Updates Management in Mobile Applications. iTunes vs Google Play

Stefano Comino
Dipartimento di Scienze Economiche e Statistiche
Universita di Udine, Udine (Italy)

Fabio M. Manenti
Dipartimento di Scienze Economiche ed Aziendali "M. Fanno"
Universita di Padova, Padova (Italy)

and Franco Mariuzzo
Centre for Competition Policy
School of Economics
University of East Anglia

CCP Working Paper 15-4

Abstract

In September 2014, more than 1.3 million apps were available in Apple iTunes and Android Google Play stores. Very low entry barriers and an extremely high degree of competition characterize these markets. In such environment one of the critical issues is how to attract the attention of users. In this paper we focus on a specific strategy that app developers may use to stimulate demand for their products: versioning management. Practitioners and developers are well aware that managing app updates (i.e. releasing new versions of an existing app) is critical to increase app visibility and to keep users engaged, disguising a hidden strategy to stimulate downloads. We develop a stylized theoretical model to describe why and when updates should be released. We then use an unbalanced panel with characteristics on the top 1,000 apps on iTunes and Google Play stores for five European countries to empirically test our theoretical predictions. Our results confirm that updates boost downloads and are more likely to be released when the app is experiencing a poor performance. We interpret this finding as evidence that app developers use updates as a "bet for resurrection" strategy.

Contact Details:
Franco Mariuzzo F.Mariuzzo@uea.ac.uk

The authors wish to acknowledge the contribution of the ESRC who supported their research through Centre for Competition Policy funding (ref: RES-578-28-0002).

ISSN 1745-9648

Updates Management in Mobile Applications. iTunes vs Google Play.*

Stefano Comino[†], Fabio M. Manenti[‡] and Franco Mariuzzo[§]

June 16, 2015

Abstract

In September 2014, more than 1.3 million apps were available in Apple iTunes and Android Google Play stores. Very low entry barriers and an extremely high degree of competition characterize these markets. In such environment one of the critical issues is how to attract the attention of users. In this paper we focus on a specific strategy that app developers may use to stimulate demand for their products: versioning management. Practitioners and developers are well aware that managing app updates (i.e., releasing new versions of an existing app) is critical to increase app visibility and to keep users engaged, disguising a hidden strategy to stimulate downloads. We develop a stylized theoretical model to describe why and when updates should be released. We then use an unbalanced panel with characteristics on the top 1,000 apps on iTunes and Google Play stores for five European countries to empirically test our theoretical predictions. Our results confirm that updates boost downloads and are more likely to be released when the app is experiencing a poor performance. We interpret this finding as evidence that app developers use updates as a "bet for resurrection" strategy.

Keywords: mobile applications, updates, downloads, iTunes, Google Play, visibility, minor updates, major updates.

JEL Codes: L10, L63, M31.

*The authors wish to acknowledge the financial support from "Progetto di Ateneo" - Università di Padova, 2012-14 and the contribution of the ESRC who funded our research assistant Martin Graffenberger from Centre for Competition Policy (funding ref: RES-578-28-0002).

[†]Dipartimento di Scienze Economiche e Statistiche, Università di Udine, Udine (Italy).

[‡]Dipartimento di Scienze Economiche ed Aziendali "M. Fanno", Università di Padova, Padova (Italy). Email: fabio.manenti@unipd.it.

[§]School of Economics & CCP, University of East Anglia, Norwich (UK).

1 Introduction

> A lot of app developers will see a large spike in downloads right at launch, and shortly after see these numbers slowly dwindle. The question I get asked in this situation is, "How do I continue growth?" The answer is simple, but the execution takes patience, practice, and a plan.
> In order to continue growth you need to provide constant value, which means everything from creating new game characters to designing a more intuitive user interface to a million things in between. But, this also means updating your app! That said, I always encourage my students to update their apps and keep iterating to get feedback, which helps boost downloads. Letting your apps collect dust is the same as letting them fail, and with the recent release of iOS 7, there is no better time than the present to update your app.
>
> ---
>
> Chad Mureta, "http://blog.appannie.com/updating-your-app-chad-mureta/"

The market for mobile applications is one of the most dynamic segments in today's economy. In December 2014, more than 3 million apps were available on the various stores which include Apple's iTunes, Android's Google Play and Microsoft's Windows Store. Looking at iTunes only, in 2014 the number of apps has grown by nearly 60%, from 890,000 available apps on 1/1/2014 to over 1.42M on 12/31/2014.[1] The growth in the number of available apps has been accompanied by an exponential increase in downloads. According to Statista.com, the cumulative number of apps downloaded from iTunes from July 2008 to October 2014 was about 85 billion. Also in terms of the number of developers and publishers involved in the app market, figures are quite impressive: according to Priori (2014), in February 2014 more than 600 thousand developers published at least one app on iTunes or Google Play, with an increase of nearly 10% with respect to the previous month. Indeed, producing and distributing a mobile application for a developer is relatively easy as it requires a small amount of investment to produce computer software.

