The Economics of App Review*

Ming Gao† Travis Ng‡

February 17, 2016

Abstract

A platform for apps, such as Apple App Store, facilitates the interaction between app users and app developers. For political, moral, and various other reasons, some apps may cause more trouble to the platform than others even if the users may not value them differently. In a two-sided market model with both "well-behaved" and "trouble-making" developers, we show that, whereas an app review is useful for screening trouble makers, the platform faces a trade-off between cost saving (from less trouble) and a loss of the attractiveness to the users (from reduced app variety). The platform's incentive to use app review depends on the additional cost it incurs from each "trouble maker", and on their proportion among all developers. Given the additional cost incurred, app review benefits the platform if and only if the proportion of trouble makers is sufficiently small, such that after screening them the platform remains fairly attractive to users. Given their proportion, however, app review helps if and only if the "trouble makers" create enough "trouble", such that the total cost saved from app review is not negligible. Our welfare analysis through simulations cautions regulators that the platform's incentive to use app review may be aligned or misaligned with the interest of social welfare, depending on the distributions of app users and developers, the strength of network effects and other parameters.

Key Words: app review, Apple, Android, two-sided market, non-price discrimination

JEL **Classification**: D42, L11, L12

---

*Preliminary and incomplete. We thank seminar participants at Shanghai Jiao Tong University and Shanghai University of Finance and Economics. Ming Gao acknowledges the financial assistance of Project 20151080389 supported by the Tsinghua University Initiative Scientific Research Program and Project 71203113 supported by the National Natural Science Foundation of China.

†Corresponding author: School of Economics and Management, Tsinghua University, Email: gaom@sem.tsinghua.edu.cn.

‡Department of Economics, The Chinese University of Hong Kong.

1 Introduction

According to ABI Research the smartphone app market was worth US$27 billion in 2013.[1] App distribution platforms for smartphones, such as App Store and Google Play, are examples of two-sided platforms where smartphone users and app developers interact with each other, and the benefits each side derives depend directly on the number of agents on the other side. That is, other things being equal, smartphone users usually prefer platforms with more apps, whereas app developers prefer platforms with more users. Rochet and Tirole (2003 and 2006) and Armstrong (2006) provide some of the canonical models for two-sided markets. Kouris (2011) models specific characteristics of app distribution platforms and studies optimal pricing strategies of such platforms. His model, however, does not analyze the effect of any non-price strategy on market outcomes.

The app review process is one non-price strategy that Apple uses. While App Store and Google Play are the two dominant app distribution platforms, they used to have a major difference, i.e. there has long been an ex ante review process on App Store, but not on Google Play until 2015.[2] Why the difference? To answer this question, this paper builds a monopoly model of two-sided market and analyzes the effect of Apple's app review process on app users, developers, App Store and social welfare.[3]

In some two-sided markets, platforms use quality as a criterion to exclude some agents on one or both sides from entering the platforms. For instance, night clubs are likely to forbid guys and girls looking lousy from entering. Hagiu (2009) provides a model of using quality as an exclusion criterion and investigates the conditions for the imposition of such an exclusion system.

Is quality the main selection criterion for the App Store's review process? We argue that it is unlikely for three reasons: First, the variety of existing apps just seems too enormous to allow for a concensus on the objective measures for their quality. Second, the more innovative (and hence different from other apps) a new app is, the more difficult it seems to gauge its quality.[4] Third, many examples suggest that apps can be rejected for reasons that are apparently unrelated to quality (e.g. Google Map was excluded for quite some

[1] Source: https://www.abiresearch.com/press/the-mobile-app-market-will-be-worth-27-billion-in-

[2] Google Play did not introduce a review process for apps until early 2015. Source: http://www.theverge.com/2015/3/17/8231125/android-apps-now-reviewed-by-google

[3] Apple describes its App Store review guidelines here: https://developer.apple.com/appstore/guidelines.html

[4] We are referring to "user quality" rather than what Apple might define as quality. Consider gambling apps, for instance, which are likely valued highly by just a small proportion of all app users. It is probably unfair to say that a new gambling app must have low quality just because most users might not even consider using it. But even if most gamblers would use this new app, is it fair then to say that it must have high quality? What about an app with a completely new function that cannot even be categorized? How should its quality be measured?

time by Apple).[5]

For example, some apps are rejected because they duplicate existing apps, their user interface designs do not meet Apple's requirements (which do not necessarily mean they are of lower quality), they contain objectionable content (which some users may value highly), or they mention other supporting platforms. Apple uses its own set of implicit standards to review apps which are not necessarily related to quality.

We construct a model different from Haigu (2009) to investigate how an app review process affects smartphone users, app developers and the platform. We derive conditions for the incentive of a monopoly app distribution platform to review apps, and study the welfare implications.

We assume there are two types of app developers who are indistinguishable to the platform before they are reviewed. For political, moral, and/or various other reasons, one type causes more trouble to the platform and is more costly to serve than the others. For instance, being associated with certain politically sensitive or controversial topics may be bad for the platform's public image, and it may therefore consider it troublesome if developers show them in their apps. Once an app is submitted for review, however, the true type of its developer will be revealed to the platform. If the platform pleases, it can use the review process to discourage participation by developers that are more costly to serve. For instance, the platform can repeatedly reject such developers' apps or simply prolong the review process such that some of them will eventually give up.

In other words, app review can be used to destroy value of developers who are more costly to the platform. But such developers are not necessarily less valuable to app users. We focus on the situation where users have a preference for variety that is reflected by the total number of developers on the platform. Therefore, when screening more costly developers through app review, the platform is faced with a trade-off between the cost saving and the loss of its overall attractiveness to app users.

We develop a simple model à la Armstrong (2006) and find that the platform's incentive to review apps depends on the additional cost it incurs due to each "trouble-making" developer, and on their proportion among all developers. Given the additional cost incurred, app review benefits the platform if and only if the proportion of trouble makers is sufficiently small, such that after screening them the platform remains fairly attractive to users. Given their proportion, however, app review helps if and only if the "trouble

[5] For some useful comments on App Store's review process, read: http://en.wikipedia.org/wiki/Approval_of_iOS_apps and http://www.tuaw.com/2008/08/07/thoughts-on-the-iphone-app-store-review-process/. The first website provides some examples of rejected apps. http://venturebeat.com/2013/02/08/9-surprising-reasons-mobile-apps-get-rejected-from-the-apple-app-store/ provides 9 surprising reasons for an app to get rejected from App Store.

makers" create enough "trouble", such that the total cost saved from app review is not negligible.

Our welfare analysis through half a million simulations cautions regulators that the platform's incentive to use app review may be aligned or misaligned with the interest of social welfare, depending on the distributions of app users and developers and other parameters, including the strength of network effects.

It remains a black box to outsiders as to why exactly an app platform like Apple imposes an ex ante review process. We are simply suggesting one arguably straight-forward way to look at it. Is it a complete view? We do not think so. But to the best of our knowledge, there is yet any work that completely decodes the platform's true incentives for app review. This may be the reason why even experienced app developers would get rejected by App Store from time to time. Even though Apple has an official website for app review rules and statistics of recent review outcomes, it does not reveal clear reasons for the majority of rejections.[6] It is plausible that some of the apps that got rejected are highly valuable to some (if not most) users (such as Google Map, Bitcoin and Google Voice). Therefore, app quality to users, though stated officially by Apple as a criterion, cannot be regarded as their sole criterion for screening. Trouble-making appears a reasonable usual suspect, where troubles to Apple by an app can be broadly defined as including everything from political position, to moral (such as apps with indecent content), to market power, etc. Appendix B gives more details.

Although we construct a monopoly model, we certainly note that Google competes with Apple in app distribution. We do not think that the competition among them is irrelevant. However it remains a challenge to us to properly incorporate the following three prominent market features into a duopoly or oligopoly model: a) multihoming by both sides of the market (such as the fact that many users carry an iphone for personal use and an Android phone for business, and that seemingly few developers work exclusively on IOS but not Android, or vice versa); b) the difference in the platforms' main revenue sources (Google's is mainly Ad revenue, as indicated in the Appendix, but Apple's is very different); c) Apple being highly vertically integrated in terms of hardware and software, whereas Google heavily concentrating on smartphone OS than hardware.

[6] It may seem that Apple has a transparent and self-explanatory app review standard, with a long list of guidelines clearly written in black and white. However in January 2016, Apple announced the very top one among 10 most common reasons for rejections from App Store, accounting for 44% of all, was "Other reasons", without revealing what exactly those other reasons were. (https://developer.apple.com/app-store/review/rejections/) And the second most common reason, taking up 16%, was "More information needed." These two obscure categories together account for 60% of all rejections. Some clear reasons for app rejections that Apple did reveal included "bugs" (12%), "complex or less than very good" (6%), "irrelevant content and functionality" (4%), and "crashes" (3%).

