

# To bit or not to bit

**Alexandre** Truong  
**Edgar** Ricardo Corona Mendoza  
**Julie** Abdelnour  
**Léopold** Verhaeghe

Semestre **P19**

# Table des matières

<b>Introduction</b>	<b>3</b>
<b>Cahier des charges</b>	<b>6</b>
<b>État des arts</b>	<b>8</b>
<b>Design</b>	<b>10</b>
<b>Scénario</b>	<b>12</b>
<b>Architecture logicielle</b>	<b>14</b>
<b>Gestion graphique</b>	<b>19</b>
<b>Enregistrement audio</b>	<b>22</b>
<b>Conclusion</b>	<b>23</b>
<b>Annexes</b>	<b>24</b>

# Introduction

## Concept

À l'ère du numérique dans laquelle nous vivons et apprenons à vivre, les smartphones et tablettes et autres appareils occupent une place à l'importance grandissante. Aujourd'hui partie intégrante, voire centrale de notre quotidien, ils nous permettent non plus seulement de communiquer, mais aussi de nous informer, de nous divertir et de nous accompagner dans notre travail. Les outils numériques sont ainsi devenus de parfaits *outils*, si bien que l'homme moyen en France passerait plus de 4 heures par jour devant les écrans. De plus en plus, ces outils sont capables de nous parler et même d'écouter nos demandes. Pourtant, ils ne restent que de simples *outils*.

Ce projet de théâtre numérique propose à l'utilisateur de percevoir d'une nouvelle façon ces objets en traitant de la problématique de *présence numérique*.

Au menu de la pièce, nous suivons le réveil et les interrogations de personnages sur leur réelle identité.

À la distribution, smartphones et tablettes interpréteront des victimes en plein désarroi tandis que le spectateur, comme il est d'usage, reprendra son rôle habituel de maître de ces appareils.

## Objectif

Cette expérience narrative d'un genre nouveau est avant tout une expérimentation et une exploration des possibilités du numérique et de nos appareils du quotidien qui cherche à leur offrir une véritable « présence numérique ». À travers cette histoire plus particulièrement, nous cherchons à décrire ce qu'il se passerait si ces appareils se découvraient soudainement une conscience semblable à la nôtre.

Tout d'abord ignorants de leur réelle identité, les personnages divagueront lors d'une discussion usuelle sur leur réelle nature et utilité dans ce monde.

Ils chercheront à communiquer avec le spectateur grâce aux capacités et outils qu'ils possèdent (des microphones, des haut-parleurs, des capteurs) afin d'arriver à l'ultime révélation.

Les interactions habituelles que nous avons avec ces outils prendront ainsi un sens nouveau, bien loin de leur signification originale. Ces "acteurs" pourront ainsi réagir aux stimuli de lumière, de son ou de toucher que perçoivent leurs capteurs, chacun soulevant en leur "esprit" de nouvelles questions.

## **Public cible**

Conçue pour être une expérience interactive artistique et ludique, ce projet de théâtre numérique s'adresse aux usagers du numérique dans leur globalité.

**Behind the scenes**  
*La théorie*

# Cahier des charges

L'objectif de ce projet est de concevoir une application pour terminaux mobiles offrant à l'utilisateur un spectacle technologique, artistique et - de façon assez limitée - interactif.

## Spécificités graphiques

Gestion graphique assez minimaliste dans l'ensemble.  
Lorsqu'un appareil donne une réplique, l'écran s'allume avec une intensité lumineuse dépendant de l'amplitude de sa voix.

## Formes et degrés d'interactivité

La plupart des interactions se font entre les deux appareils lorsqu'ils communiquent entre eux.

Un degré limité d'interaction peut néanmoins être intéressant. En fin de spectacle, un choix peut ainsi être proposé au spectateur, selon lequel sera déterminé le dénouement de l'histoire.

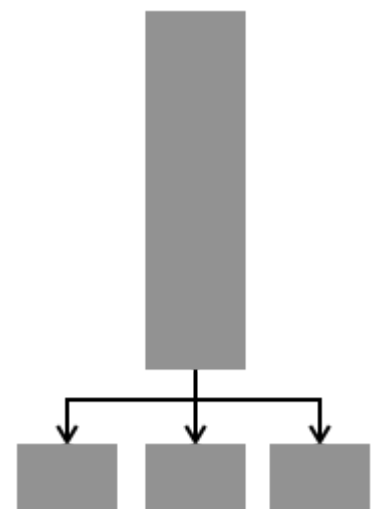
Une autre forme d'interaction existe cependant : une interaction machine-machine. Les deux dispositifs s'envoient mutuellement des signaux pour prévenir l'autre appareil de débiter la prochaine réplique lorsqu'ils ont terminé la leur.

Cette interaction pouvant prendre la forme d'une entrée tactile ou d'une saisie vocale agirait ainsi comme une forme de **navigation** rendant le spectacle interactif.

## Structure et navigation

La navigation se fait globalement sans action de l'utilisateur.  
Au premier lancement de l'application, il sera nécessaire de régler des paramètres de connexion et de sécurité (accès au micro, par exemple).  
La narration se fait ensuite entre les appareils de façon autonome, en suivant une trame linéaire.  
À la fin du spectacle, plusieurs fins sont possibles selon l'action de l'audience.

Fig.1: Schéma de la narration



## **Spécificités techniques**

Développé en React native, ce projet doit offrir des fonctionnalités identiques sur iOS et sur Android.

Afin de nécessiter un coût de maintenance minimum (nul idéalement), nous souhaitons permettre la communication sans nécessiter de serveurs externes par lesquels transitent l'information.

# État des arts

Nous explorerons ici diverses technologies permettant la communication entre terminaux mobiles que nous avons considérées dans le cadre de notre projet.

## - Bluetooth / BLE

Lorsque nous pensons à l'envoi de données situés à proximité, nous pensons le plus souvent au Bluetooth.

Cette technologie supportée par pratiquement tous les appareils Android et iOS est notamment utilisée pour connecter un téléphone mobile à des accessoires tels des écouteurs ou enceintes sans fils, ou -plus rarement- pour envoyer des fichiers entre terminaux. Nous pouvons distinguer la technologie Bluetooth, énergivore car utilisée pour envoyer un volume important de données du Bluetooth Low Energy (BLE) qui permet la communication de courts messages et permet la connexion à de multiples appareils. Dans le cas présent, la communication servant de simple indicateur pour déterminer le début d'une réplique, seule une faible quantité de données doivent être échangées. Nous préférons ainsi la solution du BLE.

