  - The mechanic for the game should be created and conveyed in Flash. While this doesn’t mean the programmer is required to have a full prototype, as the game may be subject to change, the initial idea of the game world could be shown by the programmer with basic controls to allow for presentation purposes and show the perspective.
  - The mechanic will allow the player to navigate through an area with a camera following them. The keys should be designed to not only be simple in terms of movement, but the logic behind pressing down different keys should be built during this time. It is likely that if two keys in the opposite direction are pressed, the player character should not move, but rotate in the direction of the last key pressed.

Optional Components

- The game structure could be further developed to show more gameplay depending on how soon the game design is further grounded.

Risk Assessments

- **Low:** The creation of the world should be simple, but setting up the camera and collisions may take some time to finalize and polish for future milestones.
- **Low:** The creation of the keys and binds could require an additional timer to detect multiple key presses and deal with them accordingly, but AS3 should have a feature available to detect such occurrences, and it could possibly be developed further within the input handler.
- **Low:** The development of the mechanic could require a bit of time due to the fact that the gameplay will only be somewhat finalized a few days before the end of the first milestone. Still, there's more than enough available time to complete this objective after the design meeting, so it should be an issue as long as the programmer can discuss the mechanics that will be incorporated in the prototype.

Milestone 1: Review

The first milestone was completed without any obvious difficulties from the technical perspective. During the various meetings the game idea had begun to reach a conclusive standpoint, and therefore once the actual mechanics are fully founded, it will just be a matter of assigning programming tasks for the upcoming milestones. The design of the TRA was simple as there was more than enough time available to continue to design and writing portions for the document. Furthermore, the creation of the mechanic went much better than expected, as in the process the game was not fully defined. Now that the top-down perspective is highly likely, there is a lot of space for the programming to begin developing the game's framework. Currently, it is possible to continue creating the collisions in the world and develop an item collecting system based on various spawn locations throughout the map. The overall creation of the movement and camera-based world was completed without any real issues, so further development and tasks will await on the tasks assigned by the entire team during milestone 2.

Milestone 2: Risk Expectations

For the second milestone, a basic prototype will need to be constructed to convey the core aspects of the gameplay. As a result, although the entire game will not be available at this time, a basic series of mechanics should be presentable to assist in conveying the game design. As a result, there are a certain number of features that will be essential for this milestone. Player movement, collisions with the exterior locations of the map, items, item pick-up ability, and visual goal drop-off locations must be included in the prototype gameplay. As the eventual game will involve having the player move around the map and collect certain items dictated by the colored goal points, the basic representation of these features in the game will assist in further showing the design. While these features do not have to fully work, it would be useful if the colored items could be picked up and dropped off into the goals of the respective colors. In later implementations, items will not only have a color design, but they will also be further differentiated by item types (or numbers via programming), and part of the more difficult component of the gameplay will occur when the goals will request an item of a certain type. However, for this milestone the first part pertaining to the item color will be available for presentation. This should assist the designer when giving information regarding the map design, item system, and drop-off ability for the eventual gameplay.

Required Components

- Pin Down Collision Areas for Exterior Parts of the Map
    - Simply adjust the collision boxes to prevent the player from exiting the map, while also allowing the camera to stop properly at each of the compass direction.
- Create Items
    - Create items of various colors and have them presentable on the map.
    - The player should be able to move over the items.
    - The player should be able to pick up the items, drop off the items, and switch the item with another item on the map.
        - When dropping off, the item should be placed at the player's location. When dropping off on another item, both items should simply be switched.
- Create a Spawn Manager
    - The Spawn Manager will manage all of the items within the game.
    - For the prototype, the Spawn Manager will be able to take into account a number of possible spawn locations from a text document, and generate a certain number of items that are based on the given spawn locations.
    - In this light, the game will spawn the items randomly among the given locations, and only one item will be able to spawn on a given location.