One additional benefit of our monopoly model, though, is that it can be contrasted easily with Armstrong's (2006) monopoly model and many other models in the literature that are based on his.

2 The model

Basic set-up

There is a monopoly app distribution platform which facilitates interaction between two groups of agents, where each group is represented by a continuum. Group-1 agents are app users and group-2 agents are app developers. Denote n_1 and n_2 the numbers of agents eventually joining the platform from group 1 and 2, respectively. The platform charges a fixed price p_1 and p_2 to each agent in group 1 and 2, respectively, for access to the platform.

App users have a preference for app variety. The utility that an app user derives from participation on the platform is

$$u_1 = (\alpha - \gamma)n_2 - p_1, \tag{1}$$

where $\alpha(> 0)$ represents the network benefit she enjoys from interacting with each developer, and $\gamma(\geq 0)$ represents the charge by each developer. We treat γ as an exogenous parameter.[7]

Without app review, an app developer's utility is given by

$$u_2 = \gamma(1 - \beta)n_1 - p_2, \tag{2}$$

where $\beta \in [0, 1]$ represents the commission that the platform takes from each developer's revenue.

Following Armstrong (2006), we assume that the number of app users depends solely on their utilities, denoted

$$n_1 = \phi_1(u_1).$$

where ϕ_1 is strictly increasing.[8]

Two developer types

[7]We choose not to model any developer's optimization process or the competitive situation among developers, but focus on the platform's incentives. In real life, most apps are provided for free and therefore one can set γ equal to 0. We however also allow for the possibility of a positive exogenous γ.

[8]We provide a specific functional form of ϕ_1 in section 5.

The group of developers consists of two types. Type 1 are "well-behaved" developers and their proportion is $\lambda \in [0,1]$; the remaining $(1-\lambda)$ proportion are type-2 developers who are "trouble makers." Denote n_{2k} the number of type-k developers on the platform $(k = 1, 2)$, and assume that, when a developer derives utility u_2 from participation in the platform, the numbers of different developers are given by

$$\begin{aligned} &\text{Type-1 developers:} \qquad && n_{21} = \lambda \cdot \phi_2(u_2), \\ &\text{Type-2 developers:} \qquad && n_{22} = (1-\lambda) \cdot \phi_2(u_2), \end{aligned}$$

where ϕ_2 is strictly increasing. The total number of all developers on the platform is therefore

$$n_2 = n_{21} + n_{22}.$$

The two types are equally valuable in the eyes of app users, and they only differ in terms of the cost that the platform incurs to serve them. In particular, it costs f_{21} to serve each type-1 developer and f_{22} to serve each type 2. Denote

$$\Delta f \equiv f_{22} - f_{21}.$$

Suppose it costs the platform f_1 to serve each app user. Without loss of generality, assume $f_1, f_{21}, f_{22}, \Delta f \geq 0$.

Introducing app review

We model the app review process as designed to discourage trouble-making developers' participation. In particular, app review results in a **reduction** in the utility that type-2 developers derive from the platform, denoted by $R \in [0, \bar{R}]$, whereas app review does not affect the utility of any type-1 developers.[9] The upper bound $\bar{R}(> 0)$ on R simply indicates that the platform can only reduce type-2 developers' utility within a reasonable range, but not infinitely.

Therefore, when each type-1 developer derives u_2 from the platform, the utility that each type-2 developer derives will be $u_2 - R$ as long as the platform uses app review. The number of type-2 developers is summarized as follows for a given utility u_2 derived by type-1 developers:

$$\begin{aligned} &\text{without app review:} \qquad && n_{22} = (1-\lambda) \cdot \phi_2(u_2), \\ &\text{with app review:} \qquad && n_{22} = (1-\lambda) \cdot \phi_2(u_2 - R). \end{aligned}$$

[9] In the extension in section 4 we relax this assumption and allow for the possibility that app review may imperfectly target type-2 developers.

The platform's profit is given by

$$\pi = (p_1 - f_1)n_1 + (p_2 - f_{21})n_{21} + (p_2 - f_{22})n_{22} + \beta\gamma n_1(n_{21} + n_{22}), \tag{3}$$

3 When to Review Apps?

We now focus on the impact of app review on the platform's profitability. To facilitate a clear analysis, we consider the utilities that the platform provides for different sides, u_1 and u_2, as its choice variables, and treat the utility destroyed by app review to type-2 developers, R, as an exogenous parameter. This approach allows us to show how the platform's maximized profit changes at different R. Whether or not to use app review remains the platform's choice to make. When it does, R will be deducted from type-2 developers' utility, and when it does not, there will be no deduction.

Rewrite expression (3) as a function of u_1, u_2 and R as follows:

$$\pi(u_1, u_2, R) \equiv (p_1 - f_1)n_1 + (p_2 - f_{21})n_{21} + (p_2 - f_{21} - \Delta f)n_{22} + \beta\gamma n_1(n_{21} + n_{22}), \tag{4}$$

where

$$\begin{aligned}
n_1 &= \phi_1(u_1); \\
n_{21} &= \lambda\phi_2(u_2); \\
n_{22} &= (1-\lambda)\phi_2(u_2 - R); \\
n_2 &= n_{21} + n_{22}; \\
p_1 &= (\alpha - \gamma)(n_{21} + n_{22}) - u_1; \\
p_2 &= \gamma(1-\beta)n_1 - u_2.
\end{aligned}$$

The first-order condition of (4) with respect to u_1 is

$$\underbrace{(\alpha n_2 - u_1 - f_1)}_{\text{economic profit from each user}} \cdot \underbrace{\frac{dn_1}{du_1}}_{\text{rise in } n_1} = \underbrace{n_1}_{\text{loss in revenue}} \tag{5}$$

where $\frac{dn_1}{du_1} = \phi_1'(u_1)$.

The first-order condition of (4) with respect to u_2 is

$$\underbrace{(\alpha n_1 - u_2 - f_{22})}_{\substack{\text{economic profit from}\\ \text{each type-2 developer}}} \cdot \underbrace{\frac{\partial n_2}{\partial u_2}}_{\substack{\text{rise in \# of}\\ \text{developers}}} = \underbrace{n_2}_{\substack{\text{loss in}\\ \text{revenue}}} - \underbrace{\Delta f}_{\substack{\text{cost saved by}\\ \text{each type-1}}} \cdot \underbrace{\frac{dn_{21}}{du_2}}_{\substack{\text{rise in \#}\\ \text{of type-1}}} \tag{6}$$

where $\frac{\partial n_2}{\partial u_2} = \lambda\phi_2'(u_2) + (1-\lambda)\phi_2'(u_2 - R)$ and $\frac{dn_{21}}{du_2} = \lambda\phi_2'(u_2)$.

Each developer creates a total value of αn_1 for the platform, and therefore the economic profit that a type-2 developer generates is $\alpha n_1 - u_2 - f_{22}$. When the platform raises u_2, a total of $\frac{\partial n_2}{\partial u_2}$ new developers join the platform, and the left-hand side of (6) represents the total increase in economic profit from these new developers, if they were all of type 2. On the right-hand side, the gross loss in revenue due to the additional utility offered to all developers equals their number n_2. But as a fraction of $\frac{dn_{21}}{du_2}$ new developers are of type 1 who each save the platform Δf in cost, the total cost saving from them is equal to $\Delta f \cdot \frac{dn_{21}}{du_2}$, which needs to be deducted from the gross loss in revenue.

Given some $R \in [0, \bar{R}]$, denote the u_1 that satisfies (5) and u_2 that satisfies (6) as the following

$$(u_1^*(R), u_2^*(R)) \equiv \arg\max_{(u_1,u_2)} \pi(u_1, u_2, R)$$

and denote the maximized profit as a function of R alone as follows

$$\Pi(R) \equiv \pi(u_1^*(R), u_2^*(R), R) \tag{7}$$

Assume that the profit function (4) is well-behaved such that $u_1^*(R)$ and $u_2^*(R)$ are differentiable for $R \in [0, \bar{R}]$. This implies that $\Pi(R)$ is also differentiable on this interval. We have the following useful property of $\Pi(R)$.

