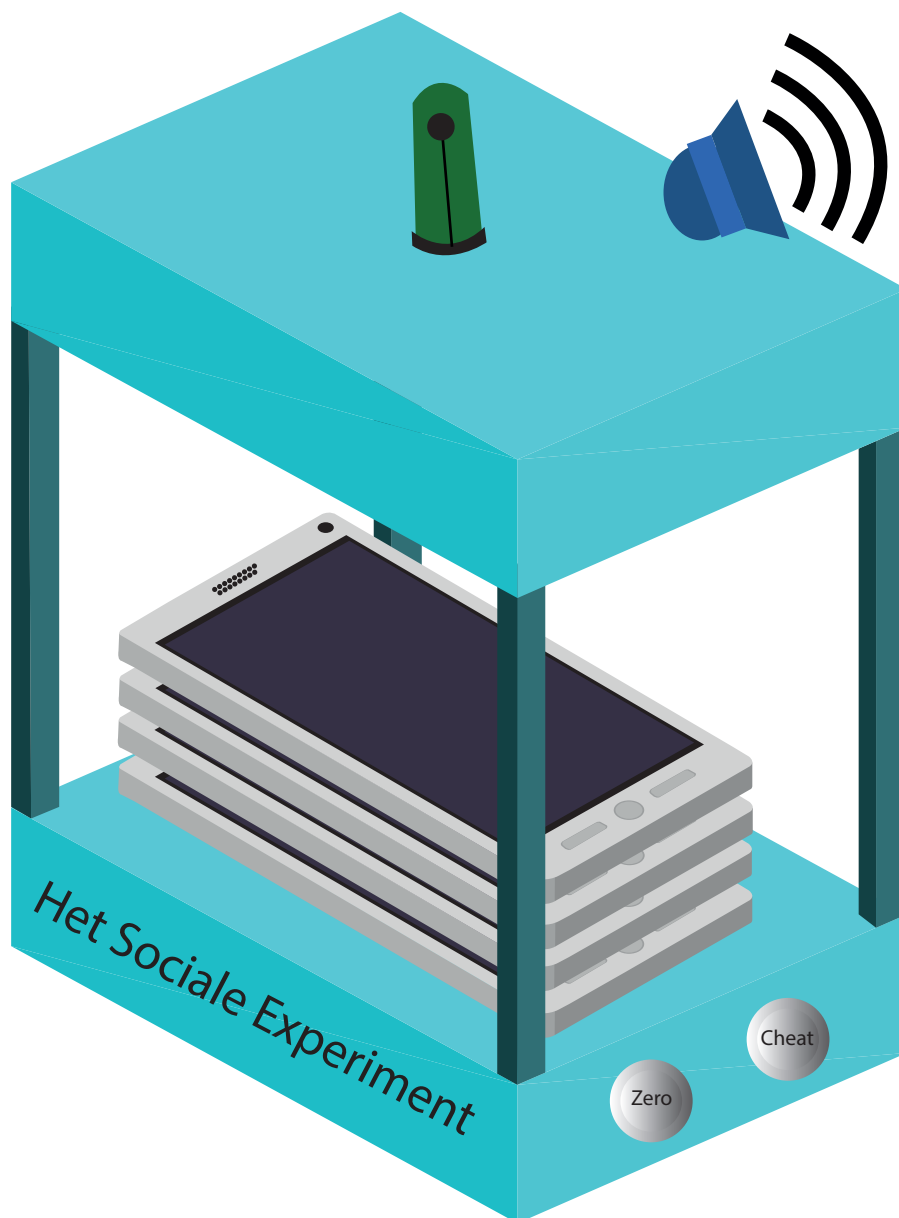


# UX Verantwoording

Ubicomp



Voorblad	Blz. 1
Inhoudsopgave	Blz. 2
'Het Sociale Experiment'	Blz. 3
Visualisatie van de ideale UX	Blz. 4
UX Principles	Blz. 5
Interactieontwerp	Blz. 6
Persuasion	Blz. 7
Systeemmodel	Blz. 8
Arduino Code	Blz. 9
Link naar filmpje van de werking	Blz. 10

