LAB DES USAGES NUMÉRIQUES

# La mobilité:

## Perspectives et enjeux de développement d'une application mobile

NOTE DE SYNTHÈSE – version finale

VOiRiN Conseil en management

TRANSFORMATION – CONSEIL SI - CONDUITE DU CHANGEMENT – USAGES COLLABORATIFS

Apporter de la valeur par les technologies digitales

Jérôme FREYERMUTH,
Colin LESPRIT –
Février 2015

# LE LAB DES USAGES

Le Lab des Usages est une cellule interne de veille et de recherche sur les usages numériques. Le Lab est ainsi aussi bien un **centre de recherche** développant des outils d'analyse qu'une structure de veille détectant les futures tendances. Le Lab des Usages est composé de deux doctorants permettant de faire le pont entre la **recherche académique** et **le monde de l'entreprise**. Le Lab des Usages, étant au cœur de VOIRIN, profite également de l'expertise de l'ensemble des consultants dans ses réflexions.

Les activités du Lab des usages se concrétisent par **divers publications** : billet de blog pour la veille, dossier d'analyse d'enquête et livre bancs pour les activités de recherche plus poussées. En tant que centre de recherche, le Lab des Usages produit également de la **R&D interne** avec la réalisation de divers outils d'analyses comme ATOM 2.0

En croisant les approches de l'ensemble des consultants et collaborateurs de VOIRIN, le regard frais de doctorants et l'expertise de chercheurs de renommée fréquemment invités à contribuer, le Lab génère et agrège l'innovation numérique pour mieux la diffuser.

Egalement membre de la **Chaire de Management de la Créativité** depuis sa création, cette chaire représente **trois entreprises** (SALM, SOCOMEC, VOIRIN) et **un laboratoire** - Le Bureau d'Economie Théorique et Appliquée (BETA). Ils collaborent ensemble sur 5 ans sur plusieurs thématiques liées à la créativité au travers de travaux de **recherche théorique** et de **recherche action.** Ces thématiques traitées couvrent la créativité sous toutes ses formes : la créativité dans la recherche de nouveaux produits, la créativité au service de l'innovation managériale, la créativité et l'entreprise 2.0 la créativité au service du marketing,…

Cabinet de conseil existant depuis plus de 33 ans, VOIRIN offre des conseils en systèmes d'information et en transformation d'organisation dans le secteur privé, public et de la santé. Ses offres se départagent dans **quatre domaines** de compétences : l'organisation numérique, la transformation des organisations, l'organisation collaborative, et le management par projet.

Depuis sa création en 1980, les équipes de consultants se sont intégrés et ont participé à l'animation de nombreuses communautés. Actuellement, des travaux de recherche et des travaux prospectifs sur les usages et outils sont ainsi menés en France avec le **BETA** (Bureau d'Economie Théorique & Appliquée à Strasbourg), le Centre de Recherche en Gestion de l'Ecole Polytechnique de Paris, le club de gestion de la connaissance, la communauté club net, et le CLUSIF dans le domaine de la sécurité des systèmes d'information. A l'étranger, des liens étroits ont été développés avec des équipes d'HEC Montréal. Cette présence est à la fois une **source de veille informationnelle**, un accès à de **nouveaux apprentissages** et le lieu de **nouvelles publications.**

# SOMMAIRE

**SOMMAIRE** ..... 4

**INTRODUCTION: LES USAGES MOBILES** ..... 5

**PARTIE 1 : PERSPECTIVES ET ENJEUX DU DEVELOPPEMENT D'UNE APPLICATION MOBILE** ..... 6

Le fonctionnement d'une application ..... 6
Le développement natif ..... 6
Le développement web ..... 7
Le développement hybride ..... 8
Les solutions C++ et MoSync ..... 9

**PARTIE 2 : LE CAS WATERAIR** ..... 14

L'application WatBook ..... 14
Les spécificités de Watbook et leurs contraintes ..... 15
Les principales réponses de l'étude de cas. ..... 16

**CONCLUSION** ..... 18

# INTRODUCTION: LES USAGES MOBILES

L'informatique est arrivée dans une ère de consommation mobile, notamment sur le plan des applications et services web. La capacité à se connecter en tout temps et tout lieu couplée à l'évolution des appareils mobiles (en puissance, fonctionnalités, etc.) est le principal moteur de ces nouveaux usages. Cependant, si les usages grand public ont connu une croissance exponentielle, le sujet est encore jeune dans les organisations. Nous pouvons y distinguer deux grands types d'usages :

- les usages relatifs à un métier spécifique (commerciaux, techniciens, etc.) ;
- les usages transverses, correspondant aux besoins quotidiens de communication et coordination (email, agenda, lecture de documents, etc.).

