Ness SES - Dharmendra Kapadia – Mar 20, 2015

NQ Mobile Adalyst Platform

Design Document – Version 0.1

Document Contact Information

|  |  |
| --- | --- |
| Ness | |
| Contact | Dharmendra Kapadia |
| Designation | Senior Director – Solutioning |
| Email | [dharmendra.kapadia@ness.com](mailto:dharmendra.kapadia@ness.com) |
| Business Phone | +1 732-986-0397 |
| NQ Mobile | |
| Contact |  |
| Email |  |
| Business Phone |  |

Confidentiality Notice/Disclaimer

This document is disclosed only to the recipient to whom this document is addressed and is pursuant to a relationship of confidentiality under which the recipient has obligations of confidentiality. This document constitutes confidential information and contains proprietary information belonging to Ness Technologies. The confidential information is to be used by the recipient only for the purpose for which this document is supplied. The recipient must obtain Ness Technologies written consent before the recipient or any other person acting on its behalf, communicate any information on the contents or subject matter of this document or part thereof to any third party. The third party to whom the communication is made includes an individual, firm or company or an employee or employees of such a firm or company.

The recipient, by its receipt of this document, acknowledges that this document is confidential information and contains proprietary information belonging to Ness Technologies and further acknowledges its obligation to comply with the provisions of this notice.

The contents of this document are provided in commercial confidence, solely for the purpose of evaluating whether the contract should be awarded to Ness Technologies.

The information contained in this document represents the views and opinions of Ness on the issues discussed, as of the date of publication. Due to the dynamic nature of the industry and the technology that it depends upon, Ness makes no warranty as to the long-term accuracy of the assessments made herein.

Copyright © Ness. All rights reserved.

All product and company names mentioned herein are trademarks or registered trademarks of their respective owners.

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ver.** | **Revision Date** | **Revised By** | **Description of Changes and Reasons** |
| 0.1 | 02/20/2015 | Dharmendra Kapadia | Initial draft |
| 0.2 | 03/20/2015 | Dharmendra Kapadia | Interim update |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[Introduction 4](#_Toc414959545)

[Adalyst Platform Overview 4](#_Toc414959546)

[Mobile Device 4](#_Toc414959547)

[NQ Mobile Application 5](#_Toc414959548)

[AdGear SDK 6](#_Toc414959549)

[SQLite 6](#_Toc414959550)

[Communicator 6](#_Toc414959551)

[PNClient 7](#_Toc414959552)

[Campaign Management 7](#_Toc414959553)

[Adalyst Controller 7](#_Toc414959554)

[RESTful Web Services 8](#_Toc414959555)

[Communicator 8](#_Toc414959556)

[Campaign Management 8](#_Toc414959557)

[PNServer 9](#_Toc414959558)

[AdGear Server 11](#_Toc414959559)

[Texting 12](#_Toc414959560)

[Main Platform Database 13](#_Toc414959561)

[Message Bus Connectivity (Kafka & Zookeeper) 14](#_Toc414959562)

[Appendix A - JSON Message Specification 16](#_Toc414959563)

[Appendix B – Development Links 17](#_Toc414959564)

[AdGear 17](#_Toc414959565)

[Appendix C – RESTful Web Services 18](#_Toc414959566)

[Appendix D – AdGear Server Call Specifications 19](#_Toc414959567)

[Appendix E – Mobile Device SQLite Design Details 20](#_Toc414959568)

# Introduction

This document highlights and details the architecture and components for the NQ Mobile Adalyst platform encompassing the pilot and commercialization phases. This architecture and the components contained therein are based on conversations and feedback from NQ Mobile as of March 20th, 2015. Any information, knowledge, updates, or feedback from NQ Mobile received after March 20th, 2015 are not included in this architecture design. That new knowledge may affect the architecture or some individual components. Also, the initial 2-week Feasibility Phase of the effort will not only verify the contents of this document, but it will also fill in any remaining gaps before the main development begins.

## Adalyst Platform Overview

The Adalyst Platform will consist of a software package which will run on mobile devices (Android & iOS) to enable the delivery and display of ad content in a seamless manner without user intervention. The loading and display of ad content will be triggered from the back-end. The content itself will display on the mobile device as an overlay on top of any currently running applications.