- Create a Goal and Goal Manager
    - The Goal Manager should be responsible for setting the number of available goal locations and monitoring the presence of the goals. The purpose of setting the number of goals will allow future implementation for level progression. Otherwise, for the prototype, the Goal Manager will be responsible for creating five goals at locations specified by a text document and managing the collisions when items are dropped in their zones.
    - The goal system should also allow items to be deleted when properly deposited. This should occur without errors and should not cause any further complications when other items are picked up.
- Preliminary HUD system
    - There are two aspects to the HUD system, but only one is required for this milestone. The HUD should contain the ability to display the item currently held in the corner of the screen, and it should update accordingly with the player's actions. Further art implementations will provide visual background and information, but for now the held item should simply be available for display.

Optional Components

- Further Improve the HUD Aspects
    - Add an additional HUD representing the goal amounts could be created to further convey the eventual gameplay. The HUD would be compromised of six basic parts, five of them representing each of the colors within the game. The sixth part of the HUD would show the number representing the total score. At the start of the game the game will show a number on each of the colored HUDS, with the sixth HUD equaling the total of all of the others.
    - Upon depositing an item into a correct goal, the HUD icon for that color should decrease, and the sixth HUD total should also decrease.
    - Further improvements to the HUD should allow for one of the five pieces to be highlighted, representing the fact that the player can only deposit an item of that color at that time. Once deposited, the score would update and another random color would be highlighted.
    - Once a color reaches zero, the HUD should no longer highlight it when choosing randomly. Furthermore, once all pieces reach zero, the sixth piece should highlight, indicating that the player has now completed the level and can return to the store.
- Implement Art Assets
    - If available, implement art assets and rework collisions.
- Implement game screens and store components
    - Allow the player to begin the game by pressing space, and allow for them to return to the start upon completing the game.
    - Apply a set of collision boundaries to the store location that becomes available at the end of the level.

Risk Assessments

- **Low:** Setting up collisions may require some work, although it is more of a task for future milestones. This is largely due to the way the art might be integrated into the game, as changes to the map design would require reworking collisions. Furthermore, the player is currently based on simple non-pixel collisions, which largely seem to work. However, it may be possible to further improve on the collision system if a pixel-based collision doesn't cause any major issues.
- **Medium:** Setting up the Spawn Manager should mostly just be a time matter. However, spawning items at various locations could require some amount of testing to ensure that all required items spawn properly and no items spawn on top of each other. This, if not fully tested and left bug-free, would cause gameplay problems and visual issues later.
- **Low:** The item pick-up system might require some further design testing in order for it to make sense. The act of dropping items in empty spaces opposed to switching items with another item might clash slightly. However, this will only be viewable after programming the two aspects. This is not so much a programming issue as a gameplay mechanic that could require polishing.
- **Medium:** The Goal Manager system should be simple to create, but generating events between the goals and the drop-off feature might cause some problems. Items must only be removed if dropped on the goal of their color, when the HUD mission is currently set to that item. This will require a system that can not only detect the color of the item, but also the type of item that is being dropped. This shouldn't be too much of a problem, but removing items from the Vector might generate some errors initially.
- **Low:** The preliminary HUD should be simple enough, since the programming concept will essentially be the reality that when an item is picked up, it will simply be moved off the camera, onto the stage, and placed in the corner of the screen. Therefore, any cases where the player can actually reach the item might cause issues, but this should never happen, and if it does, it can simply be resolved with a Boolean.
- **Low:** Implementing the art assets, although optional, might not be best to do if done shortly before the end of the milestone due to the process of reworking the collisions and checking spawn locations. Even so, it shouldn't cause any major problems.
- **Medium:** Also optional, but the HUD system will require a complex manager that can randomly assign a given quest, listen to events about items being dropped, and also set certain game states depending on the status of the quests. Simple enough, but as an optional feature, it will require a decent amount of additional time.

Milestone 2: Review

Overall, milestone 2 went as planned, and most of the optional components were added to the prototype. Various aspects, such as item types, mission additions, art implementations, level progressions, and game screens still need to be added for the remaining milestones. However, those are the primary concerns for future milestones, as everything surrounding the game's design has essentially been finished during this milestone. The expectations for this milestone were all completed and polished for testing, and most of these aspects were completed early on in the process. Item pickup and drop-off were easily added into the game, and creating the Goal Manager did not require too much reworking after implementing the features. Items were easily configured to be noticed by the goals, and goals were designed and built to accept and remove items dropped that are the same color as them. Spawning items at random locations was not too difficult to debug, as a simple array was constructed to simply keep tabs of the available locations to spawn.

