Bear Hockey

Game Functional Requirements

Professor: Nik Brown

Email: nik@ccs.neu.edu

**Bear Hockey Game**

Bear Hockey, is a variant of Air hockey with an air-hockey table, two player-held mallets, and a puck. The difference is the puck can rotate and has two fangs. If a player’s mallet touches the pucks fangs, the sleeping bear is aroused and devours him; otherwise it is identical to air hockey.

The illustrations below are from [http://theoatmeal.com](http://theoatmeal.com/)





**Contents**

[1 Requirements 3](#_Toc280952741)

[1.1 User Information 4](#_Toc280952743)

[1.2 Game Scoring/Rounds 4](#_Toc280952743)

[1.3 Dance Twerking Art/Animations 4](#_Toc280952743)

[1.4 Twerk Controller 4](#_Toc280952743)

[1.5 Single Player AI 4](#_Toc280952743)

[1.6 Robot Customization 4](#_Toc280952743)

[2 Physical Architecture 3](#_Toc280952741)

[2.1 iPhone 3](#_Toc280952742)

[2.2 iPad 4](#_Toc280952743)

[2.3 C++ 5](#_Toc280952747)

[2.3.1 Xcode 5](#_Toc280952748)

[2.3.2 Cocos2d-x 5](#_Toc280952748)

[2.4 Linux Server 6](#_Toc280952749)

[2.4.1 Email Service 6](#_Toc280952750)

[2.4.2 User Login 6](#_Toc280952750)

[2.4.3 Playing with Friends (Real-Time Network Updates) 6](#_Toc280952750)

[2.5 Communication Protocols 6](#_Toc280952751)

[2.5.1 HTTP 6](#_Toc280952752)

[2.5.2 TCP 6](#_Toc280952753)

[2.5.3 Boost.ASIO 6](#_Toc280952753)

[2.5.4 OpenSSL 6](#_Toc280952753)

[2.5.5 Encryption 6](#_Toc280952753)

[2.6 Database Model 7](#_Toc280952754)

[2.6.1 MySQL 6](#_Toc280952753)

[2.6.2 iPhone Data 6](#_Toc280952753)

[2.6.3 iPad Data 6](#_Toc280952753)

[2.7 Hardware & Software Infrastructure 8](#_Toc280952755)

[2.7.1 Ubuntu 13 8](#_Toc280952756)

[2.7.2 Amazon EC2 8](#_Toc280952757)

[2.7.3 Configuring the server 10](#_Toc280952758)

[2.7.4 Database setup 14](#_Toc280952759)

[2.7.5 Further configurations 15](#_Toc280952760)

[3 Programming 3](#_Toc280952741)

[3.1 Bear 4](#_Toc280952743)

[4 Component Analysis 3](#_Toc280952741)

[4.1 User info 4](#_Toc280952743)

[4.2 Game info 4](#_Toc280952743)

[4.3 Twerk moves 4](#_Toc280952743)

[4.4 Twerk controller 4](#_Toc280952743)

[4.5 Single Player AI 4](#_Toc280952743)

[5 Evaluation 3](#_Toc280952741)

[5.1 Metrics 4](#_Toc280952743)

[5.2 Play Testing 4](#_Toc280952743)

[6 Milestones 3](#_Toc280952741)

[6.1 November 16th 4](#_Toc280952743)

[6.2 November 30 th 4](#_Toc280952743)

[6.3 December 7th 4](#_Toc280952743)

[7 References 3](#_Toc280952741)

[7.1 Urls 4](#_Toc280952743)

[8 Contract Modifications 3](#_Toc280952741)

[8.1 E-mail Contract Modifications 4](#_Toc280952743)

# Requirements Analysis

The Bear Hockey social game is intended as a mobile air hockey.

User Information:

* User log-in (On app and over network)
* Friend list
* Friend notifications (When friends are online)

Game Scoring/Rounds:

* Keep track of turns & rounds
* Keep track of twerk sequences
* Twerk sequence matching

Dance Twerking Art/Animations :

* Map of move sets to animations.
* Set of twerk animations.