The market for mobile apps is therefore characterized on the one side by a fast growing demand from users and on the other by low entry costs, a large number of apps and developers, and high turnover rates. For these reasons the app market has been defined as a "hyper-competitive" marketplace where developers struggle to attract adopters (see Datta and Sangaralingam, 2013). In this dynamic context, with several million apps available for download, one of the most challenging problems faced by developers is to catch the attention of users (see Bresnahan et al., 2014).

In order to improve app visibility, stores created the so-called "top-ranked" apps charts, listing the most popular applications. Several scholars have shown that such charts promote the matching between users and developers with top-ranked apps enjoying a remarkable boost in downloads (Carare, 2012; Ghose and Han, 2014; Ifrach and Johari, 2014; Garg and Telang,

[1] Data taken from www.adjust.com.

2014). In turn, top ranked-charts exacerbate another feature characterizing app markets: the skewness of the distribution of downloads. According to www.appbrain.com, on Google Play about 1 million apps out of 1.4 million have less than one thousand downloads each while, by comparison, just thirteen thousand apps have more than one million downloads. Being in the top positions guarantees success. Therefore, the distribution of downloads is extremely skewed to the right with only a small fraction of applications capturing most of the market.

To climb the top-ranked charts and to improve the visibility of their apps, publishers and developers look for any possible strategy. App promotion, marketing and pricing are among the typical strategies that developers use to get noticed by customers. In this article we focus on another strategy that developers may exploit to attract users' attention, that is, the release of frequent updates.

Our data-set confirms that both in iTunes as well as in Google Play developers release updates with an extremely high frequency: in Google Play apps are updated on average every 28 days, while in iTunes this occurs every 59 days. It is widely recognized that by frequently releasing new software versions, developers are able to increase user engagement. By releasing updates developers stimulate user interest, thus improving app visibility. On top of this, developers usually promote the new versions of their products on blogs, social networks or simply in the "What's new" section of the app store. Again, the update may represent an effective tool to stimulate app visibility and, via this channel, sales.

We analyze the strategy of releasing frequent updates by exploiting the differences across the two main app stores, iTunes and Google Play. Interestingly, the two stores follow different policies regarding the publication of apps and updates. The iTunes "App store review guidelines" explicitly sets a strict screening of apps quality. For example, apps that exhibit bugs or that are in a beta/trial version are going to be rejected by the store. Similarly, applications that are considered not very useful or not providing any lasting entertainment value to users are not published. On top of this, apps that include undocumented or hidden features inconsistent with the description of the app are rejected as well. By contrast, publication on Google Play does not go through a similar quality check; updates can be published instantaneously by developers with a "simple click of the mouse". The absence of a formal screening has led several commentators to criticize Google Play for the poor quality of the apps available in the store. This issue is so critical to Google that it has stepped up its efforts to improve app quality. For instance, in February 2013, it has removed from its stores 60,000 spammy and low quality apps at once. Similarly, in October 2014 it launched a new feature that allows users to filter out all apps that are not rated at least 4 stars.[2]

In this paper, we present a stylized theoretical model investigating the developer's decision about whether to update her mobile application. We then use an unbalanced panel with characteristics on the top 1,000 apps on iTunes and Google Play stores for five European countries to empirically test the predictions we derive from the theoretical model. Interestingly, we find that the release of an update positively affects downloads in iTunes while it has no significant impact in Google Play. We interpret this finding in terms of the institu-

[2]www.appbrain.com estimates that nearly 15% of Google Play apps are of low quality.

tional differences characterizing the two stores. As argued above, only in iTunes is there a strict quality check for apps and updates. Therefore, the absence of a significant impact of updates in Google Play might be due to the fact that in this store both high and low-quality updates get published so that the overall effect of downloads is not significant. Another interesting prediction of our model that is confirmed by the empirical analysis relates to the conditions under which the developer finds it profitable to release an update. As we show in the paper, an update is more likely to be released when the developer observes a worsening of the performance of the app. Hence, in order to stimulate the attention of potential users, and "revive" the app, developers are induced to release a new version of the software. Given that the release of a new version might be risky, we interpret this finding as a sort of "bet for resurrection" strategy that developers employ when observing their apps performing poorly. We conclude our analysis by distinguishing in iTunes between major (significant changes in app functionalities) and minor updates (bug fixing and minor changes); our empirical investigation suggests that the latter are more likely to be employed by developers as a strategic tool to improve app performance on the market.