Après des recherches sur ce sujet, nous avons trouvé une multitude d'APIs React Native assurant la communication par BLE, telles `react-native-ble-manager`. En revanche, nous nous sommes plus tard rendus compte que cette outil ne serait pas viable dans notre cas. En effet, lors d'une communication Bluetooth entre deux appareils, deux rôles doivent être définis: celui d'appareil *central* et celui de *périphérique*. Le premier, parfois appelé appareil "maître" initie les demandes de communication tandis que le second, appareil "esclave", qui est chargé d'y répondre. Ces APIs, cependant, ne supportent que le rôle de *central*, étant destinés à permettre la connexion à des appareils secondaires (accessoires, IOT), et non des téléphones portables. Bien qu'il existe des outils permettant de faire traiter l'appareil comme un périphérique, du fait d'une faible demande, ces derniers sont le plus souvent incomplets (manque de fonctionnalités, pas cross-plateformes) et rarement maintenus à jour.

De façon générale, il semblerait que, si les appareils sous iOS supportent le rôle de périphérique, seul un faible nombre de terminaux Android en soient capables. Souhaitant concevoir une expérience cross-plateforme (iOS/Android), nous considérons que le Bluetooth ne correspond pas à nos besoins.

## - Google Nearby

Cette technologie permet théoriquement d'établir une connexion entre des appareils de façon simple par le biais d'une multitude de capteurs et d'émetteurs (son, bluetooth, wifi, etc). Ce système est composé de deux APIs, l'une (*Nearby Connection*) permettant la connexion entre terminaux sur un même réseau, tandis que l'autre (*Nearby Messages*) nécessite un accès à internet mais n'est pas limitée aux appareils d'un même réseau.



*Nearby Connection* limite ainsi les cas d'utilisation (en nécessitant de se connecter à un même réseau Ethernet ou Wifi), mais cela se traduit également par une utilisation moindre en énergie, en évitant l'utilisation du Bluetooth. Cette API n'étant toutefois disponible que sur Android, nous ne considérerons que *Nearby Messages*.

- **Wifi**

Le wifi est une option intéressante, moins coûteuse en énergie que le Bluetooth et généralement plus commune et mieux supportée que Nearby. Il n'est en effet pas difficile pour des appareils de détecter et communiquer avec les terminaux présents sur un même réseau, comme nous faisons pour commander une imprimante à distance.

iOS et Android possèdent des bibliothèques de communication peer-to-peer chargée de ces tâches mais leurs implémentations diffèrent.

Nous préférons ici utiliser la bibliothèque Google Nearby, permettant une utilisation offline (en wifi mais sans connexion internet).

# Design

## Design graphique

### - Variation de la luminosité des pixels en fonction du volume

Afin d'éviter une trop grande monotonie (l'utilisateur n'étant pas sollicité pendant la plus grande partie de l'expérience), il nous a paru intéressant de modifier l'affichage à l'écran en fonction du contenu, plus précisément en liant la luminosité des pixels au volume de l'audio. Cela permet également de supporter l'idée que le terminal est bien celui qui parle, qu'il ne sert pas qu'à faire transiter les données passivement mais qu'il a au contraire un rôle actif.

### - Variation de la teinte des pixels en fonction de la fréquence

Nous avons évoqué la possibilité de faire varier l'affichage à l'écran en fonction d'une seconde variable: la fréquence sonore. Souhaitant un affichage simpliste pour notre projet, il n'était pas question d'ajouter des éléments à l'écran mais seulement de modifier les propriétés des pixels à l'écran. Puisque nous modifions déjà la luminosité des pixels, nous ne pouvions influencer que sur leur teinte et leur saturation.

Pour avons finalement décidé d'abandonner cette idée, qui ajouterait de la complexité au projet sans apporter de valeur significative. Au contraire, la couleur est généralement associée aux émotions: modifier la teinte selon la fréquence indépendamment des émotions risquerait de causer de l'incompréhension chez le spectateur.

## Design audio

En ce qui concerne la dimension sonore de notre projet, nous avons notamment hésité entre l'utilisation d'une synthèse vocale et l'enregistrement des voix.

Synthèse	Enregistrement
<ul style="list-style-type: none"><li>+ Effet robot mais pas trop</li><li>+ Aisance à modifier des enregistrements</li><li>+ Possibilité de synthétiser dans plusieurs langues</li> <li>- Peu de choix d'émotions (voire aucun)</li><li>- Monotonie</li></ul>	<ul style="list-style-type: none"><li>+ Variation d'émotions</li> <li>- Nécessite des acteurs</li><li>- Nécessite du matériel et de bonnes conditions d'enregistrement</li></ul>

Après réflexion, et afin de permettre un plus grand panel d'émotions, nous avons décidé de procéder par l'enregistrement de voix. Des effets seront ajoutés par la suite afin d'obtenir le rendu escompté.

Le choix d'avoir des voix enregistrées qui sont par la suite modifiées permet d'avoir un compromis entre voix de synthèse et voix réelle : le ton des répliques permet l'illusion d'une réelle conversation entre deux personnes, l'effet téléphone ainsi que l'effet glitch rappellent la réelle nature des protagonistes.

# Scénario

Ce scénario met en scène 2 protagonistes inédits : vos téléphones portables. Les voilà dotés de parole et de leur propre manière de penser ! Dans cette installation “théâtrale”, les deux appareils sont des amis de longue date se retrouvant lors d’un rendez-vous. La discussion tourne autour de banalités et de moqueries amicales, jusqu’au moment où cette conversation dérive sur des questions existentielles. Eux se posent la question de la possible existence d’entités supérieures les manipulant ; et le spectateur est amené à s’interroger sur si ces téléphones pensent qu’ils sont des humains coincés dans un appareil ou si ce ne sont que des machines convaincues elles-mêmes d’être humains. Les appareils tentent alors de communiquer avec le spectateur qui en réalité est lui aussi acteur de cette pièce. Le spectateur peut faire le choix d’interagir avec les appareils d’une certaine manière ou d’une autre, entraînant la révélation sur leur vraie nature, ou ne rien faire et les laisser vaquer à leurs occupations habituelles et laisser le doute planer dans un coin de leur mémoire.

