CS449 Sprint 1 Report

Team **Misael's** Project Submission

Michael Cu, Elias Julian Marko Garcia, Samual Lim

October 6, 2019

Contents

1 Micro Charter

1.1 Project Name

N Men Morris

1.2 Vision Statement

Create a extensible framework for board game web apps with scalability and performance.

1.3 Mission Statement

To play Nine Men's Morris on the web browser using a composable web technology stack that allows for future modularity while not foregoing performance.

1.4 Elevator Pitch and Business Value

We are creating Nine Men's Morris on a board game framework using Express.js and Neon for Rust. This allows for a data and type safe application that is capable of composability, scalability, extensibility, and performance.

1.5 Customers and Users

- Customers: Entrepreneurs and ventures that want to deploy board games on the web with low overhead, latency, and maintenance.
- Users: Individuals who are passionate about board games and want a new online experience that they can take and play wherever they go with their friends.

1.6 Metrics

By benchmarking N Men Morris, we can compare our solution to other products on the market on:

1. latency
2. binary size
3. up-time

1.7 Milestones

1. First MVP
2. First Offline N Men's Morris
3. Player versus Player (Offline)
4. Player versus Player (Online)

1.8 Risks

1. Inherent complexity of technology stack.
2. Inability to cooperate with teammates.
3. Plausibility of orphaning project due to development team size.

1.9 Authors

- Michael Cu
- Elias Julian Marko Garcia
- Samual Lim

2 User Stories

Below you will find a table that makes up our "User Story Board", with some simplifications taken with respect to the total contents of the board. With respect to the final formal documentation, i.e. this paper, we only keep the basic qualitative and quantitative values for each story in the table while giving each user story proper its own section. This makes documenting each story less unruly while also easier to read. Each Story I.D. (SID) value is internally linked to its respective story, which also helps with navigating this section.

SID	Story Name	Priority	Time Est. (hr)	Actual (hr)	Status	Developer(s)
S1	Mills Board	high	10	4	DONE	Sam, Michael, Elias
S2	User Input and Selection	high	10	4	DONE	Sam
S3	Starting a Game	medium	10	-	TODO	-
S4	Assigning Players	medium	10	-	TODO	-
S5	Piece Placement	high	10	2	TODO	Sam, Michael
S6	Piece Movement	medium	10	-	TODO	-
S7	Mill Formation	medium	10	-	TODO	-
S8	Piece Elimination	medium	10	-	TODO	-
S9	Flying Pieces	medium	10	-	TODO	-
S10	Defining End Game	medium	10	-	TODO	-
S11	Restarting/Replaying Game	medium	10	-	TODO	-

2.1 Mills Board

Description

As a user, I need an empty board consisting of 4 expanded squares with 8 equidistant positions each to play a game of Nine Men's Morris.

2.2 User Input and Selection

Description

As a user, I need to be able to select and choose input from the web GUI of the application to be able to play and take turns at Nine Men's Morris.

2.3 Starting a Game

Description

As a user, I need a GUI to prompt me with the options to start a game with either another human or against the computer for Nine Men's Morris in order to play the game.

2.4 Assigning Players

Description

As a user, I need to be assigned the role as either the first or second player, whether against another human or the computer, in order to know my player turn (either first or second) in the game.

2.5 Piece Placement

Description

As a user, I need to place nine pieces on unoccupied positions in turn with another player to start off a game of Nine Men's Morris.

2.6 Piece Movement

Description

As a user, I need to be able to move my pieces into adjacent positions that are not occupied by the other player or adjacent to their mill in order to take a turn.

2.7 Mill Formation

Description

As a user, I need the game to recognize that I have formed a mill upon moving three of my own pieces into adjacent positions so that I may gain the future ability to attack and defend my mill pieces from being eliminated.

2.8 Piece Elimination

Description

As a user, after forming a mill, I need the ability to remove an opponent's piece of my choosing so long as either it is not in a mill or any piece given all available pieces are in a mill, so that I may appropriately attack my opponent.

2.9 Flying Pieces

Description

As a user, upon reaching three remaining pieces, I need the ability to fly (jump) my pieces across the board to any un-occupied point in order to play Nine Men's Morris according to the rules. Whether the position is guarded is a variant of the game, implementation decision TBD.

2.10 Defining End Game

Description

As a user, when either myself or the opponent reaches less than three pieces, i.e. two pieces, I need the game and to declare the respective winner in order to successfully finish a game of Nine Men's Morris.

