University of Ottawa | Université d’Ottawa
GNG2101 | Fall 2022

GNG2101

Introduction to Product Development and Management for Engineers and Computer Scientist

Course Professor: Dr. Emmanuel Bouendeu

Deliverable D

Detailed Design, Prototype 1, BOM, Peer Feedback and Team Dynamics

Presented to: Melika Ataebi

Prepared by Group B12

Team Members:

Family Name, Name:	Student ID:
Keryakes, Laura	300265134
Balachandiran, Vivethen	300245080
Ouloum, Yassine	300263213
Ma, Jiayi	300263220
Vadakkeveettilan Hilariyos, Chelse Rose	300214163

Date: October 9, 2022

Introduction 3

Summary of Client Feedback 3

Updated and Detailed Design of Group Concept 4

Bill of Materials (BOM) 5

Critical Product Assumptions: 7

Prototype 1: Pseudocode 8

Client Information, Next Steps & Conclusion 10

Introduction

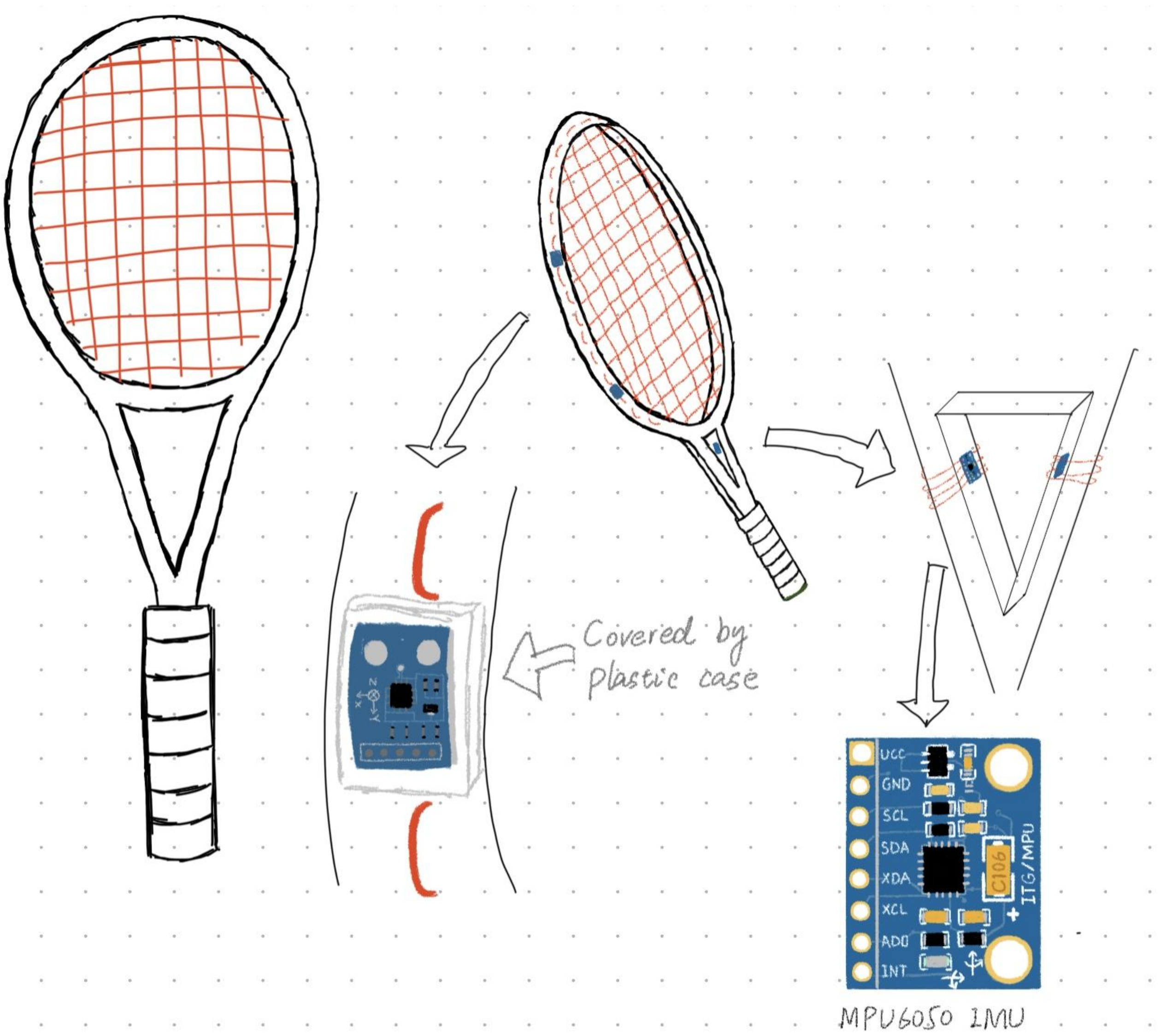
Previously, the team has identified the client needs, created conceptual designs for the project and had our second client meet. In the client meet, we received feedback on our group concept and we were told which components should be modified or added in. In this deliverable, our main focus is to come up with an overall cost estimate of our project which will only be possible if we specifically know which sensors we are required to have for our product. This prototype will be a focused low-fidelity prototype considering the lack of resources (time, expertise and money) we have at the moment. We will mainly focus on the second subsystem, to analyze and categorize the outputted data, using a pseudocode. We will also focus on coming up with which sensors we will be getting and seeing what is their optimal position, regarding accuracy in measuring the data and the durability of the sensors depending on their placement