Twerk Controller:

* Keypad
* Touch

Single Player AI:

* Generation of twerk sequences
* Replicating twerk sequences

Robot Builder:

* Color
* Head/Body Shapes
* Name
* Doilies (antenna, knobs, etc.)
* Logos

Rules:

1. First, the order of turns is decided by a coin flip.
2. The first player then makes a dance/twerk with his robot consisting of n steps, where n is the round.
3. Player 2 has to try to replicate Player 1's steps in a fixed time (or without mistakes).
4. If player 2 replicates the sequence he/she gets n points, where n is the round. Otheriwse player 1 gets the points.
5. The second player then makes a dance/twerk sequence which player 1 must try to replicate.

# Physical Architecture Analysis

The Bear Hockey Social Game is composed of several technologies that stack upon each other to compose the architecture that enables all the required services and provides the user experience. All server‐side technologies are based on Open Source components that provide flexibility and customization while proven reliable and scalable. The server‐side stack is Ubuntu linux, MySQL, Amazon EC2, Boost C++, and Boost.ASIO.

All client‐side technologies are based on Apple’s iOS for iPhone and iPad, XCode, C++, and Cocos2d-x.

The server platform runs on top of the Linux OS with the following runtime environments:

* Ubuntu Linux
* Amazon EC2
* C++
* Apache HTTP Server
* MySQL RDBMS
* OpenSSL (RTC) Server
* Boost.ASIO

The client‐side technologies are based on Apple’s iOS for iPhone and iPad, XCode, C++, and Cocos2d-x.:

* iOS
* XCode
* C++
* Cocos2d-x

## System Architecture

The system is broken up into a server-side (user data and registration) and client-side (iPhone and iPad application)

Server Side

* Ubuntu Linux
* Amazon EC2
* MySQL
* TCP

Client Side

* iPhone
* iPad

## Application Layers

The Bear Hockey application uses a game server for user and friend data and iOS for iPhone and iPad clients.

### Software Requirements

The software:

* iOS
* Xcode
* C++
* Cocos2d-x
* OpenSSL
* Boost C++
* Boost.ASIO

### Hardware Requirements

The hardware:

* Mac OS X
* iPhone
* iPad
* Ubuntu Linux

### Data Access

The databases:

* MySQL (Server)
* SQLite (iOS)

## Application Platform

### Administrative Tools

#### Linux Scripts

Bear Hockey will have a set of administrative tools written in C++ and python. Currently the following tools are included in the admin:

* Users
* Statistics
* Start/Stop game server daemon

## External Systems & services

### Email Service

All Application emails are sent by the system using a batch process.

The process sends all emails on each run using the SMTP mail protocol.

Initially, the SMTP server resides within the EC2 instance, but can be separated as server load grows.

## Communication Protocols

### HTTP

Hypertext Transfer Protocol (HTTP) is the method used to transfer or convey information on the World Wide Web. It is a patented open internet protocol whose original purpose was to provide a way to publish and receive HTML pages.

### OpenSSL

The OpenSSL Project is a collaborative effort to develop a robust, commercial-grade, full-featured, and Open Source toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols as well as a full-strength general purpose cryptography.

## Database Model

The game server and the iOS applications will all use SQL databases.

## Hardware & Software Infrastructure

Bear Hockey is currently hosted in Amazon EC2. Both environments (staging & production) are running on the same single instance.

### Standard deployment procedure

This section describes the installation of the Bear Hockey system on Amazon EC2; it is intended as a guide for a minimal system upon which further configurations can be applied.

#### Game server ports

We're setting all the default ports for OpenSSL, those can be changed but you'll probably need at least access to the web admin panel unless you connect locally from inside the instance.