Lemma 1 *$\Pi(R)$ is U-shaped in R. That is, $\Pi''(R) > 0$ if $\Pi'(R) = 0$.*

(All omitted proofs are provided in the Appendix.) Lemma 1 implies that the maximized profit as a function of R has a single trough but no peak. Because the maximized profit **without** app review is equal to $\Pi(0)$, we can use $\Pi'(0)$ and Lemma 1 to determine the platform's incentive to use app review. For $R \in [0, \bar{R}]$, apply an envelope argument for (7) and we must have

$$\Pi'(R) = \frac{d}{dR}\pi(u_1^*(R), u_2^*(R), R) = \frac{\partial}{\partial R}\pi(u_1^*(R), u_2^*(R), R) \tag{8}$$

Therefore we can find $\Pi'(0)$ by evaluating $\frac{\partial \pi}{\partial R}$ at the optimal utilities without app review.

Proposition 1 (Optimal Pricing without App Review) *Without app review, the optimal prices for two sides, (p_1^0, p_2^0), are given by*

$$\begin{cases} p_1^0 - f_1 + \gamma n_2 = \frac{n_1}{n_1'} \\ p_2^0 - (f_{22} - \lambda \Delta f) + [\alpha - \gamma(1-\beta)]n_1 = \frac{n_2}{n_2'} \end{cases} \tag{9}$$

and the optimal utilities that the platform provides to two sides, (u_1^0, u_2^0), are given by

$$\begin{cases} \phi_1(u_1^0) = \phi_1'(u_1^0)(\alpha\phi_2(u_2^0) - u_1^0 - f_1) \\ \phi_2(u_2^0) = \phi_2'(u_2^0)(\alpha\phi_1(u_1^0) - u_2^0 - f_{22} + \lambda \Delta f) \end{cases} \tag{10}$$

Without review, the platform has no way to treat the two types of developers differently, and therefore the utility it provides to them is determined by the **average** cost that the platform incurs due to each developer, $(f_{22} - \lambda \Delta f)$.

The partial derivative of (4) with respect to R gives

$$\frac{\partial}{\partial R}\pi(u_1, u_2, R) = \frac{\partial n_{22}}{\partial R}(\alpha n_1 - u_2 - f_{22}) \tag{11}$$

where $\frac{\partial n_{22}}{\partial R} = -(1-\lambda)\phi_2'(u_2 - R)$. Intuitively, raising R has the exact opposite effect on type-2 developers as raising u_2 does. The negative impact on profitability is the product between the decrease in the number of type-2s and the economic profit they each create.

Without app review, $n_2 = \phi_2(u_2^0)$ and $\frac{\partial n_2}{\partial u_2} = \phi_2'(u_2^0)$. At optimality, by the first-order condition (6) we know, the platform will choose the optimal price (and utility) for developers to balance the economic profit from type-2 developers and the net loss it incurs per developer when raising u_2, that is,

$$(\alpha n_1 - u_2^0 - f_{22}) = \frac{n_2 - \Delta f \cdot \frac{dn_{21}}{du_2}}{\frac{\partial n_2}{\partial u_2}}$$

Substitute in (11) and by (8) we have

$$\Pi'(0) = (1-\lambda)[\lambda \Delta f \phi_2'(u_2^0) - \phi_2(u_2^0)].$$

Therefore by Lemma 1, we have the following result.

Proposition 2 (Incentive to Review Apps) *The platform has an incentive to introduce app review **if and only if***

$$\lambda \Delta f \geq \frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)} \tag{12}$$

where u_2^0 is the utility that each developer obtains from the platform without app review, as given by (10).

Intuition: Compared to a type-2 developer, each type-1 developer saves the platform Δf in cost, whilst creating the same network benefits for app users on the opposite side. Because the proportion of type-1 developers is λ, $\lambda \cdot \Delta f$ represents the total cost saved due to existing type-1 developers on the platform, compared to the "worst" situation where all developers are of type 2. Proposition 2 implies that app review is profitable if and only if the total cost saving from existing good developers is significant enough. In other words, app review works in favor of the platform if and only if the problem created by trouble-making developers is **mild**.

For a given level of $\Delta f > 0$, when the proportion of good (type-1) developers, λ, is sufficiently large, it is profitable for the platform to introduce app review to discourage participation by type-2 developers. The gain comes from the cost saving due to some type-2 developers leaving the platform as a result of app review. The loss results from the reduced attractiveness of the platform to app users due to fewer developers overall. Introducing app review only makes sense when the proportion of type-2 developers is sufficiently small such that the platform will still remain fairly attractive to users even after losing some of them.

Similarly, for a given proportion $\lambda > 0$ of good developers, app review is profitable when Δf is big enough, i.e. if type-2 developers create enough trouble for the platform to justify its interference.

App review is intended to reduce the presence of trouble-making developers on the platform. When the additional cost Δf they bring is too small, or when their proportion $(1 - \lambda)$ is too large, the problem caused by them is either negligible or too severe. In either case, the cost saved due to existing type-1 developers will be too small, which renders app review ineffective.

Positive threshold

The positive threshold $\frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)}$ in (12) may explain why Google and Apple adopted different app review strategies. Google did not use any app review measures until quite recently even though there were most likely trouble-making developers on it platform all along. It is possible that Google has a smaller λ, perhaps due to the variety of developers attracted to its more open operating system.[10]

A close look at Lemma 1 reveals that, if it is possible to induce a **negative** R for type-2 developers, perhaps by some kind of "reverse review" (as compared to the normal app review that results in a positive R), the platform should do so whenever $\lambda \Delta f < \frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)}$. Intuitively, this happens either when there are too many bad developers (λ small), or the additional cost Δf they cause is too small. Given that the platform has already found the optimal $u_2^*(R)$ for good type-1 developers, it would make sense for the platform to use the negative R to further increase the utility offered to bad developers, in order to attract more app users on the opposite side. Note this only makes sense because, besides $u_2^*(R)$, the platform has no other way to further increase the utility offered to the good developers. If it had other tools, it would certainly rather use them to raise good developers' utility first. We do not really observe such "reverse review" in real life, maybe because good developers already pass the app review 100% of the time, and it is impossible to provide the bad developers an even better chance at passing it.

Also note that when $\lambda \Delta f = \frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)}$, by (9) and (10) we have

$$p_2^0 = f_{22} - [\alpha - \gamma(1-\beta)]n_1$$

Because $[\alpha - \gamma(1-\beta)]n_1$ exactly represents the total network benefits that a developer creates for the platform, $f_{22} - [\alpha - \gamma(1-\beta)]n_1$ represents the real cost that the platform incurs for each type-2 developer, and a price p_2^0 exactly equal to this real cost coincides with the "socially optimal" price level of type-2 developers.

4 An Extension: Imperfect App Review

The previous section studies the case when the platform has perfect distinguishing ability when reviewing apps, whereas in reality it may well make mistakes. If this happens, app review can also reduce the utility of good developers. Suppose the app review process can correctly identify all trouble-making developers, but with probability $\theta \in [0,1]$, it will also mistake a good developer for a trouble maker. θ can also be interpreted as the degree of "imperfection" of app review.

Therefore under imperfect app review, when a type-2 developer obtains $u_2 - R$ from the platform, a type-1 developer expects to obtain a utility of $u_2 - \theta R$ (instead of u_2 in section

[10]Note that the developers on Google and those on Apple may follow different accumulation functions (i.e. different ϕ_2), and therefore their thresholds may also be different.

3). This changes how the number of type-1 developers under app review is determined, which now also depends on R and becomes

$$n_{21} = \lambda\phi_2(u_2 - \theta R).$$

Denote $\frac{\partial n_{21}}{\partial R} = -\theta\lambda\phi_2'(u_2 - \theta R)$. Nothing else changes in the previous model set-up.

The first order conditions for the optimal u_1 and u_2 are still given by (5) and (6), and the platform's incentive to use app review still depends on $\Pi'(0)$ derived from (7), except that because n_{21} also depends on R, we now have

$$\begin{aligned}\Pi'(R) &= \frac{\partial}{\partial R}\pi(u_1^*(R), u_2^*(R), R) \\ &= \frac{\partial n_{21}}{\partial R}\cdot(\alpha n_1 - u_2 - f_{21}) + \frac{\partial n_{22}}{\partial R}\cdot(\alpha n_1 - u_2 - f_{22}).\end{aligned}$$

By (5) and (6), and let $R = 0$, we have

$$\Pi'(0) = (1-\theta)\lambda(1-\lambda)\Delta f\cdot\phi_2'(u_2^0) - (1-\lambda+\theta\lambda)\cdot\phi_2(u_2^0),$$

which implies the following result.

