Me and da Boids - Project Interim Report

A Processing Game Engine to Simulate Flocking Behavior

(70 Points)

Daniel Hatakeyama , Ben Xiang, River Jordan

This interim submission must include a video recording demonstrating the current state of your code or project. **Expect to deliver at least 50% of the functionality of your code here!**

Deliverables

Your deliverable for Project 7 is a GitHub link with the project code so far. You must include:

- a PDF document in your repo – clearly labeled “Project 7 Update.pdf” that contains:
  - * a Status Summary
  - * UML Class Diagram
  - * Plan for completion
-

Status Summary (10 points):a video recording of the state of the application

Work Done:

Mock Up Complete:

- Completed minimum viable boid simulation
  - Works with strategy pattern for arbitrary behavior and render functions
  - Boids are implemented as inheriting game objects, -> need for change

Version 2:

- Building ECS pattern from scratch with scalable design
- So far entity system, entity manager, and base components are solidly implemented
- The design of the system / system manager is almost impossible to make a simple design. This is the largest concern at the moment. Each time we find a solution, it is not scalable or expandable long term. The current ideal solution to check all of our requirements of efficient runtime and loose coupling requires quite a bit of implementation, including multiple observers and listeners, memory

caching, refactoring components, and careful planning. This is where we currently are at and need to work hard to implement a solution.

Changes or Issues Encountered:

- Understand boids behavior and how to implement their behavior correctly via Processing canvas environment.
- Implementing the ECS pattern from scratch has been extremely difficult.
- Installing Processing 4 has been challenging to get it working on certain machines due to OS differences.
- Refactoring our current implementation to run at a more efficient way.

Patterns:

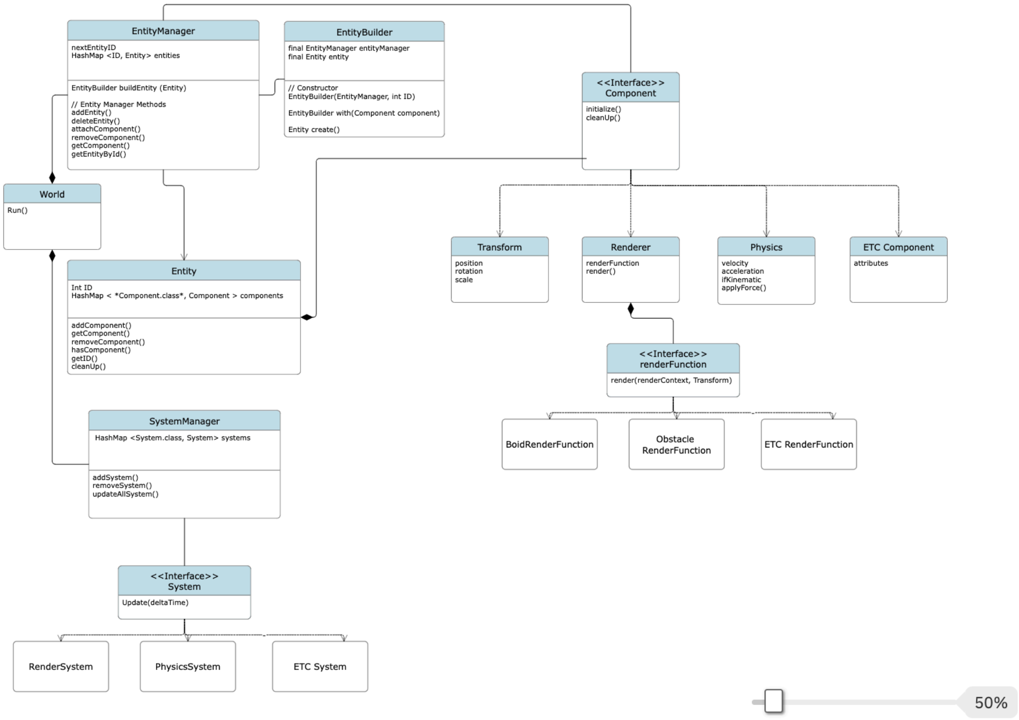
- Entity Component System - A design pattern not covered in our class
- Builder - EntityBuilder
- Extensive use of the strategy pattern
- Composition Pattern

Test Coverage Report:

- Professor Wright has given us the green light to use Processing and not strictly follow TDD.

Class Diagram (15 points)

A UML class diagram that shows all the classes and their relationships. Note which classes have been implemented and which still need to be created. Pattern use should be highlighted in this diagram.



BDD Scenarios (10 points)

You must include at least one scenario for each major function in your application. You do not need to cover all variations of these functions. Just one variation is sufficient. This will force your team to agree on exactly what you are building and have one specific scenario to work towards implementing.

You DO NOT need to implement any step definitions for these scenarios. Meaning, they don't have to be executable tests. Implementing the step definitions and running the tests will be extra credit in the final submission.

Major Components:

- Entity
    - Must act as a UUID with a collection of components, works by composition over inheritance
- EntityManager
    - Handles all entities and adds and removes components as needed.
- EntityBuilder

    - Uses builder pattern to compose entities into complex game objects easily
- System
    - Given a subset of gameobjects with components that match a query, performs behavior / action responsibilities on the component datums.
- Component
    - Data that attaches to an entity that gives it the behavior
- ComponentManager
    - Handles all data that is attached to the entity such as adding and removing data from an entity.
- PGraphics render contexts
    - Holds information about how the entity should be rendered such as the color of the entity.
- Main Game Loop
    - Uses the ‘world’ or ‘scene’ to render all entities.

Plan for Next Iteration (5 points)

Provide an estimate of how much work still needs to be done. What are your plans for the final iteration to get to the project delivery?

Plan:
Port boid code to the refactored ECS version code.

Recorded Demonstration (20 points)

The recorded video should be brief, 5 to 10 minutes; all team members should participate. Include the recording in your repo or provide an external link for viewing. Sections for the recording:

- Introduce all team members.
- Demonstrate your progress so far, showing the existing functionality.
- Identify the technologies and patterns used.
- Show some of your tests and at least two BDD scenarios.

Here is the video for our demo:
https://drive.google.com/file/d/1HbUOhYESRfGoQzbuEbhO3PCvrpVcYjmb/view?usp=sharing

NOTE: We did not show BDD because as we mentioned above, since we are using Processing as green lit by Professor Wright, he mentioned that it would be extremely difficult to create do BDD or TDD using processing and thus he understands our project and out situation thus said that we did not need to use BDD or TDD for our project.

Grading Rubric:

The point breakdown for Homework 7 is as follows:

Section	**Points**
Video Recording	20
Status Summary	10
Class Diagram	15
BDD Tests	10
Plan for next Iteration	5
Repository Submission with current code (includes required unit tests)	10
Total	70

Github Guidelines

Since this homework is a team assignment, only one repository submission is required per team. Hence, please do the following:

- Only one team member submits the repository link in Canvas.

Please do not make the repositories public since this violates the Honor Code. You will have to give access to this repository to all members of the CSCI 4448/5448 Team. Their GitHub handles are:

Shashankit Thakur	shashankit-thakur
Sudarshan Sridhar	SudarshanSridhar
Bhoomika, Prasad	bhpr6170
Puneeth Gante Hanumappa	puneeth04gh
Jinpeng Miao	JinpengMiao
Bill Wright	BillWrightCUProf

Overall Project Guidelines

Assignments will be accepted late for two days with a penalty of 10% for each day. After two days, assignments will be given a 0 and will not be graded.

Use office hours, e-mail, or Piazza to reach the class staff regarding homework/project questions or if you have issues completing the assignment.