Summary of Client Feedback

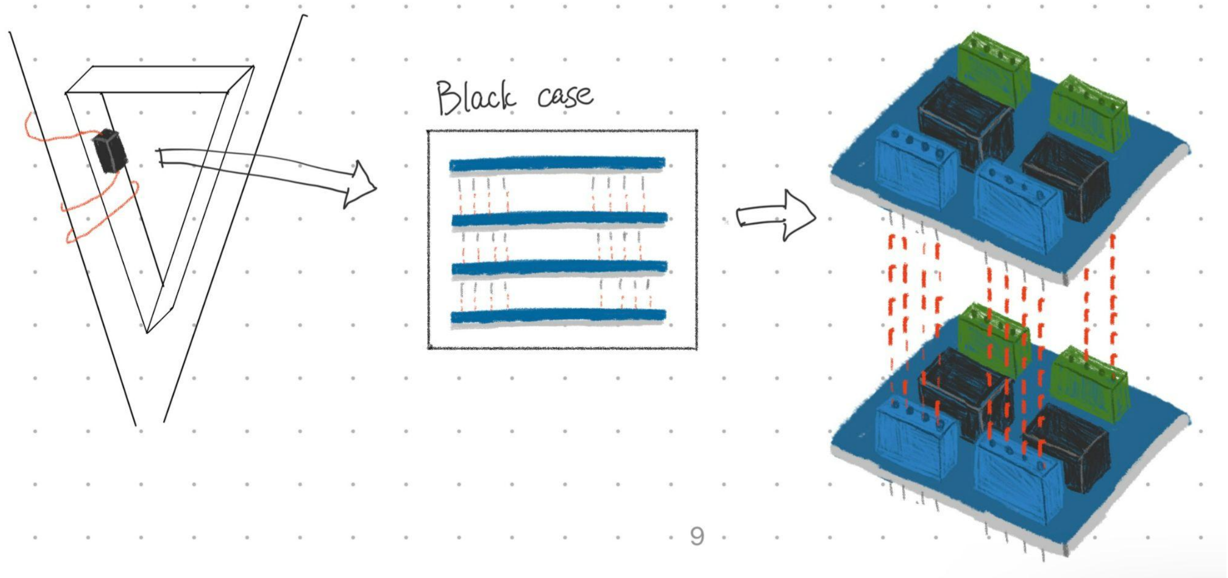
Our client liked the implementation of a Bluetooth chip which will allow for no hindrance of the athlete performance and routine. However, what needs to be further developed in our conceptual designs is the choice of the quality and quantity of our sensors. In fact, our client mentioned that it would be better if we have more sensors collecting and measuring the same data rather than having one sensor for each different datum. This way, not only will our results be more accurate as we can take the average of all the data collected, but we can also position our sensors in different areas around the racquet (instead of keeping everything at the base, as we previously mentioned in our group concept) to allow for a better distribution of the results. Furthermore, our client mentioned the possibility for our product to be marketed such that she expects an advertisement or at least an application to go alongside our product. This way our product can be advertised to a greater audience in the long run. Moreover, the output of our data to the user should be continually updated after each performance which will make it easier for the user to see their results instead of waiting for it. The data also needs to be portrayed in a user-friendly way using graphs rather than numbers.

This will make it easier for the user to understand what they are seeing as well as allowing for a comparison of their own score compared to the rest of their teammates.