Proposition 3 (Incentive to Review Apps with Imperfection) *When app review is imperfect, the platform has an incentive to introduce it* ***if and only if***

$$\frac{(1-\theta)\lambda(1-\lambda)}{1-\lambda+\theta\lambda}\Delta f \geq \frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)} \tag{13}$$

where θ is the probability that app review mistakes a type-1 developer for type 2, and u_2^0 is the utility that each developer obtains from the platform without app review, as given by (10).

Moreover, we can take the first order derivative of the left-hand side of (13) with respect to θ to find the impact of imperfection on the incentive to use app review. Because

$$\frac{\partial\frac{(1-\lambda)\lambda(1-\theta)}{1-\lambda+\theta\lambda}}{\partial\theta} = \frac{\lambda(\lambda-1)}{(1-\lambda+\theta\lambda)^2} < 0, \tag{14}$$

we have the following result.

Proposition 4 *The more imperfect the app review process is (i.e. the larger θ is), the less likely that the platform will have an incentive to use it.*

5 Welfare Analysis

5.1 General Framework

In order to clearly measure the welfare in the market, we now provide a specific way to define the accumulation processes of users and developers that will result in the demand functions from two market sides described previously.

Assuming that a user derives an idiosyncratic value t_1 if she participates on the platform, while her outside option of not participating gives her 0 utility. Therefore, given the utility that the platform provides to each user, u_1, she participates if and only if

$$u_1 + t_1 \geq 0.$$

Assuming the user population have heterogeneous idiosyncratic values that can be described by distribution F_1 on $\mathbb{R}$, the accumulation process of users can be written as

$$\phi_1(u_1) = \Pr[u_1 + t_1 \geq 0] = 1 - F_1(-u_1)$$

The function $1 - F_1(-u_1)$ is clearly strictly increasing in u_1 and therefore is consistent with our previous specification of ϕ_1. This particular accumulation process of users allows us to characterize their maximized total surplus as the following

$$v_1(u_1) \equiv \mathbb{E}_{t_1}[\max(u_1 + t_1, 0)] = \int_{-u_1}^{+\infty} (u_1 + t)dF_1(t)$$

which implies

$$v_1'(u_1) = \phi_1(u_1)$$

Similarly, assume that a developer derives an idiosyncratic value t_2 if she participates on the platform, while her outside option of not participating gives her 0 utility. Therefore, when the platform provides a utility of u_2 to each developer, she participates if and only if

$$u_2 + t_2 \geq 0.$$

Assuming the developer population have heterogeneous idiosyncratic values that can be described by distribution F_2 on $\mathbb{R}$, the accumulation process of developers can be written as

$$\phi_2(u_2) = \Pr[u_2 + t_2 \geq 0] = 1 - F_2(-u_2)$$

The function $1 - F_2(-u_2)$ is also consistent with our previous specification of ϕ_2.

Recall that a proportion of λ developers are good whereas the remainder are bad. When the platform uses app review, the utility it provides to bad developers is reduced by R compared to that provided to good developers, and therefore all developers' total maximized surplus is dependent on both u_2 and R, which we denote

$$\begin{aligned} v_2(u_2, R) &\equiv \lambda \mathbb{E}_{t_2}[\max(u_2 + t_2, 0)] + (1-\lambda)\mathbb{E}_{t_2}[\max(u_2 - R + t_2, 0)] \\ &= \lambda \int_{-u_2}^{+\infty} (u_2 + t) dF_2(t) + (1-\lambda) \int_{R-u_2}^{+\infty} (u_2 - R + t) dF_2(t) \end{aligned}$$

which implies

$$\begin{aligned} \frac{\partial}{\partial u_2} v_2(u_2, R) &= \lambda \phi_2(u_2) + (1-\lambda)\phi_2(u_2 - R); \\ \frac{\partial}{\partial R} v_2(u_2, R) &= -(1-\lambda)\phi_2(u_2 - R) \end{aligned}$$

The social welfare can therefore be represented by

$$w(u_1, u_2, R) = \pi(u_1, u_2, R) + v_1(u_1) + v_2(u_2, R). \tag{15}$$

Our goal is to compare the different levels of social welfare market when the platform does and does not use app review, given that it maximizes profit in either case, using the optimal $u_1^*(R)$ and $u_2^*(R)$ as characterized previously. That is, if we denote the social welfare when the platform has maximized profit given R as

$$\begin{aligned} W(R) &\equiv \Pi(R) + v_1(u_1^*(R)) + v_2(u_2^*(R), R) \\ &= \pi(u_1^*(R), u_2^*(R), R) + v_1(u_1^*(R)) + v_2(u_2^*(R), R), \end{aligned}$$

we will now compare $W(R)$ and $W(0)$ to determine whether app review will result in a welfare loss in the market.

The derivative of $W(R)$ is

$$\begin{aligned}
W'(R) &= \Pi'(R) + v_1'(u_1^*(R)) \cdot u_1^{*\prime}(R) \\
&\quad + \frac{\partial}{\partial u_2} v_2(u_2^*(R), R) \cdot u_2^{*\prime}(R) + \frac{\partial}{\partial R} v_2(u_2^*(R), R) \\
&= \Pi'(R) + \phi_1(u_1^*(R)) \cdot u_1^{*\prime}(R) \\
&\quad + [\lambda\phi_2(u_2^*(R)) + (1-\lambda)\phi_2(u_2^*(R) - R)] \cdot u_2^{*\prime}(R) \\
&\quad - (1-\lambda)\phi_2(u_2^*(R) - R) \\
&= \Pi'(R) + \phi_1(u_1^*(R)) \cdot u_1^{*\prime}(R) + \lambda\phi_2(u_2^*(R)) \cdot u_2^{*\prime}(R) \\
&\quad + (1-\lambda)\phi_2(u_2^*(R) - R) \cdot (u_2^{*\prime}(R) - 1) \\
&= \Pi'(R) + n_1 \cdot u_1^{*\prime}(R) + n_2 \cdot u_2^{*\prime}(R) - n_{22}
\end{aligned}$$

An obvious point of interest is the sign of

$$W'(0) = \Pi'(0) + n_1 \cdot u_1^{*\prime}(0) + n_2 \cdot u_2^{*\prime}(0) - n_{22} \tag{16}$$

We will need to know $u_1^{*\prime}(R)$ and $u_2^{*\prime}(R)$, or at least some of their properties to tell the sign of $W'(R)$ or $W'(0)$. As this is impossible without specifying the distributions of users and developers (e.g. the F_1 and F_2 defined previously), we will use simulation.

5.2 Welfare Analysis under Uniform Distribution

Suppose the distributions of users and developers are uniform distributions. In particular, users' utility is distributed uniformly from a lower bound of $a < 0$ to an upper bound of b. App developers' utility is distributed uniformly from a lower bound of $d < 0$ to an upper bound of c. Define $B \equiv \frac{b}{b-a}$, and $A \equiv \frac{1}{b-a}(> 0)$. Define $C \equiv \frac{c}{c-d}$, and $D \equiv \frac{1}{c-d}$. Then we can find the mass of users and app developers joining the platform offering u_1 and u_2 as:

$$\begin{aligned}
\phi_1(u_1) &= B + Au_1 \\
\phi_2(u_2) &= C + Du_2
\end{aligned}$$

So we have

$$\begin{aligned} n_1 &= B + Au_1 \\ n_{21} &= \lambda(C + Du_2) \\ n_{22} &= (1-\lambda)[C + D(u_2 - R)] \\ n_2 &= C + Du_2 - (1-\lambda)RD \end{aligned}$$

And we also have

$$n_1^1 = A, n_1^{11} = 0, n_{21}^2 = \lambda D, n_{22}^2 = (1-\lambda)D, n_2^2 = D.$$

where the superscript j represents taking the partial derivative with respect to u_j.