The paper is organized as follows. In Section 2 we discuss the relevant literature on mobile apps. In Section 3, we present the theoretical model and we derive the testable predictions. Sections 4 and 5 are devoted to presenting the data and the empirical strategy we employ in the estimations. The results of the empirical investigation are discussed in Sections 6 and 7. Section 8 concludes.

2 Literature review

The astonishing growth of mobile ecosystems has attracted the attention of several scholars. Despite that this is a recent phenomenon, the literature that studies the characteristics and the functioning of app markets is already quite developed. As we have pointed out above, the number and the type of apps available to users is extremely large. This raises the issue of how to attract the attention of users in order to emerge from the mass of hundreds of thousands of applications. According to Bresnahan et al. (2014) this is the crucial issue for app developers. The authors argue that app stores play a dual role: they lower the technical costs of developing and distributing applications, but also set up very high marketing costs for developers. In app stores, competition is pervasive and it does not only emerge among apps performing the same or similar tasks. Each app competes for consumer attention with all the other applications available in the store. For this reason, how to become visible to consumers is the most interesting issue discussed in the literature.

Ghose and Han (2014) present one of the first estimates of the demand for mobile applications. The authors quantify the consumer preferences for different app characteristics based on a structural model that combines a random coefficients nested logit demand model with the pricing equations of software developers. The analysis is based on daily information on the top-400 free apps and the top-400 paid apps in iTunes and Google Play platforms. The authors estimate downloads by means of a calibration exercise, relating app ranking with the number of downloads. They find that demand is larger when app description is more

accurate (measured in terms of description length and number of screenshots), when the app has the in-app purchase option, and when it is older (they measure both the age of the app and of the version). They also find that demand increases with the number of apps available from the same developer, and when the app is available on multiple platforms (iTunes and Google Play). User reviews are also found to play a significant role as their volume and effectiveness stimulates downloads. A finding which is relevant to our investigation is that demand is boosted by the number of previous versions of the same app.

Interestingly, Ghose and Han (2014) also find that cross-charting (namely, an app appearing both in the top-free and top-paid charts) has a positive impact on app demand. This suggests that being in the list of top-ranked apps may have a valuable effect in terms of stimulating additional downloads. Carare (2012) investigates in detail the role of top-ranked charts in stimulating future app demand. The work is based on the top-100 paid apps available in the US iTunes store. The author shows that the bestseller status of the top-ranked apps is a very important determinant of consumer willingness to pay, and that the effect of rank declines very steeply for the top 10 apps and becomes negligible for apps ranked higher than 50.

As argued in the introduction, developers update their apps very frequently. This strategy is not only specific to mobile applications but it is also commonly observed in the "traditional" desktop computer software (see Greenbaum, 2005). Following Sankaranarayanan (2007), the release of a new version of a "traditional" software occurs especially when the package has reached a high level of penetration so that little revenue can be collected from new customers. Software firms are therefore induced to upgrade their packages in the attempt to re-sell the software to their installed base of users.[3] This explanation for the release of frequent updates is unlikely to fit the case of mobile applications. As a matter of fact, a common rule in app stores is that updates must be made available free of charge to anyone who has previously downloaded the app.[4] As a consequence, developers cannot exploit their installed base of users by trying to re-sell upgrades of their software.[5]

In the next section we sketch a simple model to explain the strategic role of version updates.

3 The decision to update: a theoretical framework

In the market for apps, developers compete for consumer attention. With the huge mass of software available for downloads, success depends heavily on how "visible" an app is.

[3]This strategy may give rise to a classical Coasian commitment problem in durable goods, which damages producers. Sankaranarayanan (2007) suggests that software vendors may overcome the Coasian problem contractually by entitling customers to any update they are going to release during a predetermined period of time. Within this period, vendors are unable to collect revenues from their installed base of users, a constraint that decreases substantially the temptation to release updated versions.

[4]See http://digitalmediadiet.com/?p=1292.

[5]The incentives to release an update to profit from the installed base of users can be partially restored when the app comes with the in-app purchase option.