Les premiers relèvent d'usages pouvant être réalisés par des applications métiers spécifiques. A ce niveau, l'ensemble des populations mobiles d'une organisation peut potentiellement être impacté. Il s'agit dès lors d'analyser finement les besoins mais surtout d'assurer un accompagnement au changement de proximité, notamment si les fondements d'un métier sont affectés.

Dans le second cas, l'applicatif est plus standardisé  et l'adoption généralement plus facile car tirée par les usages grand public. La mobilité est souvent intégrée dans une migration plus vaste vers une plateforme collaborative.

Dans les deux cas, la gestion d'un parc mobile est bien différente d'un parc traditionnel et un certain nombre de bonnes questions doit être posé en amont du déploiement. Nous nous attacherons dans cette note au volet applicatif et ses différents types de développements possibles.

Au long de la première moitié de la note, nous détaillerons les différents éléments à prendre en compte lors d'un projet de développement d'application. Nous commencerons par définir le fonctionnement général d'une application et distinguerons trois grandes catégories de développement. Par la suite, nous exposerons les enjeux liés aux différents types d'utilisation. Nous conclurons cette partie par un comparatif des différents scénarios que pourrait envisager une organisation.

La seconde moitié de la note consiste à présenter une étude de cas réalisée par VOIRIN Consultants pour Waterair, entreprise leader dans la fabrication de piscines enterrées en kit. La problématique de cette étude était de déterminer le type de développement le plus approprié pour une application tablette afin de minimiser les coûts de maintenance face aux aléas des feuilles de routes des différents systèmes d'exploitation mobiles.

Les résultats de cette étude de cas nous permettront enfin de conclure en caractérisant les grandes lignes directrices à suivre dans le cas d'un projet mobilité.

# PARTIE 1 : PERSPECTIVES ET ENJEUX DU DEVELOPPEMENT D'UNE APPLICATION MOBILE

## Le fonctionnement d'une application

La première partie de la note a pour objectif d'expliquer brièvement le fonctionnement d'une application mobile. De manière très schématique, une application se compose d'une suite d'instructions et d'un jeu de données nécessaire aux opérations à effectuer. La suite d'instructions constitue ce qu'on appelle le code source et est écrite dans un langage de programmation spécifique. Il existe différents langages de programmation, dont le choix dépendra de l'environnement applicatif mais également des objectifs visés.

Généralement, trois façons de développer une application mobile sont distinguées : les développements de types natif, web ou hybride.

## Le développement natif

Les applications natives sont développées avec le langage natif des systèmes d'exploitation mobile. Les principaux acteurs sont iOS (Apple), Android (Google) et Windows Phone (Microsoft).

Ce type de développement est généralement celui qui obtiendra les meilleures performances, avec la meilleure ergonomie. Il a aussi l'avantage de permettre la pleine utilisation des fonctionnalités du terminal, tels que l'appareil photo, le système de notification ou la géolocalisation. Ce dernier point est essentiel et constitue un facteur de choix prépondérant.

Cependant, les applications natives posent d'importants problèmes d'interopérabilité. Le code source peut être utilisé exclusivement par le système d'exploitation concerné. Cette contrainte engendre des coûts de réécriture pour rendre l'application compatible avec les autres appareils et lors des mises à jour des différents OS.

**Swift, le nouveau langage de programmation d'Apple**

A l'occasion de la WWDC 2014 (WorldWide Developpers Conference), Apple a présenté Swift, son nouveau langage de programmation.

Initié par Chris Lattner en 2010, Swift a été conçu pour développer des applications de façon plus rapide, plus simple et plus sécurisée, sur les systèmes d'exploitation iOS et OS X.

Objective-C, le langage actuel, reste compliqué à manipuler pour une partie des développeurs, notamment dans la gestion de la mémoire. En effet, le système de pointeurs mémoire peut engendrer le crash des applications et l'ouverture de failles de sécurité.

Swift répond à ces difficultés par le biais de nouvelles particularités. Il permet notamment de voir les résultats des lignes de code en direct, grâce à son mode de compilation intelligente, *Interactive Playground*. Ajouté à cela, son code est considérablement réduit, ce qui devrait diminuer la quantité de bugs.

Une autre particularité importante est sa possibilité de coexister avec Objective-C. Ces caractéristiques devraient donc attirer une nouvelle génération de développeurs, moins spécialistes, tout en conservant les fervents utilisateurs d'Objective-C.

Nous touchons ici à l'une des principales problématiques des applications mobiles : le développement multiplateforme. Pour tous les développeurs, le saint graal est de pouvoir créer une unique application pour l'ensemble des plateformes, ce qui éviterait les coûts liés au maintien d'une application différente pour chaque plateforme. De la même manière, à chaque mise à jour du système d'exploitation, l'application peut être plus ou moins affectée. Lors d'une mise à jour majeure, la mesure du risque de devoir redévelopper une part importante de l'application se pose.