The first-order conditions for the profit-maximizing u_1^* and u_2^* are given by

$$\begin{aligned} (\alpha n_2 - u_1 - f_1) \cdot \frac{dn_1}{du_1} &= n_1 \\ (\alpha n_1 - u_2 - f_{22}) \cdot \frac{dn_2}{du_2} &= n_2 - \Delta f \cdot \frac{dn_{21}}{du_2} \end{aligned}$$

Substituting the formulas under uniform distribution, we have

$$\begin{aligned} FOC(u_1) &: -2u_1 + \alpha Du_2 = \alpha(1-\lambda)RD + f_1 + \frac{B}{A} - \alpha C \\ FOC(u_2) &: \alpha ADu_1 - 2Du_2 = C + Df_{22} - (1-\lambda)RD - \Delta f \cdot \lambda D - \alpha BD \end{aligned}$$

And the total differentiation of these equations with respect to R gives

$$\begin{aligned} dR \text{ of } FOC(u_1) &: 2u_1^{*\prime}(R) - \alpha D \cdot u_2^{*\prime}(R) = -\alpha D(1-\lambda) \\ dR \text{ of } FOC(u_2) &: \alpha A \cdot u_1^{*\prime}(R) - 2u_2^{*\prime}(R) = -(1-\lambda) \end{aligned}$$

where the unique solution is given by

$$\begin{aligned} u_1^{*\prime}(R) &= \frac{\alpha D(1-\lambda)}{\alpha^2 AD - 4} \\ u_2^{*\prime}(R) &= \frac{(\alpha^2 AD - 2)(1-\lambda)}{\alpha^2 AD - 4} \end{aligned}$$

5.2.1 The special uniform distribution with $A = D$ and $B = C$

In the special uniform distribution with $A = D$ and $B = C$, i.e. if the distribution of two sides are the same, we have

$$\begin{aligned} u_1^{*\prime}(R) &= \frac{\alpha A(1-\lambda)}{\alpha^2 A^2 - 4} \\ u_2^{*\prime}(R) &= \frac{(\alpha^2 A^2 - 2)(1-\lambda)}{\alpha^2 A^2 - 4} \end{aligned}$$

where we have the following property:

Claim i) If $\alpha A > 2$, we have $u_1^{*\prime}(R) > 0$ and $u_2^{*\prime}(R) > 0$;

ii) if $\sqrt{2} < \alpha A < 2$, we have $u_1^{*\prime}(R) < 0$ and $u_2^{*\prime}(R) < 0$; and

iii) if $\alpha A \leq \sqrt{2}$, we have $u_1^{*\prime}(R) < 0$ and $u_2^{*\prime}(R) \geq 0$.

In this special case, we can also solve for the optimal u_1^* and u_2^* in $FOC(u_1)$ and $FOC(u_2)$ shown previously, we are given by:

$$\begin{aligned} u_1^* &= \frac{1}{A^3\alpha^2 - 4A}\left(2B - AB\alpha + 2Af_1 + A^2R\alpha - A^2R\alpha\lambda + A^2\alpha f_{22} - A^2\alpha\lambda\Delta f - A^2B\alpha^2\right), \\ u_2^* &= \frac{1}{A^3\alpha^2 - 4A}\left(2B - 2AR - AB\alpha + 2AR\lambda + 2Af_{22} - 2A\lambda\Delta f + A^2\alpha f_1 - A^2B\alpha^2 + A^3R\alpha^2 - A^3R\alpha^2\lambda\right). \end{aligned}$$

The sign of $W'(0)$

Now we can calculate everything in formula (16) in order to possibly sign $W'(0)$. We are particularly interested in situations where $\Pi'(0) > 0$ but $W'(0) < 0$, such that it would be in the interest of the platform alone, not the society, to introduce app review. If this is the case, the difference between $W'(0)$ and $\Pi'(0)$ would also be negative.

A few simulated examples are provided next.

5.2.2 Simulation Results

We ran 503,119 times of simulation of the special case of uniform distribution discussed in the previous section. And the results show that the platform's incentive to review apps, $\Pi'(0)$, and the interest of social welfare, $W'(0)$, may be consistent in some cases but divergent in others. Actually all four kinds of result exist, which are $\Pi'(0) > 0$ and $W'(0) > 0$; $\Pi'(0) > 0$ and $W'(0) < 0$; $\Pi'(0) < 0$ and $W'(0) > 0$; $\Pi'(0) < 0$ and $W'(0) < 0$. Though simulated, this is a very important result as it shows that anything is possible even under rather simple uniform distributions. A regulator should therefore be very careful when it contemplates whether to interfere with a platform's app review process. Only in the situation when $\Pi'(0) > 0$ and $W'(0) < 0$ can the intervention be justified.

The four sets of parameter values that yield the four different situations are provided as follows.

Example 1. Misalignment: $\Pi'(0) > 0$ **and** $W'(0) < 0$ $\{A, B, \alpha, f_1, f_{22}, \Delta f, \lambda\} = \{0.5, 0.8, 2.4, 1, 2, 1.5, 0.1\}$

$\Pi'(0) = 0.0288281$

$W'(0) = -0.075542$

$\Pi(0) = 0.0473633.$

Example 2. Alignment: $\Pi'(0) > 0$ **and** $W'(0) > 0$ $\{A, B, \alpha, f_1, f_{22}, \Delta f, \lambda\} = \{0.5, 0.8, 1, 1, 2, 1.5, 0.1\}$

$\Pi'(0) = 0.0915$

$W'(0) = 0.0871$

$\Pi(0) = 0.0463333.$

Example 3. Misalignment: $\Pi'(0) < 0$ **and** $W'(0) < 0$ $\{A, B, \alpha, f_1, f_{22}, \Delta f, \lambda\} = \{0.5, 0.8, 3, 1, 2, 1.5, 0.1\}$

$\Pi'(0) = -0.0353571$

$W'(0) = -0.334745$

$\Pi(0) = 0.0564286.$

Example 4. Misalignment: $\Pi'(0) < 0$ **and** $W'(0) > 0$ $\{A, B, \alpha, f_1, f_{22}, \Delta f, \lambda\} = \{1, 0.9, 1.1, 2, 0.2, 0.15, 0.1\}$

$\Pi'(0) = -0.0574677$

$W'(0) = 0.0714316$

$\Pi(0) = 0.306837.$

Furthermore, for the situation where $\Pi'(0) > 0$ and $W'(0) < 0$, some simulation results also satisfy $\Pi(0) > 0$, which means the platform is profitable under this situation and it will conduct the review process, and the social welfare will decrease at the same time. There is only one situation that $\Pi(0) > 0$, $\Pi'(0) > 0$ and $W'(0) < 0$ doesn't exist at the same time, that is when $B = 0$, which means that both developers and users have a uniform distribution of negative utility derived from joining the platform. And in this situation, $W'(0) - \Pi'(0)$ is always bigger than 0, so introducing review process will always benefit social welfare first.

These results can be summarized as follows. More details are provided in the Appendix.

Proposition 5 (Welfare) *Simulation results show that the platform's incentive to use app review may be aligned or misaligned with the interest of social welfare, depending on the distributions of app users and developers and other parameters, including the strength of network effects α.*

6 Conclusion

In this paper, we provide a rather straight-forward explanation for app review - it is intended to reduce the presence of trouble-making developers on the platform. When app users have a preference for app variety despite the trouble some developers may impose on the platform, the platform faces a trade-off between cost saving from screening trouble makers and a loss of the attractiveness to the users. We have shown that the platform's incentive to use app review depends on the additional cost it incurs from each "trouble maker", and on their proportion among all developers. Given the additional cost incurred, app review benefits the platform if and only if the proportion of trouble makers is sufficiently small, such that after screening them the platform remains fairly attractive to users. Given their proportion, however, app review helps if and only if the "trouble makers" create enough "trouble", such that the total cost saved from app review is not negligible. In other cases, the platform should not use app review, or should even use "reverse review" if feasible. This may explain the different app review strategies adopted by Apple and Google.

Our welfare analysis through simulations cautions regulators that the platform's incentive to use app review may be aligned or misaligned with the interest of social welfare, depending on the distributions of app users and developers and other parameters.

Our model only considers a monopoly case. It can be applied to the smartphone market before the introduction of Google's Android and Google Play, then known as the Android Marketplace, in late 2008. At that time, Apple was the dominant player in the smartphone market. When smartphones were first introduced into the market, few people were able to develop mobile apps. Those app developers at that time were probably IT professionals with exceptional programming skills and computer literacy. Their preferences, which can be measured by their time costs of developing apps, were less diverse since they usually had very high time costs. Thus it was profitable for Apple to introduce a review process at that time.

The model may also explain why Google provides an open-source platform but Apple does not. The variety of smartphones using Android (where apps are distributed via Google Play) is much higher than that using Apple's iOS (where apps are distributed via App Store). The higher variety enables Google to attract app developers with a wider range of preferences, whereas Apple may only attract a small range of app developers. For example, less capable app developers can develop simple apps for lower-priced and lower-quality Android smartphones, whereas more capable app developers can develop complicated apps for higher-priced and higher-quality Android smartphones. Apple's profitable introduction of a review process therefore may not apply to Google. However, this analysis does not take into account the competition between the two platforms. Further research can analyze how the competition between the app distribution platforms will affect the platforms' incentive to introduce a review process, as well as their pricing strategies.