2.11 Restarting and Replaying a Game

Description

As a user, after having completed a game of Nine Men's Morris, I need the GUI to prompt me to either play again or to end the game software so that I can accordingly choose whether to keep playing or to end my game session.

3 Acceptance Criteria

The following section covers the acceptance criteria enumerated in response to the User Stories discovered and documented in §2. In a similar fashion to §2, the table documenting these acceptance criteria is in a simplified form. Every Acceptance Criterion has an Acceptance Criterion ID (`ACID`), which is associated in the table below with its respective `SID`, development status, and the developers responsible for implementing it. Each `ACID` is linked to its respective subsection below for viewing the description of each criterion.

SID & Name	ACID	Status	Developer(s)
S1 Mills Board	A1	DONE	Sam, Elias, Michael
S2 User Input and Selection	A2	DONE	Sam, Elias, Michael
S3 Starting a Game	A3	TODO	-
S4 Assigning Players	A4	TODO	-
S5 Piece Placement	A5	DONE	Sam, Michael
S6 Piece Movement	A6	TODO	-
S7 Mill Formation	A7	TODO	-
S8 Piece Elimination	A8	TODO	-
S9 Flying Pieces	A9	TODO	-
S10 Defining End Game	A10	TODO	-
S11 Restarting/Replaying Game	A11	TODO	-

3.1 Criterion 1

ACID	Description
1.0	Given a User...
1.1	When the User visits our site (IP), then an interactive board will appear.
1.2	When the User does not visit our site (IP), our board will not appear.

Further Notes

None for now.

3.2 Criterion 2

ACID	Description
2.0	Given a User using the application...
2.1	When a user clicks on an interactive button of the application's page, then the application will detect the user input event.
2.2	When a user clicks on a non-interactive button of the application's page, then the application will not detect any input.

Further Notes

None for now.

3.3 Criterion 3

ACID	Description
3.0	Given a User using the application...
3.1	When a user enters HUMAN as an opponent, then the application will allow for a second human player.
3.2	When a user enters AI as an opponent, then the application will assign an AI as a second player.
3.3	When a user chooses neither a HUMAN or AI as an opponent then the application will not choose and will re-prompt the user to choose an opponent type.

Further Notes

None for now.

3.4 Criterion 4

ACID	Description
4.0	Given a User using the application...
4.1	When a user chooses player one, then the application will assign the role of player one to the user.
4.2	When a user chooses player 2, then the application will assign the role of player two to the user.
4.3	When a user chooses neither player one or player two then the application will not will not assign a player and the player will be re-prompted

Further Notes

None for now.

3.5 Criterion 5

ACID	Description
5.1.0	Given a User playing a game with unassigned pieces...
5.1.1	When the user enters an unoccupied position, then a piece of the users color will be placed in the position.
5.1.2	When the user enters an occupied position, a piece of the users color will not be placed in the position.
5.2.0	Given a User playing a game with no unassigned pieces...
5.2.1	When the user enters an unoccupied position, then a piece of the users color will not be placed in the position.
5.2.2	When the user enters an occupied position, then a piece of the users color will not be placed in the position.

Further Notes

None for now.

3.6 Criterion 6

ACID	Description
6.0	Given a user playing the game during their turn...
6.1	When the user moves his piece to an unoccupied position not adjacent to an opponent mill, then the piece will be shifted.
6.2	When the user moves his piece to an occupied position not adjacent to an opponent mill, then the piece will not be shifted.
6.3	When the user moves his piece to an unoccupied position, adjacent to an opponent mill, then the piece will not be shifted.
6.4	When the user moves his piece to an occupied position, adjacent to an opponent mill, then the piece will not be shifted.

Further Notes

None for now.

3.7 Criterion 7

ACID	Description
7.0	Given a User is playing their turn...
7.1	When the user places a piece in a valid position adjacent to two other pieces of their color, then a mill will be formed.
7.2	When the user places a piece in an invalid position adjacent to two other pieces of their color, then a mill will not be formed.

Further Notes

None for now.

3.8 Criterion 8

ACID	Description
8.0	Given a User is playing their turn...
8.1	When the user moves a piece from his mill into an opponent's piece not in a mill, the opponent's piece will be replaced by the user's piece.
8.2	When the user moves a piece from his mill into an opponent's piece in a mill, the opponent's piece will be not replaced by the user's piece.
8.3	When the user moves a piece from his mill into a vacant space, no opponent's piece will be replaced by the user's piece.