With regards to the optional features, a lot of them were implemented in order to continue assisting with the visual presentation for the upcoming formal proposal. A HUD system was created to keep information regarding the amount of each color required to finish the level. Information is taken from the text files, and the HUD easily randomly selects a color to highlight as the current mission depending on which ones are above zero. Once all displays reach zero, the final HUD piece highlights to represent the end of the game. Once the final HUD piece highlights, an additional store feature will enable, removing collision from the store to represent that the player can now return and end the level. While the features for the actual end screen were not implemented yet, it would simply be a matter of adjusting the game state once the player reaches the store. The only difficult part of the HUD was simply the constructing of the numbered sprites, as approximately 156 sprites were created to represent the decrementing numbers on the HUD. Even this process didn't take too long due to the ability to recolor the sprites after listing out the numbers 0 to 25, but it did take slightly longer to create than the actual process of coding the HUD into the game.

The artist also made the prototype map available, which was integrated into the game without any issues due to the consistent resolution of the game map. The artist also created a foreground to create the illusion that the player can walk beneath various objects, and the shadowing aspect also works extremely well. This implementation did not take any strains through programming, and therefore went much better than expected. The only aspects of the expectations that were not implemented were the optional game screens, as it seemed better to wait for the actual layout of the game menus before constructing the system. Still, all of the expected features were created and implemented and many additional polishes and extra features were added that will make the upcoming milestones much simpler to create game components later in the development process.

Milestone 3: Risk Expectations

Due to the strides made in the second milestone, by the end of milestone 3 the final components of the game should be near completion. In this sense, at the end of milestone 3, all of the programming features should be done, thus allowing further time for two major aspects. First, the implementation of art will be possible once the core programming parts are done, since this should just be a matter of tweaking the system against the structure or the art assets and sprite sheets. Furthermore, by having the essential programming portions done by the end of this milestone, the final milestone will allow for further testing to report and correct any bugs as well as to tweak design features and gameplay. As a result, during this milestone the mission interaction system will need to be further developed and completed, thus allowing for each mission to show it's desired item, and only items of the correct color and type should be accepted by the current mission goal. Furthermore, it would be useful if a level system was correctly set in place, thus allowing for the easy creation of different levels with set amount of goals, items to be collected, and extra items. However, the level system may not be required by alpha, so it is technically an optional requirement at the current moment.

Required Components

- Mission Interaction System Completion
    - Beforehand the HUD would dictate the current color required, and the player would have to deposit an item of the correct color to continue onward.
    - Now the system should further require a certain color and type, thus forcing the player to locate a specific item on the map.
    - This system will require the game to not only ensure that enough items are spawned to complete the objectives, but it also must keep track of the values of the items spawned in order to make them mission objectives during the course of the game.
        - This implementation will ensure that all of the quest items are on the map, since otherwise the game may not be able to be completed. It should also be further noted that there is still a sense of random-ness in the manner that items still have random values assigned to them.
- Bottom HUD
    - The mission interaction system will require a HUD to be set up in the bottom section.
    - This HUD will hold the image of the current item held and it will show further information depending on the state of the player.
        - If the player is holding an item, it will show the item 'text' to the right of the image.
        - If the player is not holding an item, it will show the current mission prompt.
        - If the player is not holding an item and no missions are available, a prompt will be given for the player to return to the start location.
- Import Art Assets

    - While not required to be completed by this milestone, it should still be essential to import as much art as if available and on time.
        - Items, Cats, and HUD features are largely expected to be available by the end of the milestone, and therefore attempts should be made to integrate them into the game. This should be relatively simple given the flexibility of the system, but time will be required to ensure the transition and test the game. Therefore, any items given after a certain amount of time before the milestone may have to wait until the following milestone.