Visibility might be related to the ranking the app achieves within the top-ranked charts or it might be related to what we call the "buzz" surrounding the app, i.e. how much blogs, social networks or specialized magazines and websites "talk" about the app.

The ability of a developer to attract the attention of potential users depends on a number of elements. The intrinsic quality of the software code written by the developer and the ability to meet consumer needs/tastes are certainly crucial for app success. The prestige of the developer and/or the recognition of the brand are also very important factors; however, in order to emerge from the mass of available apps, the ability of the developer to attract the interest of bloggers and journalists of specialized magazines aimed at stimulating the buzz is also essential to succeed.

In the following pages, we present a stylized model that studies the choice of a developer about whether or not to release a new version of her app. Following our previous discussion, the key features of the model are: *i*) downloads depend both on the intrinsic quality of the software and on the buzz surrounding it, *ii*) demand (downloads) is right skewed and *iii*) the release of an updated version stimulates the buzz around the app.

3.1 The model

Consider a developer who has already published two apps in the store. The developer has to decide whether to update one of her apps (that we indicate as the "focal" app). In taking this decision the developer aims at maximizing downloads, net of further possible costs to update the software. Let v denote the visibility of the focal app perceived by potential users. We model visibility as the sum of two components: the intrinsic/true quality of the software, q, and the impact of what we call the buzz around the app, b. Both software quality and the buzz depend on the released version. Formally, we express visibility as $v(u) = q(u) + b(u)$, where $u = 0$ for the current version of the app, and $u = 1$ in the case the developer releases an update. An app can be highly visible if its intrinsic quality is high and/or it is surrounded by a positive buzz (i.e. good users reviews, positive discussions on dedicated blogs, etc.).

The crucial assumption of the model is that the decision to release an update stimulates the buzz surrounding the app. This assumption is taken on practical grounds provided that updates tend to stimulate discussions or comments in dedicated blogs/magazines/websites or on social networks. Clearly, the increased buzz can be either positive or negative: bloggers, journalists, but also regular users might positively or negatively welcome the new version of the software. In other words, the augmented buzz may improve or worsen app visibility. Formally, we assume that an update makes app visibility more uncertain.

In what follows, b is assumed as a realization of a random variable; while q, for the sake of simplicity, is assumed to be deterministic.

According to the empirical evidence outlined in the introduction, we assume that the number of downloads is highly skewed on the right. Formally, we assume that there exists a threshold level τ for app visibility such that:

$$\begin{array}{ll} \text{if } v \geq \tau & \text{downloads are } \overline{D} = \overline{d} + \rho\overline{d}, \\ \text{if } v < \tau & \text{downloads are } \underline{D} = \underline{d} + \rho\underline{d}, \end{array}$$

where $\overline{d} > \underline{d} \geq 0$. $\overline{d}$ and $\underline{d}$ represent the amount of downloads of the focal app while $\rho\,\overline{d}$ and $\rho\,\underline{d}$ measure the impact of downloads of this app on the other app distributed by the developer. In other words, $\rho\,d$ indicates the increase/decrease in the other application downloads due to the performance of the focal app. The two apps can be either complements ($\rho > 0$) or substitutes ($\rho < 0$). Therefore, an increase in downloads of the focal app can either contribute to stimulate downloads of the other app or it can cannibalize the latter. Complementarity may emerge from a "branding effect" or from cross advertising between the two apps. On the opposite, substitutability may occur when the two apps address the same or similar user needs: a crowding out effect.

The current version of the focal app has visibility $v(0) = q(0) + b(0)$, where $b(0)$ is the realization of a random variable, uniformly distributed over the segment $(B - \eta, B + \eta)$, with density function $f(b(0)) = 1/(2\eta)$. Total expected downloads/payoff generated by the current version of the focal app amount to:[6]

$$\left(\int_{B-\eta}^{\tau-q(0)} \frac{1}{2\eta} db(0)\right) \underline{D} + \left(\int_{\tau-q(0)}^{B+\eta} \frac{1}{2\eta} db(0)\right) \overline{D} = \frac{\left(\overline{D} - \underline{D}\right)(B - \tau + q(0))}{2\eta} + \frac{\overline{D} + \underline{D}}{2}. \quad (1)$$

In the case the focal app is updated, the developer incurs the (development) cost ϕ, and the new version of the software has visibility $v(1) = q(1) + b(1)$, where:

- $q(1) = q(0) + \Delta$, with $\Delta \lesseqgtr 0$. In other words, the new version of the software may have a higher or smaller intrinsic quality;[7]
- $b(1)$ is the realization of a random variable uniformly distributed over the segment $(B - \gamma\eta, B + \gamma\eta)$, with $\gamma \geq 1$, according to the density function $f(b(1)) = 1/(2\gamma\eta)$. Following the above discussion, we assume that the release of the update stimulates the buzz around the app, thus increasing the uncertainty about its visibility ($\gamma \geq 1$).