Further Notes

None for now.

3.9 Criterion 9

ACID	Description
9.0	Given a User is playing their turn...
9.1	When the user loses a piece such that they only have three pieces remaining on the board, then the application will allow them to "fly" their pieces to any open and valid position on the board.
9.2	When the user loses a piece such that they have more than three pieces remaining on the board, then the application will not allow them to "fly" their pieces to any open and valid position on the board.

Further Notes

None for now.

3.10 Criterion 10

ACID	Description
10.0	Given a User is playing their turn...
10.1	When the user eliminates an opponent's pieces down to two pieces, then the user wins.
10.2	When the user's pieces are eliminated down to two pieces, then the user loses.

Further Notes

None for now.

3.11 Criterion 11

ACID	Description
11.0	Given a user after they have completed a game...
11.1	When the user chooses to play again, then the board will be reset and game count incremented.
11.2	When the user chooses not to play again, then the board will not be reset and game count not incremented.

Further Notes

None for now.

4 Implementation Tasks

This section summarizes the details of implementation tasks for the project.
You will find in each subsection a table similar to those found in §2 and §3.

4.1 Summary of Production Code

SID & Name	ACID	Class Name(s)	Status
2 User Input and Selection	2.1, 2.1	Window, Board	Done

4.1.1 Class `Window`, `Board`

Method	Notes
1. `eventPress` 2. `at`	These functions relate to a pseudo-epic, and thus the testing will be generic.

4.2 Automated Test Code

There were no automated tests for this sprint.

SID & Name	ACID	Class Name(s)	Method Name(s)	Description	Status	Developer

4.3 Manual Test Code

SID & Name	ACID	MTID	Status	Developer(s)
S2 User Input and Selection	A2.1	M1	DONE	Samuel, Michael
S2 User Input and Selection	A2.2	M2	DONE	Samuel, Michael
S5 Piece Placement	A5.1.1	M3	DONE	Samuel, Michael
S5 Piece Placement	A5.1.2	M4	DONE	Samuel, Michael
S5 Piece Placement	A5.2.1	M5	DONE	Samuel, Michael
S5 Piece Placement	A5.2.2	M6	DONE	Samuel, Michael

4.3.1 Manual Test 1

Test Input	Test Oracle	Notes
document .getElementById("A1") .onclick	function onclick()	Checks if element clickable.

4.3.2 Manual Test 2

Test Input	Test Oracle	Notes
document .getElementById("container") .onclick	"undefined"	Checks if element clickable.

4.3.3 Manual Test 3

Test Input	Test Oracle	Notes
"A1"	elem.style.backgroundColor !== undefined	This is a GUI test. GUI will show piece placed in bottom left corner.

4.3.4 Manual Test 4

Test Input	Test Oracle	Notes
"A1", "A1"	"elem.style.backgroundColor = previousColor"	This is a GUI test. GUI will show piece placed in bottom left corner.

4.3.5 Manual Test 5

Test Input	Test Oracle	Notes
"D6"	"board = previousBoard"	This is a GUI test. GUI will show piece placed in bottom left corner.

4.3.6 Manual Test 6

Test Input	Test Oracle	Notes
"A1"	"board = previousBoard"	This is a GUI test. GUI will show piece placed in bottom left corner.

4.4 Other Manual Test Code

There were no other manual tests for this sprint.

ID	Test Input	Expected Result	Class Name	Method Name of Test	Status	Developer

5 Meeting Minutes

5.1 Meeting 2019.09.04

- Duration: 1 Hour
- Location: Miller Nichols Library

Agenda

- going over project pdf as group
  - discussing tech stack
  - going over sprint assignments
  - going over normal assignments
- discussing the actual structure of sprint 1
  - requirements
  - user stories
  - what submission might look like
  - discussion of who gets to do what
  - discussion of when to meet, general availability
    - Sam will be gone from 9th through 19th
    - Elias will be gone through the 12th - 14th

project 1 report

- want to get scrum documentation done
- get general idea down by end of this friday (2019.09.06)
  - structure of the project
  - how to use the frameworks/libraries involved (personal research/reading per individual)
    - Neon for rust
    - node.js
    - potentially express.js
- generating cards, user stories

5.2 Meeting 2019.09.06

- Duration: 1 Hour
- Location: Miller Nichols Library

Agenda

- discussing game rules
- discussing/writing user stories
- discussing tooling
- discussing design

Game Rules

- watched a video demo'ing the game
- discussed/clarify mechanics
  - whether or not to include coin flip
  - terms of loss
  - flying mechanic

User stories

- elias wrote user stories in a new org mode file called kanban.org on the repository
- discussed problem of documentation given requirements from the pdf for sprint 1
- discussed alternative means of documenting, carrying out execution of our cards for the project

Tooling & Design

- did not achieve agenda, did not get to these topics because of the time it took to discuss our epics/user stories.