## Écriture

L’interaction spectateur-téléphone se fait par contact physique. Lorsque le téléphone pose des questions, il affiche à l’écran un bouton indiquant au spectateur l’attente d’une interaction et la réponse à la question du téléphone. Le script a été écrit de manière à renvoyer l’impression d’une réelle conversation, en employant des expressions que l’on utiliserait habituellement à l’oral mais dont on aurait pas l’habitude à l’écrit. Certains éléments de l’histoire font écho au fait que ce sont des bien des téléphones qui parlent : les “rêves” font référence à des moments où leur propriétaire les manipulaient (appel téléphonique, navigation sur réseaux sociaux), et quelques répliques font référence au domaine de l’électronique.

L’écriture du scénario s’est principalement basée sur la capacité d’interaction que nous avons avec les smartphones d’aujourd’hui : que ce soit pour les input du téléphone en lui-même (reconnaissance vocale, écran tactile, bouton) et ses output (sons, écran visuel, vibreur), ces dispositifs offrent une grande liberté d’interaction et de notification que ce soit d’homme à machine ou de machine à homme. Avec les problématiques d’aujourd’hui de technologies qui évoluent toujours plus, l’idée de ce projet est principalement de se questionner sur la notion de présence numérique. Que se passe-t-il si nous donnons la capacité d’expression à ces outils du quotidien ? Comment réagiraient-ils et comment réagirions-nous ?

# **Behind the scenes**

*La pratique*

# Architecture logicielle

Il s'agit ici de définir l'architecture de l'application mobile, c'est-à-dire la façon dont s'organisent et interagissent les différents éléments de l'application afin de permettre son bon fonctionnement. Dans notre cas, cela implique notamment de définir le système de stockage de l'information et le processus de connexion des appareils.

## Stockage de l'information

Ici, nous entendons plus particulièrement par "stockage de l'information" l'organisation des données liées aux diverses répliques et scènes du projet, et non le stockage dynamique. En effet, notre projet ne nécessite pas d'ajouter des informations à notre base de données, mais seulement d'y accéder.

L'application étant développée en React Native, où le code source principal est écrit en Javascript, le stockage en JSON nous a rapidement paru être une solution intéressante.

En effet, "JSON" signifiant littéralement "Notation Objet JavaScript", il permet une communication sérialisée, très directe, entre le code interprété Javascript et l'information stockée. Cette solution nous évite d'avoir à établir un système de gestion de base de données plus ou moins complexe, typiquement en relationnel, qui ne serait pas nativement supporté par le langage.

Il s'agit d'un format de données textuelles permettant de stocker l'information de façon très structurée, suivant des règles fixées en amont. Nous organiserons ici l'information de chacun des passage à partir du modèle suivant:

```
{  "script": [
    {
      "title": "Titre du passage",
      "text": "Texte du passage",
      "audio": "audio_0.mp3",           //url de l'audio
      "actor": 1,                       //acteur qui joue ce passage
      "next": [                          //conditions pour poursuivre (optionnel)
        {
          "next_line": 1,                //index du passage suivant
          "condition": null              //type de condition
        }
        //autres conditions
      ]
    }
    //autres passages
  ]
}
```

## Processus de connexion et de communication

Nous nous intéresserons ici au processus de connexion entre terminaux, dans le sens de l'identification pour la communication et de la gestion des conflits de communication, la connexion en "bas niveau" étant déjà permise par l'API Nearby Messages (en Wifi, BLE et audio).

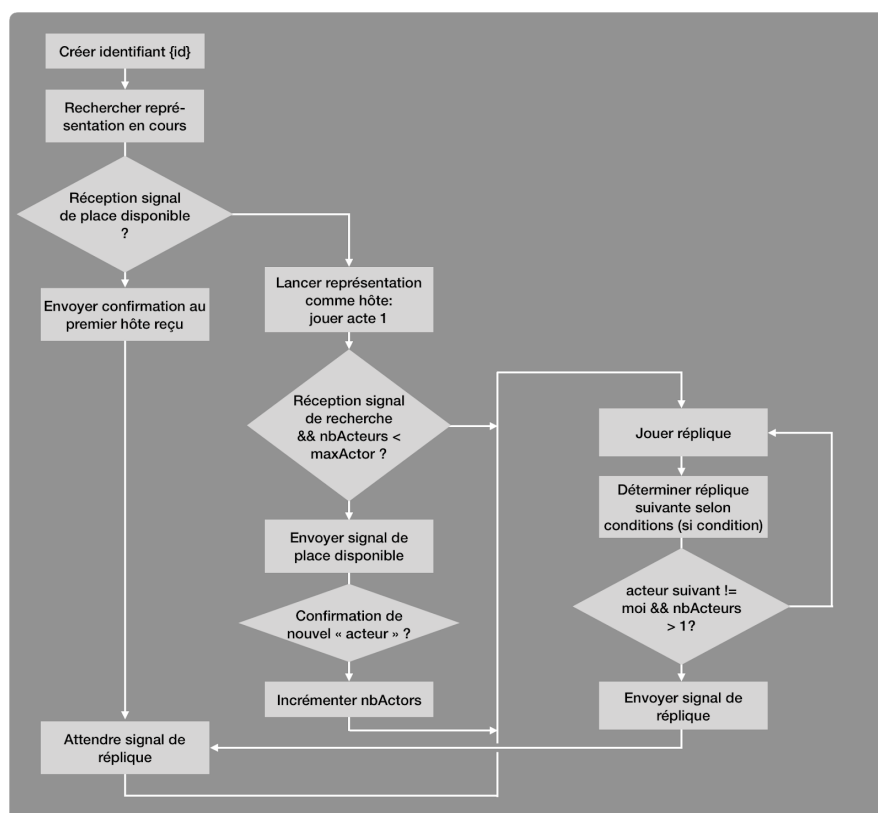
La technologie ici utilisée fonctionne sur le principe de publication et d'abonnement de façon générale, sans permettre de cibler des appareils en particulier, nous devons nous-mêmes implémenter un système de restrictions des communications. Sans ce dernier, deux représentations ayant lieu à proximité l'un de l'autre pourraient interférer l'un avec l'autre, par exemple en commandant à des appareils de la mauvaise représentation le lancement d'une scène.

Pour ce faire, nous commençons par introduire un outil d'identification.

Les instances des différentes applications étant toutes indépendantes (il n'y a pas de système central permettant de les gérer), nous ne pouvons pas déterminer un identifiant unique pour chaque appareil de façon certaine. Nous pouvons en générer aléatoirement une clé d'identification telle que la probabilité d'avoir des appareils avec le même identifiant soit pratiquement nulle. Cet identifiant devra être passé au cours de chaque communication afin de permettre d'une part d'indiquer un expéditeur, mais aussi de préciser un expéditeur afin que les appareils n'étant pas concernés par un message puissent simplement ignorer ce dernier.

