NEW MEDIA &
COMMUNICATION
TECHNOLOGY

New Media
Project

howest
De Hogeschool West-Vlaanderen
Lid van de Associatie Universiteit Gent

NEW MEDIA AND COMMUNICATION TECHNOLOGY

Brainstorm

Zodra we onze opdracht kregen, zijn we begonnen met allerlei ideeën te verzamelen. Eens we behoorlijk wat hadden nagedacht, na onze brainstorm op te schrijven en niet veel inspiratie meer over te hebben, zijn we tot deze ideeën gekomen:

Makey Makey

http://www.makeymakey.com/

Arduino

https://www.arduino.cc/

Raspberry Pi

https://www.raspberrypi.org/

Humanoïde robot + spraakherkenning

Leap motion of kinect + robot

Beacon + hue (spel)

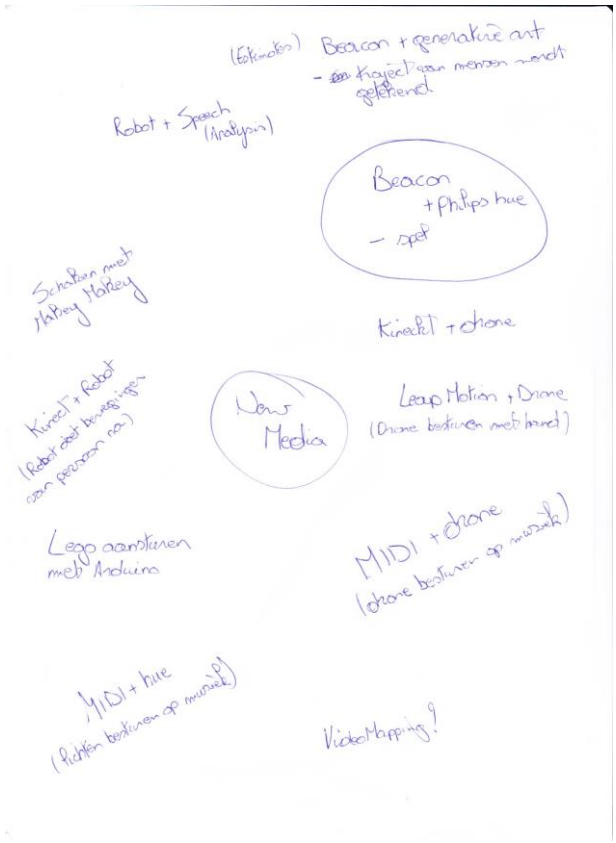
Schaken met Makey makey

Leap motion of kinect + drone

MIDI + drone (drone besturen op muziek)

LEGO???

Muziek (speech analysis?) + hue



Na het denkwerk over wat we allemaal konden doen, zijn we samen tot het besluit gekomen dat het ons het leukst lijkt, om een combinatie van beacons met de Philips Hue te gebruiken. Ons doel is om hiermee een interactief spel te maken, dat we nog zullen uitdenken. Er zijn alvast veel soorten spelen mogelijk met de beacons.

Het project

Eens het brainstormen voorbij was, en de keuze om met de Estimote Beacons en de Philips Hue te werken gemaakt was, zijn we concreter beginnen nadenken over wat voor spel we wouden maken. We zijn tot de conclusie gekomen, om een interactief spel te maken waarbij verschillende team of individuen het tegen elkaar opnemen. Door middel van Beacons (de checkpoints) op verschillende punten te zetten, bijvoorbeeld in een sporthal, kan een server zien wie zich op welke plaats bevind. Zo kan het team dat het meeste checkpoints veroverd/'afgepakt' heeft van het andere team winnen. Ons spel staat nu in grote lijnen vast, maar de regels en details moeten nog bedacht worden.

We willen ook de Philips Hue betrekken bij ons project. Zo kan er visueel getoond worden (met verschillende kleuren) wanneer je een checkpoint bereikt hebt.

Het Spel

We hebben als spel, een basis gemaakt, met een uitbreiding. Voor de uitbreiding zullen we echter niet genoeg tijd hebben, dus het is momenteel gebleven bij de basis. Eventueel zouden we de uitbreiding ooit nog kunnen maken, en er dan ook voor zorgen dat we de app op iOs & Android kunnen gebruiken. Dit zou het spel makkelijker maken om te spelen.