References

[1] **Armstrong, Mark**. 2006. "Competition in Two-sided Markets." *RAND Journal of Economics*, 37(3): 668-691.

[2] **Hagiu, Andrei**. 2009. "Quantity vs. Quality and Exclusion by Two-sided Platforms." *Harvard Business School Working Paper*, 09-094.

[3] **Kouris, Iana**. 2011. "Unified Two-sided Market Model." *Working paper*.

[4] **Rochet, Jean-Charles, and Jean Tirole**. 2003. "Platform Competition in Two-sided Markets." *Journal of the European Economic Association*, 1(4): 990-1029.

[5] **Rochet, Jean-Charles, and Jean Tirole**. 2006. "Two-sided Markets: A Progress Report." *Rand Journal of Economics*, 37(3): 645-667.

Appendices

Appendix A Proofs and Derivations

Proof of Lemma 1

The partial derivative of (4) with respect to R gives

$$\frac{\partial}{\partial R}\pi(u_1, u_2, R) = \frac{dn_{22}}{dR}(\alpha n_1 - u_2 - f_{21} - \Delta f)$$

where

$$\frac{dn_{22}}{dR} = -(1-\lambda)\phi_2'(u_2 - R)$$

At the optimal utilities $(u_1^*(R), u_2^*(R))$, by (6) we have

$$(\alpha n_1 - u_2^*(R) - f_{22}) = \frac{n_2 - \Delta f \cdot \frac{dn_{21}}{du_2}}{\frac{\partial n_2}{\partial u_2}}$$

which implies

$$\begin{aligned} \Pi'(R) &= \frac{\partial}{\partial R}\pi(u_1^*(R), u_2^*(R), R) \qquad (17) \\ &= \frac{dn_{22}}{dR} \cdot \frac{n_2 - \Delta f \cdot \frac{dn_{21}}{du_2}}{\frac{\partial n_2}{\partial u_2}} \\ &= \frac{-\frac{dn_{22}}{dR}}{\frac{dn_2}{du_2}}(\Delta f \cdot \frac{dn_{21}}{du_2} - n_2) \end{aligned}$$

Therefore

$$\Delta f \cdot \frac{dn_{21}}{du_2} - n_2 = (\frac{\frac{\partial n_2}{\partial u_2}}{-\frac{dn_{22}}{dR}}) \cdot \Pi'(R)$$

and

$$\begin{aligned} \Pi''(R) &= \frac{\partial^2}{\partial R^2}\pi(u_1^*(R), u_2^*(R), R) \qquad (18) \\ &= \frac{\frac{dn_{21}}{du_2}}{(\frac{\partial n_2}{\partial u_2})^2} \cdot (-\frac{d^2 n_{22}}{dR^2}) \cdot (\Delta f \cdot \frac{dn_{21}}{du_2} - n_2) + \frac{(\frac{dn_{22}}{dR})^2}{\frac{\partial n_2}{\partial u_2}} \\ &= \frac{\frac{dn_{21}}{du_2}}{(\frac{\partial n_2}{\partial u_2})^2} \cdot (-\frac{d^2 n_{22}}{dR^2}) \cdot (\frac{\frac{\partial n_2}{\partial u_2}}{-\frac{dn_{22}}{dR}}) \cdot \Pi'(R) + \frac{(\frac{dn_{22}}{dR})^2}{\frac{\partial n_2}{\partial u_2}} \\ &= \frac{\frac{dn_{21}}{du_2}}{\frac{\partial n_2}{\partial u_2}} \cdot (-\frac{d^2 n_{22}}{dR^2}) \cdot (\frac{1}{-\frac{dn_{22}}{dR}}) \cdot \Pi'(R) + \frac{(\frac{dn_{22}}{dR})^2}{\frac{\partial n_2}{\partial u_2}} \end{aligned}$$

where

$$\frac{d^2 n_{22}}{dR^2} = (1-\lambda)\phi_2''(u_2 - R)$$

Because $\frac{\partial n_2}{\partial u_2} > 0$, and $\frac{dn_{22}}{dR} \neq 0$, we know $\Pi''(R) > 0$ whenever $\Pi'(R) = 0$. ■

Proof of Proposition 1

(9) and (10) are found by letting $R = 0$ in (5) and (6). ■

Proof of Proposition 2

Let $R = 0$ in (17) and we have

$$\Pi'(0) = (1 - \lambda)[\lambda \Delta f \phi_2'(u_2^0) - \phi_2(u_2^0)]$$

As $\phi_2'(u_2^0) > 0$, we know $\frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)} > 0$. Therefore, whenever $\lambda \neq 1$ (i.e. there exist some type-2 developers), we have

$$\Pi'(0) > 0 \text{ if and only if } \lambda \Delta f > \frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)}.$$

Now consider the case when $\lambda \Delta f = \frac{\phi_2(u_2^0)}{\phi_2'(u_2^0)}$, that is, when $\Pi'(0) = 0$. By Lemma 1 we know $\Pi''(0) > 0$, therefore $R = 0$ is a minimizer of $\Pi(R)$. Because introducing app review raises R from 0, it will make $\Pi'(R) > 0$ and is hence profitable. ■

Appendix B Background Information on App Review

Apple has built a "walled garden" for its app store by pre-screening all apps and their updates before putting them onto its shelf.[11] In this paper we will discuss 4 motivations behind Apple's app review policy, including property protection, enhancement of brand loyalty, preservation of market power and upholding of the goodwill of the its own business. We will further highlight 2 catalysts that help to reinforce Apple's determination to install its app review policy. These are pickier consumers and lower training cost for staff. We will go through relevant examples and data to support all the arguments made.

B.1 Introduction

A "walled garden," or a closed platform, is a jargon used widely in the information technology industry, describing a software platform introduced by a service provider or carrier to control apps in respect of their content, media and applications, and to restrict easy access to ineligible content or applications.[12] Roughly speaking, "walled garden" can be a phrase to describe some sort of isolated technology or platform, which is the opposite of an open source platform. Apple's iOS is a typical example of a walled garden. Since Apple launched its exclusive Apple App Store in 2008, it has required app developers to obtain its prior approval before putting their apps onto its shelf . All apps have to be pre-screened and Apple has full authority to reject or remove apps from its app store. Apple contends that its ex ante app review policy will ensure that the approved apps are reliable, free of offensive content and at a level of high performance in order to guarantee consumer satisfaction and to satisfy social responsibility.[13] However, profit maximization, instead of consumers' utility maximization, has long been the core postulate of economics, making people query whether the reasons as advanced by Apple are really the true motives of Apple. Nevertheless, we cannot deny that maximizing the profit of a firm may largely hinge on maximizing consumers' utility. With this in mind, we seek to analyze in this paper the true motivations behind Apple's app review policy.

[11] This part is the modified version of Douglas Cheng's term paper entitled "The Walled Garden: Motivations behind Apple's app review policy," under the supervision of one of the authors.

[12] This definition comes from Wikipedia.

[13] "We review all apps submitted to the App Store and Mac App Store to ensure they are reliable, perform as expected, and are free of offensive material. As you plan and develop your app, make sure to use these guidelines and resources," extracted from https://developer.apple.com/app-store/review/

B.2 Description of the app review

B.2.1 Background

Apple launched its Apple App Store in July, 2008, through an update to iTunes. As of October, 2014, Apple App Store accumulated more than 85 billion downloads. Around the same time, there were around 1.2 million apps available in Apple App Store, making it the second largest app store in the world, behind Google Play which had 0.1 million more apps.[14]

B.2.2 App review guidelines

Apple has officially promulgated a set of strict guidelines, under the title of App Store Review Guidelines, for its app review policy, under which both content and design of the apps will be examined.[15] App developers who fail to comply with any of those guidelines will have their apps rejected. I will not list the guidelines and describe them one by one as it is not the interest of this paper. Instead, I summarize and generalize Apple's focus during its ex ante app review process on the basis of the App Store Review Guidelines. The next section notes some observations regarding the emphasis placed by Apple on upholding the review policy.

Basic guidelines can be largely generalized with reference to their functionality, content appropriateness and legality. Apps with bugs, having non-functional hyper-links, or crash will be rejected or removed from Apple App Store. Apps with inappropriate content (insult to humanities, racism, pornography or personal attack) will be rejected. Also, the Apple review team will reject apps that infringe personal privacy, replicate other apps or fail to comply with local laws. It seems Apple is cautious of any sensitive or legal issues that will lead itself into trouble.