OpenSSL default ports are:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Interface | Port |  | Type | Description |
| All addresses | 5555 |  | Client (iOS) to Server | The port for iOS clients to connect to the server using OPenSSL. Connections may or may not be encrypted. You can update the security settings for this port. |
| All addresses | 80 |  | HTTP Binding | The port used for unsecured HTTP client connections. |
| All addresses | 443 | S | HTTP Binding | The port used for secured HTTP client connections. |

Allow all those as TCP connections.

# Programming Analysis

Nik Bear Brown will be responsible for all programming of Bear Hockey.

# Component Analysis

User Class:

The User class will keep track of:

* User name
* Actual Name
* User e-mail
* Password
* Friend list

The User class will allow:

* User log-in (On app and over network)
* Friend notifications (When friends are online)
* User time and log-in logs

Game Class:

The Game class will keep track of:

* Game Time
* Score
* End Game Conditions

The Game class will allow:

* Time display
* Score display
* Game Mode (Single or Multiplayer)
* Game Difficulty
* Parental Controls (turn off blood splatters)
* Spawning Paddles and Puck
* Victory or Defeat
* Game Boundaries (table edges etc…)
* Paddle restrictions on board

Puck Class

The Puck class will keep track of:

* Puck location
* Fang Position
* Puck Speed
* Puck Rotation

The Puck class will:

* Handle Puck display
* Handle Puck collision
  + Paddles / Walls
* Handle Fang collision
  + Bounce off walls
  + Spawn the Bear on Paddles

Paddle Class

The Paddle class will keep track of:

* Paddle Position

The Paddle class will allow:

* Being eaten by a Bear
* Display paddle

Paddle Controller

The Paddle Controller class will keep track of:

* Input from the iPad/iPhone

The Paddle Controller class will:

* Move paddle based on input from the iPad/iPhone
* Movement in allowed area
  + Won’t move outside of allowed area

AI Class

The AI class will keep track of:

* None

The AI class will allow:

* Computer control of second paddle
* Levels of difficulty to play against

Bear Class

The Bear class will keep track of:

* Image
* Eating Animation
* Deafening Roar
* Name
* Amount of blood in teeth

The Bear class will allow:

* The generation of customized bears from template
* GUI for bear customization
* Eating the loser
* In-App purchases for bear hats



(Image from https://wiki.teamfortress.com/w/images/6/6c/Bear\_Necessities.png?t=20130711152823)

# Evaluation

The following tests will be used to evaluate the quality of the software:

Unit Tests

* A positive assertion for all class methods
* A negative assertion for all class methods

Play Tests

* Bear will play/user test any component with user input

Simulations

* Data will be simulated when missing or sparse

Security tests

* Automated stress and password hacking will profile each component

Metric Logs

* User and play statistics will be logged

Dynamic Profiling

* Valgrind will be used to profile all components

# Milestones

The following milestones will be due at the following dates:

November 16th

* The User Class and Game Class

November 30th

* The Paddle and Paddle Controller Class

December 7th

* A working prototype of a playable game on the iPhone

# References

The Bear Hockey Social Game is composed of several both server, and application components. It is written using C++, Cocos2d-x, OpenSSL, Boost.ASIO and linux.

## References

* Apache: <http://httpd.apache.org/>
* Python: <http://www.python.org/>
* Ubuntu Linux: <http://www.ubuntu.com/>
* Django: <https://www.djangoproject.com/>
* Amazon EC2: <http://aws.amazon.com/ec2/>
* mod\_wsgi: <http://code.google.com/p/modwsgi/>
* Cocos2d-x: <http://www.cocos2d-x.org/wiki/Cocos2d-x>
* OpenSSL: <http://www.openssl.org/>
* Boost C++: <http://www.boost.org/>
* Boost.ASIO: <http://www.boost.org/doc/libs/1_54_0/doc/html/boost_asio.html>
* Xcode: <https://developer.apple.com/xcode/>

# Contract Modifications

Any changes to this document will be e-mailed to the professor with the changes highlighted and the date. The professor must approve of the most recent changes,