Voorbereiding

Basis

Er worden 2 teams gemaakt van 5 personen. Er is een rood team en een blauw team. Eenmaal er 2 teams gemaakt zijn, wordt er voor beide teams een basis gekozen. Dit is het punt waarnaar ze terug moeten keren als iemand dood is. Eens beide basissen gekozen zijn moeten de beacons geplaats worden. Deze moeten eerlijk geplaatst worden (op gelijke afstanden, 1 dicht, 1 midden en 1 ver). De beacons moeten samen met een Philips hue ergens gezet worden. Of je deze verstopt of niet, mag je zelf kiezen. De Philips hue moet echter wel goed zichtbaar zijn.

Uitbreiding

Er worden 2 teams van 6 personen gemaakt, met telkens 5 puntenvangers en 1 aanvaller. De aanvaller mag niet in de buurt komen van de beacons, en moet proberen om de puntenvangers van het andere team aan te tikken.

Doel

Basis

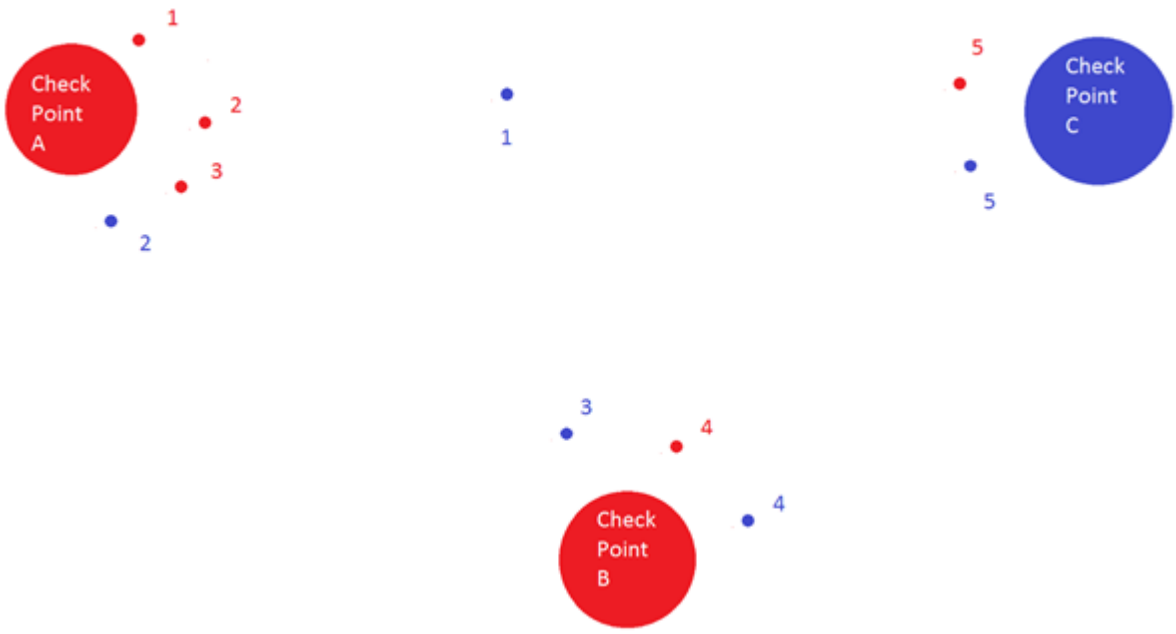
Het doel van het spel is zo snel mogelijk 500 punten verzamelen. Punten kunnen verzameld worden door een beacon voor jou team te winnen (uitleg volgt).

Een checkpoint kan neutraal, van het andere team, of van jouw team zijn. Voor een checkpoint te winnen, moet jouw team in de meerderheid zijn aan een checkpoint (< 0.5m). Als je in de meerderheid bent, wordt de checkpoint langzaam van jouw team. Als de checkpoint neutraal is, moet je 3 seconden lang in de meerderheid zijn aan een checkpoint. Als de checkpoint van het andere team is, moet je eerst 1 seconden in de meerderheid zijn om hem neutraal te maken. Als een checkpoint van jouw team is, en er staat niemand op blijft hij van jouw team. Elke 2 seconden krijg je 1 punt per checkpoint die van jouw team is.

Uitbreiding

Er is in beide teams ook een aanvaller aanewezig. Deze moet uit de buurt van de beacons blijven. Als deze toch in de buurt komt van de beacon krijgt zijn team -25 punten. Als de aanvaller een puntenvanger van de tegenstanders kan aantikken, dan moet die puntenvanger terug naar zijn basis lopen, en mag pas vanaf dan weer punten beginnen vangen.

Voorbeeld



In bovenstaande situatie heeft het rode team twee checkpoints en het blauwe team één checkpoint. Dit betekend dat het rode team twee punten per twee seconden krijgt, en blauw maar één punt.

