plAsync

A framework for asynchronous mutiplayer games on mobile devices

plAsync Objectives

- Support playing turn based games with friends across Android, iOS, and Facebook
- Provide a software platform that is independent of any cloud service (i.e. can be deployed on any server)
- Abstract away the details of messaging and game data exchange from the game developer
- Facilitate the processes of inviting players, hosting games, joining games, and notifying players when it is their turn and other relevant game data

Why?

- Mobile platforms are ideal for turn based gaming
  - Game with friends without location or time constraints
- There are a number of subscription based services that provide messaging
  - Urban Airship, OpenFeint
  - Have to monetize app to cover subscription costs
  - Most are very focussed on in-app purchase (i.e. microtransactions)
  - Free devs from the tyranny of these providers so they can decide which cloud service to use and whether and how to monetize.
- The communications and messaging requirements for most games is very similar, easily made generic, and ripe for a framework
- Have fun and learn some cool new technologies

Notification Technologies

A key aspect of asynchronous multiplayer gaming is asynchronously notifying a user that it is their turn and outcome of other player's turns.

- Fortunately, each of the supported platforms has support for such notifications
  - Android - Google Cloud Messaging
  - iOS - Apple Push Notifications
  - Facebook - Facebook notifications

Google Cloud Messaging

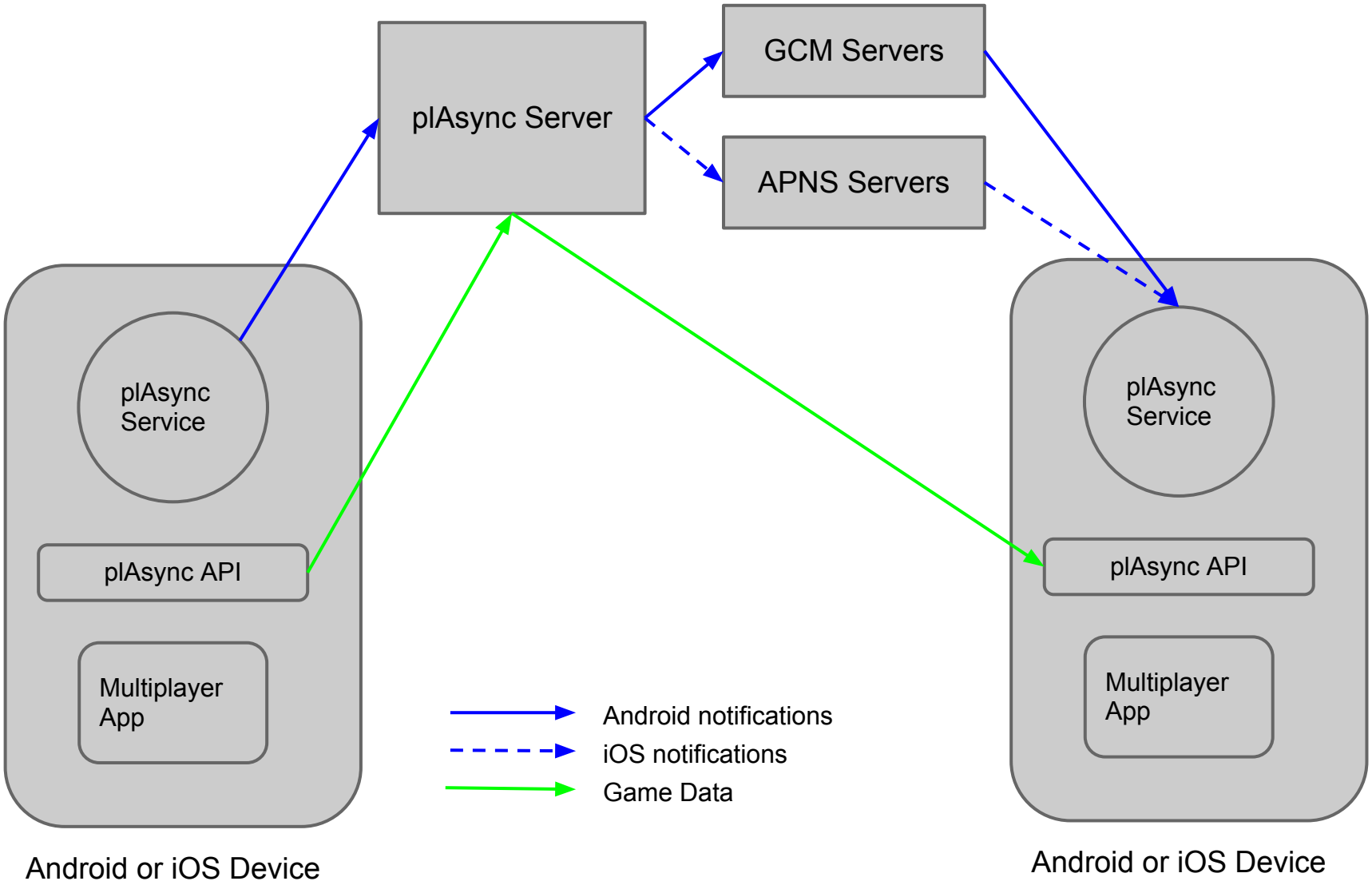
- The most battery and data efficient way to send messages to android devices
- Achieves this by leveraging existing messaging between android devices and google servers
- Not intended for transfer of data, just to send a message to the device to check the server for new data

Top Level Requirements

Asynchronous multiplayer games require

1. Async notifications for invites, joins, resigns, start of turn, and in some cases out of turn actions.
2. Sharing of game data and state between players such as player pawn locations, important events during player turn, effect of player action on other players.
3. Work across android, iOS, and Facebook (not everyone's friends have the same device)

Technical Concept



plAsync Components

- plAsync Server
  - Implements messaging and RESTFul services to allow apps to be notified of game events and exchange game data
  - Can be hosted on any cloud service
- plAsync Android SDK
  - client side components for Android devices
- plAsync iOS SDK
  - client side components for iOS devices
- plAysnc Facebook SDK
  - client side components for Facebook apps

plAsync Android Components

- Android service
  - Manages GCM registration and communication
  - One per device - can be used by multiple apps on a device
- Android client api
  - Provides the service interface for apps that use the service

Architecture TBDs

- plAsync Server -
  - One per app, or many apps can use the same
  - Platform - Grails?, Java with Spring?, something else?
  - User management strategy
- iOS analog to GCM?
  - Looks like it's Apple Push Notification Services (APNS)
- plAsync Service Distribution
  - If there is to be one per device we need to have the users install if from the store
  - Apps need to check if the service is installed
- Scope/role of the framework
  - How much should the framework do vs the app?

plAsync Android Service and Client API Functions

1. Register application
2. Register users
3. Update users(i.e. add application, add email, change registrationId)
4. Remove users
5. Create game (new game session, invite players, etc...)
6. Start game
7. endTurn
8. setGameState
9. getGameState
10. endGame
11. receiveJoinGameNotification
12. receivePlayerTurnCompleteNotification

plAsync Server Functions

1. Register users
2. Update users(i.e. add application, add email, change registrationId)
3. Remove users
4. Create game (new game session, invite players, etc...)
5. Start game
6. endTurn
7. setGameState
8. getGameState
9. endGame

Next steps

Define the architecture
Define the server API
Define the client side API for Android