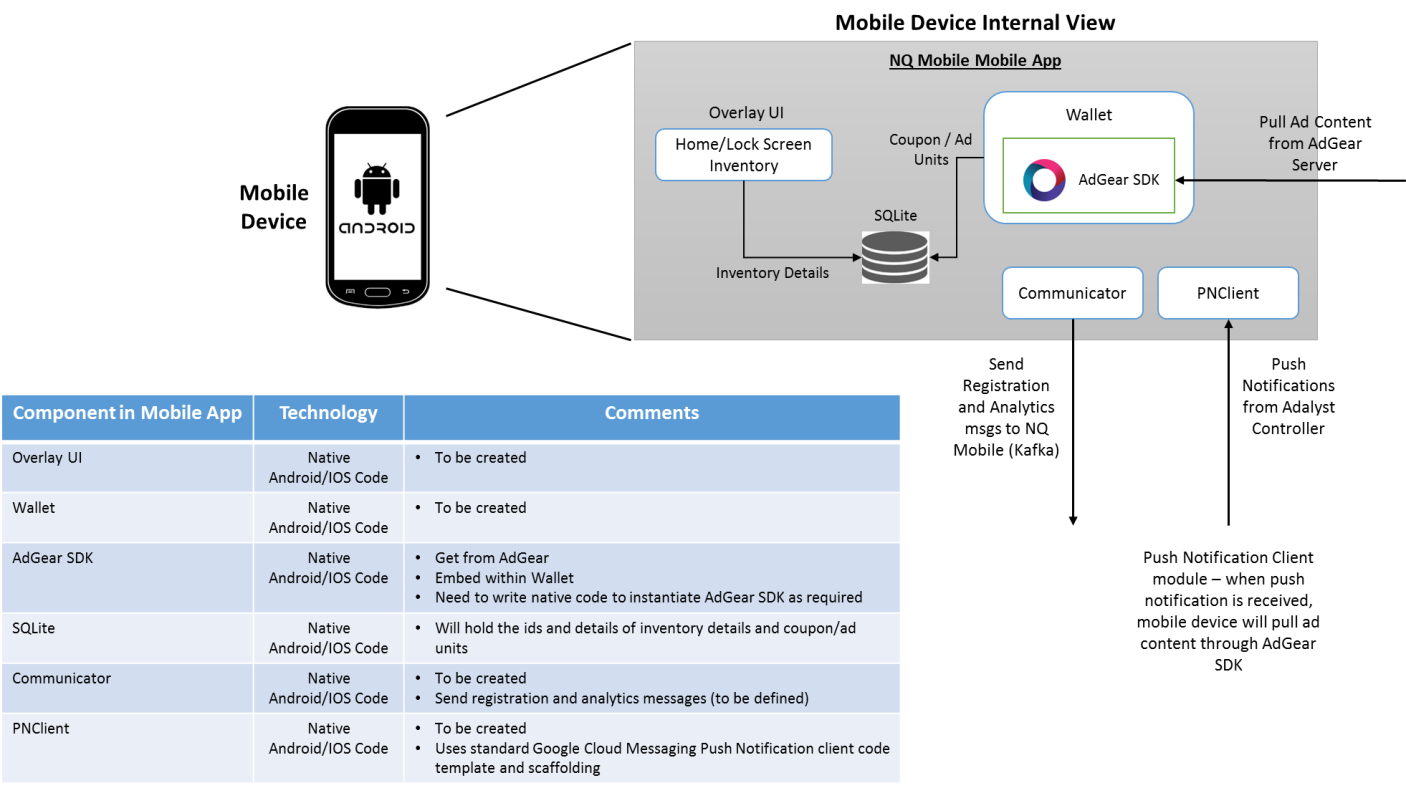
Not all devices with the Adalyst Platform software will display all ad content. There will be a “campaign” run and generated from the back-end. The campaign parameters and attributes will help select the subset of devices that will be a part of the campaign. This will allow for targeted and focused ad delivery. Device, campaign and ad analytics will be gathered part of the campaign from the device.

# Mobile Device

A mobile device is a smartphone running either Google Android (versions 4.1+) or Apple iOS (version 7+). The mobile device will have the NQ Mobile software package installed on it by the carrier. The user will not be able to uninstall this package. The package will be created in native mobile device code:

* Android – Android SDK (http://developer.android.com/sdk/index.html)
* Applie – iOS (https://developer.apple.com/devcenter/ios/index.action)

Figure 1 shows a detailed view of the view of the individual components running within the mobile device. In addition, it lists the technologies used for each components.



**Figure 1 - Mobile Device component details**

Besides the required respective mobile device SDK’s, the package will contain the following components:

* NQ Mobile Application
  + Overlay UI
  + Wallet
    - AdGear SDK
  + SQLite
  + Communicator
  + PNClient

## NQ Mobile Application

The NQ Mobile Application is the main container package for all of sub-components contained within. The application package will be written in the respective native code for each platform. It consists of the following:

* Overlay UI code – this code will do the actual rendering of the ad content on the Home or Lock Screen when directed. It will also house the inventory details.
* Wallet – this code will contain the coupon and ad unit details. It will also house the AdGear SDK as an embedded component, through which all AdGear platform calls will be made by both the mobile device and the Adalyst back-end.