Op checkpoint A zijn er drie mensen van het rode team aanwezig, en één iemand van het blauwe team. Dit betekend dat deze checkpoint behouden blijft door het rode team.

Je ziet echter ook dat er op checkpoint B twee mensen van het blauwe team zitten, terwijl er maar één speler zit van het rode team. Dit betekend dat het punt na twee seconden niet meer van rood zal zijn, maar neutraal zal worden (wit licht). Over verloop van tijd (aanpasbaar naar hoe je het wilt, momenteel 6 seconden) zal checkpoint B niet meer neutraal zijn, maar van blauw zijn.

Op checkpoint C, staan er evenveel leden van beide teams, wat resulteert in een behouden stand van een checkpoint. Hetzelfde zou gebeuren als er van beide teams niemand aanwezig is.

Speler 1 van het blauwe team, zit tussen de drie checkpoints in, en telt dus nergens mee.

Ondanks speler 1, is dit een goede situatie voor het blauwe team aangezien ze, als het rode team geen actie onderneemt, twee punten zullen hebben.

Estimote Beacons

De beacons zijn oorspronkelijk gemaakt om enkel met iOS en Android devices te werken. We zullen eerst moeten bekijken welk OS (iOS, Android of Windows) we zullen gebruiken. Aangezien geen van ons twee ervaring heeft met iOS laten we dit besturingssysteem alvast vallen. We zouden graag met Windows werken omdat we daar het meest ervaring mee hebben, maar voor dat OS is er geen ‘officiële’ code om mee te werken. We zoeken dus naar een alternatief. Met Android hebben we niet veel ervaring dus we nemen dat OS als plan B.