Apple puts much emphasis on user interface design as it wants to make sure that all the apps are designed to suit its product but not the other way round.[16] App developers need to take care of even minor details when they create the user interface of their apps. Apple's Design and Trademark Guidelines stipulate requirements for layout, interactivity and integrality with iOS. For layout, app developers need to make sure that content views, temporary views, bars and controls of an app are placed at the right place with the right size. For instance, the navigation bar and the tab bar have to be placed on the top and bottom of the screen respectively, while the toolbar button has been placed at the

[14] http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/
[15] The full list is at: https://developer.apple.com/app-store/review/
[16] Supplementary guidelines for user interface design is at: https://developer.apple.com/design/tips/

bottom right comer. Interactivity means the users of the app can see the primary content without scrolling or zooming, and can comfortably make touch gestures. The text needs to be at least 11 points and the hit targets need to be at least 44 points x 44 points. Integrality means that the app should incorporate as many general system-supported technologies as it can, like VoiceOver and multitasking.

Also, the app should be able to integrate with some specific system-supported technologies or functions where necessary, for example, Game Center, Passbook, iCloud, etc. By and large, it is preferable, if not necessary, for the app to integrate with iOS as much as possible.

B.2.3 Common app rejections

It seems that Apple has a transparent and self-explanatory app review standard, with a long list of guidelines clearly written in black and white. However, the mechanism may not be as transparent as one imagines. In late-2014, Apple announced the top 10 common reasons that apps were rejected.[17]

The most common reason, taking up 42% of the apps rejected app from App Store was "Other reasons." However, Apple did not further explain what exactly those other reasons were. The second most common reason, taking up 14%, was equally obscure, for apps were rejected because of "more information needed."

Some other common reasons of app rejections included non-functional or broken apps (8%), poorly designed user interface (6%) and irrelevant names, descriptions or screenshots (5%).

B.2.4 Process of app review

As Apple has not officially disclosed how the app review mechanism works, so we can only imagine the process with assistance of a disc10sure from an Apple's employee.[18] Mike Lee, a former senior engineer at Apple, suggested that Apple's app review team was understaffed. He said, "like every other part of Apple, they can't get enough really good people." Lee also claimed "the app reviewers are essentially tasked with managing a slush pile of apps" and "there is so much garbage." With a small app review team as claimed, we can expect that each app reviewer has to deal with a multitude of apps each day. When app reviewers reject apps in haste, it is not surprising that hidden gems may be dumped accidentally as waste. Also, it takes around 8 days for Apple to review an

[17] The whole list is on https://developer.apple.com/app-store/review/rejections/

[18] More information is at: http://www.businessinsider.com/heres-why-it-really-sucks-to-be-an-app-reviewer-

app.[19]

B.3 Motives and examples

This section proposes Apple's motivations behind its walled garden and lists the relevant examples. There are 4 motivations behind Apple's policy: property protection, enhancement of brand loyalty, preservation of market power and maintenance of the goodwill of the company.

B.3.1 Property protection

Apple is not a monopoly in the IT world, but it is the monopoly in the world of iOS. People who buy Apple products are actually buying a bundle of Apple hardware and iOS. Safari and iTunes are pre-installed, while Apple devices and iOS are exclusive to each other. Apple is exceptionally mindful of protecting its technology, and app review is a way to combat property rights invasion. This idea is similar to other efforts and schemes to protect the "secret formula" of some franchises, like those adopted by Coca-Cola and Panadol.

Apple's restrictions on app-discovery apps is a clear example to support the above argument. An app discovery app is an application programming interface (API) that serves as a search engine for apps. It facilitates access to app database in different platforms and then gathers, organizes and analyses the apps information for users. Examples of app-discovery apps include Mevvy API, App Shopper, Appidemia and App of the Day. Such apps have not had a good fate in iOS as opposed to Android. According to Melanie Haselmayr, the CEO of Mevvy Limited, none of these apps were available before the late-2013.[20] All app-discovery apps were rejected, as Apple was too cautious to allow a third-party app to gain access to its database. In 2014, app-discovery apps started to appear in the iOS market, because Apple started to provide those apps with its own set of app data. This means those app-discovery apps would no longer need to write an API to extract information from Apple's app database, and instead they would have limited information provided by Apple. In comparison, there are a lot more app-discovery apps on Google Play and these apps can have access to Android's database.

[19] Information extracted from http://appreviewtimes.com

[20] Information from Melanie Haselmayr comes from an email interview that 1 conducted in February, 2015.

B.3.2 Enhancement of brand loyalty

The fully integrated iOS ecosystem is a phenomenal feature of Apple Inc., i.e., all iProducts are branded in harmony together with Apple's exclusive operating system. All Apple products look similar, so are their functionality. For instance, iPad is an enlarged iPod where iPhone is an iPod with telecommunication functions. Apple wants to ensure that an app is functional on one kind of device and is equally functional on another product of the company. This explains a reason for using the app review policy to eliminate apps that fail to observe the design user interface standard or are not compatible with general system-supported technology, as in the case of Siri. Thus, there will be easier and tighter inter-operability with the OS. As a result, Apple will increase consumers' switching cost as users are too used to Apple products as well as iOS. Apple is achieving a 76% customer retention rate which is the highest in the mobile industry.[21] On the contrary, if the app design is too random, or too gadget-specific, consumers will be more prone to switching to another brand or another platform operated by a rival company.

B.3.3 Preservation of market power

The app review policy can be anti-competitive as it can be a gimmick to drive away potential competitors. In particular, there may be a business field into which Apple is desirous of making inroads. It may therefore take ear1y steps to reject or remove competitors' apps in that field. Bitcoin and Google Voice are two landmark examples as explained in the following paragraphs.

In early-2014, Apple started removing Bitcoin apps without any explanation. This was a controversial move.[22] Bitcoin is a virtual currency that people can use to pay for goods and services and it is becoming popular worldwide. Many suspected that Apple withdrew Bitcoin apps from App Store because Apple was developing its own way of virtual payment, i.e. the Apple Pay.[23] Bitcoin could be a big threat Apple Pay. But later in the same year, Bitcoin apps were available on iOS again. It was because after Apple removed all Bitcoin apps, Google became the only platform allowing Bitcoin apps. Apple did not want to want to lose out to Google, and had therefore relented in the case of Bitcoin. It is all about money.

Another example is rejection of Google Voice from App Store in 2009. Apple even pub-

[21] Information extracted from: http://news.xerox.com/news/WDS-A-Xerox-Company-Mobile-Loyalty-Audit-Results

[22] For more information see http://www.coindesk.com/apple-removes-blockchain-bitcoin-wallet-from-app-store

[23] For more information see: http://www.pcworld.com/article/2095060/apple-removes-blockchain-last-bitcoin-wallet-app-from-mobile-store.html

licly gave an explanation.[24] Following up on Apple's explanation, Google Voice sought to replace a lot of core functions of iPhone, but Apple did not appreciate that. Apple is least keen to enable its users to experience another user interface or core functions other than iOS. Now that Google Voice is available on iOS because the design has been modified to suit the iOS system. Therefore, app review is a powerful and potentially anti-competitive practice.

B.3.4 Maintenance of the goodwill of the business

Apple puts more emphasis on end users experience than Google because hardware sales are its mainstream revenue.[25] Hardware sales hinge much on the goodwill of the business which can be improved through better customer experience. If there are some sensitive apps that maybe regarded as offensive to religion or humanities, the offended individuals may boycott Apple's products, leading to a big loss of sales revenue for the company. That is why Apple is very careful when it comes to eliminating troublemakers from App Store in order to sustain and not to tarnish its reputation. Generally, apps with obscene, racist, gambling or violence content will not be allowed.

There are many more relevant examples regarding sensitive content being rejected. Here, I am going to briefly discuss two.

In 2011, "The Unpleasant Horse" was rejected from iOS because the app was seen to be promoting violence against animals.[26] This app is a game that when a player reaches the stage of "Game Over", the horse will be grinded into mince meat. Apple did not allow such an app to exist in its app store to avoid complaints from animal lovers. Another example is "Baby Shaker," a game where players shake the crying animated baby to death.[27] Apple initially approved this app but it then caused a lot of controversy as the game was seen to be promoting infanticides. Many parents expressed their discontent by suggesting that they would boycott Apple products. Apple then quickly removed the app with an apology. This illustrates that problematic apps can harm a company's image.