### AdGear SDK

The AdGear SDK is an API created by AdGear which is used to integrate any product or platform to the AdGear ad platform. There are API calls specific to a mobile device. There are API calls that would be used by middleware or server components.

The AdGear API page for developers has details on all of the SDKs and their capabilities:

* <https://adgear.atlassian.net/wiki/pages/viewpage.action?pageId=20217995>
* iOS specific - http://adgear.com/mSDK/files/adgear-ios-txt.html

## SQLite

SQLite is an open-source, on-device data store mechanism which runs under both Android and iOS. It allows any application properly written to manage its own private database. For the Adalyst Platform, the local SQLite instance will persist inventory details, coupon/ad units, and any other application specific data. The design details for the database on the mobile device is specified in [Appendix E](#_Appendix_E_–).

SQLite specifications for the Android and iOS platforms are specified at:

* Android - <http://developer.android.com/reference/android/database/sqlite/package-summary.html>
* iOS - <https://developer.apple.com/technologies/ios/data-management.html>

## Communicator

The Communicator piece will be written in native code. All communications with the NQ Mobile back-end will happen through this module and will be via JSON messages being passed back and forth. The message types that will be sent and received are as follows:

|  |  |  |
| --- | --- | --- |
| Message | Originator | Comments |
| DeviceRegistration | Device | JSON message sent by mobile device to register itself with the Adalyst platform. Details of the message itself are TBD. |
| AnalyticMessages | Device | JSON messages containing campaign and other analytic data. Details of the messages are TBD. |

## PNClient

The PNClient component will receive push handle push notifications from the Adalyst Controller. For Android, push notifications will be handled by the Google Cloud Messaging platform (). Push notifications will go from the Adalyst Controller to the mobile device to trigger ad content pull from AdGear and then ad display through the Overlay UI.

# Campaign Management

The Campaign Management component of the NQ Mobile Platform consists of a set of components within the NQ Mobile environment to execute the campaign itself.

## Adalyst Controller

The Adalyst Controller is the heart of the campaign management system. Figure 2 below shows the internal components which make up the Adalyst Controller as well as the points of outside message integration.

