Alex Speasmaker Ben Mays Kyle Jones Omeed Ashtiani

**Software Requirements Specifications (SRS) for a Rigid Body Simulator**

**1. Introduction**

**1.1. Purpose**

1. The purpose in writing this document is to specify the requirements for a simple physics and game engine.

2. The current intended audience is the software engineering team and developers to view various physics simulations.

**1.2. Scope**

The project, Atlas, is a physics and game engine. The objective of this project is to simulate the physics and gravity between worlds and other objects in space. This involves moving the objects, giving them forces, and creating new objects. For this to occur, physics of rigid bodies (involving Kinematics, rotation, energy) must be available, along with collision detection. Atlas will be used as a base to build simple games with 2D graphics.

**1.3. Definitions, Acronyms, and Abbreviations**

2D = 2-dimensional

ccw = Counter Clockwise

cw = Clockwise

\* = any

**1.4. References**

1. IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.

**1.5. Overview**

1. The SRS is divided into three main sections: Introduction (Section 1), Overall Description (Section 2), and Specific Requirements (Section 3).

2. Section 2 contains an overall description of the physics simulator. It includes the Product Perspective (Section 2.1) which describes how this simulator compares to other simulators and game engines. Product Functions (Section 2.2) outlines the primary functions of the software. User Characteristics (Section 2.3) provides the reasoning behind the general user requirements that are specified in section 3. Constraints (Section 2.4) specifies any regulatory, system, user, or other restrictions that affect the software development. The Assumptions and Dependencies (Section 2.5) section lists any assumptions or dependencies that were made about the user operating environment and its capabilities. Apportioning of Requirements (Section 2.6) sub-section lists any requirements that may be implemented in later versions.

3. Section 3 contains the specific requirements of the software. The External Interfaces subsection (Section 3.1) gives the details of any software interface communications between the software and external software or systems. Functional Requirements are defined in section 3.2 as a list of requisite operations for the software. In the Performance Requirements section (3.3) are the calculable software or user related performance requirements. Specification of the types of information used and stored in the database is described in the Logical Structure of Data section (3.4). The Design Constraints (section 3.5) lists any known standards, or hardware restrictions that affect website development. Software System Attributes section (3.6) describes the attributes of good system software and the details of how they relate to the physics simulator.

**2. Overall Description**

**2.1. Product Perspective**

This product is completely standalone product.

The physics simulator will use pyglet as a low level abstraction over OpenGL for graphics rendering. Pyglet will also handle window and event management.

**2.2. Product Functions**

This subsection explains the various product functions.

**2.2.1 World**

**The system shall provide a world in which the player may move and interact.**

Rationale: the world is the environment within which all rigid body physics and user interactions will be staged.

**2.2.2. Scene Translation**

**The system shall allow the user to view the world freely.**

Rationale: free movement is a necessary element to observe everything that is happening in the system.

**2.2.2. Interactions**

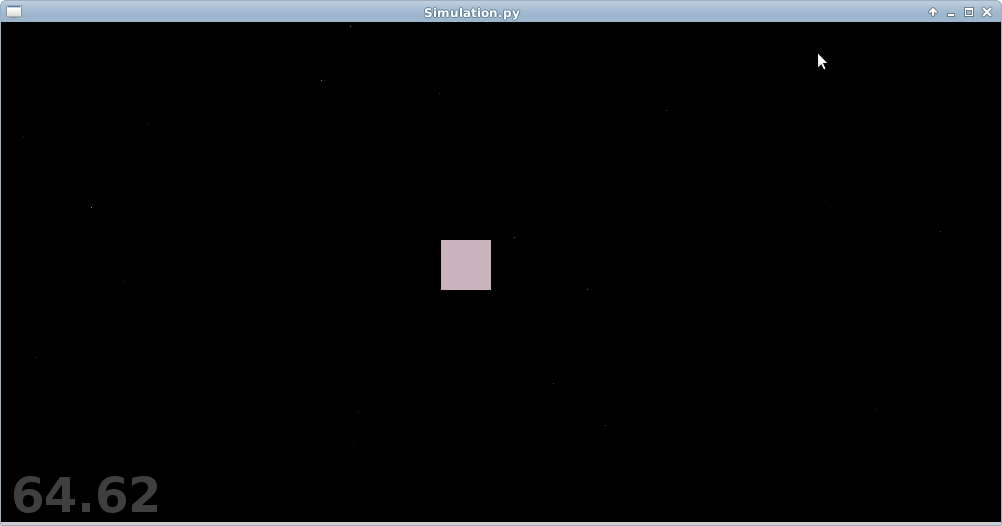
**The system shall allow the user to freely interact with other rigid bodies.**