Based on the above assumptions, the expected payoff associated with the decision to release an update is:[8]

$$\left(\int_{B-\gamma\eta}^{\tau-q(0)-\Delta} \frac{1}{2\gamma\eta} db(1)\right) \underline{D} + \left(\int_{\tau-q(0)-\Delta}^{B+\gamma\eta} \frac{1}{2\gamma\eta} db(1)\right) \overline{D} - \phi = \frac{\left(\overline{D} - \underline{D}\right)(B - \tau + q(0) + \Delta)}{2\gamma\eta} + \frac{\overline{D} + \underline{D}}{2} - \phi. \quad (2)$$

[6] We focus on the most interesting case where both $v(0) \geq \tau$ and $v(0) < \tau$ occur with positive probability. Therefore, parameter τ satisfies the condition $q(0) + B - \eta < \tau < q(0) + B + \eta$.

[7] The case $\Delta < 0$ might occur when the new version comes with bugs or when it includes new features that users do not appreciate or that worsen the usability of the software.

[8] We assume that also, when the update is released, both $v(1) \geq \tau$ and $v(1) < \tau$ occur with positive probability. Therefore, parameter τ satisfies the condition $q(0) + B + \Delta - \gamma\eta < \tau < q(0) + B + \Delta + \gamma\eta$.

In this case, the payoff is composed of two elements, the expected total downloads generated by the release of the new version of the software, and ϕ, the cost of developing the update.

By comparing expressions (1) and (2) one can easily determine the condition under which the developer chooses to release an update:

Proposition 1 *The developer releases a new version of the focal application when the increase in total expected downloads is higher than the cost of developing the update; formally,* $u = 1$ *when:*

$$(\overline{D} - \underline{D})[(\tau - q(0) - B)(\gamma - 1) + \Delta] \geq 2\gamma\eta\phi. \tag{3}$$

A necessary condition for the release of a new version of the software is that the update generates an increase in total expected downloads. Formally, this occurs when the term within the square brackets of expression (3) is positive. A simple inspection of expression (3) reveals that this is more likely to happen when the expected visibility of the current version of the focal app $(q(0) + B)$ is small relative to the threshold level τ.

In other words, when the developer expects a poor performance of the current version of her app, she might be induced to release an update in the hope of stimulating positive buzz around it and, via this channel, downloads. This is a risky strategy as app visibility becomes more uncertain. This is why we reinterpret the decision to release the update as a sort of "bet for resurrection" strategy. Clearly, this decision is profitable provided that the development cost ϕ is small compared to the increase in expected downloads.

Looking more closely at expression (3), it is possible to verify that an update is more likely to be published whenever its intrinsic quality is large (Δ is large) and when the impact on downloads, $\overline{D} - \underline{D}$, is sizeable. This latter condition is more (less) likely to occur when the apps of the developer are complements (substitutes), that is, when $\rho > 0$ ($\rho < 0$).[9]

An interesting implication of Proposition 1 is the following:

Corollary 1 *The developer may decide to release an update even if it does not improve the intrinsic quality of the app,* $\Delta \leq 0$.

Proof. Suppose that $\Delta = 0$. A necessary condition for inequality (3) to be satisfied is $\tau > q(0)+B$. Notice that with $\Delta = 0$, the model is defined for $\tau \in (q(0)+B-\eta, q(0)+B+\eta)$. Therefore, when $\tau \in (q(0) + B, q(0) + B + \eta)$, condition (Equation 3) holds provided that $\overline{D} - \underline{D}$ large enough/ϕ small enough. By continuity, it follows that updates with lower intrinsic quality ($\Delta < 0$) might also be profitable.■

[9] A larger γ (greater variance of the visibility of the update) may induce the developer to release an update more or less often depending on the value of the threshold. When $\tau > q(0) + B + \Delta$, then a larger γ makes condition (3) more likely to be satisfied. In other words, when the expected perceived quality of the app is very low with respect to the threshold τ, then an increase in γ makes the update more profitable. By contrast, when τ is smaller than $q(0) + B + \Delta$, then a larger γ makes condition (Equation 3) less likely to be satisfied.

