

# Building Security Simulator

## Requirements Definition Document

### Version 1.2

#### Project Members

- Matthew Fulgo
- Lane Haury
- Craig Szczesiul
- Tal Ron

#### Table of Contents

- A. Core Idea
- B. Essential Project Features
- C. Peripheral Project Features
- D. Inter-Group Commonality
  - 1. Infrastructure Group
  - 2. Island Simulator
  - 3. Traffic Simulator
- E. Functional Requirements
- F. Non-Functional Requirements
- G. Peripheral Functional Requirements
- H. Peripheral Non-Functional Requirements
- I. Glossary

#### Core Idea

The Building Security Simulator is a program designed to realistically imitate a competition between security guards and intruders. The simulation consists of a model of a building, its security features, and the resources and characters within the building. Players can control characters within the simulation environment in order to complete a given objective. These objectives can vary among simulations as determined by the environment creator. This program could be extended to train personnel for emergency situations such as medical or fire response.

This approach is most useful for evaluating the security of a building. The program can also potentially train people for building security and infiltration. Given a set of in-game resources, players can evaluate the effectiveness of a building's security system. Furthermore, the simulation can build teamwork and promote coordination among players.

Characters will accomplish their objectives through interactions with the simulation environment. These interactions include movement, communications, resource management, and coordination. Ultimately, the players' decisions and actions will result in their team's success or failure.

## **Essential Project Features**

At its essence, the simulation consists of a two-dimensional map/model of a building with sensory data for guards and intruders. Within the environment, sensors assist the characters in achieving their objective(s). Additionally, models will represent physical objects within the simulation environment. These will minimally include: doors, cameras, boxes, windows, keys, models for people (at least one for each team), walls, and stairs.

## **Peripheral Project Features**

Ideally, the simulation will be a three-dimensional environment in which characters can interact with all physical objects. The ability to modify the security system (e.g. Sensors, camera locations, locks) would enable the dynamic improvement of the modeled system and present more challenges for players. Additionally, players can modify the environment's physics model before each round.

## **Inter-Group Commonality**

There are several aspects of our program that may be similar to both the Island and Traffic Simulators. All three simulators will require a map which can contain other objects and be visually represented. Additionally, we will need a flexible time model (with 1:alpha mapping) to allow the server to set the simulation speed.

## **Infrastructure Group**

We require an infrastructure with a two-dimensional visual representation of a mapped environment. A client-server platform capable of object- and message-passing must be provided. The client must allow the display of the building, objects, and characters within the environment. Additionally, the platform must detect collisions of objects and/or characters (i.e. two objects/characters intersecting). The framework must include a customizable, extensible physical model for the simulation. This model would minimally include the following features: gravity, friction, and basic motion.

Illegal movements, such as walking through walls, characters, or other impassable objects, should be prevented by the framework.

## **Island Simulator**

The Island Simulator will share several features with the Building Security Simulator. The collection/capture of sensory data within the simulation environment are required by both programs. (We refer to this as the detection of actions: sounds and movement.) Secondly, the environments will contain physical objects with which characters can interact. Lastly, characters must have the capability to move within the environment, while adhering to physical laws (e.g. two objects cannot occupy the same location).

## **Traffic Simulator**

Shared features between the Traffic Simulator and the Building Security Simulator concern time. Both programs require the ability for random, scheduled, or recurring events. Furthermore, both simulators need a representation of the time of day. It is conceivable that the Island Simulator also requires these features.

## Functional Requirements

The simulation must maintain a list of characters and objects and data specific to each of them. This includes:

- character status
- location in map coordinates
- environmental information the character can currently access

The following data must be transferred from the client to the server over a network protocol:

- location updates
- actions

The following data must be transferred continuously (streamed) from the server to the client over a network protocol:

- location and status updates of objects and characters within the character's sphere of influence.
- sensor data to which the character has access.
- actions marked as required for the given client

The following data must be transferred once from the server to the client at the initialization of each round:

- map information
- object list
- properties for each object and character

Actions must be declared as either required or not-required to be sent to the client. Public actions are defined as those which have a sensory effect on the game environment. Private actions, such as viewing a camera monitor, do not potentially affect other characters or the game environment. Universal actions are marked as required for distribution to every character.

Actions will include:

- opening, closing, locking, and unlocking doors
- changing the location of movable objects
- acquiring keys (including the stealing of keys from guards)
- watching camera monitors
- walking
- communicating

Actions can cause

- sounds, which have a location and magnitude and may be represented visually
- visual cues
- changes in character status
- changes in object properties (e.g. opening doors, moving boxes, blocking hallways)

Actions should also have limitations (e.g. can't open a door without an access card)

All actions within a character's sphere of influence must be relayed to that client.