Rationale: interactions with the environment provide visual feedback and a visual representation of rigid body simulations

**2.2.3. Rigid Body Physics**

**The system shall simulate rigid body physics on all rigid bodies.**

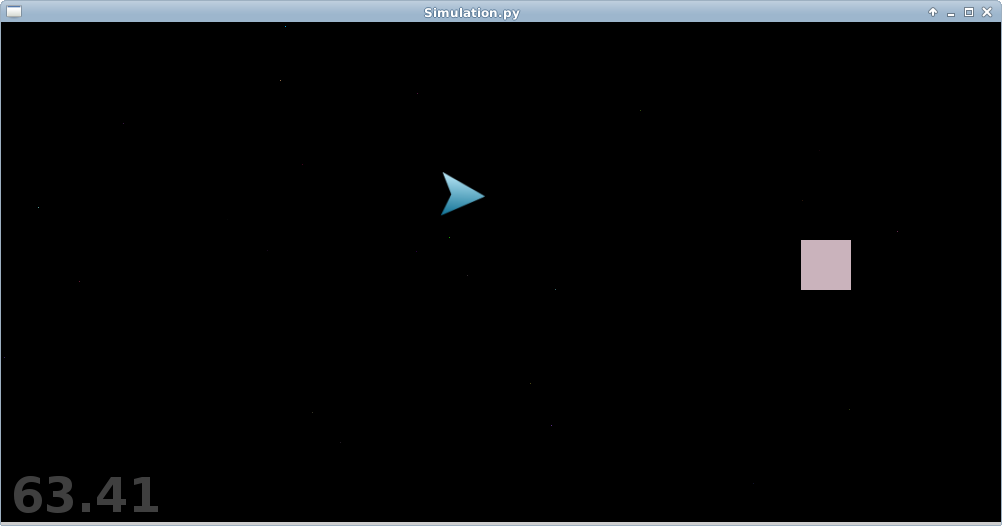
Rationale: rigid body physics simulation is the overall goal of the system and will be the main feature.



**2.2.4. Player Sprite**

**The system shall have a moveable sprite that is controllable by the user that follows the physics rules of the system.**

Rationale: Having a moveable sprite that follows the systems physics rules allows the user to view how the system would act on the objects they wish to add or move.



**2.3. User Characteristics**

The expected user should be experienced using computers and simple video games. If the user wishes to build upon the engine, they should be experienced with the Python programming language.

**3. Specific Requirements**

**3.1.2. User Interfaces**

**3.1.2.1 Display**

The system shall be displayed on a window in which the world and all character interactions and rigid body simulations will take place.

**3.1.2.2 Keyboard**

The system shall be manipulated using the previously specified keyboard commands. These commands may change the view, state, or both of the world.

**3.1.2.3 Mouse**

The system shall be manipulated using the previously specified mouse. The mouse shall be used for direct state manipulation of the world.

**3.2. Functional Requirements**

**3.2.1. Stimulus: Simulator start**

1. The system shall generate the world and all objects for rigid body physics simulation.

2. The system shall play background music.

**3.2.2. Stimulus: Translate Scene**

**3.2.2.1. Stimulus: Press UP, LEFT, DOWN, or RIGHT**

1. The system shall allow the user to translate the scene using “UP”, “LEFT”, “DOWN”, or “RIGHT.

2. The system shall allow the user to translate the scene upward when “UP” is pressed.

3. The system shall allow the user to translate the scene to the left when “LEFT” is pressed.

4. The system shall allow the user to translate the scene downward when “DOWN” is pressed.

5. The system shall allow the user to translate the scene to the right when “RIGHT” is pressed.

**3.2.3. Stimulus: Object interaction**

1. Based on the conditions of one object interacting with one or more objects the system shall appropriately simulate the interactions based on the rules for rigid body physics.

**3.2.4. Stimulus: Move Player**

**3.2.4.1 Stimulus: Press W, A, or D**

1. The system shall allow the user to move the player sprite using the “W”, “A”, “D”.

2. The system shall allow the user to rotate the player sprite ccw when “A” is pressed.

3. The system shall allow the user to move the player sprite cw when “D” is pressed.

4. The system shall allow the user to move the player sprite forward when “W” is pressed.

**3.2.5 Stimulus: Click**

1. The user shall be able to select the type and size of the object to be inserted using a menu.

2. The system shall allow the user to place an object on screen at mouse location.

3. The user shall be able to ‘throw’ rigid bodies, using the velocity of the mouse pointer, to interact with the simulation.

