Name\_Nathan Sawyer Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

I will be implementing the offense within Retro Hockey. I will be responsible for creating a functional offense system. This system will be able to pass the puck, skate in reasonable patterns, and shoot when the opportunity arises.

This system must have reasonable passing and skating such that the offense does not cause a quick turnover. The offense will also take into account where the defense is and adjust the movements accordingly. This system is crucial to the games functionality, and much of the “fun” the player has playing the game depends on the functionality of this system.

## Use case diagram with scenario \_\_14

[Use the lecture notes in class.

Ensure you have at least one exception case, and that the <<extend>> matches up with the Exceptions in your scenario, and the Exception step matches your Basic Sequence step.

Also include an <<include>> that is a suitable candidate for dynamic binding]

Example:

### Use Case Diagrams



### Scenarios

**[Decides where to skate]**

**Name:** Skate\_Direction

**Summary:** Calculate and decide the best play to make depending on the defense positions

**Actors:** Deffense, Offense, puck position, goalie position

**Preconditions:** Offense has possession of the puck

**Basic sequence:**

**Step 1:** Get current position and direction

**Step 2:** Get Defense position and Offense position

**Step 3:** Get Current “playcall”, evaluate if it still fits

**Step 4:** Decide where to skate next

**Exceptions:**

**Step 1:** Puck is traveling towards player: go towards puck

**Step 2:** Player collides with other player

**Post conditions:** Continues until turnover

**Priority:** 1\*

**ID:** C01

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.



**[Pass the puck]**

**Name:** Passes

**Summary:** Calculate and decide the best pass to make or to make a pass at all

**Actors:** Deffense, Offense, puck position, goalie position

**Preconditions:** Offense has possession of the puck

**Basic sequence:**

**Step 1:** Get current position and direction

**Step 2:** Get Defense position and Offense position

**Step 3:** Decide if user should pass or continue with puck

**Step 4:** Complete action

**Exceptions:**

**Step 1:** Passing lane blocked

**Step 2:** Puck intercepted

**Post conditions:** Continues until turnover

**Priority:** 2\*

**ID:** C02

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.



**[Shoot the puck]**

**Name:** Shoots

**Summary:** Calculate and decide the best shot to make or to make a shot at all

**Actors:** Deffense, Offense, puck position, goalie position

**Preconditions:** Offense has possession of the puck

**Basic sequence:**

**Step 1:** Get current position and direction

**Step 2:** Get Defense position and Offense position

**Step 3:** Decide if user should shoot or continue with puck

**Step 4:** Complete action

**Exceptions:**

**Step 1:** Shooting lane blocked

**Step 2:** Puck intercepted

**Post conditions:** Continues until turnover

**Priority:** 2\*

**ID:** C03

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.