The server will process sensor data in order to send that data to clients designated for receiving that information.

Characters must be able to access sensor data by using monitors.

- Use of monitors will be limited by a characters proximity to the monitor
- Camera monitors will grant players a view of an area.
- Monitors for motion sensors and infrared beams will convey a map location if triggered
- Monitors will be configured to determine which sensors' data they will show.

Players must input control of character actions to the game. Players will control character movement, communication, and interactions with the environment and each other. This control will be achieved through human input devices such as a keyboard and mouse.

Characters must be able to move throughout the environment: they must be able to rotate left and right and move forward, backward, left, and right. Characters must be able to ascend and descend stairs, escalators, and elevators.

Objects may not move through other objects.

Characters must be able to acquire core resources. These include:

- keys / access cards
- radios (peripheral feature)

Characters will be able to communicate with each other through a chat mechanism, subject to constraints on the mode of communication used. These include:

- talking (default)
- text chat (peripheral)
- shouting (peripheral)
- radios (peripheral)
- whisper to nearby characters (peripheral)

Include support for time-triggered events. Characters can initiate a one-shot timer for the following events:

- police arriving – the level of response/number of police characters will be determined by the properties of the map
- fire alarms (peripheral)

Maps shall contain sensor locations and orientations for the start of each round.

Intruders will have a set of objectives in each round. "Victory" is obtained upon the fulfillment of those objectives.

Sensors and characters must be able to detect the following events:

- sounds
- movement

Each character starts a round with a base set of objects. These objects will be defined by the map. These include:

- keys / access cards
- flashlight (peripheral)
- radio (peripheral)

Each player's client display will indicate/list what objects their character currently carries.

The server shall be an instance of the platform that contains all of the data in the simulation and does not allow direct control by players.

## **Non-Functional Requirements**

Passage of time uses a 1:alpha mapping

Maps shall have realistic sensor quantities and placement.

Sensors shall model real-life, having limitations on use/abilities, failures, sensitivity levels, and the potential for bypass.

Guards shall not know the intruders' objectives.

Characters' actions govern what information the character can access. This information will have realistic limitations based on:

- field of view
- sound propagation
- map location
- proximity to objects

## **Peripheral Functional Requirements**

- Report statistics at the end of the round to help evaluate the guards' performance (high priority)
- Sensors/objects must be placeable for a map/scenario
- Aspects of players (health, movement speed, etc.) must be able to be changed within the game
- An inventory system of what players can carry / are carrying
- Players can trade inventory items with each other
- Model night and day situations
- Guards may have a prioritized list of things to guard and/or objectives
- If any character in the game is not controlled by a player, it will be controlled by a component that handles artificial intelligence (A. I.). The A. I. characters will have all the capabilities and limitations of player-controlled characters (e.g. an objective, sphere of influence, performing actions).

## **Peripheral Non-Functional Requirement**

Limited funds should be given to the players (guards and intruders) to reflect real-world budgets for sensors (cameras, door alarms, etc.)

# Glossary

## Action

Happenings within the simulation that are directly caused by characters. Interactions with physical objects, the environment, or characters (e.g. walking, opening doors, and arresting intruders).

## Character

Representation of a person within the game environment. A guard, intruder, etc.

## Client

The program a player uses.

## Event

Happenings within the simulation that are not directly caused by a character's action

## Guard

Characters that attempt to detect, oppose, and capture or otherwise incapacitate the intruders.

## Intruder

Characters whose objective(s) are given at the beginning of each round, usually involving destruction, theft, or other crimes.

## Monitor

An in-game object used to relay sensor information to a character.

## Objective

A goal presented by the game to a specific character or group of characters at the beginning of each round.

## Player

A Real Life™ person.

## Round

The period of game time beginning when objectives are given to the intruders and ending at the completion or failure of these objectives.

## Sensor

An object within the game environment that captures information and is processed by the server.

## Sphere of Influence

The spherical radius within which a character or sensor can detect events.

## Status

An enumeration of a character's state, which defines their abilities and limitations (i.e. active/normal, incapacitated, captured, dead).

## Attachments

- [Requirements.odt](#) (19.0 kB) - added by fulgo on 02/11/07 18:35:44.
- [Requirements.pdf](#) (73.4 kB) - added by fulgo on 02/11/07 18:36:15.