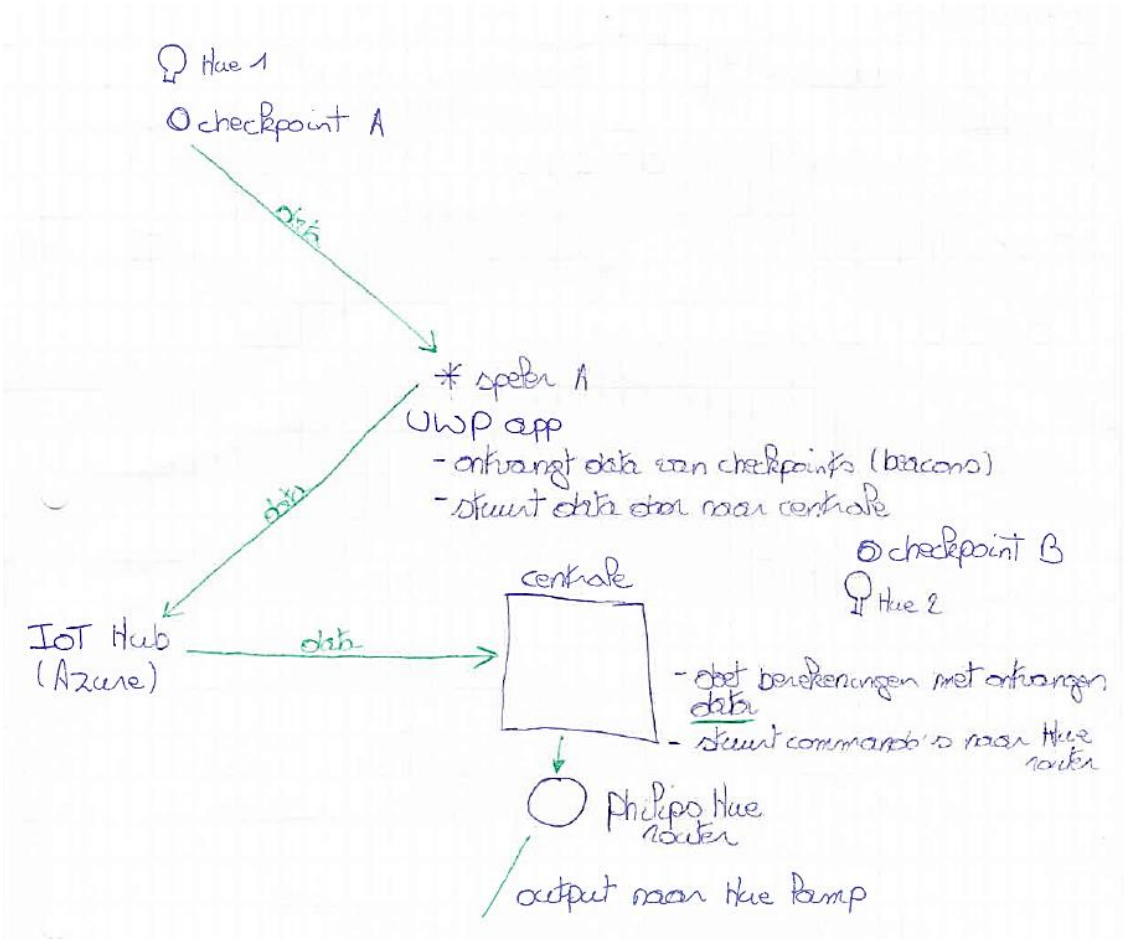
Philips Hue

In de les hebben we reeds met Philips Hue gewerkt d.m.v. ze met Processing te programmeren. Met de Hue kunnen we de status van de checkpoints op een visuele manier weergeven aan de spelers.

Infrastructuur

Om dit spel te kunnen spelen zijn er spelers nodig met elk een Windows 10 phone. Daarop draait onze UWP applicatie. Die app ontvangt het signaal van de Estimote Beacons (checkpoints) en berekent de nodige gegevens. Daarnaast kunnen spelers met die app ook zichzelf registreren in een team en het spel starten.
Met behulp van de Azure IoT Hub staat die UWP app in contact met een C# Console applicatie, deze noemen we onze 'Centrale'.

De Centrale doet alle berekeningen die nodig zijn om het spel in juiste banen te leiden. Dit houdt onder andere in dat hij berichten ontvangt verzonden door de meerdere UWP apps. Met deze data weet de Centrale welke speler bij welk checkpoints staat, hoeveel punten elk team krijgt en welke Hue opgelicht moet worden met welke kleur.



Waarom C#?

Voor hetgeen we willen doen met ons project, is het niet mogelijk Processing voor alles te gebruiken. Ten eerste kunnen we geen Windows smartphone applicatie bouwen met de taal. Een tweede rede is de afwezigheid van Beacon libraries voor Processing. Daarbij zou communicatie tussen een Processing applicatie, die de Hue aanstuurt, en een Windows app zeer moeilijk verlopen. Om deze redenen hebben we gekozen alles te programmeren in C# en dus Windows applicaties te ontwikkelen en gebruiken.

Planning & tijdsbesteding

21 maart (labo 1)

Jan

- Hoe zullen we communiceren tussen smartphone en server? 45 min
- IIS webserver ontdekken 60 min
- Hoe gebruik maken van IIS als local server in ASP.NET? 60 min
- Uittesten beacons 30 min
- Uitdenken spel 45 min

Stijn

- Onderzoeksdocument schrijven 15 min
- Kunnen de beacons gebruikt worden met Windows? 60 min
- Experimenteren met de Estimote Beacons 40 min
- Experimenteren met de library, een UWP schrijven 60 min
- App op een Windows phone proberen zetten 60 min

7 april

Jan

- Code schrijven voor het spel 240 min
- 'Centrale' van UWP naar console 60 min

Stijn

- IoT Hub testen 90 min
- IoT Hub integreren in het project 240 min

8 april

Stijn

- Communicatie tussen smartphone app & centrale (met IoT Hub) 120 min

Jan

- Logica & code spel verfijnen en afwerken 120 min

16 april

Stijn

- Communicatie tussen smartphone app & centrale (met IoT Hub) 180 min

Jan

- Afstandberekening onderzoeken met signaal beacons 90 min

18 april (labo 2 + namiddag 4u)

Jan

- Beacons ontvangen op Windows phone en afstand berekenen 120 min
- Philips Hue aansturen met centrale 150 min
-

Stijn

- Communicatie D2C, C2D, Console to Hue 180 min
- App laten werken op een (andere) Windows 10 phone (lukt niet) 120 min

samen

- Bug fixes 300 min

25 april (labo 3 + namiddag 4u)

Jan

- App deployen op een tweede Windows 10 Phone 60 min

samen

- Communicatie testen met twee phones i.p.v. maar 1 180 min
- Spel testen en bugs oplossen 300 min

26 april

samen

- Laatste tests 60 min

Resultaten

21 maart (labo 1)

In het begin van het labo hebben we definitief afgesproken hoe we ons project precies zullen aanpakken.
We kozen voor Windows als OS omdat we hiervan het meest kennis en hardware hebben. Het nadeel is dat de Estimote Beacons Windows niet officieel ondersteunt. Na wat opzoek werk vonden we een library: 'The Universal Beacon Library'[1] die de mogelijkheid biedt om op een gemakkelijke manier beacons te lokaliseren.