**[Skate the puck]**

**Name:** Skate\_with\_puck

**Summary:** Calculate and decide to skate with the puck or not.

**Actors:** Deffense, Offense, puck position, goalie position

**Preconditions:** Offense has possession of the puck

**Basic sequence:**

**Step 1:** Get current position and direction

**Step 2:** Get Defense position and Offense position

**Step 3:** Decide if user should skate with puck

**Step 4:** Complete action

**Exceptions:**

**Step 1:** Defender blocks lane

**Step 2:** Defender steals puck

**Post conditions:** Continues until turnover

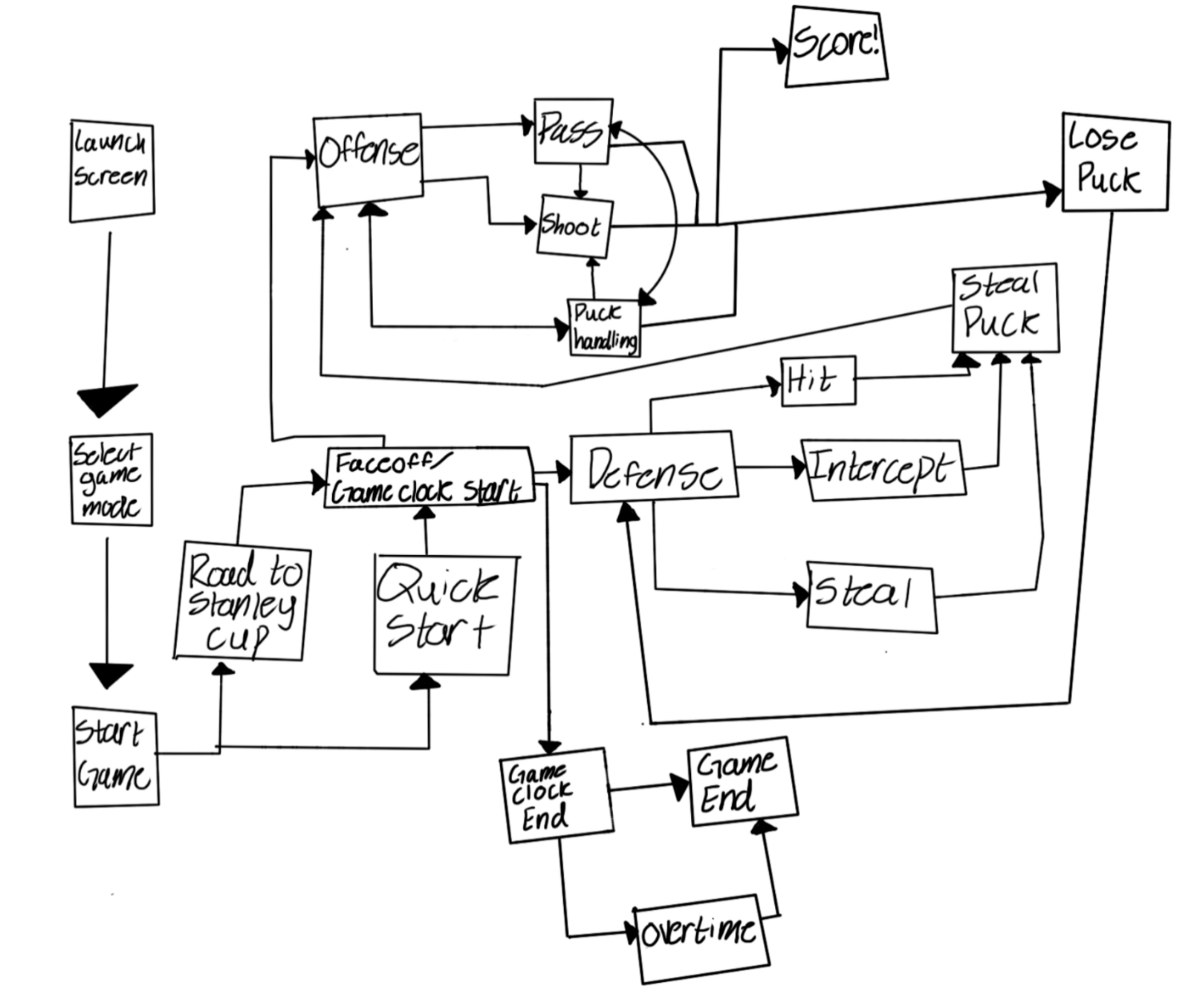
**Priority:** 1\*

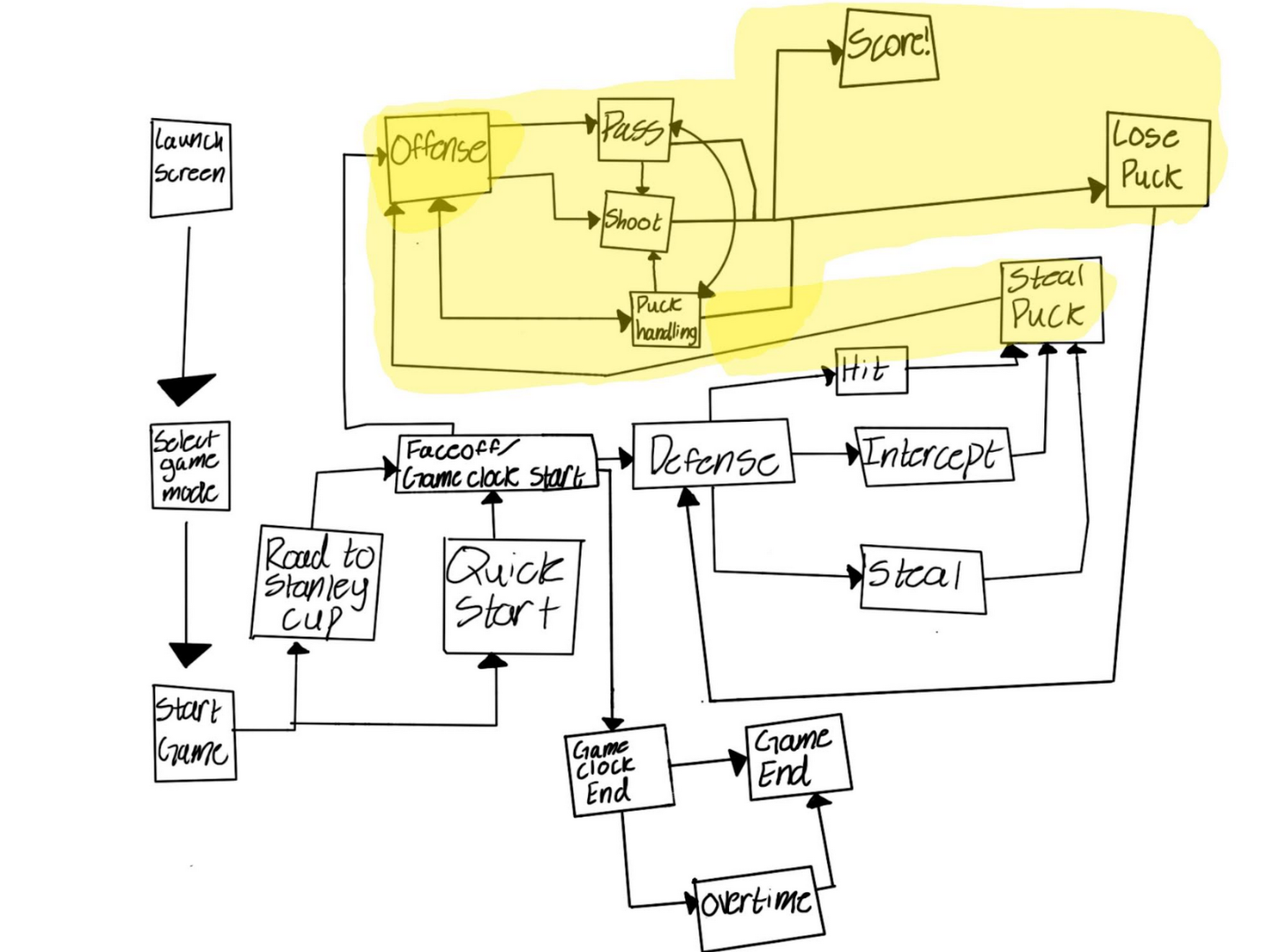
**ID:** C04

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

## Data Flow diagram(s) from Level 0 to process description for your feature

### Data Flow Diagrams





### Process Descriptions

On-Puck:

Pass If: No open net shot is possible, or defender is blocking path to goal

Shoot if : Shoot if there is clear path to goal and player is close

Else: Skate with puck to differing position

Off-puck:

If playcall not decided:

If: player positioning is in man coverage, send offense in a crossing pattern to get free

If : defense is in a zone coverage, skate to “holes” in zone

While: defender is blocking path, skate around defender

If two or more players get blocked, switch playcall

If: puck is passed:

Closest player head towards the puck

If : puck is shot and deflected

All players “crash” on puck

## Acceptance Tests \_\_\_\_\_\_\_\_9

Offense must be able to score at a bare minimum of 5% of the time without player interaction. If the offense is turning over the puck consistently, this would make the game extremely difficult and frustrating to play.

Offense must prioritize passing back to the user when the user is open. This must happen at a minimum of 50% of the time when the user is uncovered.

Offense must pass the puck with an average of 2 times per possession at a minimum. In a fast break, this clearly would not be the case, but those should not be too common.