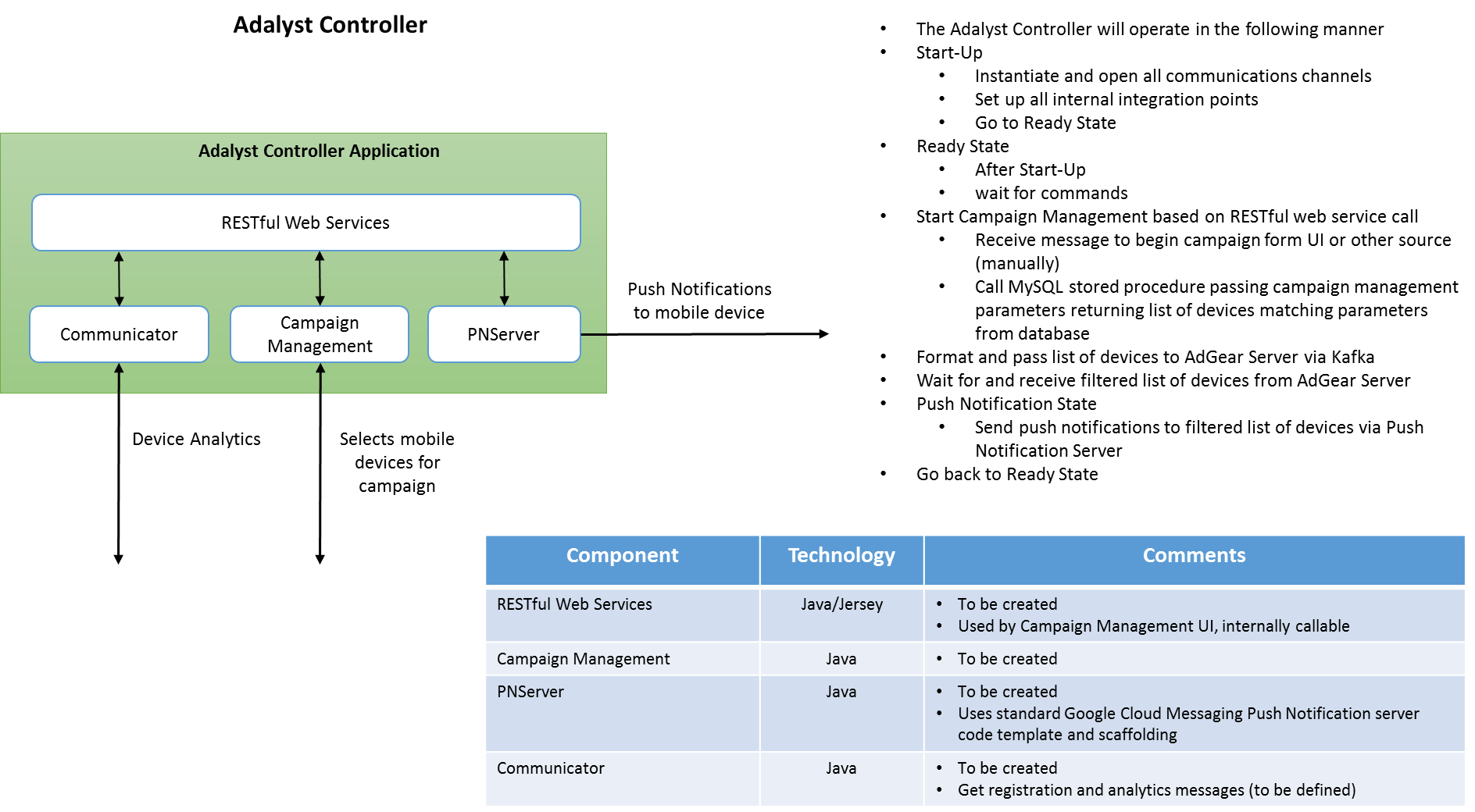


Figure - Adalyst Controller component details

### RESTful Web Services

The Adalyst Controller itself will have a set of RESTful web services to make it perform various functions, including starting a campaign, selecting mobile devices, etc. The RESTful web services will be written in Java and Jersey (<https://jersey.java.net/>) and are documented in [Appendix C](#_Appendix_C_–).

### Communicator

The Communicator module in the Adalyst Controller will be the channel by which analytic data will enter. It will make the appropriate database calls to retrieve data as necessary from the database to perform the appropriate calculations.

### Campaign Management

The Campaign Management piece of the Adalyst Controller will get kicked off by one of the RESTful web services calls. That will move the Adalyst Controller through the various states that it supports as shown in Figure 3 below.