# 'Het Sociale Experiment'

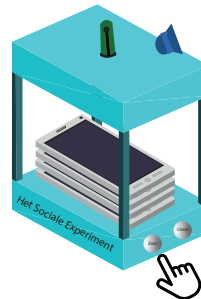
Je kent het wel, je zit 's avonds met z'n allen op de bank TV te kijken. Halverwege een TV-show of film gaat een van je familie leden op zijn of haar telefoon zitten Whatsappen of Facebooken. Is dat niet irritant? Nu is daar een oplossing voor! 'Het Sociale Experiment'.

Het zogenaamde Sociale Experiment zorgt ervoor dat je familieleden nooit meer op zijn of haar telefoon zitten tijdens het gezellig samen TV-kijken. Het Sociale Experiment Meet of alle telefoons in de machine liggen. Als alle telefoons er liggen is de machine blij. Maar zodra een familielid ook maar even zijn of haar mobiel uit de machine pakt om iets te doen op zijn of haar telefoon begint de machine te tellen. De machine meet hoelang deze persoon op zijn of haar mobiel zit te staren. Na enkele minuten wordt de machine geïrriteerd en er klinkt een naar geluidje. Als dan een tweede familielid zijn of haar mobiel pakt uit de machine wordt het 'Sociale Experiment' steeds meer geïrriteerd. Als er na een tijdje nog steeds niet alle telefoons in de machine liggen of er ligt zelfs helemaal geen telefoon, gaat de TV uit. Zo simpel is het.

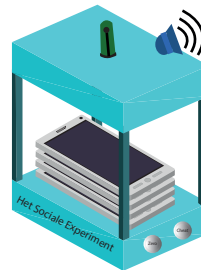
# Visualisatie van de ideale UX

De machine heeft verschillende states:

- **Zero-state:** Leg alle telefoons in de machine voordat je TV gaat kijken en de machine weet hoeveel mensen er TV gaan kijken. Druk op de knop voor het vaststellen.



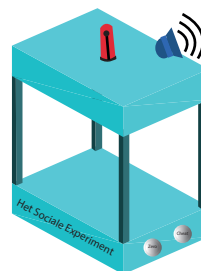
- **Blij-state:** Alle telefoons liggen in de machine. Er klinkt een vrolijk melodietje



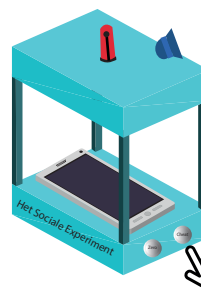
- **Geïrriteerde state:** 1 of meer telefoons liggen niet meer in de machine en de machine raakt geïrriteerd en maakt irritante geluidjes.



- **Boze-state:** Na een tijd liggen alle telefoons er nogsteeds niet of er liggen helemaal geen telefoons, de machine wordt boos, er klinkt een boos-melodietje en de TV valt uit.



- **Cheat-state:** Druk op de cheat knop en het is toegestaan om je telefoon uit de machine te halen als er een noodgeval is.



UX Principles van Donald Norman:

## **Affordance:**

'Het Sociale Experiment' is een eenvoudige machine. De gebruiker wordt de opdracht gegeven om alle telefoons in de machine te leggen zodat de machine niet boos wordt en de TV uitvalt. De machine heeft een plek waar telefoons op kunnen liggen. Boven de telefoons hangt een afstandmeter die meet aan de hand van de afstand hoeveel telefoons er liggen.

## **Mapping:**

De machine heeft 2 knoppen: Zero (om het aantal telefoons te bepalen) en Cheat (om de machine tijdelijk op pauze te zetten in noodgevallen).

## **Visibility:**

De gebruiker weet door het gering aan knoppen makkelijk hoe de machine moet worden gebruikt. Ook is duidelijk te zien waar de output-devices van de machine zitten en wat de doen. Het RGB-ledlampje steekt aan de voorkant uit. En de speaker waar het melodietje uitkomt zit aan de bovenkant.