Nadat we een universal Windows applicatie geschreven hebben, kon onze Windows Phone met Windows 10 Mobile de beacons detecteren.
Nu we de beacons kunnen 'zien' kan het programmeren van het spel beginnen.

Volgende code laat ons toe beacons te detecteren met een Windows apparaat:

```
private readonly BluetoothLEAdvertisementWatcher _watcher;

public MainPage()
        {
            this.InitializeComponent();
            _beaconManager = new BeaconManager();            _watcher = new
BluetoothLEAdvertisementWatcher { ScanningMode = BluetoothLEScanningMode.Active };
            _watcher.Received += WatcherOnReceived;
            _watcher.Start();

            ReceiveC2dAsync();
            CheckNearbyBeacons();
        }
private void WatcherOnReceived(BluetoothLEAdvertisementWatcher sender,
BluetoothLEAdvertisementReceivedEventArgs eventArgs)
```

[1] https://github.com/andijakl/universal-beacon

```
        {
            _beaconManager.ReceivedAdvertisement(eventArgs);
        }
```

Hier na kan er gemakkelijk met een foreach statement de gegevens van de gedetecteerde beacons gelezen worden door de lijst te overlopen.
Niet enkel onze Estimote Beacons worden gezien door de code. Ook andere bluetooth apparaten zitten in de lijst. Door middel van te filteren op MAC-adres zullen we enkel de gegevens van onze beacons in een lijst weergeven.

```
private void CheckNearbyBeacons()
        {
            lstConsole.Items.Clear();
            lstConsole.Items.Add("Beacons discovered so far\n--------------------");

            foreach (var bluetoothBeacon in _beaconManager.BluetoothBeacons.ToList())
            {
                if (bluetoothBeacon.BluetoothAddressAsString == CHECKPOINT1
                    || bluetoothBeacon.BluetoothAddressAsString == CHECKPOINT2
                    || bluetoothBeacon.BluetoothAddressAsString == CHECKPOINT3
                    )
                {
                    if (bluetoothBeacon.BluetoothAddressAsString == CHECKPOINT1)
                        lstConsole.Items.Add("paars");
                    else if (bluetoothBeacon.BluetoothAddressAsString == CHECKPOINT2)
                        lstConsole.Items.Add("mint");
                    else
                        lstConsole.Items.Add("blauw");
                    double afstand = CalculateAccuracy(bluetoothBeacon.Rssi);
                    lstConsole.Items.Add("Distance: " + afstand);
        }
```

Ook hebben we een guide[2] gevolgd, om een IIS Webserver op te zetten zodat we de lokale communicatie kunnen verzorgen tussen de smartphones, computer en Philips hue. Hiermee hebben we echter veel problemen ervaren, waardoor we zijn beginnen zoeken naar een alternatieve oplossing.

22 maart

We hebben hulp gevraagd over hoe we de communicatie tussen smartphone, computer & hue het best aanpakken. We zijn van plan veranderd en gaan niet meer met een IIS server werken, maar wel met PHP via een USBWebserver.

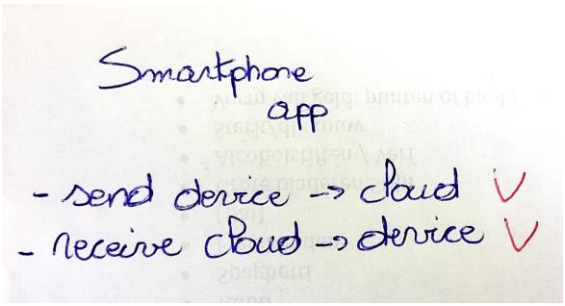
7 april

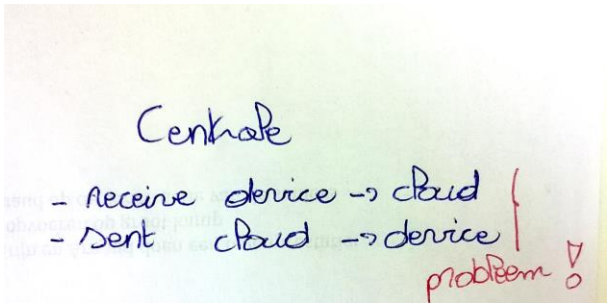
We zijn opnieuw van plan veranderd, en hebben beslaten om met de IoT Hub[3] te werken. Deze hebben we vandaag de IoT Hub uitgeprobeerd en berichten uitgewisseld tussen een device en de cloud (IoT Hub) en van de cloud naar een device. Dit is goed gelukt d.m.v. een handige tool[4]. Maar als we de IoT Hub dan in ons project wouden integreren botsten we op enkele problemen waar we heel lang op vast hebben gezeten en waarvoor we nog steeds geen oplossing hebben. De smartphone app kan zonder problemen berichten sturen naar de cloud (Device to cloud). Dan willen we de centrale, een Universal Windows applicatie, al die berichten laten ontvangen. Dit lukt blijkbaar enkel met een Windows Console applicatie, en niet met UWP. We zoeken nog steeds naar een oplossing.

