## Frameworks

* Rhodes (Guillaume)
* Titanium (Xavier)
* Cordova (Max)
* Natif

# Points de comparaisons

## Compétence de l’équipe dans le langage

### Natif

iOS: Objective-C, Compétence très légère d’un membre de l’équipe

Android: Java, Bon niveau de la part de l’équipe

### Titanium

Javascript: Excellent niveau de certains membres de l’équipe, et compétences suffisantes pour le reste

### Rhobes

Ruby: l’équipe à les connaissances de base du langage mais aucun membre ne l’a expérimenté dans le cadre d’un application mobile.

### Cordova

Cordova utilise des technologies Web pour le développement, à savoir HTML, CSS et Javascript. D’une manière générale, l’équipe à de bonnes compétences en la matière.

## Crossplateforme

### Natif

Aucun portage possible

### Titanium

Android/iOS/Tous les OS de Desktop

**Rhobes**

iPhone/Android/RIM/Windows/Mobile/Windows Phone 7

### Cordova

iOS/Android/Windows Phone/Blackberry/Web OS/Symbian.

## Intégration continue (outils d’analyse de code, tests unitaires, …)

### Natif

Jenkins pour iOS/Android

### Titanium

Jasmine pour les tests unitaires, node-cow (outil personnel) pour réaliser les tests automatisés.

**Rhobes**

Un [débuggeur](http://docs.rhomobile.com/rhodes/test-log-debug) existe, mais seulement sous Mac OSX, cela pose problème dans la mesure ou toute l’équipe ne possède pas de Mac.

### Cordova

Le debuggage sous Cordova est un peu compliqué à mettre en place sous Windows, mais fonctionne bien sous les autres plateformes. Il s’effectue grâce à WebKit, et utilise Chrome.

## Réactivité

### Natif

Réactivité totale

### Titanium

Accès aux objets natifs de l'environnement, complètement transparent pour l’utilisateur final.

**Rhobes**

Possibilité d’intégrer des librairies javascript (jQTouch, jQuery Mobile...), malgré cela le rendu d’une application Rhodes ne sera pas celui d’une application native.

Manque parfois de réactivité due à la structure complexe des applications Rhobes.

### Cordova

Une application Cordova est affichée dans un WebView, ce qui en fait une technologie potentiellement peu réactive, solution qui ne parait pas adéquate avec le pilotage d’un engin aérien.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Natif | Titanium | Rhodes | Cordova |
| Compétences | - | ++ | -- | ++ |
| Crossplateforme | -- | ++ | ++ | ++ |
| IC | + | ++ | - | -- |
| Reactivité | ++ | ++ | - | - |
| Moyenne => | 0 | ++ | - | + |

## Média de communication

### Bluetooth

Il possède une portée standard de 10 m, ce qui ne convient pas aux besoins du client.

Le Bluetooth reste très économe en termes de consommation.

L’équipe possède une faible expérience dans ce domaine.

### Wifi

Il possède une portée standard de 100 m environ.

Le wifi est plus consommateur que le Bluetooth mais il est nettement plus performent.

L’équipe a déjà réalisé plusieurs projets avec ce média de communication.

|  |  |  |
| --- | --- | --- |
|  | Bluetooth | Wifi |
| consommation | ++ | - |
| portée | -- | ++ |
| débit | ++ | ++ |
| complexité | + | + |
| expérience | - | + |

**Après comparaison le média de communication le plus approprié semble être le wifi.**

## Architecture

### Client (mobile)/serveur (embarqué)

Cela simplifie l’application sur le point de vue de l’architecture.

En cas de perte de connexion une seule synchronisation est nécessaire.

Par contre il est nécessaire d’implémenter une routine afin d’interroger le dirigeable sur sa position.

### [Client/serveur] / [Client/serveur]

Optimisation des messages : chaque message apporte une information utile.

Ce choix complique l’architecture, car il y a mise en place d’un client serveur sur chaque plateforme.

### Client (embarqué)/serveur (mobile)

Ce choix parait peu pertinent car le mobile devra commander le dirigeable il est donc préférable qu’il soit le client.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Client (mobile)/ serveur (embarqué) | [Client / serveur] / [Client / serveur] | Client (embarqué)/ serveur (mobile) |
| Complexité architecture | + | - | + |
| Complexité connexion | + | - | + |
| Complexité communication | - | + | - |
| Pertinence de la solution | ++ | + | - |

**Après comparaison l’architecture retenue est client (mobile) et serveur (embarqué).**