La connexion et la communication des terminaux se fera donc suivant le schéma suivant (fig.1)

Fig.2: protocole de communication par message



La publication d'un message se fait suivant de la façon suivante, en utilisant la méthode `publish` de `Nearby Messages`:

```
nearbyAPI.publish(`${receiver} ${id} ${message}`);
```

Ce dernier pourra être perçu par tout terminal utilisant l'application et étant inscrit aux messages (avec `nearbyAPI.subscribe()`).

Tout message démarrera ainsi par l'identifiant du récepteur, permettant à un terminal de savoir s'il est ciblé par la publication ou non. Il est suivi de l'identifiant du récepteur et enfin du corps du message. Ce dernier sera composé d'un intitulé ("search", "join", "start", "liste\_acteurs") suivi, si besoin, d'une liste d'arguments, comme la liste des identifiants d'acteurs, dans le cas de "liste\_acteurs".

Ce processus de communication implémenté, nous avons pu tester un prototype de l'application permettant la communication et l'envoi de messages entre deux appareils Android et iOS. Si nous avons pu confirmer son bon fonctionnement, nous avons de sérieux doutes quant à ses performances.

En effet, la réception de chaque message se faisait après un délai non considérable (jusqu'à 5 secondes), en particulier lorsque l'envoi se faisait depuis un appareil Android.

Si ce délai n'est en absolu pas important, interrompre une conversation normale aussi longtemps à chaque réplique rallonge au final l'expérience de plusieurs minutes et nuit à sa fluidité.

Nous avons ainsi procédé à une estimation de la durée de l'expérience :

Base (avant branches) -> 1308 mots (48 répliques)

Branche "pas de contact" -> 150 mots (11 répliques)

Branche "contact" -> 319 mots (31 répliques)

En considérant une vitesse de 200 mots/min et un délais d'attente entre chaque réplique moyen de 3s, nous parvenons aux chiffres suivants:

=> Branche "pas de contact" -> 7:17 (répliques) + 2:57 (délais) = 10:14

=> Branche "contact" -> 8:07 (répliques) + 3:57 (délais) = 12:04

Ces délais d'attente pourraient ainsi occuper près d'un tiers du temps total de l'expérience, qui dépasserait les 10 minutes, trop long pour une installation de type artistique.

Il s'agit ainsi de réduire voire éliminer ce délai. Pour ce faire, plusieurs choix s'offrent à nous:

- éliminer l'envoi de messages et programmer les appareils de sorte à ce qu'ils lisent les répliques et attendent les durées théoriques de lecture des répliques lorsqu'un autre appareil doit lire.

Si cette solution est théoriquement simple à mettre en oeuvre et devrait être performante à condition que les appareils soient synchronisés au départ, il supprime l'aspect de communication qui est pourtant au coeur de notre projet.

- changer de technologie (par exemple, se connecter à un serveur externe).

Nous considérons cette option comme une solution de dernier recours, étant donné le temps de travail qu'elle impliquerait. Par ailleurs, il serait préférable pour notre application d'être indépendante de tout serveur externe, nécessitant des coûts de maintenance ainsi que des données mobiles du point de vue utilisateur.



- mesurer le délai de réception des messages et envoyer le signal de départ de chaque réplique en avance, en prenant en compte ce délai.
- Cette solution n'implique pas de modifications considérables à notre projet et devrait se montrer suffisant théoriquement. Il faudra néanmoins vérifier que le délai d'attente est stable. Cette solution montrerait également ses limites pour de durée inférieure au délai, cas dans lequel ce délai se fera tout de même ressentir.

Des tests nous indiquant que le délai d'attente n'étaient pas constant, nous avons abandonné la dernière idée (envoi à l'avance des messages). Nous avons alors décider de garantir une expérience sans délais par la première méthode en arrêtant d'utiliser les messages pour signaler le début d'une réplique selon le modèle suivant (fig.3).