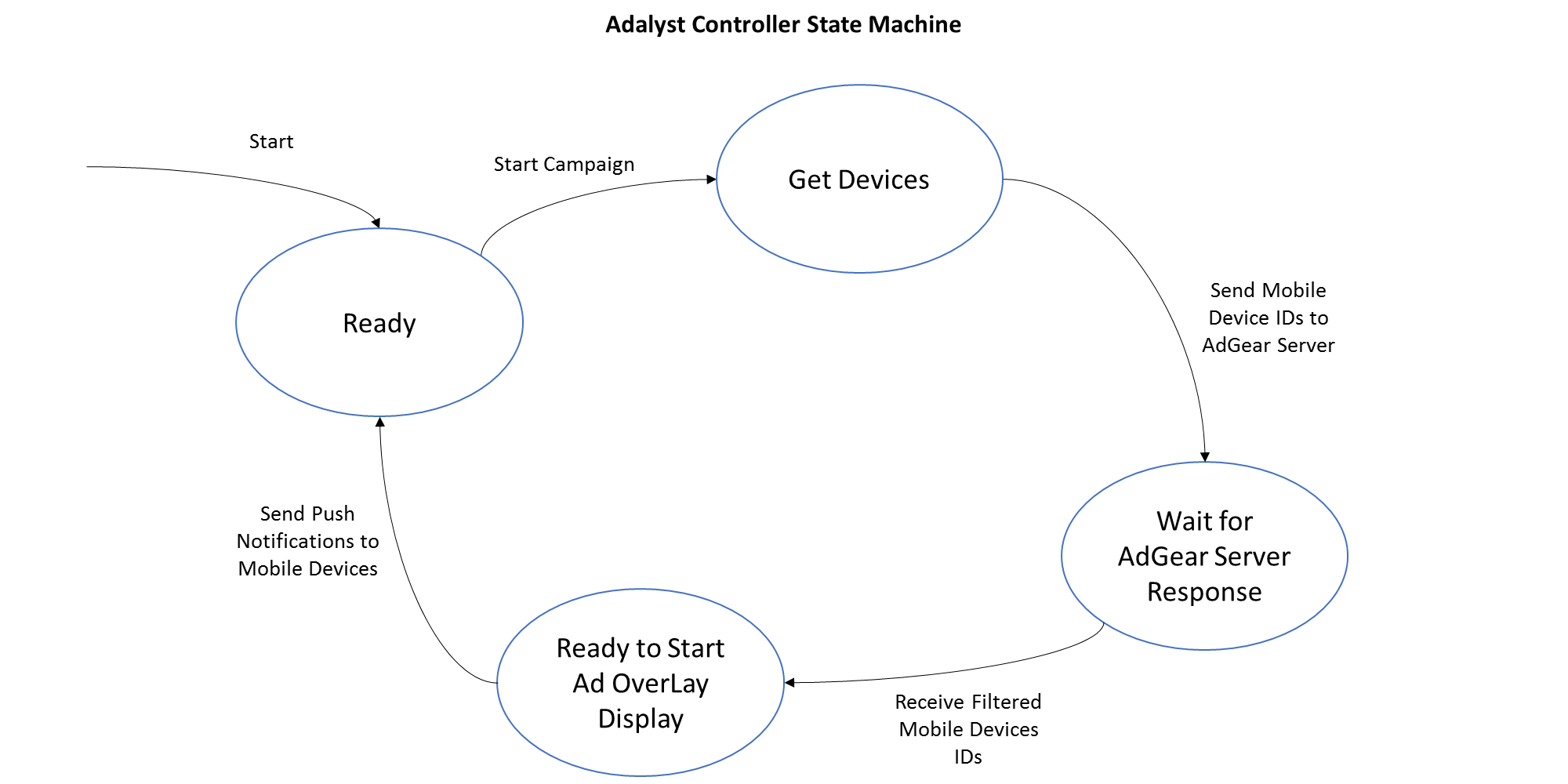


Figure - Adalyst Controller State Machine diagram

A typical campaign will have the following steps:

* Start-Up
  + Instantiate and open all communications channels
  + Set up all internal integration points
  + Go to Ready State
* Ready State
  + After Start-Up
  + wait for commands
* Start Campaign Management based on RESTful web service call
  + Receive message to begin campaign form UI or other source (manually)
  + Call MySQL stored procedure passing campaign management parameters returning list of devices matching parameters from database
* Format and pass list of devices to AdGear Server via Kafka
* Wait for and receive filtered list of devices from AdGear Server
* Push Notification State
  + Send push notifications to filtered list of devices via Push Notification Server
* Go back to Ready State

### PNServer

The main communication method between the Adalyst Controller and the mobile device is the Push Notification Server (PNServer). That server will be a Java-based component running on the backend. It will accept instructions in the form of messages from the Adalyst Controller.

The PNServer will be the server analogue to the PNClient module embedded in the mobile device. It will ‘forward’ the push notifications to send to a set of mobile devices. There are different methods of push notification implementation for Android and iOS as detailed in the table below.

|  |  |  |
| --- | --- | --- |
| Mobile OS | Method | Link Details |
| Android | Google Cloud Messaging (GCM) | <https://developer.android.com/google/gcm/index.html> |
| iOS | Apple Push Notifications (APN) | <https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Chapters/ApplePushService.html> |

Note: GCM can also be used to push notifications to iOS devices as well as Android devices. It may be desirable to use GCM to push notifications to both Android and iOS mobile devices from one platform component because it reduces the development effort. The instructions on how to do that are:

* <https://cloud.google.com/developers/articles/ios-push-notifications/>

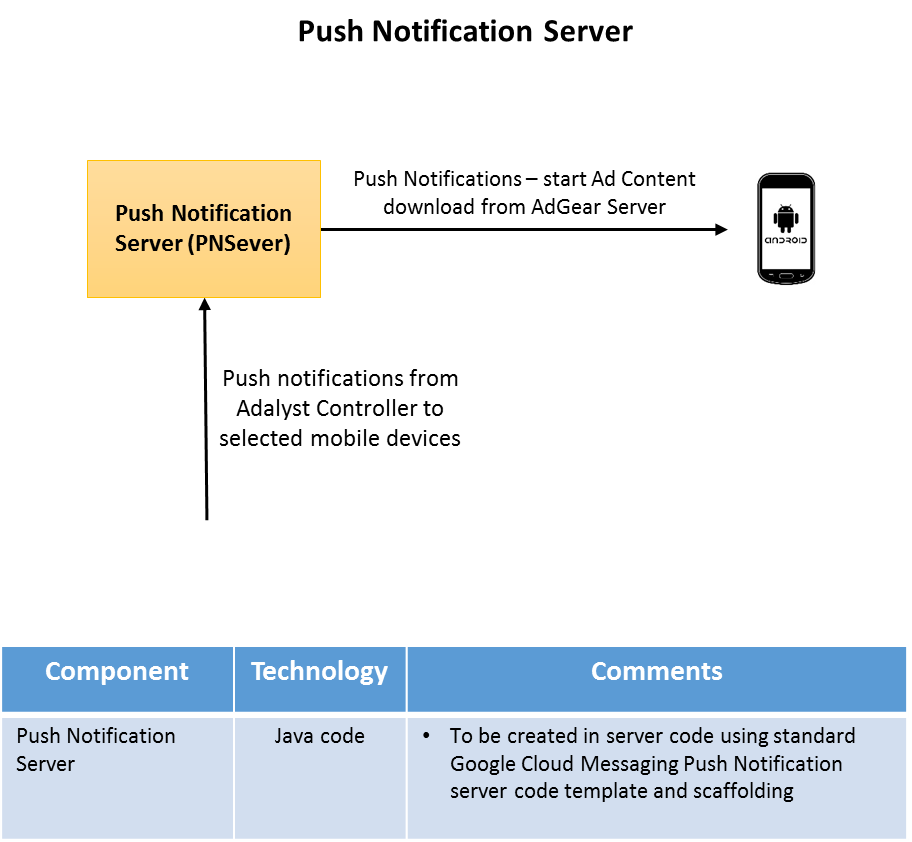


Figure - Push Notification Server details

# AdGear Server

The AdGear Server will be located outside of NQ Mobile’s network or environment. It will have a set of API calls which will be exposed to the Adalyst Controller and the [AdGear Server SDK embedded](#_AdGear_SDK) within the NQ Mobile application installed on the mobile device. Figure 5 below shows a block diagram of the interactions related to the AdGear Server. The AdGear Server will be called to perform the following functions:

* Receive a list of mobile devices from the Adalyst Controller as part of a campaign
* Return a filtered list of mobile devices back to the Adalyst Controller
* Called by the mobile device to pull ad content as part of a campaign after it receives a push notification generated from the Adalyst Controller

The relevant AdGear Server call specifications are listed in [Appendix D](#_Appendix_D_–).

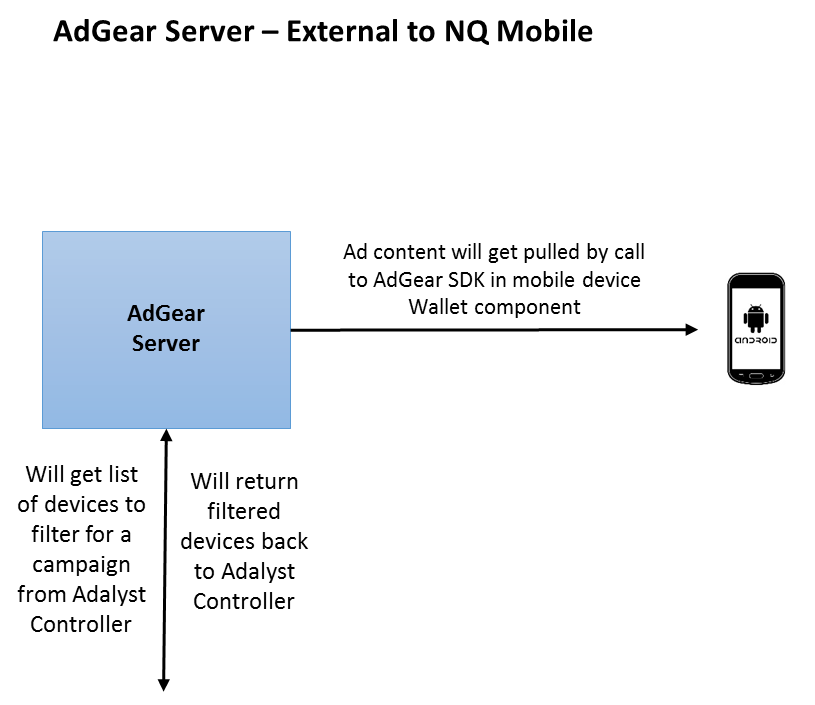


Figure - AdGear Server Interactions

# Texting

Texting is another communication avenue between the Adalyst Controller and the mobile device. It is, however, not considered a primary or desirable method of communication and is not in-scope for the pilot phase.

If and when texting is realized on the NQ Mobile platform, it would be in the form described in Figure 6 below. The Adalyst Server will generate the text messages to be sent to the mobile devices. Those messages will get sent through the Kafka message broker to the carrier’s text messaging infrastructure which will be utilized to send the actual texts.



Figure - Platform Texting interactions

# Main Platform Database

The NQ Mobile backend will have its own persistent stores to satisfy all data capturing requirements. There will be a NoSQL-based store (MongoDB) for device analytic messages, all push notification traffic, and other runtime data. There will also be a relational-based store (MySQL) for reporting and device profile data.