Optional Components

- Implement Level System
    - While the level components are currently in place, there is no direct system to refresh / delete old content and build into the next level. While the current system should allow for it, it will take some time to make sure that within each manager, the proper aspects are deleted or reset.
    - The end goal will allow for data to be extracted from a text file at the beginning of each new level to set the proper HUD, mission, and item features. At the end of all of the levels, the game will reset to the start level.
- Screen System
    - A basic screen system could be created that would show a start screen at the opening of the game and a complete screen after each level. At the end of the game it would show a complete screen, followed by a start screen, thus allowing for the replay of the game. This should be kept as basic, since information surrounding the design and art of the screens is still unknown, but a basic screen system that maintains the current game state and allows for the current key inputs should be created for future use.

Risk Assessments

- **Medium:** The actual implementation of the mission interaction will not be simple. This is simply because each goal will be required to be aware of the 'ensured' items that were created at the beginning of each level. The HUD, mission prompt, items on screen, and goals all must be in communication to ensure that the correct items are selected as mission prompt and the goal is set to accept that item. While this isn't too much of a risk, it will take a lot of time to debug, test, and figure out the process to allow for such Events to connect the different managers.
- **Low:** The bottom HUD shouldn't be too difficult, but once again, it must correspond to the given item held and the current quest provided. Therefore, when the player picks up an item, the correct information box must be presented according to the item held. Furthermore, when an item is not held, the mission prompt requesting a specific item must be displayed.
- **Low:** Importing the art assets shouldn't be too difficult. However, depending on the type of assets provided, the programmer may either be required to rework the system or the artists may have to revisit the structure in rare cases. Due to the specifications already set out, it is more likely that the programmer will simply have to readjust the system to accept the organized sprite sheets. This will not be difficult by any means, but depending on the time when the art becomes available, it may be a bit of a task to prepare the art implementation by the end of this milestone.
- **Medium:** The implementation of the level system will be somewhat difficult. This is simply because at the end of each level the information must be cleaned up, and therefore there is a lot of room from bugs

and issues when entering new levels. At the end of each level all graphics, except for the complete screen, must be removed. All managers must have the required information reset. Then, the game must return to a certain state in order to readjust the game according to the next level specifications in the embedded text file, and continue on from there. While simple in concept, ensuring that the correct features are deleted or reset will take some time, as many components, such as the spawn locations and goal locations, will not need to be removed while other aspects, such as the items and current HUD notations, will need to be removed at the end of each level.

- **Low:** The level system should cause any outstanding difficulties, but triggering the game states at the correct time will require some testing. This is simply because it will be required to ensure that at the end of each level that the screen manager does not get altered and still is requesting further key input, even if the update function is temporarily halted.

Milestone 3: Review

Given the amount of objectives already completed in milestone 2, milestone 3 went according to plan and most of the optional features were implemented. At this point in the development stage, all of the mechanics and gameplay aspects have been implemented through programming. Therefore, all that's required at this point is further testing and the implementation of the art and mechanics. This will allow for the final week to largely be based in the testing of the design and visuals, and therefore programming will only be required for tweaks in the current system. There is still a low to manage for milestone 4 in terms of testing and implementation of final art pieces, but it will be less stressful given that the core aspects of the game are all in place. The only components that require further tweaking from the milestone are largely based in the art sections, largely because the artists may want to further polish or develop on the art that was made available.

With that in mind, the majority of the art made available was easily placed in the game due to the adherence to the structure set up by the prototype sprite sheets in the game. In other words, by matching the design on the sprite sheets already found in the Lib folder in the game's content directory, the programmer was able to easily replace the old file with the new versions in order to easily place the content in the game with little to no adjustments in the code. This was largely due to the benefit of communication between the artists and programmer over the course of the project's development, as the artists requested and were made knowledgeable of the best specifications and design for the art sheets. Therefore, while some art still requires further development or testing to properly implemented, a great majority surrounding the items, map, and HUD pieces were added to the game during this milestone.