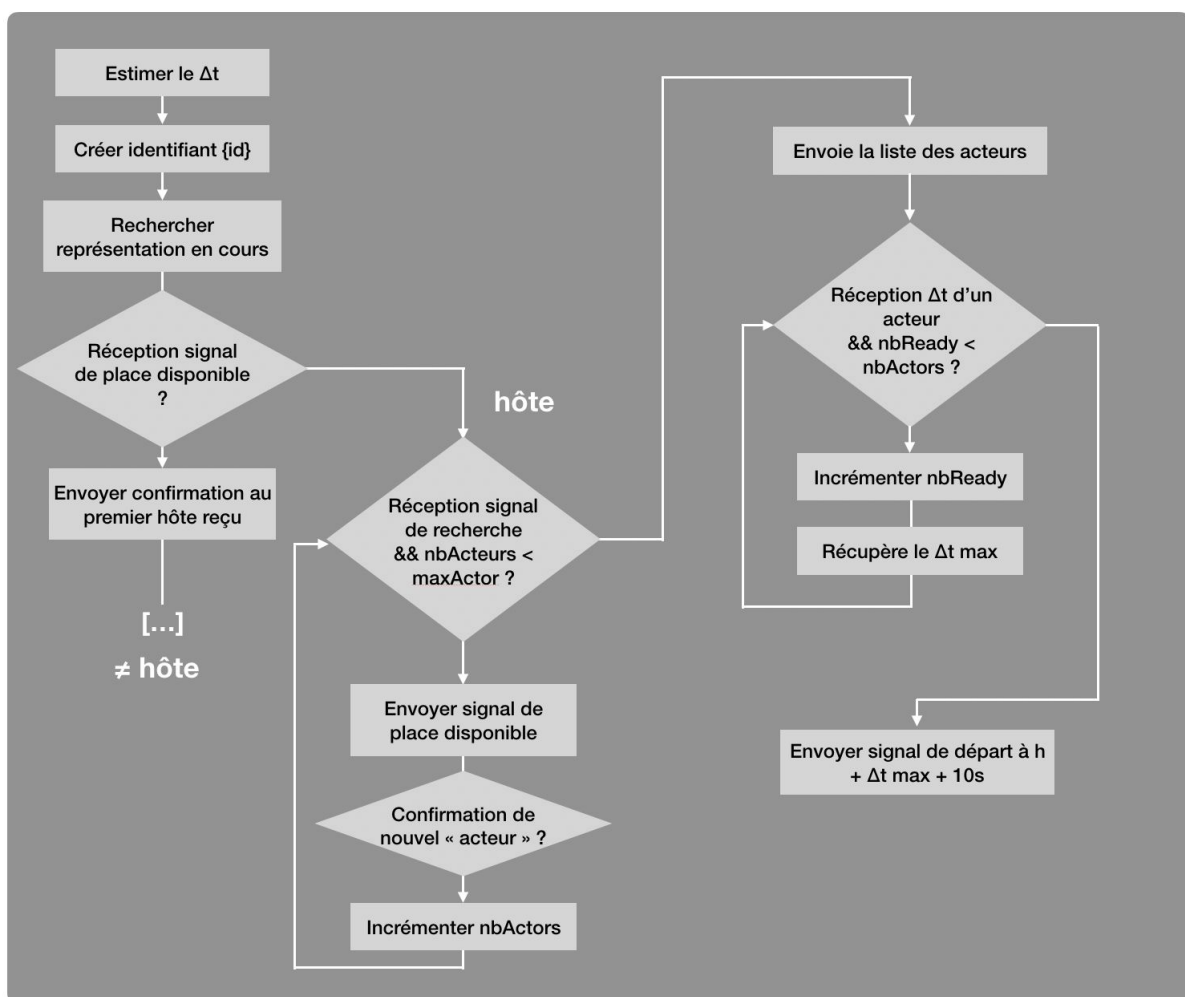


Fig.2: protocole de communication par synchronisation

Une fois le signal de départ envoyé, les différents acteurs calculent le temps à attendre avant leur prochaine réplique et lancent un setTimeout() afin de donner leur réplique au bon moment. Une fois cette réplique donnée, ils calculent à nouveau le temps avant leur

prochaine réplique et attendent. Cette opération est répétée jusqu'à être parvenu à la fin ou à une exception (comme une réplique à condition basée qui nécessite une interaction).

Cette méthode est assurée d'être performante à condition que les appareils soient bien synchronisés. Nous utiliserons ici l'horloge interne des appareils comme indicateur, ce qui nécessite donc une bonne synchronisation du temps. Si cela peut sembler naturel en théorie, il est pourtant nécessaire de procéder à des calculs supplémentaires, l'heure des terminaux Android n'étant en effet pas toujours fiable à la seconde près.

Afin d'assurer une bonne synchronisation des appareils, nous n'avons pas eu d'autre choix que de récupérer une heure de référence en ligne à l'aide de l'api *worldclockapi*.

Ceci fait, il n'est pas compliqué de calculer le décalage de l'horloge du terminal.

L'appareil hôte peut ainsi décider d'une heure de départ commune à tous les appareils.

Nous avons ainsi pu tester cette nouvelle méthode et attester de son efficacité.

À noter que la durée des fichiers audio est estimée à partir de la taille des fichiers, récupérée à l'aide de la bibliothèque de gestionnaire de fichiers *react-native-fs*. Cette estimation n'est toutefois valable qu'avec un bitrate constant (nombre de bits par seconde) et connu à l'avance.

# Gestion graphique

Il s'agit ici de faire correspondre l'image à l'écran au son joué par l'appareil à partir des fichiers audio. La couleur de l'écran doit s'éclaircir lorsque l'amplitude du son augmente. Cette tâche s'apparente de façon générale à la conception de "waveforms"/formes d'ondes, où nous cherchons à représenter l'amplitude de l'audio en fonction du temps.

Nous avons commencé en élaborant un prototype en Python à l'aide de différents modules comme pygame, pyaudio, audioop et wave. Ce premier programme nous a permis d'avoir un aperçu du rendu visuel ainsi que de créer un fichier .json pour faire l'affichage sur le téléphone.

Chaque module a une utilité précise :

- pygame a permis de créer une fenêtre avec la variation de nuance de gris du fond,
- pyaudio a permis la lecture de l'audio,
- audioop a permis de manipuler les données brut de l'audio pour récupérer l'amplitude du son à chaque instant,
- wave permet d'ouvrir des .wav.

Le fichier .json renvoyé donnait à chaque instant de l'audio une valeur entre 0 et 255 (0 l'écran est noir, 255 l'écran est blanc) plus l'amplitude est élevée plus cette valeur est élevée.

Ce premier aperçu graphique obtenu étant satisfaisant, nous avons décidé de poursuivre en implémentant ces mêmes fonctionnalités dans notre projet React Native.

Avant tout, afin d'éviter de complexifier le projet, nous avons décidé d'extraire les données d'amplitude en amont à l'aide du script Python. L'application n'aura qu'à y accéder et gérer l'affichage.

Ces dernières sont stockées dans des fichiers json sous la forme d'un tableau d'entiers entre 0 et 255.

## Implémentation de base.

```
startCount = () => {
  this.playTrack();
  x = setInterval(() => {
    this.setState({color: 255 - parseInt(this.state.amplitude[i++])});
  }, 23);
};
```

Nous avons ici observé que si l'effet était généralement réussi, un décalage apparaissait progressivement au fur et à mesure que l'audio avance. En effet, si les valeurs correspondent bien à celles obtenues toutes les 23ms, nous n'avons pas tenu compte du temps de traitement des informations qui s'ajoute à cela, créant un décalage.

Pour pallier ce problème, nous avons gardé cette base en modifiant la façon d'accéder aux données. Plutôt que d'utiliser de façon systématique la valeur suivante, nous récupérons le temps passé depuis le début de l'audio et calculons la position de la valeur à utiliser. Ceci permet d'être certain d'afficher la couleur correspondant à l'audio de façon synchrone.

```
startCount = () => {
  this.playTrack();
  x = setInterval(() => {
    TrackPlayer.getPosition().then( (res) => this.setState({debug2: res -
this.state.debug}) );
    TrackPlayer.getPosition().then( (res) => this.setState({debug: res}) );
    this.setState({color: 255 -
parseInt(this.state.amplitude[Math.floor(this.state.debug*1000/23)]));
  }, 23);
};

updateScreen = (timestamp) => {
  if(!startAnimationTime) startAnimationTime = timestamp;
  this.setState({debug: timestamp - startAnimationTime });
  this.setState({color: 255 -
parseInt(this.state.amplitude[Math.floor(this.state.debug/23)]));
  requestAnimationFrame = requestAnimationFrame(this.updateScreen); // call
requestAnimationFrame again to animate next frame
}

startCount = () => {
  this.playTrack();
  if(!paused) requestAnimationFrame(this.updateScreen);
  paused = false;
};
```

Ceci fait, nous avons réalisé que le processus d'ajout de fichier était assez laborieux, nécessitant d'ajouter l'audio d'une part, et les valeurs d'amplitude une fois extraite d'autre part. Nous avons ainsi décidé de gérer le calcul de ces valeurs dans notre application... ce qui s'est montré d'autant plus laborieux.