Initially, all the raw messages will be captured in MongoDB. A module will listen to the MongoDB instance for particular messages, process them, and put them into the MySQL store (reporting, device registration, etc.). Figure 7 below captures the details of the database stores.

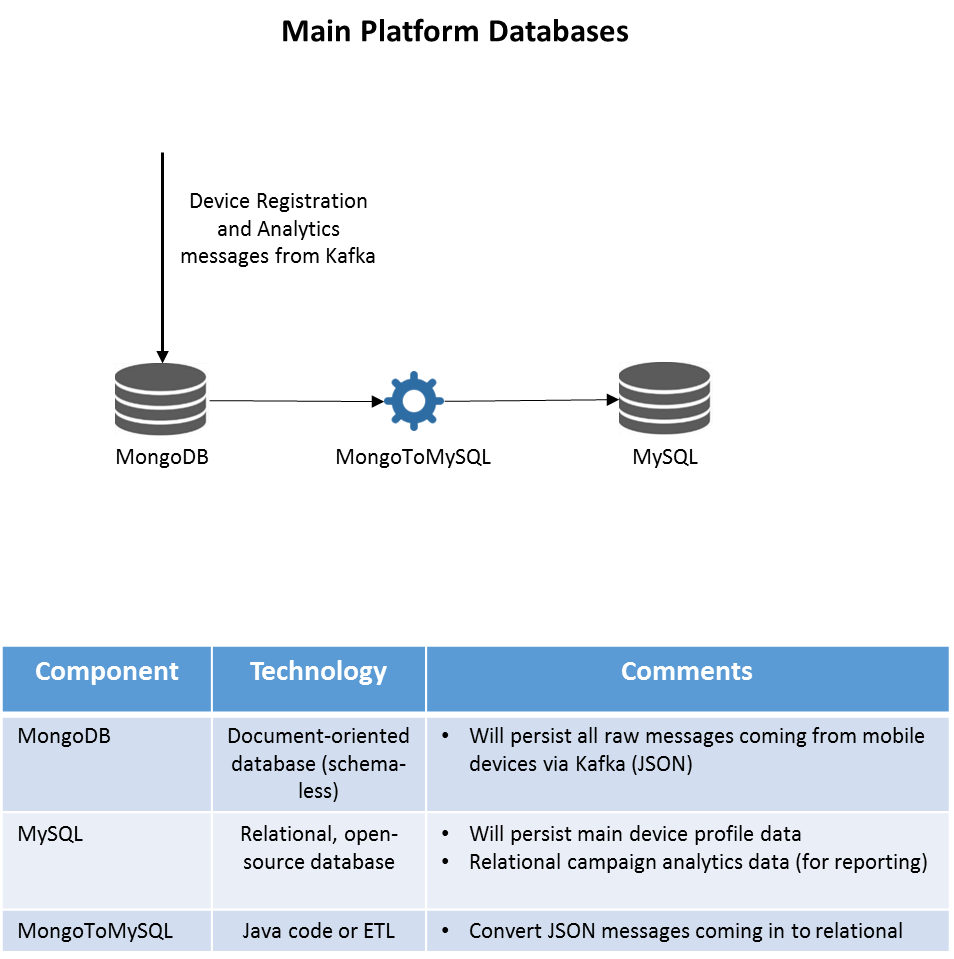


Figure - Platform Persistent Store Details

# Message Bus Connectivity (Kafka & Zookeeper)

Kafka is a high-speed message broker which uses publish-subscribe to send and receive messages. It is a robust, open source, Java-based component which can be easily scaled to handle hundreds-of-thousands of messages per second without requiring exotic hardware. ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services. It coordinates the messages among the Kafka nodes.

For the NQ Mobile platform a 3-node Kafka cluster will be provisioned to handle the messages between the NQ Mobile platform and the mobile devices. There will be topics created for the Adalyst Controller to publish messages to. The mobile devices will subscribe to the appropriate topic so that it can consume messages from the Adalyst Controller. The consumption of the message will direct the mobile device to perform the appropriate function, such as pulling ad content from the AdGear Server.

Figure 8 below details the high-level component interactions between the AdGear Server, mobile devices, and NQ Mobile platform over Kafka.