Pour pallier à ce problème, la solution est soit de s'affranchir du système d'exploitation, soit de programmer dans un langage compatible avec toutes les plateformes. Les deux types de développement suivants ont justement pour objectif de répondre à ces problématiques.

## Le développement web

Le développement web consiste à programmer avec les standards du web (HTML, CSS, Javascript) afin de pouvoir utiliser l'application via un navigateur web et s'affranchir du système d'exploitation. Les coûts de migration ou de mise à jour se trouvent considérablement diminués n'étant plus que liés aux évolutions du navigateur web.

**Le Responsive Web Design (RWD)**

Le RWD est une technique qui permet d'optimiser son site automatiquement à l'appareil qui s'y connecte. En effet, en détectant les dimensions de l'écran de l'appareil, la structure, l'ergonomie et même le contenu du site vont s'adapter afin de rendre la transmission de l'information plus agréable et pertinente.

Cette technique peut permettre d'améliorer les applications web et de leur donner l'avantage de s'adapter au terminal et à l'utilisateur.

Ainsi, certains sites ont un design et une arborescence différents sur tablettes et smartphones afin de correspondre aux attentes des internautes et à leurs caractéristiques.

Les évolutions récentes d'HTML5 apportent de plus en plus de crédibilité à ce type de développement en fournissant des API permettant de se connecter aux fonctionnalités du terminal. L'efficacité est encore loin d'atteindre celle d'un développement natif mais peut suffire dans le cas d'applications simples.

L'autre grande évolution concerne l'ergonomie, notamment le responsive web design (voir l'encadré). Il est dorénavant possible de rendre une application web à la fois très attractive en termes de rendu et adaptable automatiquement suivant le type de support.

L'une des grandes faiblesses du développement web reste néanmoins l'accès aux fonctionnalités du terminal. L'utilisation de la caméra ou de l'appareil photo par exemple est encore relativement peu performante. En parallèle, certaines fonctionnalités complexes comme le dessin en temps réel sont extrêmement difficiles à coder en HTML5.

Le second inconvénient réside dans l'utilisation hors ligne. Bien que HTML5 commence à intégrer ce fonctionnement, il reste très limité. En effet, le stockage des données se fait dans le cache du navigateur web, ce qui restreint fortement l'espace disponible et réduit les performances globales de l'application. Ainsi, si l'application utilise une base de données volumineuse, le développement web sera probablement impossible.

## Le développement hybride

Les applications hybrides sont développées avec une solution mixte entre le natif et le web. Elles sont la combinaison des deux premiers types, l'objectif étant de combiner le meilleur des deux mondes.

Il existe plusieurs outils de développement hybride, dont les plus connus sont PhoneGap et Titanium.

### *PHONEGAP.*

Le principe de PhoneGap est de fournir un environnement de programmation web (HTML5, CSS) avec des API (*Application Programming Interface*) en JavaScript afin de pouvoir accéder aux fonctionnalités natives du terminal. Le code de l'application est encapsulé dans une application native (spécifique à chaque OS) qui utilise le rendu web afin d'afficher l'interface.

Néanmoins, les contraintes de PhoneGap sont sensiblement les mêmes que pour une application web, à savoir les limitations dues à la mémoire cache du navigateur.

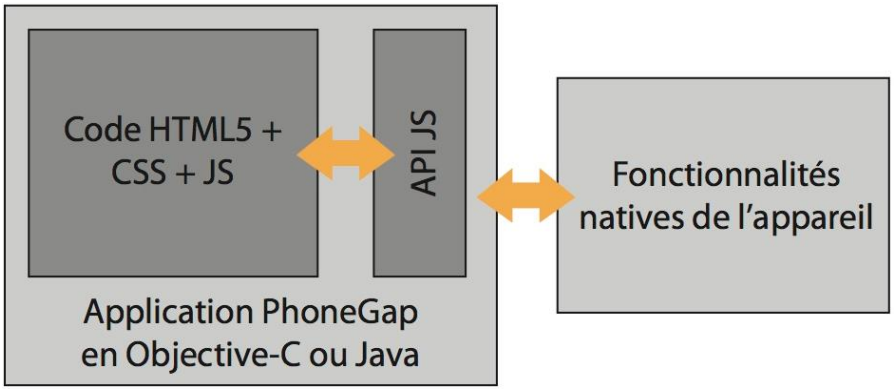


**Figure 1 : fonctionnement de PhoneGap**

### *TITANIUM.*

Le principe de Titanium est de fournir une machine virtuelle JavaScript permettant d'accéder aux fonctionnalités natives de l'appareil mobile. L'application est développée en JavaScript avec les API spécifiques de Titanium.