With that in mind, all of the other features were also finished on target. The mission system, while taking a considerable amount of time as expected, was completed in such a manner that will allow for all requested quest items to be set at the start of each level, and each goal point will only accept the desired item color and type. A basic prototype HUD was set up on the bottom, thus showing the mission prompt and information surrounding the current item held. While taking some time to integrate the HUD, spawn manager, and goal manager with each other, it certainly was all possible during this milestone. The level system, while taking time due to a number of bugs, was also implemented at the same time that the screen manager was added. There were little to no complications with regard to the screen manager, and while the level system required some time to set-up, it should allow for each level design during the final milestone.

Milestone 4: Risk Expectations

At the end of the third milestone, almost all of the programming content was completed in order to allow for more time during the fourth milestone to simply test the game and add additional art assets. Screens were implemented; however, this milestone may call for additional title screens to convey instructions and information. This isn't at all a time-consuming integration into the game structure, but since the rest of the programming is already completed, it marks the only 'real' aspect of programming that will need to be tackled to allow for the final pieces of the art to be integrated into the game. Therefore, once completed, all that will need to be done in the process of awaiting the art availability, implementing the art, testing the ensure that the visuals and mechanics work, and then completing various aspects of paperwork that might be available. Due to the limit of time during the fourth milestone, and the fact that the individual and team postmortems will be due shortly after, it may be best to allot some time for those documents and meetings in order to ensure that they are all prepared and uploaded in the short timespan that marks the end of the last milestone and the due date for the postmortem papers.

Required Components

- Screen Manager Adjustments
    - The screen manager should simply be altered to allow for two additional screens to follow the start screen.
- Implement Art Assets
    - The final art assets will need to be implemented into the game. These include:
        - The store
        - The store wall
        - The finalized title screens
        - The finalized description and quest texts
        - The finalized return text
            - This may take some reworking as it will not be included in the description or quest spritesheet.
- Documents
    - The individual postmortem, team postmortem, and TRA will need to be updated before the final milestone.

Risk Assessments

- **Medium:** Implementing some of the art assets could be an issue, but only if last minute structures need extreme changes in situations such as the HUD, item descriptions, or quest prompts. As is, most of the art made available should simply replace older versions, but if design changes in the HUD result in new additions, some further coding structure may need to be implemented.
- **Low:** Finding time to complete the individual and team post mortems near the end of the last milestone, but before the postmortem due date should at least be acknowledged by the team to help people submit the documents in on time and meet to discuss the team document.

Milestone 4: Review

Milestone 4 was largely based on the team meetings that occurred, as the only thing that really needed to be implemented through code was the minor HUD adjustments and the implementation of art assets. Most of this content was completed during the team meeting, as it was there that the team was able to discuss the final specifics for features such as the top HUD, bottom HUD, item texts, and screen content. Still, there's was a decent amount that was altered and improved outside of the meetings, as it did take a small amount of time to alter the screen manager to essentially progress through the three initial start screens, and there were multiple renditions of the map art that was tested over the course of the milestone.

The only aspects that took time were the final documents, and even those were not too time-consuming. The TRA took less time than usual due to the fact that the amount of risks to report were slim and there were not any present optional features to record. The written version for the individual post mortem also took some time, but it was done after all the other features were just about set. During the middle of the milestone, the basis for the team postmortem was created to save time at the end of the week, simply because the amount of space between the end of the project and the deadline for the team postmortem wound up being extremely small. Therefore, although not directly related to the success of the project, the documents finished during this milestone ended up taking the majority of the time, which only ranged from two to three hours.

For the rest of the milestone the programmer simply updated the art when made available, and since all of it was set and ready by Wednesday evening, it was possible to finalize the game, provide some additional testing, and upload the TRA and .SWF up onto the SVN for the producer to mark down for the final paperwork and wiki information. A small issue occurred with the process of cutting down on the amount of lag that was appearing on older computers, but this was largely fixed by adjusting the amount of bitmap data made available by decorative images. Alex Beauchesne assisted reworking the graphics system to manage less buffer data during the game processes, and as a result the game no longer lagged on older systems. Overall, due to the previous mechanics finalized in milestone 3, all the expectations were met in milestone 4 since they were all mostly based on art implementation and documentation.