3.2 Testable predictions

The theoretical model can be used to derive some testable predictions. The first prediction follows directly from Proposition 1. As discussed above, according to Proposition 1 the developer releases an update whenever the expected visibility of the current version is low compared with the threshold. It is reasonable to assume that the developer forms expectations on the visibility of the current version of their app by looking at its past performance. In this case, Proposition 1 implies that:

Conjecture 1 *Developers are more likely to release an update when they observe a worsening of the app performance.*

As discussed above, condition (3) is more likely to hold when the developer distributes more than one complementary application. On the contrary, when apps are substitutes, developers are less prone to update. Based on these arguments, we predict that:

Conjecture 2 *Developers distributing more than one application are more (less) likely to release updates when apps are complements (substitutes).*

According to Corollary 1, developers may also decide to release qualitatively worse updates, hoping to stimulate downloads via the effect on buzz. The corollary suggests an interesting prediction related to the institutional differences between the iTunes and Google Play stores. As we argued in the introduction, while iTunes has a formal quality check for publishing apps and related updates, such a check is not implemented in Google Play. Therefore, in principle, quality-worsening updates can be released in Google Play but they cannot in iTunes. This amounts to saying that Corollary 1 may apply only to Google Play apps; based on this reasoning we expect that:

Conjecture 3 *The effect of the release of an update on downloads is stronger in iTunes than in Google Play.*

The theoretical model assumes that developers base their choice about whether to release an update just looking at expected future downloads (and development costs). Therefore, in the model, we implicitly assume that developers only profit from new users of their apps. This is a well taken assumption given that developers must make new versions of the software available free of charge to earlier adopters of the app. However, a common commercial strategy used by developers is to include so-called in-app purchases, that is the option to pay in order to access improved functionalities of the software (or, in the case where the app is a game, access to more challenging parts of the story). When apps come with the in-app option, developers profit not only from new consumers but also from the installed base of users. In this case, it is even more compelling for developers to adopt strategies that increase users' engagement. In terms of updating strategies, when the app has the in-app purchase option, this translates into stronger incentives to release new versions. Therefore, we expect that:

Conjecture 4 *Developers of applications with the in-app purchase option are more likely to update their apps.*

4 The data

The data used in this study is a combination of information obtained from the consulting analytics Piori and data downloaded from the web-site AppAnnie.com. Priori provided us with monthly data on the top 1,000 most downloaded apps in iTunes and Google Play in five European countries (Germany, France, Italy, Spain, and the UK) for the period September 2013-February 2014. According to Priori (2014), the top 1,000 apps cover about 60% of the market in each country (e.g. in October 2013 our Priori data cover 55.3% of downloads in iTunes in UK and 62.09% in Italy).

For each app, the Priori dataset provides the following information: name of the app, name of the publisher, app monthly and overall number of country downloads, the worldwide average customer rating of the app (in a scale from 1 to 5), the number-of user ratings, the date when the app has been published in the store, the overall number of updates released, the price of the app, when suitable, and whether the app has the in-app purchase option.

With the exception of the number of downloads, all the information gathered by Priori is taken directly from the app stores. Downloads, instead, are computed combining publicly available information (financial statements and other reputable or verified press sources) with Priori proprietary metrics establishing a relationship between downloads and user ratings, ranking (i.e. position in the app stores top-ranked charts) and number of reviews. For a sample of apps, Priori cross-checked these estimates by using real downloads data provided by partner developers.[10] Only first-time installations are counted as downloads in our data; updates of already installed applications are not counted as downloads.

For iTunes apps, we complement Priori data with additional information taken from the App Annie web site. Specifically, for each app we collected the following information: the type of compatibility (app for iPhone/iPod touch, for iPad only, for iPhone only or Universal), the size of the code (data collected in May 2014), the age restrictions (4+, 9+, 12+, 17+) and whether the app is available in the language of the country or not. On top of this, we identified in the sample the so-called corporate apps, namely apps used by companies as additional distribution channels (e.g. airlines or banking apps, newspaper or TV network apps etc.) or to spread information about their services. We also collected additional information on the monetization strategy followed by the publisher; for each free app we checked whether the publisher also distributes a pro version or, in case of a paid app, a free version (usually with few functionalities or with in-app advertising).