Titanium ne transforme pas le code JavaScript (JS) en code Objective-C ou en Java comme cela est parfois déclaré. Il transforme le code en bytecode et utilise un moteur d'exécution JS pour interpréter le code. Au contraire de PhoneGap, il n'y a pas de vue web mais un accès aux interfaces natives et tout est écrit en JS.

Le résultat étant que l'application Titanium est très similaire aux applications natives. Cependant, si en théorie JS est multiplateforme, en pratique, le même code marche rarement pour l'ensemble des OS à cause des spécificités de ceux-ci dans la manière d'interpréter et d'exécuter JS.

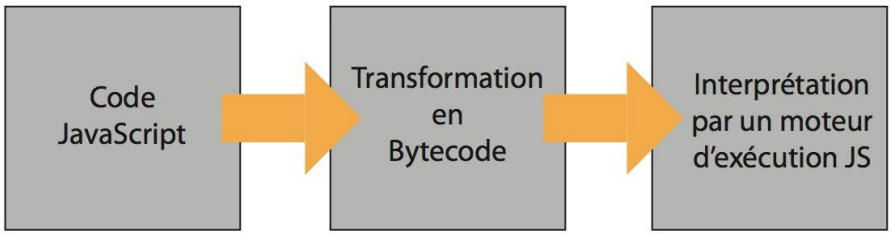


Figure 2 : fonctionnement de Titanium

Le tableau ci-dessous synthétise les avantages et inconvénients des deux principales solutions hybrides :

| | Avantages | Inconvénients |
|---|---|---|
| PhoneGap | ● Développement HTML5 peu coûteux | ● Rendu Web<br>● Besoin de revoir les API pour chaque OS |
| Titanium | ● Rendu proche du natif<br>● JavaScript, répandu et connu par beaucoup de développeur | ● Contraintes JavaScript<br>● Besoin de revoir les API pour chaque OS |

Figure 3 : comparatif PhoneGap/Titanium

Au final, les solutions hybrides sont prometteuses sur le papier mais se révèlent souvent problématiques en pratique. Afin de maximiser l'efficacité de ce type de solution, il est primordial de penser le multiplateforme en amont et de créer le programme en conséquence. Rendre multiplateforme une application pensée pour un seul OS est souvent voué à l'échec.

## Les solutions C++ et MoSync

Une dernière grande catégorie de solution est le développement en C++. Ce langage permet de faire des développements natifs tout en étant supporté par iOS, Android et Windows Phone. De surcroît, il permet de développer des applications plus performantes notamment au niveau de l'utilisation de la batterie et du CPU. Cette solution semble donc idéale. Cependant, sa bibliothèque standard (STL) manque cruellement de fonctionnalités pour les mobiles. Il est donc nécessaire de les créer soi-même, ce qui peut représenter un investissement non négligeable. De plus, C++ étant un langage de bas niveau, il est nécessaire de tenir compte des différentes architectures CPU.

Il existe cependant un framework permettant de combler certaines de ces lacunes : MoSync. MoSync est un kit de développement multiplateforme avec le grand avantage de proposer énormément de fonctions C/C++ permettant d'exploiter au maximum les fonctionnalités du terminal. De plus, il embarque une couche HTML5/CSS/JS de tel sorte qu'il n'est plus nécessaire de rentrer dans la couche C/C++.

## Les critères de choix

La problématique centrale de choix entre les différents développements est fortement liée au contexte d'utilisation de l'application en question. Chacun de ces contextes demande des caractéristiques différentes au développeur, en termes de stockage, de fonctionnalités, d'ergonomie ou d'interopérabilité. Nous pouvons cependant noter quelques bonnes questions à se poser en amont d'un projet mobilité.

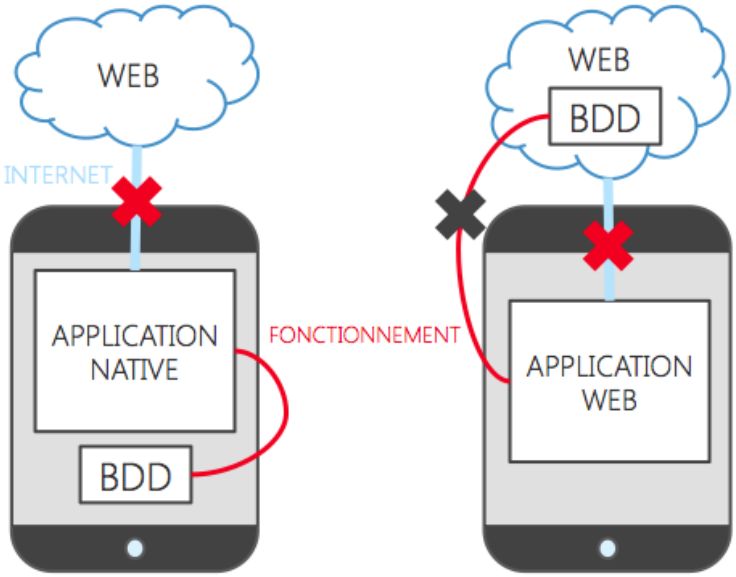


**Figure 4 : fonctionnement en ligne/hors ligne**

L'utilisation hors-ligne tout d'abord qui concerne toutes les utilisations que l'on peut faire en restant déconnecté. Pour leur réalisation, l'application a besoin de données étant enregistrées en local et donc accessibles depuis l'appareil natif. Ceci n'empêche cependant pas ce dernier de devoir faire régulièrement des mises à jour des bases de données des applications à partir d'internet, afin de les garder opérationnelles.