## **Constraints:**

De gebruiker kan met de machine drie simpele dingen doen. Door op de zero-knop te drukken het aantal telefoons vast te stellen. Door op de Cheat knop de machine tijdelijk stil te zetten. En door telefoons in de machine te leggen of eruit te halen waardoor de machine van state veranderd.

# Interactieontwerp

## Micro-Interaction Template

Device: 'Het Sociale Experiment'  
 User Goal: Met de hele familie sociaal TV kunnen kijken zonder dat er iemand asociaal op zijn telefoon gaat zitten.

	 TRIGGER	 RULES	 FEEDBACK
State 1: Zero-state	Het Zero-knopje wordt ingedrukt.	Meet aan de hand van de afstand tot de bovenste telefoon hoeveel telefoons er liggen.	Het groene lampje brandt en er klinkt een vrolijk melodietje.
State 2: Blij-state	Alle telefoons liggen in de machine.	De afstand tot de bovenste telefoon is hetzelfde als bij de Zero-state.	Het groene lampje brandt en er klinkt een vrolijk melodietje.
State 3: Geïrriteerde-state	1 of meer telefoons liggen niet meer in de machine.	De afstand tot de bovenste telefoon is groter dan bij de Zero-state.	Het blauwe lampje knippert en er klinkt een irritant melodietje.
State 5: Boze-state	Na een tijdje liggen nog steeds niet alle telefoons in de machine of er ligt helemaal geen telefoon.	De afstand tot de bovenste telefoon is net zo groot als 1 telefoon of net zo groot als geen telefoons dan bij de Zero-state.	Het rode lampje knippert en er klinkt een boos melodietje. De TV valt uit.
State 1: Cheat-state	Het Cheat-knopje wordt ingedrukt.	De machine wordt tijdelijk uitgeschakeld totdat er weer op de knop wordt gedrukt.	Het groene lampje brandt.

De Persuasion principes die toegepast zijn op 'Het Sociale Experiment' die gedragsverandering op een positieve manier veranderen zijn:

**Trigger:**

De gebruiker van 'Het Sociale Experiment' wordt getriggerd nadat de machine een geluidje maakt. De gebruiker hoort een irritant melodietje en krijgt een trigger om zijn of haar telefoon weer in de machine te leggen om het melodietje te stoppen.

**Surprise:**

De gebruiker van 'Het Sociale Experiment' wordt verrast als alle telefoons in de machine liggen. De machine speelt een leuk melodietje af als alle telefoons aanwezig zijn. Dit doet de machine herhalend om de aantal minuten om aan te geven dat iedereen sociaal bezig is.