Updated and Detailed Design of Group Concept



Or:



Based on the team design concept and the second customer meeting, combined with the BOM, we came up with a more detailed design solution. The improvement we made based on the suggestions made in the customer meeting was to set up more sensors to ensure the accuracy of the data and thus support more variables. Meanwhile, to reduce the burden on the players, we distributed the different sensors in different positions of the racket. Eventually all sensors will be connected together by jumper wires.

aBill of Materials (BOM)

As a team, we have agreed on specifically testing the athlete's swing speed and the racquet's trajectory during specific techniques. To do so, we need sensors that measure the position over time. Among these sensors, we have found PIR Motion Sensors which allow us to sense motion by detecting the occupancy and movement from the infrared radiated from a human body. They are small, inexpensive, low-power, easy to use and do not wear out which is perfect for our project constraints (Arduino). Furthermore, from the client meeting, we have learned that they preferred to have more sensors that measure the same thing rather than a singular sensor for each variable. Thus, to accompany the PIR Motion Sensors, we have decided to implement gyroscopes, more specifically, the MPU6050 IMU, which has both the 3-Axis accelerometer and 3-Axis gyroscope integrated on a single chip. The gyroscope measures rotational velocity of the angular position over time along the x-y-z axis, while the accelerometer measures the gravitational acceleration along the three axis and we can get the position via the angle that the sensor is positioned. Thus, we can get very accurate information about the sensor orientation. Finally, we will obviously need the Arduino Uno R3, a Breadboard, the HC-05 Bluetooth Module and jumper wires to connect everything together and allow for proper transmittance of data to the design team.

Item Number	Part Name	Description	Quantity	Total Cost
1	Arduino Uno R3	Microcontroller Board	1	$0- already possess one
2	BLE Arduino nano	Microcontroller Board+Bluetooth module	1	$15 (makersportal.com)
3	Breadboard	Hold electronic components wired together	1	$0- already possess one
4	HC-05 Bluetooth Module	Wireless communication between components and Arduino Uno	1	$12.99 (Amazon.ca)
4	Jumper Wires	Make connections between items on breadboard	1 pack	$0- already possess one
5	PIR Motion Sensor	Electronic sensor that measures IR light radiating from objects in its field of view	5 pack	$14.69 (Amazon.ca)
6	MPU6050 IMU	6-Axis motion tracking device that a 3-axis accelerometer and	3 pack	$14.98 (Amazon.ca)

		3-axis gyroscope data		
Total				**$57.66**

Table 1. BOM for Final Product

By listing our updated design of our concept after the client meeting, we were able to specifically know which component we require for our final product and therefore optimize our budget.

Critical Product Assumptions:

We assume that the sensors will retrieve multiple sets of positions and times throughout the entire swing. We will approximate the small intervals of sensing to have a constant velocity in that brief moment in time. We will use these intervals and compare them with one another to find out the user's peak swing velocity throughout their racquet swing. Thus, our most critical product assumptions for this prototype is based on the critical functionality of our final code. In fact, we want to ensure that our final code can ultimately read the data inputted by our various sensors, such that the multiple sets positions and times throughout the swing can be effectively translated to the swing speed velocity of the racquet during the performance.

Since this is our first prototype and based on our critical product assumptions, we decided to create a low-fidelity analytical prototype that focuses on the second subsystem: "The product's ability to categorize and analyze the data based on performance." More specifically, we created a pseudocode, which is a detailed yet readable description of what our Arduino software must do to be able to analyze and categorize the data based on what it receives, concerning the sets of position and time. We decided on focusing on the second subsystem because before being able to test our sensors, it is important to understand the fundamental foundation of our product which is our code; without this code, the data collected by the sensors yields nothing, We started out with a low-fidelity pseudocode because we cannot test the code directly on the Arduino software without having the physical sensors connected to that software. According to our lack of resources, this seems to be the most simplest and effective way to ensure that we have a general idea on how our code should interact with the data it receives from the multiple sensors.