C'est ici que réside un point essentiel quant au choix du type de développement. En effet, les applications développées en natif n'ont aucun problème à fonctionner hors-ligne puisque leurs bases de données (BDD) sont enregistrées dans la mémoire de l'appareil mobile. A l'inverse, comme nous l'indiquions précédemment, les applications web ont des limitations à travailler en cet état puisque leurs BDD se trouvent sur des serveurs en ligne. Elles peuvent tout de même fonctionner ainsi, en se servant de la

mémoire cache du navigateur, qui toutefois reste limitée, ce qui pose des problèmes d'ergonomie et de performance.

Comme nous l'avons vu précédemment, le développement web peut tout à fait convenir pour une application simple, avec peu d'interactions avec le terminal et une base de données relativement légère. En revanche, une application complexe demandera souvent un développement natif, voire en C/C++ avec MoSync. La complexité de l'application est donc un autre critère à prendre en compte.

Enfin, un troisième grand critère est celui de l'accès aux fonctionnalités natives du terminal. Bien que HTML5 progresse considérablement dans ce domaine, il est encore loin d'égaler les développements natifs. En terme de performance, ces derniers seront toujours un cran au-dessus.

De manière générale, plus l'application est indépendante de l'OS, plus les contraintes d'usages seront importantes (hors ligne, accès aux fonctionnalités, complexité des bases de données, etc.) mais moins les coûts de maintenance et mise à jour seront élevés. Tout l'enjeu est ainsi de trouver le bon arbitrage entre fonctionnalités et type de développement.

Le schéma exposé ci-dessous récapitule les différentes situations possibles :

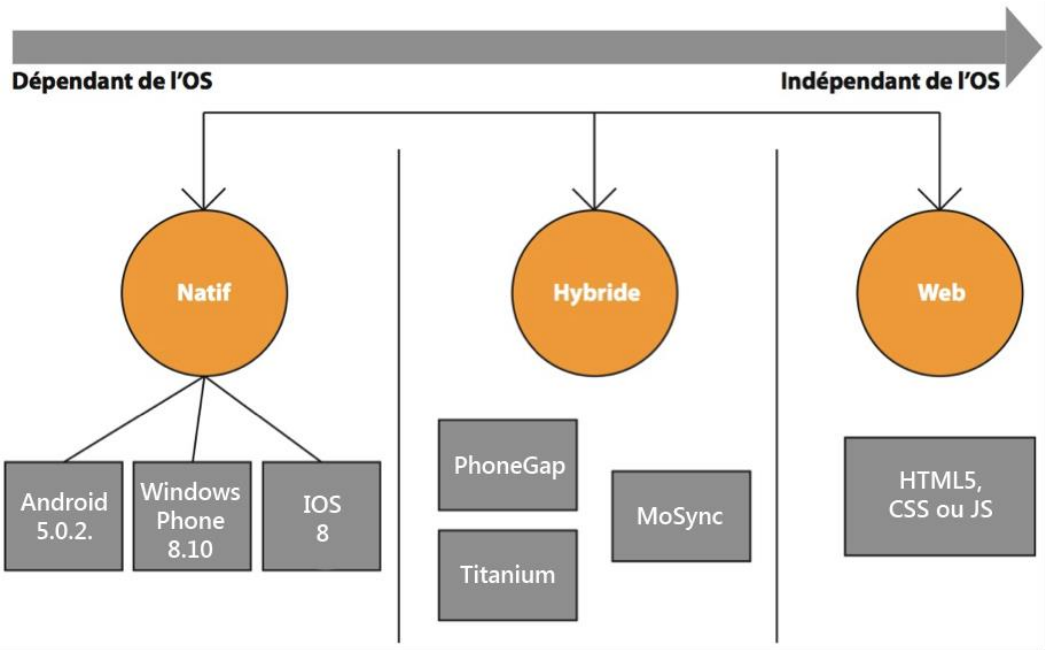


**Figure 5 : les différents types de développement d'application**

Pour conclure cette sous-partie, nous dressons un bilan « avantages/inconvénients » de chaque technologie au sein des tableaux présentés ci-dessous.