[2] https://www.youtube.com/watch?v=-ymjT8C-tHc

[3] https://azure.microsoft.com/en-us/documentation/articles/iot-hub-csharp-csharp-getstarted/#create-a-simulated-device-app

[4] https://github.com/Azure/azure-iot-sdks/blob/master/tools/DeviceExplorer/doc/how_to_use_device_explorer.md





Receive C2D & Sent C2D (https://azure.microsoft.com/en-us/documentation/articles/iot-hub-csharp-csharp-c2d/)

Receive D2C & Sent D2C (https://azure.microsoft.com/en-us/documentation/articles/iot-hub-csharp-csharp-getstarted/#receive-device-to-cloud-messages)

Code

Deze code bevind zich in de UWP applicatie.

Deze globale variabelen zijn gegevens waarmee de iothub en device herkent wordt.

```
static DeviceClient deviceClient;
        static string iotHubUri = "NewMediaBeaconGame.azure-devices.net";
        static string MyFirstDeviceKey = "0FwB40mUJ4VwPsrgbgbE7aZexHpSaceEV1zdxQI/rac=";
        static string DeviceName = "MyFirstDevice";
```

Dit is de constructor van de MainPage en wordt als eerste uitgevoerd. Hier wordt de deviceClient gekoppeld aan de iothub en het device waaruit verstuurd wordt meegegeven.

```
public MainPage()
        {
            this.InitializeComponent();
            deviceClient = DeviceClient.Create(iotHubUri, new
DeviceAuthenticationWithRegistrySymmetricKey(DeviceName, MyFirstDeviceKey));
            ReceiveC2dAsync();
        }
```

sent D2C

Deze methode verzend berichten naar de IoT hub.

```
private static async void SendDeviceToCloudMessagesAsync(int CheckpointId, double distance)
        {
            var D2Cmessage = new
            {
                DeviceId = DeviceName,
                PersonName = PersonName,
                Checkpoint = CheckpointId,
                AfstandTotCp = distance,
                Team = TeamId
            };
            var messageString = JsonConvert.SerializeObject(D2Cmessage);
            var message = new Message(Encoding.ASCII.GetBytes(messageString));

            await deviceClient.SendEventAsync(message);
        }
```

receive C2D

Met deze methode kunnen berichten verstuurd vanuit de Centrale ontvangen worden.

```
        public async void ReceiveC2dAsync()
        {
```

```
            while (true)
            {
                Message receivedMessage = await deviceClient.ReceiveAsync();
                if (receivedMessage == null) continue;

                lstTeams.Items.Add(Encoding.ASCII.GetString(receivedMessage.GetBytes()));

                await deviceClient.CompleteAsync(receivedMessage);
            }
        }
```

8 April

De smartphone applicatie is in staat data te versturen naar de IoT Hub (sent D2C) d.m.v. een Console applicatie te gebruiken. Het lukt om met de centrale die data op te halen (receive D2C). De centrale kan data versturen naar een device (sent C2D). En daarna is het gelukt om met het device de verstuurde data van de cloud terug op te halen (receive C2D).

Code

Code centrale:

```
static string connectionString = "HostName=NewMediaBeaconGame.azure-
devices.net;SharedAccessKeyName=iothubowner;SharedAccessKey=ohRIpIsXLPhwiChZMmik2i6Lc1pYaoRvFXK2M8jRaII
=";
        static string iotHubD2cEndpoint = "messages/events";
        static EventHubClient eventHubClient;
```