[24] The whole statement is at: https://www.apple.com/hotnews/apple-answers-fcc-questions/

[25] In Q4 of 2014, around 86% of Apple's revenue came from hardware sales. Information extracted from http://www.macstories.net/news/apple-q4-2014-results-42-1-billion-revenue-39-3-million-iphones-12-3-million-ipads-sold/

[26] More information on http://www.digitaltrends.com/mobile/apple-rejects-popcaps-unpleasant-horse-due-to

[27] More information at:http://www.theguardian.com/technology/2009/apr/23/apple-iphone-baby-shaker

Table 1: Comparing iOS and Android apps

	iOS	Android
Apps count	1,384,043	976,444
Mean user rating	3.6337	3.9918
Mean rating count	112.40	1,591.34
Percentage of free apps	71%	85%
Number of common apps	221,321	
Mean price for common apps	\$0.6886	\$0.34002

B.4 Data and statistics

This section examines Apple's app review policy against some data. App data from iOS and Android were given by Mevvy Ltd on 1st March, 2015. Below are some statistics:

We have almost all app data pertaining to iOS, while only a subset of data for Android. As of March 2015, Android had around 0.1 million more apps than iOS. Sti11, the close number of apps between the two platforms is surprising, probably due to several reason. Firstly, Android apps are easier to develop in programming sense, and app developers mostly agree with this. Secondly, trial versions, beta versions, light versions are allowed on Android but not on iOS. Thirdly, the world market share of Apple products is only one-third of Android devices. Fourth, and perhaps most importantly, Android has no pre-screening of apps. Therefore, for iOS to have nearly as many apps available as Android is quite unexpected, and such phenomenon is worthy of another specific study to examine the causes and to ascertain answers.

Mean user rating is a 5-star scale rating which measures the average consumer satisfaction. From our sample, the mean user rating for iOS was 0.36 lower than that of Android. This may imply that the iOS users were more difficult to serve, or in other words they were pickier. Mean rating count shows how many users rate the app on average, acting as a proxy for the number of downloads per app. Since iOS did not publicly disclose to or provide Mevvy Limited with the data for downloads per app, we can only use a proxy. It is not surprising to see Android had much higher mean rating per app as it had a higher market share worldwide. Furthermore, there were 221,321 common apps across the 2 platforms, while the mean price for those common apps on iOS was almost double that of Android.

Figure **??** graphs that plot price range against average rating count. For the sake of easier data handling, 1 have grouped the apps into 6 price ranges: $P = \$0$; $\$0 < P \leq \$5$; $\$5 < P \leq \$10$; $\$10 < P \leq \$15$; $\$15 < P \leq \$20$, and $P > \$20$.

From the above graphs, we can see there is a steeper slope for the "demand curve" for iOS. This may imply that Apple users were more willing to pay for apps than Android users. It should be noted that both x- and y-axis are only proxies for P and Q of a normal demand curve. Thus this implication is not definite.

From all above graphs and numbers, we can derive one implication. Apple users and Android users do not seem to share common characteristics and consumer behavior. The former group comprises those more willing to pay for apps, and more demanding. This means that problematic apps will trigger more problems for iOS than Android. This may be a justification, or some may say a driving force, for Apple to implement an ex ante app review policy.

B.5 Counter-example: Google's Android

Sometimes when we think ”Why A?”, it may be equally if not more inspiring to think ”Why not B?”. In this context, "Why not Google?" The previous sections have explored a couple of motivations behind Apple's app review policy, where such a policy can protect Apple's business. One may ask why Google does not follow suit and practice the same pre-screening policy.

I propose 2 main reasons: Google's main source of revenue is not from hardware sales and the fixed cost incurred to train and operate a review team is too high.

To further elaborate on the main reasons, let us briefly examine the main differences between Apple's and Google's business models. Apple has a couple of roles in its business model, where it started as a computing hardware developer. Subsequently, it opened the iPhone line with a unique operating system, thus becoming a mobile device manufacturer as well as a computing software developer. Also, Apple is an e-commerce engine and retailer. The business roles of Apple are well vertically integrated and more versatile than Google which is mainly an internet service provider.

Unlike Apple, Google's main source of revenue comes from advertising (around 40%), where its hardware sales only take up a slight portion of its earnings (less than 10%).[28] Android is not exclusive to Google's devices as there are more gadget producers like Samsung, Sony, LG that also support the mobile operating system. Owning only Motorola, Google is far from a dominant competitor in the market of mobile devices. The opportunity cost of selling some potentially troublesome apps will be lower for Google than Apple. Therefore, Google is less concerned about losing sales owing to some sensitive apps, while the company is more inclined to allowing more apps on Google Play to generate advertising revenue through Google Mobile Ads.

[28] As of Dec 31, 2013, hardware sales only take up 7% of Google's revenue for a quarter. More information at: https://investor.google.com/financial/tables.html

Secondly, 1 am not saying that Google would never remove problematic apps from its app store. Indeed, Google tends to do that as well. If some users discover and report that some apps are scam, broken, or likely to cause legal trouble, Google will seek to remove them. For example, in 2014, there was a fake anti-virus app called Virus Shield which once topped the Google Play's sales chart.[29] Google subsequently removed the app and even provided refunds to its customers. This illustrates that Google does care about the goodwill of its brand name but it is also undeniable that Google will incur huge fixed cost for training an app review team for the sake of pre-screening apps. As mentioned in the previous paragraph, there are a wide variety of mobile devices that support Android while the majority of them are not owned by Google. An app works well on one gadget is no guarantee that it works or looks equally well on another gadget. This means if Google seeks to practice an ex ante app review policy, there will be huge fixed training cost incurred. So it is better for Google to remove problematic apps as per customers' requests. In comparison, App Store is exclusive to Apple products, where Apple knows its devices well, enabling it to spend much lower fixed cost than Google to train a team of app reviewers. Moreover, as mentioned above, Apple has the role of a retailer in its business model. As a retailer, Apple has a front line that constantly keeps in touch with customers, enabling mutual communication and collection of customer opinions. As a result, Apple has a better chance of making the right decision when it comes to approving apps for inclusion in the Apple App Store.

B.6 Conclusion

To sum up, we have gone through 4 motivations, i.e., property protection, enhancement of brand loyalty, preservation of market power and maintenance of the goodwill of the business, and a further 2 driving forces, i.e., pickier consumers and lower training cost for staff in implementing Apple's app review policy. There are fundamental differences between Apple's and Google's business models that make only the former practices a pre-screening policy.

Steve Job once said, "We do believe we have a moral responsibility to keep porn off the iPhone. Folks who want porn can buy an Android phone." In a similar line, the Apple official website says "People who want to criticize a religion and culture can choose write a book." Apple may profess to want to fulfill social responsibility, but the prime incentive for a profit-making firm has always been profit-making. If it is more profitable and less risky to carry out an app review policy, a company will have incentives to implement one.

[29] More information at: https://nakedsecurity.sophos.com/2014/04/22/google-refunds-android-users-who-bought-fake-virus-shield-app/

B.7 References

[1] Arthur, C. (2009, April23). " 'Shaker' game pulled from Apple's iPhone App Store." Retrieved from http://www.theguardian.com/technology/2009/apr/23/apple-iphone-baby-shake

[2] Fiegerman, S. (2012, July 3). "Here's Why It Really Sucks To Be An App Reviewer For Apple." Retrieved from http://www.businessinsider.com/heres-why-it-really-sucks-to-be

[3] Kirk, J. (2014, February 6). "Apple removes Blockchain, last Bitcoin wallet app, from iOS App Store." Retrieved from http://www.pcworld.com/article/2095060/apple-removes-bl html

[4] Munson, L. (2014, April22). "Google refunds Android users who bought fake Virus Shield app." Retrieved from https://nakedsecurity.sophos.com/2014/04/22/google-refunds-android-users-who-bought-fake-virus-shield-app/

[5] Southurst, J. (2014, February6). "Apple Removes Blockchain Bitcoin Wallet Apps from its App Stores." Retrieved from http://www.coindesk.com/apple-removes-blockchain-bitcoi

[6] Spencer, G. (2014, October 21). "Apple Q4 2014 Results: $42.1 Billion Revenue, 39.3 Million iPhones, 12.3 Million iPads Sold." Retrieved from https://www.macstories.net/news/apple-q4-2014-results-42-1-billion-revenue-39-3-million-iphones-12-3-million

[7] Van Camp, J. (2011, April 8). "Apple rejects PopCap's Unpleasant Horse due to excessive pony meat grinding." Retrieved from http://www.digitaltrends.com/apple/apple-rejects-popcaps-unpleasant-horse-due-to-violence/

[8] Xerox. (2014, February 24). "Apple and Samsung vs the rest." Retrieved from http://www.wds.co/apple-samsung-vs-rest/