Finally, and relevant for this article, for each iTunes app we also gathered information on the type of update. Conventionally, software developers keep track of the different versions of their products by means of a three-digit sequence, where the first digit identifies major updates and the second and third digit minor updates of decreasing significance. So, for example, the current version of a given app can be 2.12.4 meaning that there have been two

[10] According to Priori's statement, this internal validation study, based on 2,000 Android apps, returned a mean absolute error of +/- 24.6 per cent the level of downloads. In our regression analysis we employ the growth rate of downloads rather than their absolute level, and in this way we smooth out the underlying issue of nonrandom measurement error.

major updates and sixteen minor ones (12+4).[11] It is customary to consider minor updates new versions of the software aimed at fixing bugs (e.g. crashing) or at including minor additional features, while major updates are aimed at distributing software with significant jumps in functionalities. It deserves to be noticed that once published the new version of the software is available worldwide. In other words, the developer cannot choose to update the app for certain countries but not for others.

For each iTunes app in the sample we are able to distinguish between minor and major updates and for each major update when it was published in the store. Once published, updates are available worldwide, namely they are not country-specific, but are a time varying attribute of the app.

4.1 Descriptive statistics

All variables in our data-set, but downloads, are country invariant. Given this feature and given the purpose of our research, we aggregate the data from the five European countries; following this aggregation, monthly downloads (and the growth in downloads) are the sum of the downloads in the five countries in a given month. Summary statistics for the two stores are provided in Table 1. On the top panel we display the aggregated/pooled data and on the bottom panel the restricted sample on which we base our empirical analysis. From the original 30,000 observations per store (1,000 apps, during 6 months, in 5 countries) we are left with 15,985 observations for Google Play, and 14,765 for iTunes. This drop in the number of observations is due to the fact that, in several cases, the same app appears in the top 1,000 ranking in more than one country in a given month.

The bottom panel in the table highlights that only a small proportion (about 20 per cent) of the sample is of use for our regression analysis. The reason for this second drop in sample size is that the econometric specification which we use requires the apps to be observed in at least three subsequent periods. As we highlight below, a significant number of apps enter the top 1,000 ranking during one or two months and, therefore, it cannot be employed in the econometric analysis (see Table 2). The final sample considered in the regressions is composed of 2,956 observations for Google Play and 3,660 for iTunes.

The main characteristics of the aggregated/pooled sample are:

- In Google Play nearly all the top 1,000 apps are free; only 17 observations represent paid apps. Free apps are also prevalent in iTunes, although paid apps are more frequent than in Google Play (8.3% of the observations have a positive price).

[11] Developers may deliberately skip multiple versions at a time (for instance jumping directly from version 2.12.4 to version 5.0.0 or to version 2.15.0) to signal that a significant number of new features have been added. The discrete jump in the digit number reflects the subjective evaluation of the developer about the importance of the new feature that have been released. Because of this subjective flavour in our empirical analysis we will concentrate on updates and not much on the number of digits that are skipped from version to version.

Table 1: Summary statistics from the five countries

	Google Play			iTunes		
	N.	mean	std. dev.	N.	mean	std. dev.
			Full sample			
			Priori data			
free	15,985	0.999	0.033	14,675	0.917	0.276
price (if free=0)	17	2.447	1.148	1,219	2.760	2.916
in-app purchase	15,985	0.297	0.457	14,675	0.562	0.496
local	15,985	0.371	0.483	14,675	0.349	0.477
user rating	15,985	4.077	0.449	14,675	4.067	0.651
user rating count	15,985	80,320	375,114	14,675	47,645	174,580
age (in months)	15,985	15.036	13.993	14,675	19.227	15.868
age version (in months)	15,985	1.836	3.100	14,675	2.290	3.781
number versions	15,985	33.430	78.487	14,675	10.044	9.860
update version*	7,991	0.542	0.498	8,469	0.453	0.498
apps same developer	15,985	7.182	17.343	14,675	8.925	18.089
number countries	15,985	1.877	1.395	14,675	2.027	1.490
monthly downloads	15,985	249,837	1,244,139	14,675	58,243	166,898
growth downloads*	8,225	0.476	5.335	8,591	0.115	1.808
			App Annie data			
age major version (in months)				13,645	10.523	9.913
number major versions				13,773	2.050	2.005
update major version				8,113	0.037	0.188
size				13,771	64.088	149.987
			Selected sample			
			Priori data			
free	2,956	0.999	0.032	3,660	0.958	0.200
price (if free=0)	3	3.873	0.203	152	3.199	2.631
in-app purchase	2,956	0.349	0.477	3,660	0.611	0.487
local	2,956	0.361	0.480	3,660	0.337	0.473
rating	2,956	4.119	0.413	3,360	4.151	0.564
rating count	2,956	207,489	699,731	3,660	79,820	235,946
age (in months)	2,956	20.664	13.849	3,660	23.614	15.294
age version (in months)	2,956	2.134	3.352	3,660	2.774	3.794
number versions	2,956	54.908	91.956	3,660	13.304	10.744
update version	2,956	0.507	0.500	3,660	0.410	0.492
apps same developer	2,956	11.066	23.898	3,660	11.828	23.139
number countries	2,956	2.398	1.667	3,660	2.374	1.632
monthly downloads	2,956	476,337	2,189,596	3,660	69,856	157,927
growth downloads	2,956	0.204	1.284	3,660	0.003	1.241
			App Annie data			
age major version (in months)				3,539	12.625	9.794
number major versions				3,557	2.269	2.192
update major version				3,557	0.034	0.181
size				3,557	64.457	149.715