TODO

- ☐ discuss tooling
  - need to finalize what our stack will look like and frameworks to be used.
  - elias has experimented with Neon and reports that it works well, seems viable for the product.
- ☐ discuss design
  - need to discuss how the actual product will be packaged and its architecture.

5.3 Meeting 2019.09.27

- Duration: 1 Hour, 30 minutes
- Location: Miller Nichols Library

Agenda

- discuss project structure
- acceptance criteria
- work assignment
- remaining TODOs

Project Structure

- express.js has a lot of dependencies, only really need connect.js
  - might try just using connect.js, which would be a lot simpler
  - will continue with using Neon
- board
  - gui
    - js renders the fontend
    - logic/data is all handled on back
  - data structure/representation

    * two choices:
        1. one big board object that includes methods for both resolving where players are **and** where things like mills are
        2. two object entities, one is purely for the GUI (tracking positions on the board), the other would be some kind of graph structure that allows position nodes to check peers for occupation and whether it is the same or opposing players

- movement and move validity
    * need to track flying
        · proposition: flying is a universal property, merely constrained until player count is reduced.
        · need some kind of getter/setter between board and entity management system
        · mill detection
        · if going with entity system, would merely be a graph traversal from any given node
        · another idea: create a mill entity system that tracks active mills and checks each mill upon each turn(?) and modifies or destroys the mill as necessary.
        · could save a lot of checking
        · as for organization/logical membership, would keep such a mill entity system independent of other objects in the system for simplicity, at least for now.
        · Checking for attack
        · if a mill entity system is used, we natively have a means to detect valid attacks. so long as the node is not in one of the mills, do not attack **unless** all available nodes are in mills.

- game driver
    * Will have some kind of Game entity/manager object that drives the game event loop.
        · will take inputs from players, run them as game moves
        · however, internal logic to the entity management system is what will ultimately validate moves

        - game manager will have no logic for why this happens, only passes back and for game inputs and the results of moves.
        - consequentially, need to codify where and how game validation logic happens
    - validation logic
        * as of now, think it will be handled by the main entity management system
        * will have a set of logic checking methods defined over the system that verify whether a given move is allowed

Acceptance Criteria

- realized we need to add numbering to the board GUI (a-g, 1-7)
- (deferred, Sam will begin working on before next meeting)

Work Assignment

- elias will begin on exploratory work for the backend (board, entity management, etc)
- sam, michael will begin exploratory work for the frontend (GUI, communicating with backend)

TODO

- ☐ kanban board setup, finalization of workflow for documentation
    - can probably just use github for real time management, but keep organizational and notes in `kanban.org` file on the repo.
- ☐ defining test cases for stories and acceptance criteria
- ☐ refining stories
    - same case applies with above: refine stories, and put them on github's project management board accordingly; actual refinement can be delegated to within `kanban.org` file.

5.4 Meeting 2019.10.02

- Duration: 1 Hour, 40 Minutes
- Location: Miller Nichols Library

Agenda

- addressing tagged issues generated on GitHub
- settling on how front-end talks to back-end
- documentation/design stuff

Issues on GitHub

- issue #3: determine communication channel between js and rust
  - event polling seems overkill for what we need
  - even handler on front-end which speaks to an entity Manager-Glue, which will be the JS that talks to rust backend
    * There will be a manager in the back-end, which will generate game state, and return that to the front-end
    * back-end will also have triggers (flags? Enums?) which signal to front-end when certain actions are no longer needed or valid, i.e. button inputs or game state continuation
  - JSON seems like a good enough medium for message passing between front and back components
  - issue #4 is largely tagged to #3, so this resolves that
    * `State`: Input Handle + BoardStruct + Trigger)
    * `BoardState`
      · this is what gets sent back to the JS
      · 1D array of the `State` struct

        - this array will be handed off as a NeonJS object, whatever it's called in neon
        -