Globale Variabelen:

```
public static int lamp1 = 14; // paars
public static int lamp2 = 12; // mint
public static int lamp3 = 10; // blauw

static bool TrueOrFalse = true;

private static List<String> DeviceIds = new List<string>();

private static int cp1 = 50; // Checkpoint 1 stand, 0 = team rood (1), 100 = team blauw (2)
private static int cp2 = 50; // Checkpoint 2 stand, 0 = team rood, 100 = team blauw
private static int cp3 = 50; // Checkpoint 3 stand, 0 = team rood, 100 = team blauw

private static List<Speler> cp1t1 = new List<Speler>(); // Array van ID's van teamleaden team 1 in de
buurt van checkpoint 1
private static List<Speler> cp2t1 = new List<Speler>(); // Array van ID's van teamleaden team 1 in de
buurt van checkpoint 2
private static List<Speler> cp3t1 = new List<Speler>(); // Array van ID's van teamleaden team 1 in de
buurt van checkpoint 3

private static List<Speler> cp1t2 = new List<Speler>(); // Array van ID's van teamleaden team 2 in de
buurt van checkpoint 1
rivate static List<Speler> cp2t2 = new List<Speler>(); // Array van ID's van teamleaden team 2 in de
buurt van checkpoint 2
private static List<Speler> cp3t2 = new List<Speler>(); // Array van ID's van teamleaden team 2 in de
buurt van checkpoint 3

private const int INEUTRAL = 50;
private const int ICAPTURED = 25;

private static int scoreT1 = 0;
private static int scoreT2 = 0;
```

Constructor:

```
static void Main(string[] args)
        {
```

```
            //Sent C2D
            serviceClient = ServiceClient.CreateFromConnectionString(connectionString);

            //Receive D2C
            eventHubClient = EventHubClient.CreateFromConnectionString(connectionString,
iotHubD2cEndpoint);

            var d2cPartitions = eventHubClient.GetRuntimeInformation().PartitionIds;

            foreach (string partition in d2cPartitions)
            {
             ReceiveD2CBegin(partition);
            }
         }
```

Sent C2D

```
private static async Task SendCloudToDeviceMessageAsync(string data)
        {
            MessageObject mo = JsonConvert.DeserializeObject<MessageObject>(data);
            string name = mo.PersonName;
            int team = mo.Team;
            var C2Dmessage = new
            {
                naam = name,
                team = team
            };

            var messageString = JsonConvert.SerializeObject(C2Dmessage);
            var commandMessage = new Message(Encoding.ASCII.GetBytes(messageString));
            foreach(var d in DeviceIds)
            {
                await serviceClient.SendAsync(d, commandMessage);
            }
        }
```

Receive D2C

```
private static async Task ReceiveD2CBegin(string partition)
        {
            var eventHubReceiver = eventHubClient.GetDefaultConsumerGroup().CreateReceiver(partition,
DateTime.UtcNow);
            while (true)
            {
                EventData eventData = await eventHubReceiver.ReceiveAsync();
                if (eventData == null) continue;

                string data = Encoding.UTF8.GetString(eventData.GetBytes());
                Console.WriteLine(string.Format("Message received.", partition, data));
            }
        }
```

Speler toevoegen

```
private static void VoegSpelerToe(MessageObject mo)
        {
            int count = team1.Count() + team2.Count();
            if (mo.Team == 1 && mo.PersonName != null && mo.Team != -1)
            {
                Speler s = new Speler(count, mo.PersonName, 1);
                team1.Add(s);
            }
            if (mo.Team == 2 && mo.PersonName != null && mo.Team != -1)
            {
                Speler s = new Speler(count, mo.PersonName, 2);
                team2.Add(s);
            }
        }
```

Kleur Hue bepalen:

```
private static void LampStatus()
```

```
        {
            if (cp1 == 100)
            {
                WriteToHue(47920, 50, 255, lamp1);
            }
            else if (cp1 == 0)
            {
                WriteToHue(50, 50, 255, lamp1);
            }
            else
            {
                WriteToHue(0, 0, 0, lamp1);
            }
  }
```

Van wie is checkpoint

```
private static void WhoHasCp1()
        {
            if (cp1t1.Count > cp1t2.Count)
            {
                if (cp1 <= 50) cp1 = cp1 - ICAPTURED;
                else cp1 = cp1 - INEUTRAL;
            }
            if (cp1t1.Count < cp1t2.Count)
            {
                if (cp1 < 50) cp1 = cp1 + INEUTRAL;
                else cp1 = cp1 + ICAPTURED;
            }

            if (cp1 < 0)
            {
                cp1 = 0;
            }
            else if (cp1 > 100)
            {
                cp1 = 100;
            }

            if (cp1 == 0) scoreT1++;
            if (cp1 == 100) scoreT2++;

            //anders evenveel mensen van beide op het punt
        }
```

