Problem

This problem specification will be on the inefficiency of many android developers, it will be from the perspective of a business student. Looking through the android play store, there are many apps that overlap; for example, there are many music player apps that do the exact same thing such as Poweramp, Google Play music, Apollo Music. Perfect competition is a good thing because it avoids a monopoly and it keeps most of the apps free because no major player has the market power to the set price of a product. The problem occurs when developers, who have the objective of creating one product that competes in one industry, try to create one product that competes in many industries. For example, apps which provide music downloading services are built with their own mediocre music player; therefore they are competing in the music downloading market and music playing market. The argument can be made that the developer has included a music player for convenience; however this benefit comes with many costs and the costs far outweigh the benefit. The costs of including features that do not fit with the main objective are:

1. More time spent on creating the final app: There is a lot more time spent on planning and programming when other app features are introduced. For example, introducing the music player feature in the app would require the developer to spend more time drawing wire frames, doing user interface tests, programming and debugging. As a result, research and development costs increase and time to produce a product that satisfied the main objective increases.
2. Lack of competencies: The programmer might not have the competencies and knowledge required to make the extra features; therefore, the programmer must either rely on other libraries and open source projects, learn how to create the feature themselves or hire an individual with the required knowledge. The first option requires the programmer to spend time learning how to implement the library or open source project into their application, the time for this can vary as some libraries are easy to implement while others are more daunting. The second option has a much steeper learning curve compared to the first option, and would most likely require more time than any of the other options. The latter option will result in an increase in costs since a new individual must be hired; however, the time to produce the application would decrease.
3. Decrease in user experience: Usually when new functions are added it decreases user experience because it increases the steepness of the learning curve for new users to learn how the application works. As a result, users may not use the application because it seems confusing or too overwhelming to learn. In addition, if the programmer is creating an application for an industry which they lack all competency, they will most likely not be able to compete against the major players in terms of usability and user experience.
4. Dilution of the brand and the program: By adding extra convenience features that do not help solve the main problem the application is solving, the application dilutes its main objective and confuses the users on what the application actually does. For example by adding a music player to the music downloading application, the user may perceive the application’s main objective to be a music player and they may not use the application for the problem it was meant to solve.

This problem though stems from applications not being able to allow their functionality to be increased through third party developers. In other words, most applications on the Google Play store do not allow plug-ins to be built for their applications. In some scenarios, the plug-in architecture would not be plausible but in many others it would be plausible. Developers have failed to see the importance of plug-ins, and how it could be beneficial for their applications and other third party developers. Using the previous example, if Google Play had a supported plug-in engine, it would allow third-party developers to easily create a plug-in which downloaded music. This would solve all the problems listed above and plus the developer can now focus on their competency, reduce research and development costs and produce applications much faster.

The android framework has supported all the fundamental tools required to make a successful plug-in engine since API level 1. The fundamentals are: Services, Inter-Process Communication, Broadcast Receivers and the API access to Package Manager. Even though there have been applications such as Dachclock who have supported this ideology, they have not fully embraced the idea. They only allow the plug-ins to communicate with the host application and vice-versa; they do not embrace the idea of plug-ins communicating with each other. The main objective of creating a plug-in engine should be that they allow bidirectional communication with host application, and that they allow other plug-ins to communicate with each other in order to solve a bigger problem.

This project is still a work-in-progress and shows how to utilize this plug-in philosophy. I will update this document when I finalize the design of the plug-in engine.