

- » AMELIORER LA PERFORMANCE
- » AUGMENTER LA SOUPLESSE
- » ASSURER LA TRANSPARENCE
- » REDUIRE LES COUTS
- » AMELIORER LA RELATION CLIENT
- » ACCELERER LA MISE SUR LE MARCHE
- » INNOVER
- » AMELIORER L'EFFICACITE
- » ACCROITRE L'ADAPTABILITE
- » GARANTIR LA CONFORMITE



CONSULTING > SOLUTIONS > OUTSOURCING

Architecture Android

Les spécificités de l'OS

Philippe Prados

Solution Linux – Mai 2011

ADVANCE YOUR BUSINESS »

- » Philippe Prados
- » Architecte Senior
- » Responsable d'un pool R&D sur Android



Les fondamentaux d'Android

- » Quels sont les principes ayant guidé à la conception d'Android ?
- » Pourquoi le framework est-il conçu comme cela ?
- » Quels sont les avantages des choix initiaux ?

La stratégie

- » Exécution dans un environnement extrêmement contraint :
 - » CPU peu puissante
 - » Peu de mémoire
 - » Réseau chaotique
 - » Batterie à économiser au maximum
 - » Traitement prioritaire (la réception d'un appel)
- » Il faut économiser la moindre ressource !
- » Cela apparaît dans tous les choix d'architecture du framework

Les composants d'architectures

- » Les activités (similaire à des pages Web)
- » Les fournisseurs de contenus (similaire à des ressources REST)
- » Des services (similaires à des traitements en tâches de fond)
- » Des receveurs de messages « broadcast », destinés à toutes les applications

Les activités

- » Basées sur le modèle des pages Web.
- » Identifiées par une « Intention », sorte de requête HTTP enrichie (URL, action, type mime, extra)
- » Cycle de vie fragile.
- » Une activité meurt lors de l'affichage de la suivante ou du basculement de l'écran.
- » Mais, si Android est sympa, il peut la garder en cache avant de la reprendre.
- » Les données d'une activité ne sont pas pérennes
- » Android mémorise l'historique de navigation entre les activités (d'où le bouton « retour »)
- » Ainsi, une activité peut déclencher l'affichage d'une activité d'une autre application.

Les fournisseurs de contenus

- » Permettent d'exposer des données aux autres applications via API de type REST.
- » Toutes les données sont identifiées par une URI (content://contacts/people/1)
- » Doivent offrir les quatre services de manipulations classiques (création, modification, consultation, effacement)
- » La consultation doit répondre à une requête et retourner un Curseur.
- » L'implémentation est libre (SQLite interne, fichiers plats, requêtes Web, etc.)
- » Ne permet pas les jointures entres fournisseurs de contenus.

Les receveurs broadcast

- » Composants à l'écoute d'événements génériques
 - » Activation / désactivation Wifi ou Bluetooth
 - » Installation d'une application
 - » Push
 - » Etc.

- » Permettent de réveiller l'application lors d'événements majeurs.

Les services

- » Deux usages :
 - » Offrir des traitements en tâche de fond
 - » Permettre la communication entre objets de différents processus
- » La communication entre processus est très utilisée par le framework Android (même si les développeurs l'ignorent)
- » Utilise un module noyau spécifique.

Déclaration des composants

- » Dans le fichier AndroidManifest.xml de chaque application.
- » Permet d'identifier les applications à déclencher, sans avoir à les exécuter (enregistrement passif)
- » Ainsi, les applications peuvent réagir aux événements sans consommation de ressources

Cycle de vie

- » Android gère le cycle de vie des applications :
 - » Évite autant que possible de tuer une application
 - » Mais n'hésite pas à le faire s'il a besoin de ressources complémentaires
 - » (Les killapps sont contraire à Android, et peuvent avoir des effets de bords dommageable pour les applications)
- » A minima, comme sur d'autres plate-formes mobiles, il n'y a qu'une seule application active à la fois, celle présentée à l'utilisateur.
- » Lorsque l'utilisateur retourne à une application (bouton retour, sélection de l'application, etc.) il peut être nécessaire de la ré-initialiser si elle n'était plus active (d'où un délai parfois long).

Processus

- » Android sépare la notion de processus et la notion d'applications
- » Une application peut être portée par plusieurs processus
- » Plusieurs applications peuvent partager le même processus
- » Il est même possible de lancer un Android dans un unique processus
- » Les processus s'exécutent avec des utilisateurs différents
- » Android peut tuer un processus après l'avoir prévenu. Il en contrôle la consommation des ressources.
- » C'est un élément de sécurité important pour isoler les applications.

system_app

- » Le processus system_app porte le framework Android.
 - » Contient les informations de tous les fichiers AndroidManifest.xml des applications
 - » Permet de gérer le cycle de vie des autres processus
 - » S'il meurt, le téléphone reboot.
- » Pour accélérer le démarrage d'un processus, un machine virtuelle java (Dalvik) est initialisé avec une liste de classes, puis attend une commande pour se dédoubler, associer un utilisateur Unix, et démarrer l'application.
- » Ainsi, une partie du framework est partagée entre les processus

Optimisations

- » De très nombreuses astuces sont utilisées pour optimiser le code
 - » Une machine virtuelle spécifique (Dalvik)
 - » L'utilisation de constante numérique à la place de chaîne de caractères
 - » Une communication optimisée entre les processus
 - » Etc.

Conclusions

- » Garder toujours à l'esprit, les règles fondatrices de la conception d'Android, si on souhaite faire une application pour Android.
- » Conception et développement dirigés par l'économie de ressources.

- » AMELIORER LA PERFORMANCE
- » AUGMENTER LA SOUPLESSE
- » ASSURER LA TRANSPARENCE
- » REDUIRE LES COUTS
- » AMELIORER LA RELATION CLIENT
- » ACCELERER LA MISE SUR LE MARCHE
- » INNOVER
- » AMELIORER L'EFFICACITE
- » ACCROITRE L'ADAPTABILITE
- » GARANTIR LA CONFORMITE



CONSULTING > SOLUTIONS > OUTSOURCING

Pour plus d'informations, contacter :

Philippe PRADOS
+33 (0)6 70 03 89 60
philippe.prados@atosorigin.com

Atos Origin
www.atosorigin.fr

ADVANCE YOUR BUSINESS »