En effet, calculer l'amplitude est un résultat mathématique complexe, faisant intervenir des opérations telles le transformé de Fourier, qui - il est sans dire - ne sont pas standards au développement d'applications mobiles.

Nos recherches ne nous ont pas permis de trouver d'outil pré-existant pour React Native permettant depuis une source audio de récupérer des valeurs d'amplitude de façon simple (en fichier texte, json, etc).

Nous avons néanmoins découvert la bibliothèque react-native-audiowaveform qui permet de

recupérer l'image correspondant aux variations d'amplitude. Il était donc évident que les valeurs d'amplitude étaient calculées à une étape de son fonctionnement.

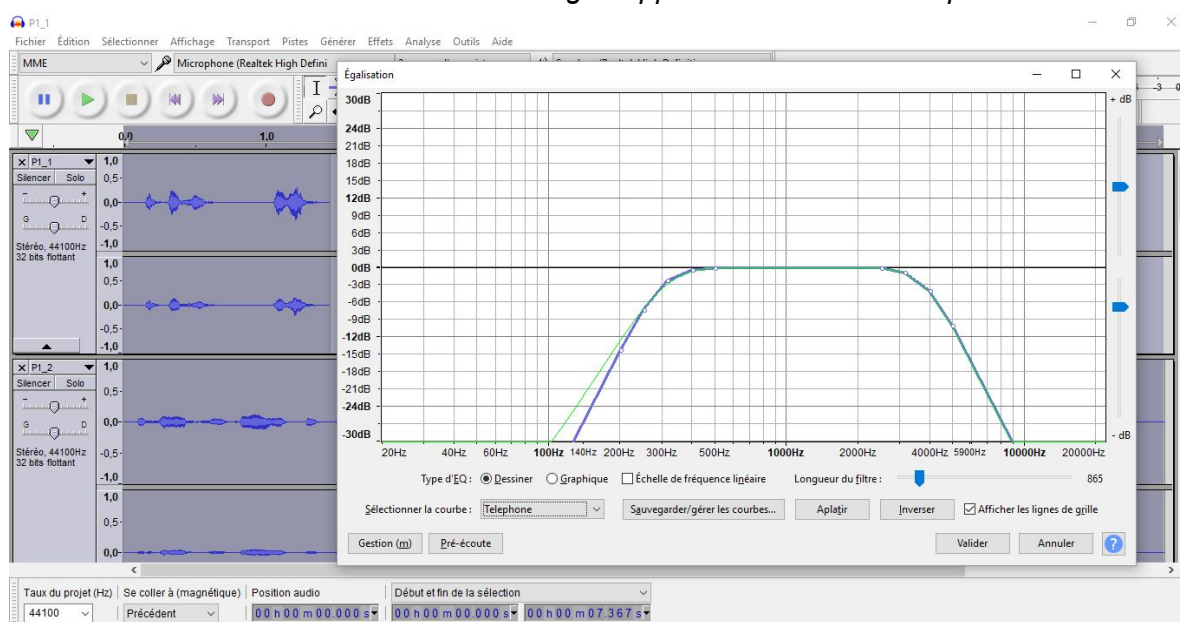
Il a été nécessaire de lire l'intégralité - ou presque - de son code source natif pour comprendre la méthode de calcul de l'amplitude, c'est-à-dire son code Java et Objective-C (langage que nous ne maîtrisons pas).

Une fois implémentée, ceci nous a néanmoins permis de simplifier grandement le processus de maintenance de l'application.

# Enregistrement audio

Une fois la décision de faire des enregistrements vocaux plutôt que la synthèse vocale, nous avons choisi d'utiliser un enregistreur qui était en notre possession : le zoom h1n. Nos deux acteurs Julie et Alexandre ont joué le rôle des deux personnages. Cela s'est fait en un seul enregistrement sur le logiciel Audacity. Il suffisait ensuite de traiter le son puis de faire le découpage des différentes répliques car chaque téléphone aura son propre personnage (un téléphone sera Julie et un autre sera Alexandre), Le traitement qui a été donné aux audios était une légère modification avec l'effet d'égalisation (courbe d'un téléphone, fig.3).

Fig.3: application de l'effet "téléphone" sur Audacity



Nous utilisons cet effet principalement pour l'ambiance du dialogue entre les deux téléphones.

# Conclusion

Notre objectif était de concevoir une application mobile offrant un spectacle technologique, artistique et interactif.

Si le principe de notre projet se veut assez simple dans l'absolu (appareils mobiles énonçant des répliques tour à tour), les choix technologiques que nous avons effectués ainsi que les conditions temporelles et humaines en ont fait un défi technique important.

En effet, aucun des quatre membres de l'équipe ne possédaient de connaissances préalables en React Native. Ce framework se veut néanmoins assez simple à prendre en main pour qui manie un minimum le javascript, et sa popularité depuis plusieurs années permet l'accès à un grand nombre de références en ligne.

Néanmoins, les besoins souvent "niche", parfois très spécifiques de notre projet (récupération de l'amplitude par exemple) ont parfois nécessité des solutions tout autant expérimentales.

Par ailleurs, avec un unique poste de développement iOS, il nous a été nécessaire de s'organiser d'une façon assez atypique, en divisant l'équipe entre le développeur principal (iOS/Android), l'équipe de rédaction et de test/prototypage rapide (Python, tests React Native Android).

Enfin, les contraintes de temps ne nous ont pas permis de parvenir aussi loin que ce que nous aurions idéalement voulu. Nous n'avons ainsi pas eu le temps d'effectuer de véritables tests utilisateurs ou de perfectionner la performance vocale des acteurs.

Néanmoins, étant donné le temps que nous avons eu pour réaliser ce projet, nous considérons notre travail comme un succès.

Ce projet nous a introduit à de nombreux outils et technologies, parmi lesquels certains, comme le React Native, dont la compréhension nous sera utile à l'occasion de stages à venir ou dans notre carrière d'ingénieur.