- issue #6: Front-end/GUI Skeleton, Basic Design
    1. Neon builds a node module
    2. This is sent to express.js
        - accepts it as a bunch of js functions
    3. Express takes this, as a bunch of objects, and then saves as strings to JS files, in turn statically served to end user (i.e. browser)
        - express.js interaction is a one-off affair
    4. Stretch goal: being able to set different themes on the front-end
- issue #8: CI/CD
    - GitHub has native CI/CD now via it's Action's service.
    - can impl for both Rust and Node.js

TODO

- ☐ Design docs(?)
    - at least 3 needed:
        1. event diagram
        2. general UML diagram for total project
        3. class hierarchy/component diagram

5.5 Meeting 2019.10.03

- Discord: 1 Hour, 53 Minutes
- Location: Video Call

Agenda

- how to branch
- branching $\text{basic}_{\text{gui}}$
- GitHub PR format
- styling format

GitHub PR format

- Show Michael how to create a branch
- name the branch and pull from remote
- push the branch from local
- sync branches
- checkout a branch

Branching `basic_gui`

- created a branch `basic_gui`
- set up an issue with the branch for PR
- push a commit from local to remote branch

GitHub PR format

- went through how to form a PR from different branches
- how to further commit to the compare branch

Styling Format

- no bootstrap, no jquery
- setup proper layouts for the GUI
- discussed how we want to handle events onclick

TODO

- push scaffolds for the website GUI
- handle basic logic for pushing items to back-end storage
- create mock of Rust functionality in TypeScript for further discussion

5.6 Meeting 2019.10.05

- Duration: 1 Hour, 16 Minutes
- Location: Video Call

Agenda

- CSS Grid
- SASS
- TypeScript
- build script compilation and runtime
- proper layout for GUI

CSS Grid

- teach Michael about CSS Grid
- pure CSS, not bootstrap (Elias)
- use columns properly
- no need for floats / flexbox

SASS

- transpiler for CSS
- allows nested functionality
- separate compiled/uncompiled folders
- use `watch` script to sync changes

TypeScript

- better able to handle equivalence mocking to Rust
- easy to push onto browser
- separate folders (see above)
- push to public folder for site access

Build Script Compilation and Runtime

- use watch and start scripts to build site
- separate scripts will be run for Rust beforehand
- build scripts allow for synced changes between folders (see above)

Proper Layout for GUI

- use column areas in CSS Grid
- main column for game
- nested grid for board layout (tentative)
- proportion text for board side-by-side

TODO

- design docs
- microcharter
- mocking TS => Rust
- event keys on front-end (browser)

5.7 Meeting 2019.10.06

- Duration: 7 hours
- Location: Video Call

Agenda

- Tying up loose ends with respect to documentation and write up
- Tying up loose ends with respect to UI/JS end of the application
- Discussing what is left to do with the project

Documentation and Write-up

- Figured out how to format the tables given that many of the ones provided do not play well with latex/org-mode markdown
- Similarly, decided on how to interconnect documentation components between sections
- Discussed the remaining things left undocumented, particularly pair ratings.

UI/JS Loose Ends

- Complete manual testing of interacting elements
- Finalize positions of clickable elements on the board grid.
- Alternating player logic for placement of pieces.
- Limiting piece placement to nine.

Discussing Future Sprint/Direction of Project

- Current User Stories are pseudo-Epics and need to be refined into better User stories aside from S1. As they stand, discussing the current user stories makes for overly generic/abstract discussion and doesn't meaningfully translate into logic/behavior to implement and actual engineering tasks.
- Currently, the front end mocks all of the behavior/functionality that would otherwise be provided by the backend. In sprint 2, this is where the real meat of programming will come in as we learn to make the back-end and front-end interface, particularly with translating data types across the FFI boundary through Neon.
- We need to improve the current state documentation massively.

  - Design diagrams.
  - Docstrings across software code base.
  - Event diagrams.

- Translate the above issues into their proper documentation for the master documentation and write-up file
- How to test more of the functionality given that a major component of this application is running directly on the browser.

6 Team Ratings

Submission document does not specify scale, so it is assumed out of 5 with 1 being "Worst" and 5 being "Excellent".

	Elias Julian Marko Garcia	Michael Sy Cu	Samuel Chia Ern Lim
Elias Julian Marko Garcia	-	5	5
Michael Sy Cu	5	-	5
Samuel Chia Ern	5	5	-
Average	5	5	5