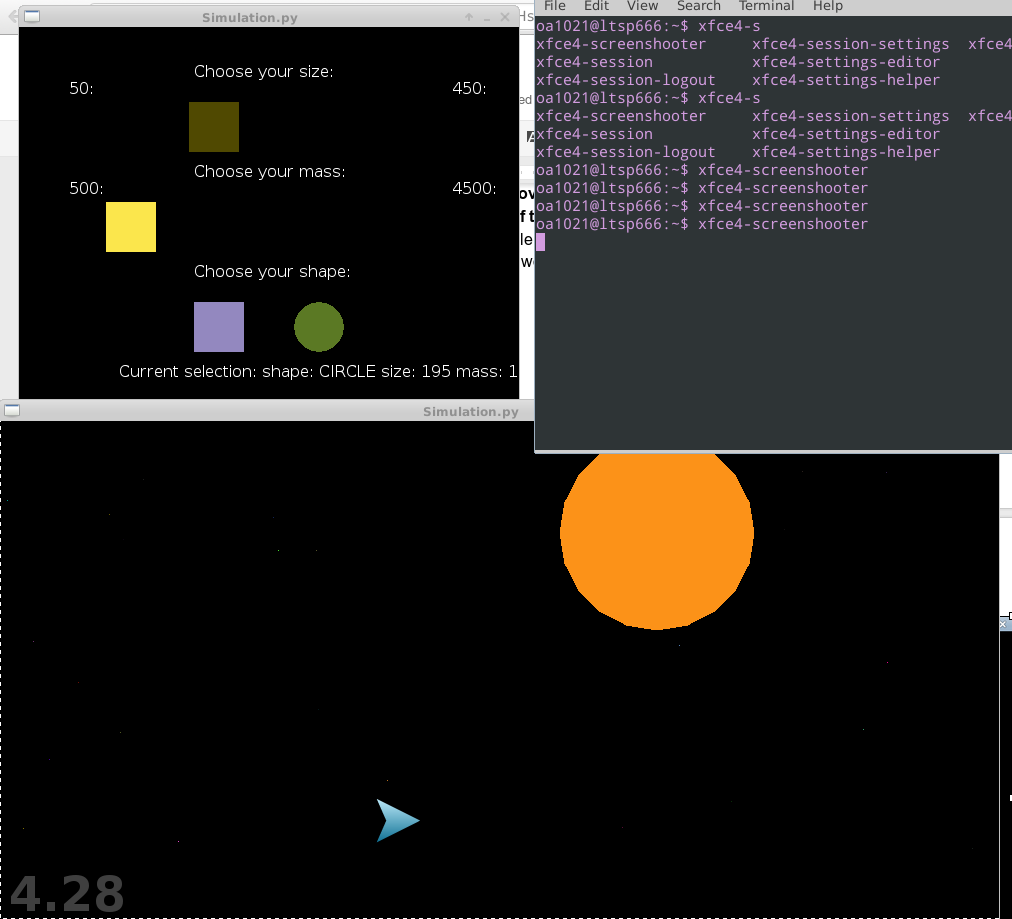
**3.2.6 Stimulus: Press escape**

1. The user shall be able to press the escape key to access a menu which will change the type of entity created on click stimulus.

2. The simulation will halt while the menu is open.

3. If the escape key is pressed while in the menu, the menu will close, and the simulation shall resume.

3. Settings from the previous menu shall be retained if the menu is opened again.



**3.3 Performance Requirements**

The system shall support one instance and user at a time. Interactions shall be processed before the next time the physics state is updated each 1/60 second.

**3.4 Logical Database Requirements**

There are currently no database requirements.

**3.5 Design Constraints**

There are currently no design constraints.

**3.6 Software System Attributes**

**3.6.1 Reliability**

The system should be able to handle physics calculations within the 1/60 second update period.

**3.6.2 Availability**

The system should be available at all times when required platforms are installed.

**3.6.3 Security**

The system does not handle any sensitive information, so there is no encryption on any data. The system inherits the security of all programs running on the Python interpreter.

**3.6.4 Maintainability**

The system is adaptable to changes, and features a modular design to aid in maintainability.

**3.6.5 Portability**

The system may be ported to any computer capable of running Python 2.7.\* and pyglet.

**4. Design and Implementation Constraints**

**4.1 Requirement**

The simulator shall be run using Python 2.7.\* and the pyglet library installed on the system on which it is run.

**4.2 Constraints**

The software must work at a framerate that does not interfere with the simulation of rigid body physics.

**5. Resources**

**Source:** Essential Mathematics for Games, ..., pg 601

**Source:** Game Physics, Eberly, David H., 2010