WriteToHue

```
private static void WriteToHue(int hue, int brightness, int saturation, int lamp)
        {
            try
            {

                var httpWebRequest =
(HttpWebRequest)WebRequest.Create("http://172.23.190.22/api/janholbrouck/lights/" + lamp + "/state");
                httpWebRequest.ContentType = "application/json";
                httpWebRequest.Method = "PUT";

                using (var streamWriter = new StreamWriter(httpWebRequest.GetRequestStream()))
                {
                    string json = "{\"on\":true, \"hue\":" + hue + ", \"bri\":" + brightness + ",
\"sat\":" + saturation + ", \"transitiontime\":1}";

                    streamWriter.Write(json);
                    streamWriter.Flush();
                    streamWriter.Close();
                }

                var httpResponse = (HttpWebResponse)httpWebRequest.GetResponse();
                using (var streamReader = new StreamReader(httpResponse.GetResponseStream()))
                {
                    var result = streamReader.ReadToEnd();
```

```
            }
        }
        catch
        {
            Console.WriteLine("Error WriteToHue!");
        }
    }
```

Persoon aan checkpoint toevoegen:

```
if (team == 1)
            {
                if (team1.Contains(speler))
                {
                    switch (cp)
                    {
                        case -3:
                                if (cp3t1.Contains(speler))
                                {
                                    cp3t1.Remove(speler);
                                }

                            break;
                        case -2:
                                if (cp2t1.Contains(speler))
                                {
                                    cp2t1.Remove(speler);
                                }
                            break;
                        case -1:
                                if (cp1t1.Contains(speler))
                                {
                                    cp1t1.Remove(speler);
                                }
                            break;
                        case 1:
                            if (!cp1t1.Contains(speler))
                            {
                                cp1t1.Add(speler);
                            }
                            break;
                        case 2:
                            if (!cp2t1.Contains(speler))
                            {
                                cp2t1.Add(speler);
                            }
                            break;
                        case 3:
                            if (!cp3t1.Contains(speler))
                            {
                                cp3t1.Add(speler);
                            }
                            break;
                    }
                }
            }
```

18 April

We kunnen de Philips Hue aanspreken met de ‘Centrale’ console applicatie. Dit gebeurd met C# en niet met Processing, omdat onze applicatie werkt in C#. Met Processing kunnen we de IoT Hub niet aanspreken dus hebben we ons volledige project in C# moeten programmeren. Nu lichten de Hue’s op vanuit die console app.
Hierdoor kunnen we het signaal, als een smartphone in de buurt van een beacon komt, omzetten in de Hue die oplicht. Een probleem waarmee we te maken hebben is de volgende: De beacons zenden hun signaal niet snel genoeg uit waardoor onze lichtjes pas na enige tijd veranderen van status.

Na wat opzoekwerk blijkt het dat we een andere configuratie kunnen geven aan de beacons. Dit zullen we in een volgend labo bekijken.

Als we ons project grondig willen testen hebben we minimum twee spelers nodig. Daarom hebben we een Windows 10 Phone bij Rita uitgeleend. Het lukte echter niet om onze app op daarop te deployen. Na uren opzoekwerk en tests is dit nog steeds niet gelukt.

25 April

Om het spel goed te kunnen testen hebben we minstens twee Windows 10 phones nodig. We hebben er eentje geleend bij Rita, maar het lukte niet om onze app daarop te deployen. Na veel opzoekingswerk en updates uit te voeren van usb-drivers en Visual Studio lukte het uiteindelijk wel.

Het testen van de communicatie tussen device en console (cloud) met twee smartphones was heel wat complexer dan met maar één apparaat. Enkele aanpassingen in de code hebben dit probleem verholpen.

Als de communicatie goed zat konden we eindelijk beginnen met het testen van het volledige project. Hierbij zijn we op behoorlijk wat problemen gestoten. Dit waren vooral logica foutjes in de code van het spel. Er werd bepaalde data verkeerd doorgestuurd naar de console.
Het vinden van deze fouten was niet gemakkelijk en vergde heel wat debug werk.

26 April

Voor de zekerheid nog een eindtest gedaan zonder de Hue's. We hebben de smartphone geleend van iemand uit onze klas, en een eind test gedaan. Alles werkte vlot en er zijn geen problemen meer opgetreden.

Conclusie

We vonden het een zéér leerrijk en interessant project. Dit is ook de reden dat we er meer dan dubbel zoveel tijd ingestoken hadden dan er voorzien was (3 labo's). We hebben in dit project met allerlei nieuwe zaken leren werken, zoals de IoT hub en de beacons. Hier zijn uiteraard soms problemen bij komen kijken, maar dat hoort erbij. We hebben van dit project een zeer uitgebreid iets gemaakt, met een basis waarop we eventueel na deze lessen mee voort kunnen. Het ontwikkelde spel is perfect speelbaar, en na nog wat aanpassingen zouden we het nog beter kunnen maken. We hebben hier zeer veel tijd en moeite ingestoken, maar zijn dan ook zeer trots op ons eindresultaat.