## APPLICATION NATIVE

| AVANTAGES | INCONVÉNIENTS |
|---|---|
| + Meilleure performance ;<br>+ Visibilité de l'application sur les stores ;<br>+ Accès aux données rapide ;<br>+ Tout le contenu est disponible sur l'appareil. | - Le déploiement sur chaque plateforme demande un développement particulier ;<br>- L'application peut être très lourde. |

## APPLICATION WEB

| AVANTAGES | INCONVÉNIENTS |
|---|---|
| + L'application fonctionne sur tous les OS ;<br>+ Les mises à jour du contenu sont simplifiées ;<br>+ L'application est légère. | - L'usage hors ligne encore limité ;<br>- La performance dépend fortement du débit ;<br>- Les fonctionnalités natives de l'appareil ne son pas facilement accessibles. |

## APPLICATION HYBRIDE

| AVANTAGES | INCONVÉNIENTS |
|---|---|
| + Bon compromis entre la disponibilité et la flexibilité du contenu ;<br>+ Susceptible de synthétiser les avantages des deux premiers types. | − Les solutions restent à perfectionner ;<br>− Subsiste des problèmes de compatibilité. |

**Figure 6 : bilan avantages/inconvénients des solutions**

# PARTIE 2 : LE CAS WATERAIR

Au cours de cette partie, nous présenterons une étude de cas réalisée pour Waterair afin d'illustrer les éléments mis en avant dans la première partie.

Waterair est une entreprise leader dans la fabrication de piscines enterrées en kit. De son expérience de 42 ans et de plus de 90 mille piscines installées, elle est reconnue pour son savoir-faire dans la fabrication, la sécurité et la fiabilité des piscines.

Waterair a développé une application mobile pour ses commerciaux afin de disposer d'un outil d'aide à la vente permettant de simuler l'intégration d'une nouvelle piscine ainsi que le devis associé.

## L'application WatBook

WatBook est une application mobile d'aide à la vente développée sur iOS. L'objectif du projet était de munir les commerciaux de tablettes et d'une application permettant de présenter aux clients les différentes offres du catalogue et réaliser des simulations d'implantation personnalisée. Suite à cette simulation, l'idée était de pouvoir également réaliser un devis et l'imprimer par le biais d'une imprimante portable compatible avec la tablette.

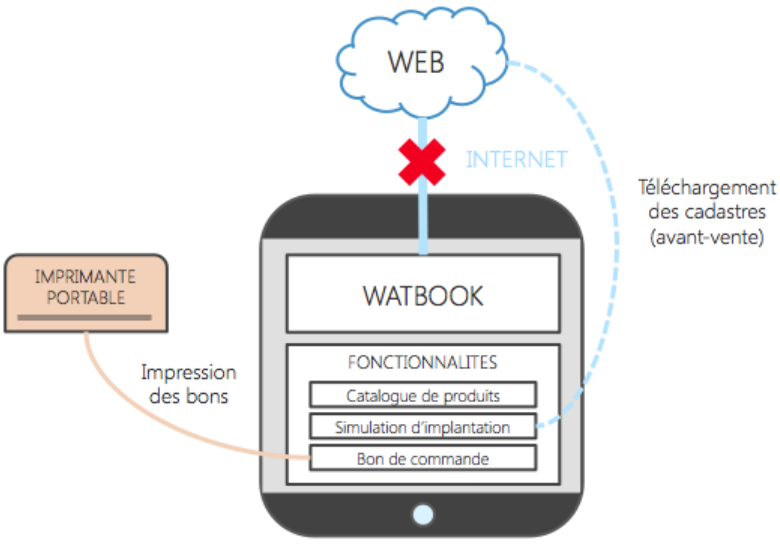


**Figure 7 : contexte de l'application Watbook**

## Les spécificités de Watbook et leurs contraintes

Initialement développé sur iOS 5 en natif, le passage à iOS 6 c'était révélé relativement couteux. Avec l'arrivé d'iOS 7 (au moment de l'étude), la question des coûts de mise à jour s'est à nouveau posée. L'objectif a donc était d'étudier si un affranchissement de l'OS par un développement web ou hybride était possible.

L'application Watbook devait répondre à plusieurs objectifs :

- Permettre le travail hors ligne et l'enregistrement en local des données nécessaires à l'application,
- Permettre une synchronisation en ligne avec les bases de données relatives aux piscines de Waterair,
- Disposer de certaines fonctionnalités comme la mémorisation de plusieurs projets, le dessin en temps réel, l'enregistrement de devis, etc.
- Accéder aux fonctionnalités de la tablette comme l'appareil photo ou l'imprimante via USB.