Defenders must be avoided and cannot be skated straight into. No more than 20% of puck positions should end in big collisions

**Inputs vs Outputs**

| Input | Output |
| --- | --- |
| Puck passed | Puck Received |
| Defender blocking route | Defender avoided |
| Defender blocking offender | Defender avoided |
| Offense receives puck | Offense scores > 5% |
| User is open | Puck is passed > 50% |
| Offense receives puck | Offense passes puck > 2 times (excluding fast breaks) |

## Timeline \_\_\_\_\_\_\_\_\_/10

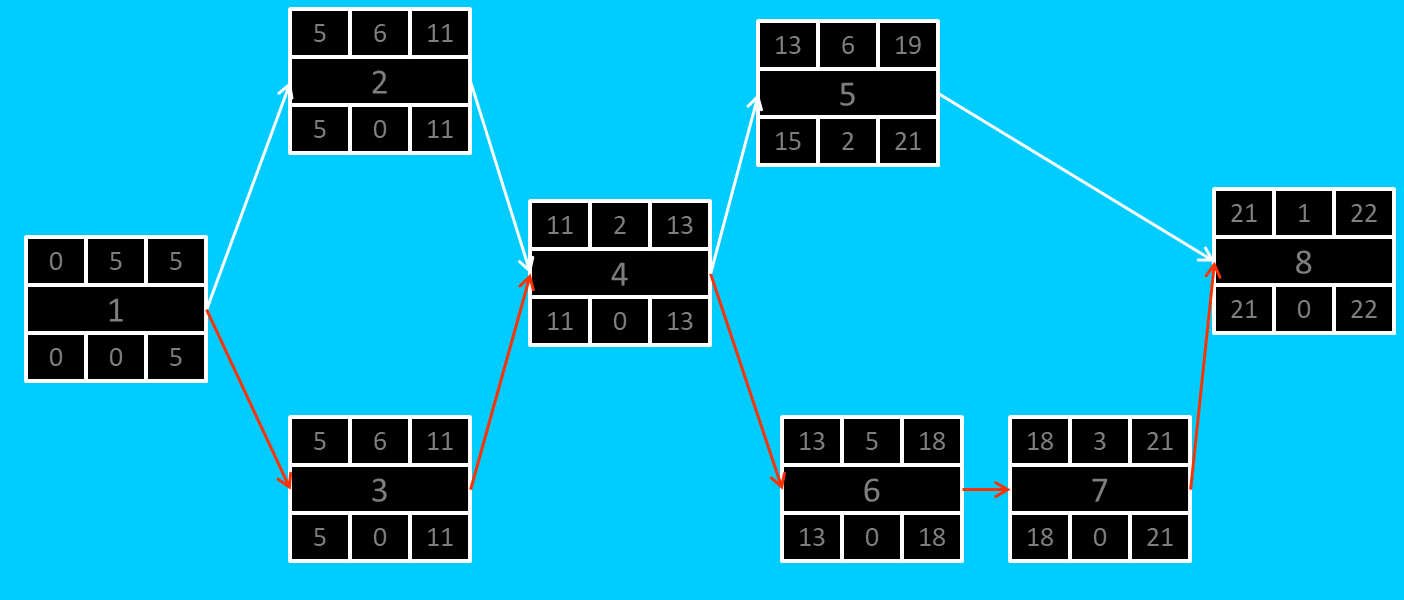
[Figure out the tasks required to complete your feature]

Example:

### Work items

| Task | Duration (PWks) | Predecessor Task(s) |
| --- | --- | --- |
| 1. Define actions and workflow | 1 | - |
| 2. Skating logic | 6 | 1 |
| 3. Passing Logic | 2 | 2 |
| 4. Shooting logic | 2 | 2, 3 |
| 5. Documentation | 1 | 4 |
| 6. Testing | 4 | 4 |
| 7. Bug Fixing | 6 | 6 |
| 8. Finishing Touches | 1 | 5, 7 |

### Pert diagram



Gant timeline:

Key = 1 = hours spent working

| 1 | 1 | 1 | 1 | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 |  |  |  |  |  | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  | 1 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  | 1 | 1 |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  | 1 | 1 | 1 | 1 |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 | 1 | 1 | 1 | 1 |
| 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |