MICHIGAN STATE UNIVERSITY

Project Plan

Continuous Improvement of Boeing Assembly Lines

The Capstone Experience

Team Boeing

Ross Blakeney

Dave Grabowski

Sean Heider

Kyle Kotulak

Department of Computer Science and Engineering

Michigan State University

Fall 2013

From Students…
…to Professionals

Project Overview

- Create a 3D simulation of a Boeing assembly line.
- Compile important data about the construction process.
- Use this data to optimize the design of the assembly line, improving safety and efficiency.

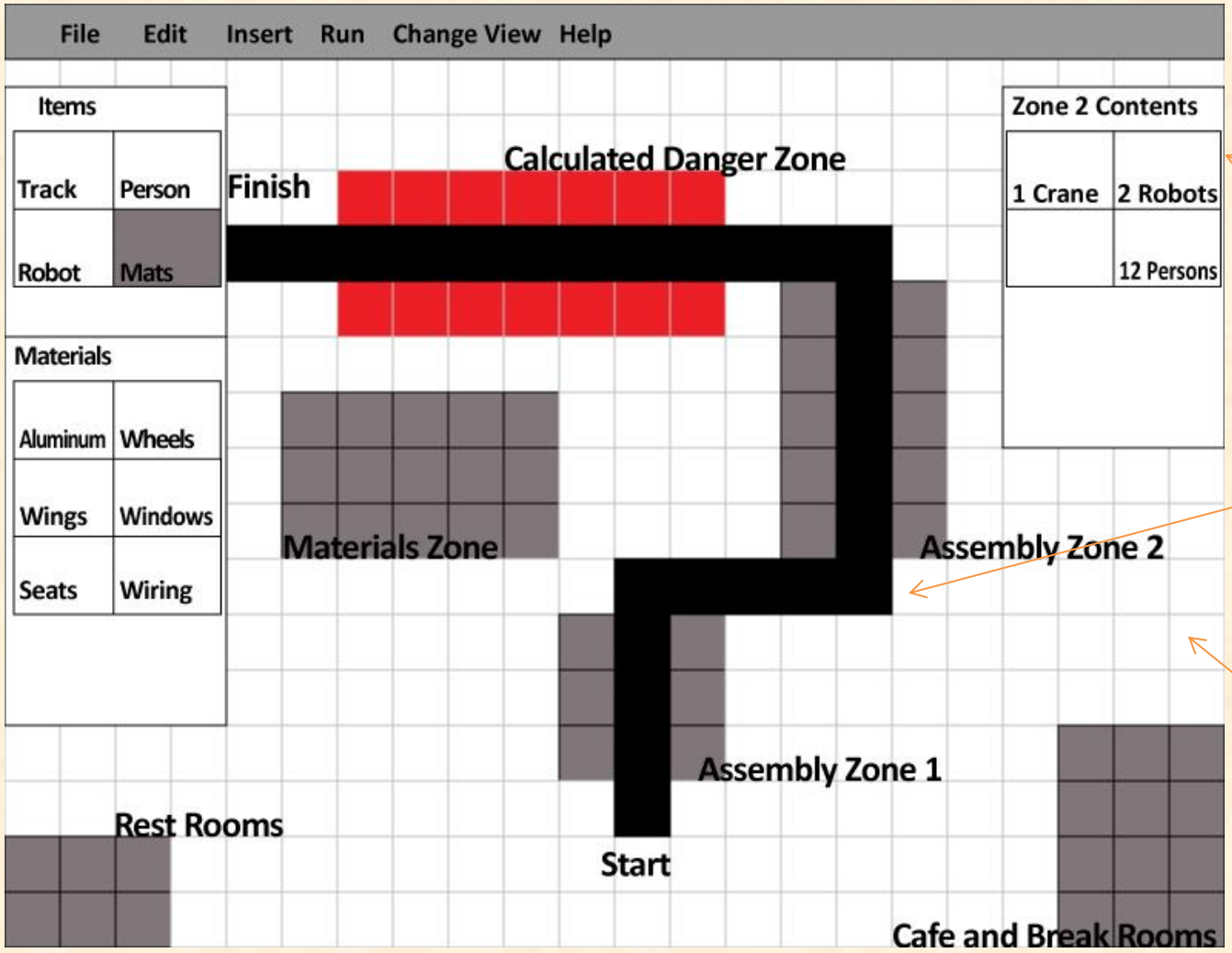
Functional Specifications

- Simulate and monitor a Boeing assembly line
  - Teams of people and robots working together
  - Multiple levels of construction
    - Beneath, above, and inside the aircraft
- Simulate realistic limitations of the workers
  - Limited sight distance
  - Limited hearing distance
  - Fatigue
  - Breaks / idle time
  - Walking distances
  - Etc…
- Must analyze the simulation metrics to identify:
  - Safety concerns
    - Dangerous situations for workers
    - Time spent in dangerous situations
  - Assembly line efficiency
    - Overall Idle time of assembly zones
    - Idle time of individuals and robots

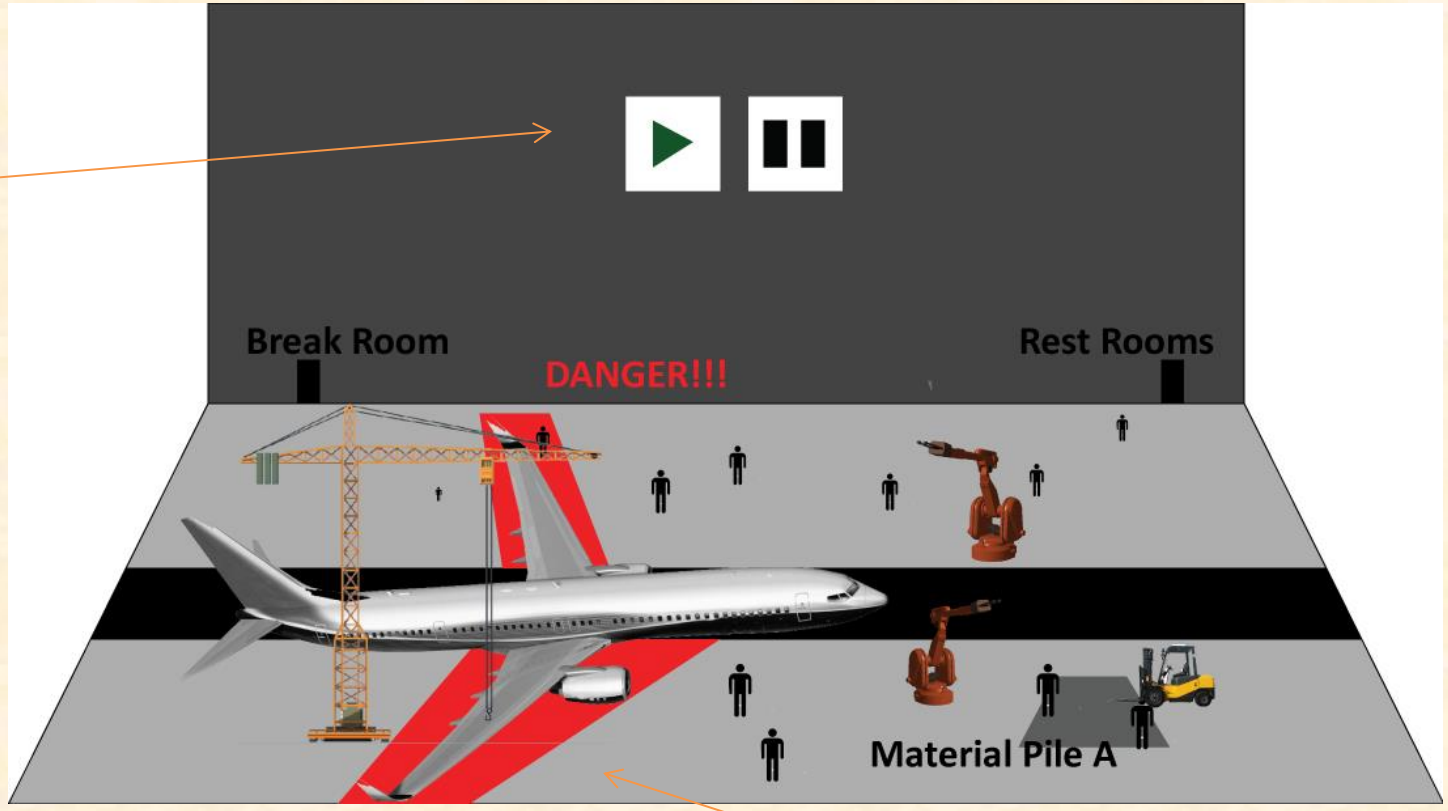
Design Specifications

- 3D Graphical Visualization
  - Top-Down View
    - Must include:
      - ❖ Grid based system for placing the various modules and zones that will make up the assembly line
      - ❖ GUI for selecting and placing the various modules
  - Free-Range Third Person View
    - Must include:
      - ❖ Dynamic information display based on current location along the assembly line.
        - » Current percentage of work completed
        - » Safety concerns
  - Free-Range First Person View
    - Same requirements as the third person view but from a higher perspective
- Quality metrics
  - Must be used to analyze safety and efficiency
  - Must identify specific points of danger in the line

Screen Mockup: Construction View

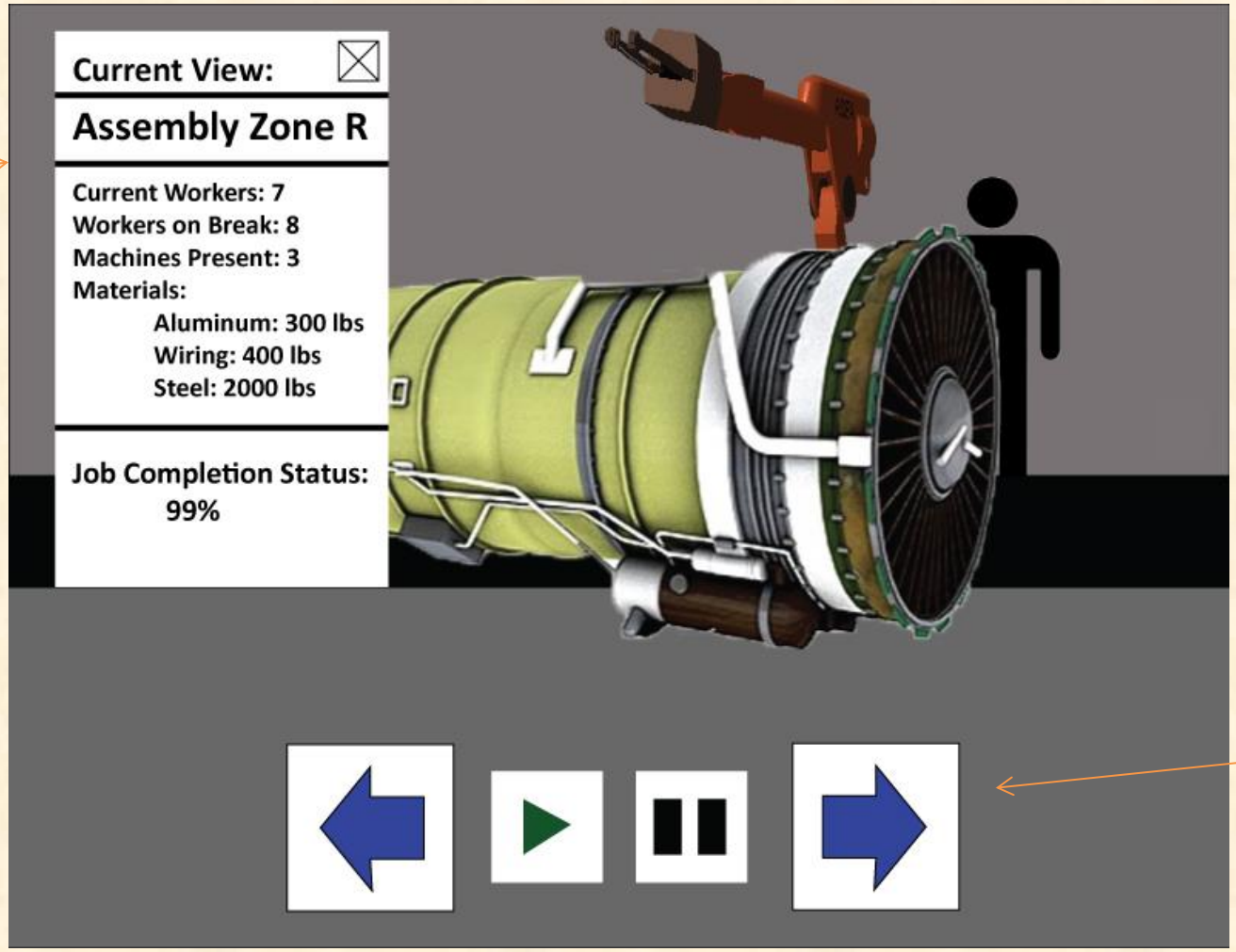


Screen Mockup: Third Person View



Screen Mockup: First Person View

Zone Contents and Job Completion Menu



Simulation Controls

Screen Mockup: Metrics Breakdown

Simulation Metrics

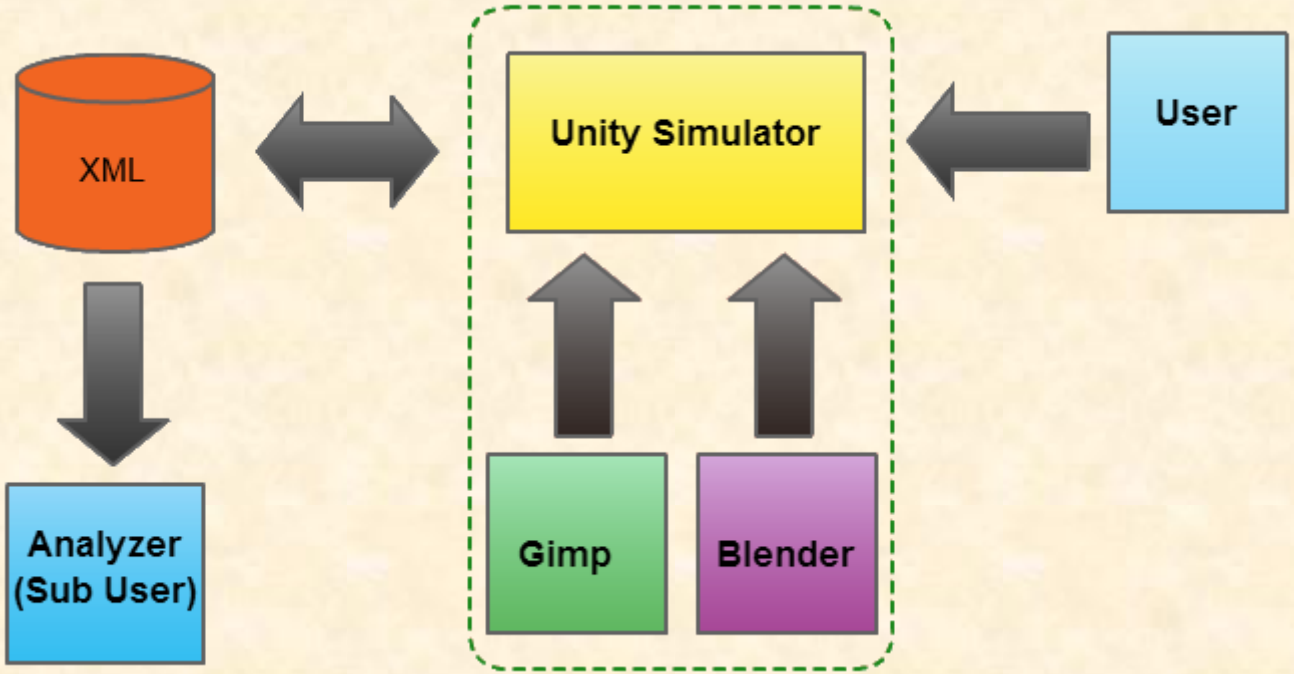
Technical Specifications

- Software Technologies
  - Unity
  - Blender
  - Microsoft Visual Studio 2012
- Development Tools and Methodologies
  - C#
  - XML
  - SQL
  - Test Driven Development
  - Code Reviews
  - Inverse Kinematic Algorithms

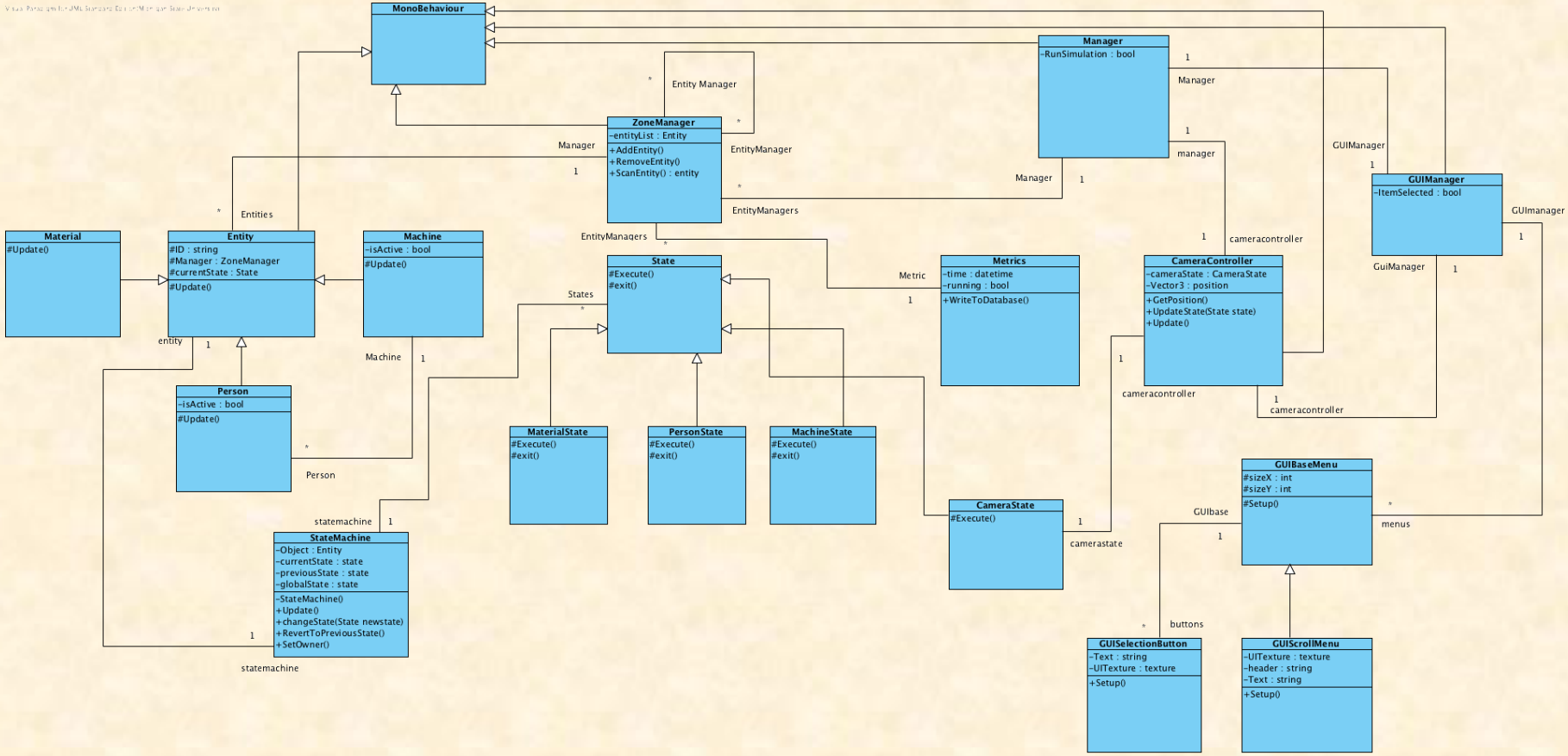
Technical Specifications – continued

- Schedule:
  - 8/27 – 9/15
    - Initial Setup
    - Initial Research
    - Initial Documentation
  - 9/16 – 9/22
    - Prototyping
    - Specifically Research Assembly Lines
    - Implement Initial Project Layout (skeleton)
  - 9/23 – 10/14
    - Development/Testing
    - Alpha
  - 10/15 – 11/11
    - Development/Testing
    - Beta
  - 11/12 – 12/2
    - Development/Testing
    - Release
  - 12/6
    - Design Day

System Architecture



UML Diagram


MonoBehaviour
Manager
-RunSimulation : bool
Entity Manager
ZoneManager
-entityList : Entity
+AddEntity()
+RemoveEntity()
+ScanEntity() : entity
Manager
EntityManager
EntityManagers
GUIManager
-ItemSelected : bool
GuiManager
GUImanager
Material
#Update()
Entity
#ID : string
#Manager : ZoneManager
#currentState : State
#Update()
Entities
Machine
-isActive : bool
#Update()
State
#Execute()
#exit()
States
Metric
Metrics
-time : datetime
-running : bool
+WriteToDatabase()
CameraController
-cameraState : CameraState
-Vector3 : position
+GetPosition()
+UpdateState(State state)
+Update()
cameracontroller
entity
Machine
Person
-isActive : bool
#Update()
MaterialState
#Execute()
#exit()
PersonState
#Execute()
#exit()
MachineState
#Execute()
#exit()
CameraState
#Execute()
camerastate
statemachine
StateMachine
-Object : Entity
-currentState : state
-previousState : state
-globalState : state
-StateMachine()
+Update()
+changeState(State newstate)
+RevertToPreviousState()
+SetOwner()
GUIBaseMenu
#sizeX : int
#sizeY : int
#Setup()
GUIbase
menus
buttons
GUISelectionButton
-Text : string
-UITexture : texture
+Setup()
GUIScrollMenu
-UITexture : texture
-header : string
-Text : string
+Setup()

UML Diagram – continued

- The Zone Manager has many entities within it, each with their own State Machine
- The Manager of the program has many Zone Managers and allows for communication between them
- Entities are divided into three derived classes, person, material, and machine
- The Camera Controller class will be in charge of handling input for movement and will contain an instance of the GUI manager class

System Components

- Hardware Platforms
  - PC
- Software Platforms / Technologies
  - Windows
    - Unity
  - Mac
    - Unity
    - Blender

Testing

- Nunit
  - Testing framework for C#
- Uunit
  - Testing framework for Unity
- Test Driven Development
  - Red – Green – Refactor

Testing: Red – Green – Refactor

- Red
  - Write a test case so that it fails (the functionality is not yet implemented)
- Green
  - Write the code so that the test case passes
- Refactor
  - Clean up redundant and spaghetti string code

Risks

- Purchasing of Unity License
  - We will require access to Unity Pro, which will require the purchase of a license after 30 days.
  - $129 license available through http://www.studica.com/unity
- Familiarity with the concept of Inverse Kinematics
  - Wikipedia: **Inverse kinematics** refers to the use of the kinematics equations of a robot to determine the joint parameters that provide a desired position of the end-effector.
  - Become more familiar with the concept of inverse kinematics.
- Knowing which metrics to measure
  - There are hundreds, possibly even thousands of factors that go into measuring safety and efficiency on an assembly line
  - Speak with Jayson, and decipher which metrics are the most relevant, and which aren't
- GUI for Unity
  - We are trying to use Unity, a game developing tool, to create a useful simulation "game", however limitations with unity's built-in UI functionality will force alternate approaches to be considered.
  - Figure out if it's possible, and if it's not, change our approach