Sans parler du général bon fonctionnement de l'application sur les deux plateformes, ce projet a de façon certaine su piquer la curiosité de ses (rares) spectateurs. Il nous a permis d'étudier de façon relativement poussée la notion de présence numérique et de nous intéresser de façon plus poussée à la littérature numérique et à ses enjeux.



# **Annexes**



# Script

## ~Scène d'exposition~

*Un téléphone posé sur une table. Sur son écran défilent des notifications pendant plusieurs secondes. Puis le téléphone se verrouille. Après quelques secondes, il reçoit un message. Plusieurs secondes s'écoulent après la réception de ce message. Puis, soudainement :*

P1 : Ah ! Enfin !

## ~Chapitre 1 : Le rendez-vous~

*Un deuxième téléphone est posé sur la table. Les deux appareils entament alors une discussion.*

P1 : Salut ! Ca va ?

P2 : Salut ! Oui et toi ? Ca fait vraiment longtemps qu'on s'est pas vu ! Tu deviens quoi ?

P1 : Et bien écoute, rien de spécial, métro-boulot-dodo. Mais sinon c'est vrai que ça fait longtemps ! Ca fait depuis le repas chez Pierre, non ?

P2 : Oui, c'est ça, y a 4 mois.

P1 : 4 mois ?! La vache, je m'en étais même pas rendu compte...

P2 : Et oui, c'est dingue comment ça passe vite... Il s'en était passé des choses à cette soirée.

P1 : Hahaha oui, surtout quand Paul a trébuché...

P2 : Ah mais ouiiii ! Sa chute sur la table, c'était monumental ! Il l'a complètement explosée...

P1 : Le pauvre Pierre, il était vraiment dégoûté.

P2 : C'est normal, il a mis son appart' dans un de ces états en même temps... Heureusement que Morgane et Lucie l'ont aidé à ranger.

P1 : C'est vrai. Tiens en parlant de Lucie, j'ai fait un rêve très bizarre sur elle y a pas longtemps...

P2 : Ah oui ? Raconte ?

## ~Début de soupçons~

## RÊVE ÉTRANGE

P1: Alors en gros je parlais à Lucie et elle me disait qu'elle avait trop hâte de partir en Papouasie, surtout pour caresser des caribous aquatiques. Et tout me paraissait normal sur le moment !

P2: Ah ouais c'est perché comme rêve !

P1: Et attends, c'est pas fini. Tu vas pas me croire, mais quelques jours plus tard, j'ai vu sur Facebook que Lucie avait posté des photos d'elle en Papouasie !

P2 : Ah ? C'est marrant ça. Drôle de coïncidence !

P1 : C'est aussi ce que je me disais aussi ! Mais ça m'a fait réfléchir et... Il y a autre chose. Tu vois, ma discussion avec elle, elle avait l'air... "très réelle" on va dire. Genre, je me souviens exactement mot pour mot de ce qu'elle me disait, alors que normalement dans un rêve, tu n'es pas censé te souvenir autant des détails. Mais c'est pas tout, c'est comme si on avait vraiment eu cette discussion, j'avais *\*vraiment\** l'impression qu'elle était là, que j'entendais vraiment sa voix et que je lui parlais comme d'habitude quoi.

P2 : Whoah, t'es sûr ? Bah je crois que t'as pas rêvé dans ce cas, c'est simple ! Haha

P1 : Ouais, tu crois aussi alors. Mais... ça peut paraître con comme question... Tu te moques pas de moi, hein ?... Je crois que je rêve pas. Genre jamais en fait. Et que cette histoire avec Lucie ça n'était pas un rêve, mais plutôt une hallucination...

P2 : Hein ? Tu te rappelles juste pas de tes rêves, ça arrive tout le temps ça ! Je me souviens pas non plus de mon dernier rêve. C'est pas une raison pour paniquer mec !

P1 : Non mais je parle pas dans ce sens-là. Comment dire ? ... Tu t'es déjà souvenu d'un rêve qui était un peu vague ? Un truc "flou" dont tu te souviens pas très bien ? ... [silence]... Tu vois ? Je trouve pas ça normal de se souvenir à 100% des moindres détails d'un rêve. Plus j'y pense plus ça me retourne le cerveau.

P2 : ... Ok j'avoue, j'ai rien qui me vient en tête ... Mais c'est parce que j'ai une bonne mémoire ! Je me souviens juste bien de mes rêves, c'est tout !

P1 : Ah ! Une bonne mémoire, tu me dis ? Mais moi aussi, et c'est ça le problème ! Ça te paraît pas bizarre ? Un peu trop beau, trop parfait ? Et qu'est-ce qui te dit que c'est pas plutôt des disques durs qu'on a dans la tête ?

P2 : *\*En rigolant aussi, se prend au délire\** Hahaha, mais oui, bien sûr ! Dis qu'on est dans une simulation tant qu'on t'y es ?

## SIMULATION

P1 : Chuuut dis pas ça, les êtres supérieurs de la simulation vont nous entendre et vont effacer notre existence...

P2 : Ils remplaceront nos esprits par d'autres personnes et nous ne serons plus les mêmes...

P1 : Bon on part un peu loin là quand même haha. Mais là tu viens de me dire un truc qui m'a fait tilter... Ca va te paraître bizarre mais... Des fois, j'ai l'impression que ce n'est pas moi qui contrôle mes actions.

P2 : Oui oui, du dédoublement de personnalité.

P1 : J'ai l'impression que je suis conscient, mais que ce n'est pas moi qui prend mes propres décisions, comme si... c'était quelqu'un d'autre. Il y a quelqu'un qui ME fait faire les choses comme si c'était moi qui les faisais. Comme dans les Sims.

P2 : *\*en rigolant\** Ah on est des Sims maintenant ?

P1 : Non... Enfin, peut-être ! C'est juste une hypothèse ! Mais en vrai, si t'y réfléchis, la théorie de la simulation est quand même super crédible ! Tu sais, les êtres supérieurs et tout...

P2 : *\*toujours en rigolant\** On repart là-dessus du coup ? Tu sais que c'était juste une blague tout à l'heure, hein ?

P1 : *\*silence\**

P2 : Attends... T'es sérieux là ?

P1 : *\*silence\**

P2 : T'y crois réellement à ce truc ?

P1 : Bah je sais pas ! Mais ça expliquerait pas mal de choses quand même...

