Mock Design Review

Personal Tennis Trainer for Beginners

Overall Block Diagram

Legend
- Wi-Fi
- Audio Prompts
- Physical strokes
- Analog signal
Figure 4: Software Components Block Diagram
Web Application:
Our web application will comprise of the following components:

1) SQL Database: Tables will be used to store user progress, Physical simulation results and training data for ML Depth Model.

2) Physical Model Simulation and ML Model: Based on the Newtonian mechanics of our mechanical system and the time series values generated from the sensor we will determine the direction and depth of an actual tennis ball on a real court. A decision trees model will be introduced for directional determination using a training set manually classified by us. A Gaussian kernel regression model for depth estimation will be used in accordance with the physically expected depth value.

3) APIs:
- getUserProgress(Activity): This API call will return user scores for the selected activity over a period of time. It will be called by the Android application that would then use this data to display user progress as a plot.
- setUserScore(User, Activity, Score): This API method will be called from the mobile application that would return the user score for the particular activity that they just played. The method will store this data into the database.

4) Publish Subscribe Server: In order for the direction data to be processed in real time, we will have to provide the mobile app with directional data as soon as readings are collected. This will be done by using a publish subscribe protocol. We will be using a pub sub server, with the web app acting as a publisher and the phone app as a subscriber.

We will be developing an Android Application with the following functionality:

1) UI design and development: An interactive UI will be designed to allow easy navigation and fun gameplay through the phone.

2) Audio Prompts: Specific audio prompts will be given via the speakers of the phone depending on the activity being played by the user.

3) Subscriber to Pub/Sub messaging: The phone app will retrieve directional and depth information for each shot hit by the player.

4) Score Calculation and Storage: At the end of each activity, a score will be determined based on the performance of the user. This score will be supplied to the web application via an API call.

A web application is a natural choice for managing users and their progress on the platform. We don’t want to have individual databases storing a user’s information locally that requires too much space and doesn’t persist between app installations. Further the Machine Learning algorithms will be iterative and will require extensive computational resources and historic data, thus we simply cannot have this logic on a mobile app. The only other choice is to provide a centralized location that contains this logic in the form of a RESTful web service.

We will use Google Cloud Engine for hosting our application.
Wifi is a requirement and we cannot use something like Bluetooth since the web application will be computing the depth and direction, so the measured readings from the sensor need to be sent over the network to our service. Adding Bluetooth has no advantages for our use case.

**Requirements & Verifications**

<table>
<thead>
<tr>
<th>Component</th>
<th>Requirement</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Application &amp; communication protocols</td>
<td>Correctness of depth calculations done by the webapp to a 95% accuracy limit.</td>
<td>Running unit tests with varying input datasets. These datasets will be collected by using a ball throwing sling shot. The unit tests will compare results obtained by throwing an actual tennis ball using the slingshot on an actual tennis court with computation results from our system using the webApp and pass only if the directions match. Results will be stored and accuracy will be calculated for 100 runs.</td>
</tr>
<tr>
<td></td>
<td>The computational analysis of depth must be completed under 500ms.</td>
<td>Must pass unit tests with varying input datasets will which time out if computation exceeds 500ms.</td>
</tr>
<tr>
<td></td>
<td>The web application must receive the tennis shot data within 500ms (communication latency).</td>
<td>Must pass unit tests where timestamps on the incoming data are compared with the timestamp of data received on the web app, allowing for a maximum of 500ms difference. Using the Precision Time Protocol for synchronization over WLAN we will alter the WiFi source accordingly to ensure correct rates of data transfer.</td>
</tr>
</tbody>
</table>

**Privacy and Security**
It is imperative that we maintain the highest standards of security and build access control in our web application design. Our POST requests will be built tolerant to SQL Injection attempts. We will implement basic authorization with requests to build access control of data. We build concurrency control as a fault tolerance mechanism against multiple user access.

Citations