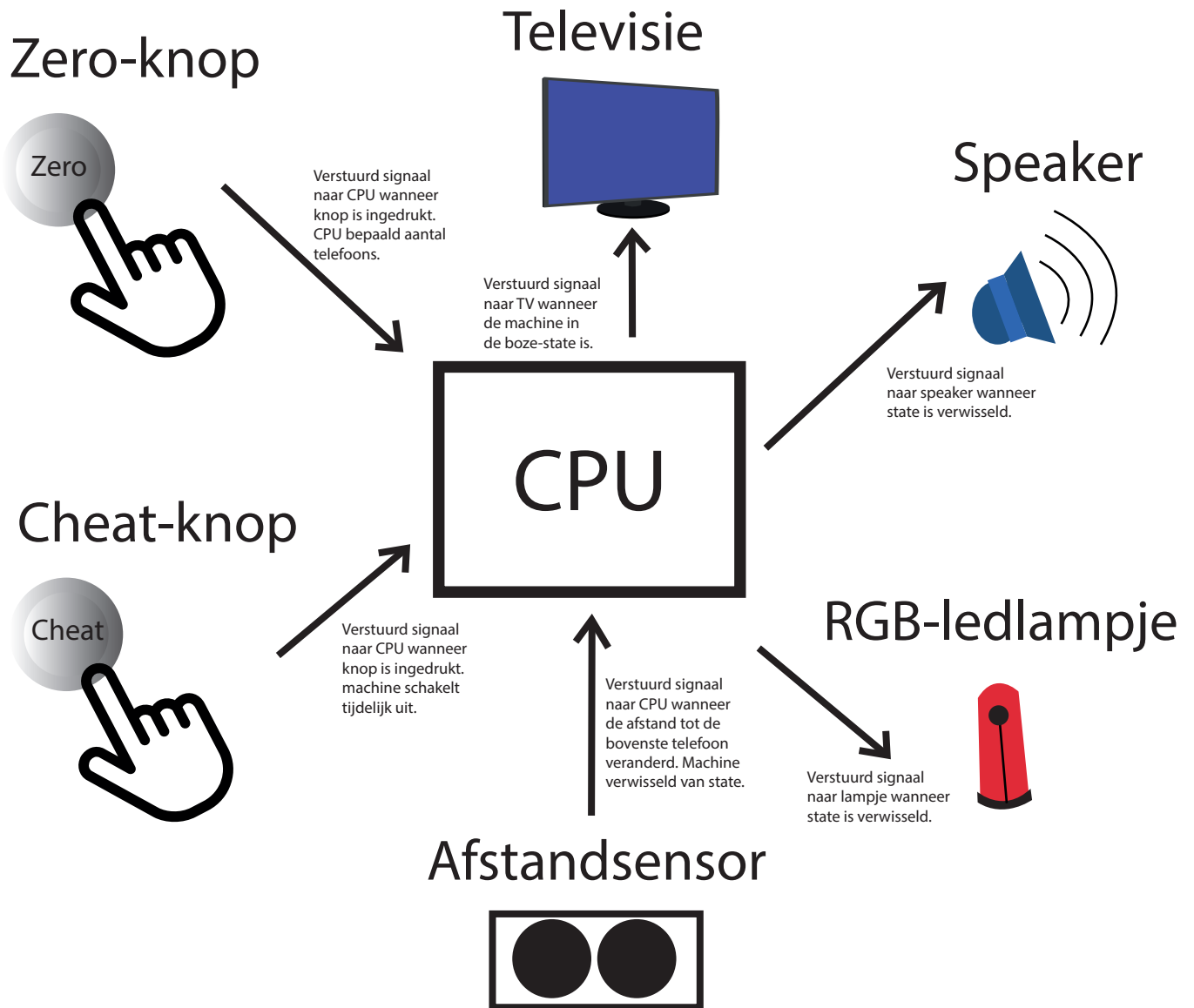
**Commitment & Consistency:**

Als de machine in de geïrriteerde-state zit speelt hij om de aantal seconden een irritant melodietje af. Dit doet de machine om aan te geven dat niet iedereen sociaal is en dat de machine dus geïrriteerd raakt. De machine stopt pas als alle telefoons weer in de machine liggen.

**Sensory appeal:**

De machine speelt melodietjes af maar heeft ook een RGB-ledlampje om aan te geven in welke state de machine is.