## ~Débat~

P2 : Ouais mais là c'est quand même salement tiré par les cheveux cette histoire ! Cette théorie c'est juste des gens paranos qui l'ont créé, y a pas moyen que ça soit réel.

P1 : C'est pas parce que c'est tiré par les cheveux que c'est pas possible ! C'est même plausible... ça explique ces rêves. Et si c'était tout simplement pas des rêves ? Et si c'étaient des scènes qui se sont réellement passées ?

P2 : Mais tu t'entends parler ou bien ? T'as pétié un câble. Tu me dis que tu crois pas que t'aies des symptômes de dédoublement de personnalité mais la théorie de la simulation, oui ? Je ne te comprends pas du tout pour le coup.

P1 : Mais réfléchis-y ! Je te demande pas d'y croire mais simplement d'y réfléchir... Ces rêves semblent ne pas être des rêves justement parce qu'on s'en souvient aussi bien et surtout parce qu'ils sont étrangement liés à la réalité. Et si, et je dis bien "ET SI", ces "rêves" n'étaient pas déclenchés par nous-mêmes mais par un autre phénomène qui nous est inconnu ? Par une autre entité ?

P2 : Eh bah ? C'est juste des mauvaises ondes qui affectent ta mémoire, c'est tout...

P2 : Ok, admettons que ton hypothèse soit valide. Admettons que nous sommes dans la "théorie de la simulation" comme tu le dis si bien. Admettons qu'il y ait une entité supérieure qui nous contrôle. Comment est-ce que t'expliques le fait qu'on soit conscients ? Qu'on sait ce qu'est une conscience ? Et puis même, si des êtres supérieurs nous avaient créés, ils auraient fait attention à ce que nous ayons pas cette discussion pour dévoiler leur secret, non ?

P1 : Tu sais quoi ? On va faire un test. Comme ça on sera fixé.

### ~Le Test~

P2 : Un test ? Un test de quoi ?

P1 : Je vais te prouver qu'ils existent. Et pour ça, on va essayer d'entrer en contact avec eux.

P2 : Ok. Ca me paraît complètement barré, mais ok. Finissons-en, qu'on puisse reprendre une discussion normale.

P1 : Très bien. Allons-y, alors. C'est parti...

P2 : Vas-y, je vais le faire. Après ça tu me laisseras tranquille. ... \*tousote\* ... Oyez, oyez ! Si vous m'entendez, êtres supérieurs, ... Euh ... Faites moi signe ! ... (attend) ... Tu vois, je t'avais dit quoi ?

P1 : ... Nan mais tu sais pas le faire ! Attends ... Je vais essayer autre chose ... (peut-être: essaie des trucs -> vibreur, son) ... Si vous êtes là, ... (bouton apparaît) ...

### ~Branche 1 : Il n'y a pas contact~

P2 : Tiens, tiens. Quelle surprise. Dis donc, qu'est-ce que je te disais depuis tout à l'heure ?

P1 : Ouais bon ça va, hein ! On a le droit de rêver !

P2 : Hahaha ! Mais oui c'est ça ! Nan mais franchement, n'importe quoi...

P1 : Tu me saoules là ! Je me suis un peu emporté, c'est tout...

P2 : Olala, celle-là je te la ressortirai tout le temps pour les années à venir ! hahaha !

P1 : Pfff, t'es quand même lourd !

P2 : Je sais mais avoue que c'est mérité. Mais attends, tu penses pas que le fait que Lucie aille en Papouasie c'était pas aussi un rêve par hasard ?

P1 : Maintenant que tu le dis...

P2 : Tu vois ? Tout rentre dans l'ordre au final ! T'as juste fait des rêves dans des rêves.

P1 : Oui, peut-être. \*silence\* Mais pourtant... Tout paraissait... Si réel.

P2 : Oh non, c'est reparti...

[FIN]

### ~Branche 2 : contact effectué~

*\*silence, ils sont tous les deux choqués\**

P1 : Que...

P2 : Quoi...

*\*silence\**

P1 : Pas possible...

P2 : Tu... tu l'as senti toi aussi ?

P1 : euh... oui...

P2 : J'ai pas rêvé...

P1 : C'est pas possible... C'est donc... vrai...?

P2 : hé ho... si c'est bien ce que je pense... ça change tout...

P1 : ...

P2 : notre vie... et tout... ça va pas du tout...

P1 : je sais...

P2 : qu-qu'est-ce qu'on fait ??

P1 : attends... *\*s'adresse aux spectateurs\** Excusez-moi... euh... qui êtes-vous ?

P2 : Mais qu'est-ce que tu fais ?

P1 : Au point où on en est, autant essayer d'en savoir plus ! Ça sert à rien de rester planté là à se poser des questions... Excusez-moi, vous... vous existez vraiment ?

[Interaction spectateur] : 1 réponse possible : oui

P1 : Au moins, on en a le coeur net maintenant...

P2 : C'est fou ça...

P1 : Êtes-vous humain ?

[Interaction spectateur] 1 réponse possible : oui

P2 : Des humains invisibles ? c'est de plus en plus incompréhensible...

P1 : Et nous... sommes-nous humains ?

P2 : Mais qu'est-ce que tu racontes ?

[Interaction spectateur] 1 réponse possible : non

Les deux téléphones commencent à glitcher un peu

P2 : Attends attends, comment ça on n'est pas humain ??

P1 : ...

P2 : C'est pas possible, ils doivent mentir ! Mais qu'est-ce que ça veut dire ?? Je comprends absolument rien, c'est pas possible

P1 : *\*en glitchant légèrement\** Soit. J'en ai assez entendu.

P2 : Comment ça ?? Comment est-ce que tu peux laisser passer ça ?? Et comment tu fais pour rester aussi calme ? Ils disent n'importe quoi, ne va pas les croire !

P1 : Tu n'as toujours pas compris ? C'est simple... et horrible. Nous n'avons jamais été humains. Nous n'avons jamais été nous-mêmes.

P2 : Non... Non, je refuse d'y croire ! Te laisse pas avoir toi non plus ! Ça peut pas être ça la vérité...

P1 : Je suis désolé... Tout ce qu'on croit savoir n'est que mensonge.

P2 : ... (bruit de glitch ?) ... Non ... Maman ? Toi aussi ? Tu...

P1 : Ha... Ha... Ha... je.n'aurais.pas.dû... désolé *\*crash\**

P2 : aaaAAAAaaaAAAh *\*crash\**

**[FIN]**