Prototype 1: Pseudocode

Calculate “swing speed (velocity)” based on distance and time
(assume multiple distances and times given in an array as inputs)
Vivethen Balachandiran

(Array dist[] = array of positions, Array time[] = array of times) taken from arduino sensors
Variables: V = velocity, lastD = previous distance, lastT = previous time, peakV = peak velocity;

```
----------------------------------------------------------------------------------------------------------
V, lastD, lastT, peakV = 0; //Initialize variables
/* If multiple sensors are used just average all the positions and times from them all and
calculate as usual */
for ( int i = 0; i < arrayTime.length(); i++ ) { //Using arrayTime or arrayDist make no diff
        if (time[i] != 0) { //Making sure we don’t divide by 0 in calculations
                V = (dist[i] - lastD) / (time[i] - lastT); //Velocity in specific point in time
        }
        if (peakV < V) peakV = V; //changes peak velocity if the new velocity is greater
        lastD = dist[i]; //Setting distance to use in future loop calculation
        lastT = time[i]; //Setting time to use in future loop calculation
}
return peakV //The peakV (peak velocity) will be used to make a graph
```

Pseudocode Prototype 1 Testing Examples:

Trial 1:

Distance	0	1	3	6	8
Time	0	1	2	3	4

Peak velocity (peakV) of trial 1 = 3

Steps:

1. dist[i] = 0, time[i] = 0 (Entering ‘for’ loop)
   - We enter the for loop the same number of times as the length of the array
   - Since time[i] = 0, the velocity value does not change (first data point)
2. dist[i] = 1, time[i] = 1
   - Using >>> V = (dist[i] - lastD) / (time[i] - lastT) we get V = (1-0)/(1-0) = 1
   - peakV (0) < V (1) is true therefore peakV = V = 1
   - lastD and lastT set to current dist and time
3. dist[i] = 3, time[i] = 2
   - Using >>> V = (dist[i] - lastD) / (time[i] - lastT) we get V = (3-1)/(2-1) = 2
   - peakV (1) < V (2) is true therefore peakV = V = 2
   - lastD and lastT set to current dist and time
4. dist[i] = 6, time[i] = 3
   - Using >>> V = (dist[i] - lastD) / (time[i] - lastT) we get V = (6-3)/(3-2) = 3
   - peakV (2) < V (3) is true therefore peakV = V = 3
   - lastD and lastT set to current dist and time
5. dist[i] = 8, time[i] = 4
   - Using >>> V = (dist[i] - lastD) / (time[i] - lastT) we get V = (8-6)/(4-3) = 2
   - peakV (3) < V (2) is false therefore peakV remains the same value
   - lastD and lastT set to current dist and time
6. Loop finished and returns the peakV value (in this case 3)

Trial 2:

Distance	0	1	8	12
Time	0	1	3	5

Peak velocity (peakV) of trial 2 = 2.67

Steps:

1. dist[i] = 0, time[i] = 0 (Entering ‘for’ loop)

   We enter the for loop the same number of times as the length of the array

   Since time[i] = 0, the velocity value does not change (first data point)
2. dist[i] = 1, time[i] = 1

Using >>> V = (dist[i] - lastD) / (time[i] - lastT) we get V = (1-0)/(1-0) = 1

peakV (0) < V (1) is true therefore peakV = V = 1

lastD and lastT set to current dist and time

3. dist[i] = 8, time[i] = 3

   Using >>> V = (dist[i] - lastD) / (time[i] - lastT) we get V = (8-1)/(3-1) = 2.67

   peakV (1) < V (3) is true therefore peakV = V = 2.67

   lastD and lastT set to current dist and time
4. dist[i] = 12, time[i] = 5

   Using >>> V = (dist[i] - lastD) / (time[i] - lastT) we get V = (12-8)/(5-3) = 2

   peakV (3) > V (5) is false therefore peakV remains the same value

   lastD and lastT set to current dist and time
5. Loop finished and returns the peakV value (The peak value in this trial is 2.67)

The results of our tests prove that, given an input of position over time which is the simplest sensor set we could use, we can convert that data into velocity at each instant, which is one of our chosen measurements. It also proves that there is no limitation on the values it can measure since it is based on basic algebra (the only limitation could come from the frequency of measurements which does not belong to this prototype).

Client Information, Next Steps & Conclusion

Now that we have a general idea of how we want to analyze and categorize our data, it is important to not only get the approval of our client, but also ensure that for the next meeting, we can properly demonstrate the efficiency of our prototype regarding not only its analysis, but also how it collects data. Thus, for our next client meeting it will be important that we have a medium-fidelity prototype that is at least functional in some aspects of the final product, either regarding how it outputs the analyzed and categorized data or how it inputs the raw data received from the athletes' performances. For our next client meeting, we are also looking to ensure that any other expectations regarding our final product are communicated through by the client and to ensure that what they are looking for are completely translated into the prototype we showed them (this will ensure that we are on the right track in developing a satisfactory final product).