## Functie: Het Sociale Experiment





## Arduino Code

```

#define trigPin D5 // Het definiëren van de trigger-pin van de HCSR-04 afstandmeter, deze staat op D5.
#define echoPin D6 // Het definiëren van de Echo-pin van de HCSR-04 afstandmeter, deze staat op D6.

void setup() {
  Serial.begin(115200); // De setup.
  pinMode(trigPin, OUTPUT); // Het aantal baud dat ook in de Serial Monitor staat.
  pinMode(echoPin, INPUT); // Duidelijk maken dat de Trigger-pin als output dient.
  pinMode(D1, OUTPUT); // Duidelijk maken dat de Echo-pin als input dient.
  pinMode(D2, OUTPUT); // Rode kleur van het RGB-ledlampje.
  pinMode(D3, OUTPUT); // Groene kleur van het RGB-ledlampje.
  pinMode(D4, OUTPUT); // Blauwe kleur van het RGB-ledlampje.
}

void loop() {
  long duration, distance; // De loop.
  digitalWrite(trigPin, LOW); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.
  delayMicroseconds(2); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.
  digitalWrite(trigPin, HIGH); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.
  delayMicroseconds(10); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.
  digitalWrite(trigPin, LOW); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.
  duration = pulseIn(echoPin, HIGH); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.
  distance = (duration/2) / 29.1; // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.

  Serial.print(distance); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.
  Serial.println(" cm"); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.

  delay(500); // Aangeleverde code in het 'Aansluiten sensoren (input)'-document.

  if (distance < 8) { // If-statement met de waarde 'distance' is kleiner dan 8cm (alle telefoons).
    digitalWrite(D1, LOW); // Rode kleur van het RGB-ledlampje is uit.
    digitalWrite(D2, HIGH); // Groene kleur van het RGB-ledlampje is aan.
    digitalWrite(D3, LOW); // Blauwe kleur van het RGB-ledlampje is uit.

    tone(D8, 2000, 200); // Eerste toon van het vrolijke melodietje. D8 is de pin, 2000 is de frequentie en 200 is het aantal miliseconden.
    delay(300); // Een delay van 300 miliseconden voordat de volgende toon klinkt.
    tone(D8, 2200, 200); // Tweede toon van het vrolijke melodietje. D8 is de pin, 2200 is de frequentie en 200 is het aantal miliseconden.
    delay(300); // Een delay van 300 miliseconden voordat de volgende toon klinkt.
    tone(D8, 2400, 200); // Derde toon van het vrolijke melodietje. D8 is de pin, 2400 is de frequentie en 200 is het aantal miliseconden.
    delay(300); // Een delay van 300 miliseconden voordat de volgende toon klinkt.
    tone(D8, 2600, 200); // Vierde toon van het vrolijke melodietje. D8 is de pin, 2600 is de frequentie en 200 is het aantal miliseconden.
    delay(10000); // Een delay van 10000 miliseconden voordat het melodietje weer klinkt.
  }

  else if (distance > 13) { // Else if-statement met de waarde 'distance' is groter dan 13cm (1 of geen telefoons).
    digitalWrite(D1, HIGH); // Rode kleur van het RGB-ledlampje is aan.
    digitalWrite(D2, LOW); // Groene kleur van het RGB-ledlampje is uit.
    digitalWrite(D3, LOW); // Blauwe kleur van het RGB-ledlampje is uit.
    delay(100); // Een delay van 100 miliseconden voordat het rode lampje weer uitgaat.
    digitalWrite(D1, LOW); // Rode kleur van het RGB-ledlampje is uit.
    delay(100); // Een delay van 100 miliseconden voordat het rode lampje weer aangaat.

    tone(D8, 700, 400); // Het boze melodietje. D8 is de pin, 700 is de frequentie en 400 is het aantal miliseconden.
    delay(700); // Een delay van 700 miliseconden voordat het boze melodietje weer klinkt.
  }

  else { // Else-statement zonder waarde (2 of meer telefoons).
    digitalWrite(D1, LOW); // Rode kleur van het RGB-ledlampje is uit.
    digitalWrite(D2, LOW); // Groene kleur van het RGB-ledlampje is uit.
    digitalWrite(D3, HIGH); // Blauwe kleur van het RGB-ledlampje is aan.

    tone(D8, 500, 400); // Eerste toon van het nare melodietje. D8 is de pin, 500 is de frequentie en 400 is het aantal miliseconden.
    delay(500); // Een delay van 500 miliseconden voordat de volgende toon klinkt.
    tone(D8, 300, 1000); // Tweede toon van het nare melodietje. D8 is de pin, 300 is de frequentie en 1000 is het aantal miliseconden.
    delay(6000); // Een delay van 6000 miliseconden voordat het melodietje weer klinkt.
  }
}

```

# Link naar filmpje van de werking

<https://youtu.be/u4ryYP1UoKc>