La principale contrainte rencontrée par Waterair réside dans le caractère natif du développement initial de Watbook. En effet, l'application a été développée en Objective-C afin de pouvoir remplir de manière performante les objectifs d'usages. Comme nous l'avons vu dans la première partie, ceci pose de grands problèmes en terme d'interopérabilité. Dès lors qu'iOS subit une mise à jour majeure, les coûts de migration peuvent très vite monter (estimés à 80 000 euros sur iOS 7).

La seconde contrainte est que Waterair couvre des territoires où l'accès à internet n'est pas forcément convenable. Par conséquent, les commerciaux devaient pouvoir travailler hors ligne. Pour cela, ils doivent faire obligatoirement une mise à jour avant de partir en clientèle afin de stocker les plans cadastraux utiles.

La troisième contrainte est le besoin de pouvoir connecter une imprimante portable à la tablette, permettant d'imprimer les devis et bons de commande aux clients. Ce point est indispensable au fonctionnement de ce projet car sans l'imprimante le vendeur ne peut plus finaliser la vente.

Enfin, une dernière contrainte est le besoin de fonctionnalités avancées comme le dessin en temps réel ou le positionnement de certains éléments d'une piscine sur un cadastre. En parallèle, l'application devait également pouvoir accéder aux fonctionnalités de l'iPad, dont notamment l'appareil photo et l'USB (pour l'imprimante).

Sachant ces contraintes, était-il possible d'envisager un développement s'affranchissant de l'OS ?

## Les principales réponses de l'étude de cas.

Afin de répondre, il est nécessaire d'analyser le contournement des grandes contraintes avec un développement web. Nous pouvons distinguer dans ce cas deux grandes contraintes :

### L'usage hors ligne :

- Comme nous l'avons vu dans la première partie, l'usage hors ligne peut être envisagé si les bases de données n'ont pas un volume trop important et n'ont pas des interactions trop complexes avec les instructions. Or Watbook dispose à la fois d'une base de données importante (les images des piscines notamment) et des interactions complexes (mises à jour régulières des bases de données, réductions automatisées, enregistrement de projets, etc.). Le développement web se trouve donc fortement limité dans ce cas.

### Les fonctionnalités avancées :

- Lorsqu'ils simulent l'implantation d'une piscine aux clients, les commerciaux doivent accéder à des fonctionnalités avancées comme le dessin en temps réel. Si théoriquement HTML5 permettrait de réaliser cette fonctionnalité, en pratique cela peut vite devenir un casse-tête de programmation et donc augmenter les coûts. De même pour la fonctionnalité de positionnement d'un escalier sur un cadastre qui est bien plus facilement gérable avec un développement natif.

Face à ces contraintes, le développement web pourrait théoriquement contourner les problèmes mais cela demanderait une étude approfondie de faisabilité et donc des coûts supplémentaires non négligeables.

Une deuxième option est le développement hybride. Nous évoquons ici les principales solutions hybrides qui sont PhoneGap et Titanium. Idéalement, ces types de développement sont censés rassembler le meilleur du natif et du web. Malheureusement, PhoneGap se rapproche des applications web et Titanium des applications natives, ils ont respectivement les mêmes défauts (bien qu'amoindris) que les deux premiers types d'application. Ajouté à cela, le développement d'une application avec Titanium est à prévoir et planifier en amont. Cette option n'est donc au final pas conseillée.

Le dernier scénario de développement est la solution C++ améliorée par MoSync. Comme nous le disions dans la première partie, ce scénario a l'avantage d'avoir un langage compris par tous les OS et permettant d'améliorer la performance des applications. Cependant, le langage C++ est complexe, ce qui rend son développement très coûteux malgré la présence de MoSync. De plus, des mises à jour seront possibles avec l'évolution du matériel, notamment concernant le CPU.

Au final, iOS reste la meilleure solution. Le résultat aurait pu être différent si les solutions hybrides ou HTML5 avaient été plus matures. Il est intéressant de noter que le développement multiplateforme est ici

une solution analysée de manière détournée. En effet, il ne s'agissait pas de savoir si l'application pouvait être rendue compatible pour différents OS mais bien de s'affranchir de l'OS afin d'éviter les coûteuses mises à jour sur un même OS d'un développement natif.

Nous pouvons enfin remarquer que redévelopper Watbook pour un autre OS aurait également pu être une solution à envisager. Cependant, le redéveloppement depuis zéro aurait également eu un coût considérable car aucun langage de programmation ne permet d'économiser suffisamment de jours de développement pour devenir attractif.

# CONCLUSION

En conclusion, un développeur doit évaluer précisément le contexte d'utilisation de son application avant de faire le choix du type de langage de programmation et de plateforme cible.

Le développement natif permet à l'utilisateur de profiter pleinement des fonctionnalités de son smartphone ou de sa tablette mais aussi de travailler hors-ligne. Néanmoins, il pose des problèmes en termes d'interopérabilité et de maintenance.