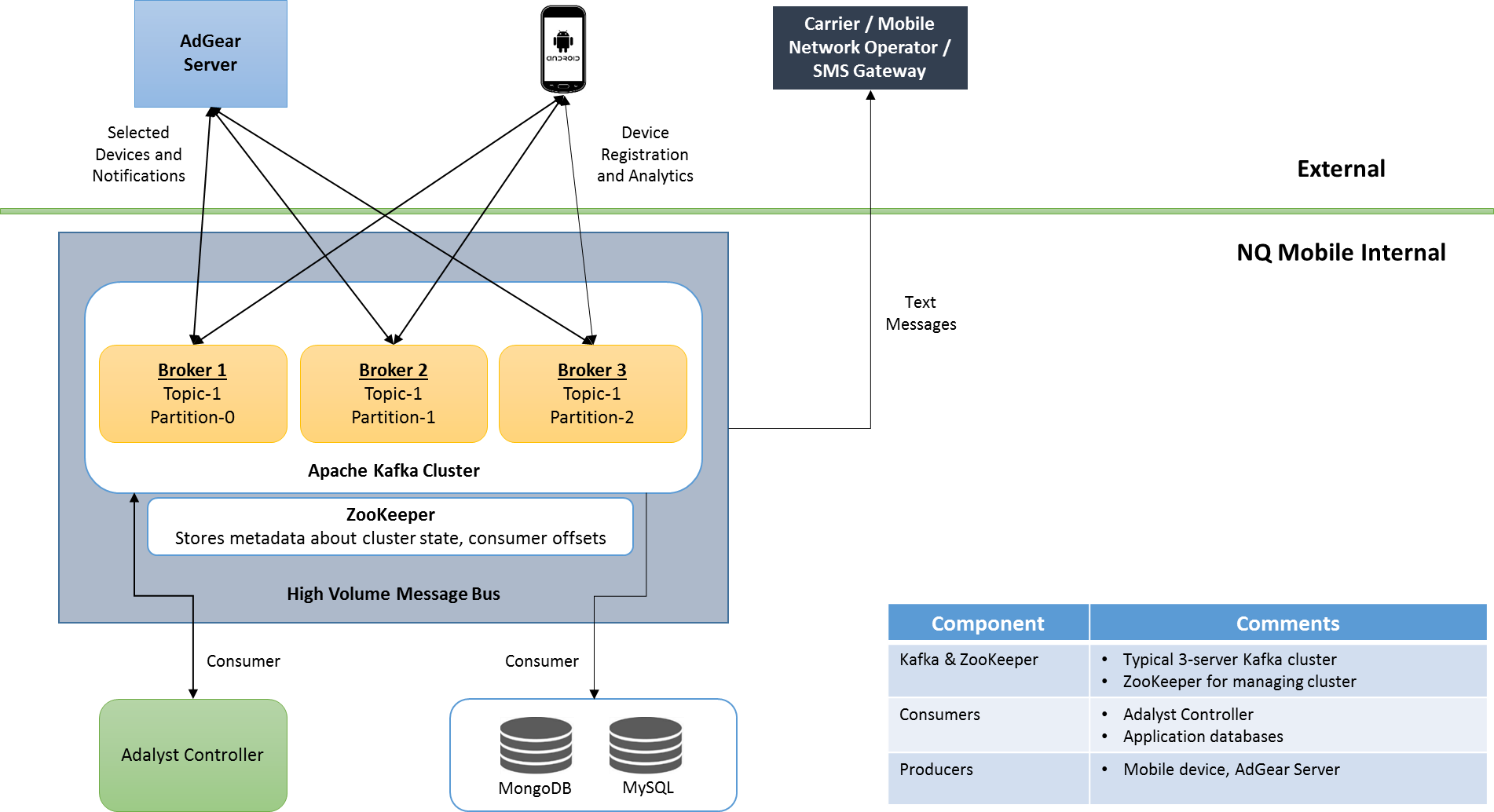


Figure - Kafka/Zookeeper Details

# Appendix A - JSON Message Specification

**Message Number:** 1

**Message Name:** RegisterDevice

**From:** Mobile Device

**To:** Adalyst Platform

**Details:** Msg sent to register the device with the Adalyst Platform. Once registered, the device does not have to register again.

**Example:**

{“RegisterDevice”: {

“msgId”: “1”,

.

.

.

}}

# Appendix B – Development Links

## AdGear

Integrating AdGear Android SDK with Your App

<https://adgear.atlassian.net/wiki/display/adgear/Integrating+AdGear+Android+SDK+With+Your+App>

Integrating iOS SDK with Your App

<https://adgear.atlassian.net/wiki/pages/viewpage.action?pageId=4194349>

AdGear API Usage Examples

<https://adgear.atlassian.net/wiki/display/adgear/API+Usage+Examples>

# Appendix C – RESTful Web Services

# Appendix D – AdGear Server Call Specifications

# Appendix E – Mobile Device SQLite Design Details