Le développement web a l'avantage d'avoir un langage universel, ce qui permet d'utiliser l'application avec n'importe quel type de mobile. Cependant, il connaît des limitations quant à l'utilisation hors-ligne de l'application et d'accès aux fonctionnalités de l'appareil.

En théorie, les applications hybrides (PhoneGap, Titanium, C++/MoSync) sont censées combler les désavantages du natif et du web. Néanmoins, la pratique montre que ce type de développement reste encore à perfectionner.

L'étude de cas pour Waterair nous montre bien la complexité de ce type de projets. Il est primordial de planifier les différents objectifs de l'application en amont et de définir précisément l'environnement et le contexte d'utilisation car il est difficile de rectifier la trajectoire de développement après le lancement du projet.

Dans tous les cas, ce sont bien les usages qui guident les choix techniques et non l'inverse. L'objectif de cette note était ainsi de mettre en avant le besoin de bien connaître les usages afin de pouvoir sélectionner la solution technique la plus adaptée et permettant de minimiser les coûts.

## LES AUTEURS

Doctorant depuis 2012 au sein du Cabinet, Jérôme a précédemment effectué un master 2 d'économie et management de l'innovation à l'université de Strasbourg. Aujourd'hui responsable du Lab des Usages, il contribue régulièrement au blog interne, à la rédaction de livres blancs, d'enquêtes liées aux thèmes de l'innovation des technologies web 2.0, aux usages de médias sociaux et au poste de travail du futur. Jérôme est également consultant junior pour le cabinet, et effectue notamment des missions de refontes Intranet pour des comptes publics et privés.

Titulaire d'un master 2 d'économie et management de l'innovation à l'université de Strasbourg, Colin a commencé à travailler au sein du Cabinet en 2014. Il consacre une partie de son temps à l'animation du Lab des Usages, en publiant des articles de blog qui traitent les thèmes des nouveaux usages numériques et de l'innovation en général. Il intervient aussi dans la réalisation de notes de synthèse et d'autres documents de recherche.

## PUBLICATIONS RECENTES


**SharePoint 2013, la maturité de l'Intranet social**

Avec cette note de synthèse, nous souhaitons partager les premiers retours d'expérience de mise en œuvre dans le cadre de projets d'intranet sociaux avec SharePoint 2013. Les atouts de SharePoint 2013, outre un périmètre iso fonctionnel avec les autres solutions, sont à chercher du côté de son architecture – principalement le positionnement et les capacités du moteur de recherche - mais également en tant que brique d'une infrastructure de gestion des contenus et d'espaces collaboratifs et opportunité de valorisation de l'expertise déjà acquise en interne.


**Développer les usages des applications informatiques de gestion, une réelle opportunité pour les entreprises**

Cette note a pour objectif de valoriser l'activation des réservoirs d'usages qui existent dans les entreprises, alors que bien souvent ces dernières ignorent l'existence de telles ressources. Les usages existent dans les organisations dès lors que les individus membres de cette organisation doivent accomplir des tâches, et sont au cœur de la dynamique de transformation numérique et digitale des entreprises.


**Le moteur de recherche**

Ce livre blanc a pour objectif d'éclairer le lecteur sur le fonctionnement et les enjeux de la recherche d'information en entreprise en proposant une vue d'ensemble sur les technologies et principes régissant ce domaine ainsi que sur le marché actuel de la recherche.


**Institut Montaigne : Pour un New Deal numérique**

Dans une période de crise économique affectant sévèrement l'Europe, le numérique représente une source d'innovation et de croissance pour la France. La dernière étude de l'Institut Montaigne démontre que le numérique, devenu un outil incontournable au développement, n'est pourtant pas encore pleinement exploité dans les organisations publiques et privées françaises. Cette opportunité à saisir par l'État et les entrepreneurs a encouragé Gilles BABINET, multi-entrepreneur et ancien Président du Conseil national du numérique, et Fréderic CREPLET, Directeur Général de VOIRIN, à publier une nouvelle étude intitulée 'Pour un 'New Deal' Numérique'.

Retrouvez toutes nos publications sur notre site web : **www.voirin-consultants.com**

**Suivez-nous !**
@cabvoirin

**Le Lab sur Twitter :**
@LabdesUsages

**Abonnez-vous** à notre **newsletter** mensuelle

**Retrouvez-nous** sur www.voirin-consultants.com

...soyez informés des **évènements à venir** sur notre site !


**BUREAU DE STRASBOURG**
(siège social)
42 Route de Bischwiller
67300 Schiltigheim
Tél. 03 88 62 23 00
Fax : 03 88 33 38 23
info@voirin-consultants.com

**BUREAU DE PARIS**

171 quai de Valmy
75010 PARIS
Tél. 01 40 38 61 10
info@voirin-consultants.com