*Updates and growth are in first differences and hence have less observations

- There is a significant difference between the two stores in terms of the number of apps with in-app purchases: 29.7% of the Google Play sample has in-app purchases, while in iTunes the same occurs in 56.2% of apps.

- The average user rating is over 4 (out of 5) in both stores with higher cross-sectional dispersion in iTunes. User rating count (number of users providing a rating) is double in Google Play.[12]

- iTunes apps are on average older than apps in Google Play, thus suggesting a higher turnover rate in the top 1,000 apps in the latter store. In iTunes the average app age is 19.227 while in Google Play it is 15.036.

- On both stores, apps are updated frequently. On average in Google Play apps are updated about 33 times since their publication while this figure reduces to about 10 in iTunes. This difference may be due to the aforementioned divergent regulations on the publication of apps and updates implemented by the two stores; the absence of a strict quality check may partially explain why we observe so many more updates in Google Play. A possible additional explanation to these figures is related to the fact that Google Play apps run on the Android mobile operating system which can be installed on several different devices, and that may require a closer management of updates by developers;

- Downloads are much higher in Google Play than in iTunes, with Google Play apps on average downloaded nearly five times more than iTunes apps (249,837 compared with 58,243).

- In both stores, roughly 35% of apps are local. An app is named as local in a given country when at least 40% of its all time downloads occur in that country.

- About 5 per cent of the apps in our sample are multi-homed, i.e. they are available both stores.

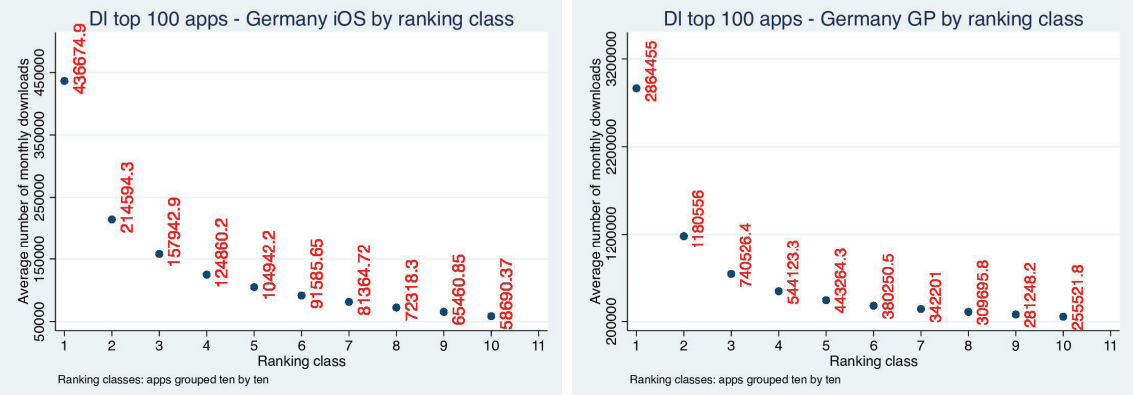
- On average, in Google Play, a developer distributes 7.182 top-ranked apps. The figure for iTunes is 8.925.

- On average an app enters the top-1,000 ranking in about 2 out of 5 countries in both stores (1.877 in Google Play and 2.027 in iTunes).

- 54.2% of Google Play apps and 45.3% of iTunes apps are updated every month during the six months period of observation.

[12]Neither user rating nor user rating count are employed in our econometric application because the former tends not to vary much over time, and the latter is problematic as it refers to a period that may differ from the one used in our data, leading to serious measurement error. For these reasons both variables are excluded from our econometric work, though they are important and appealing variables that would have been ideal to proxy for visibility and quality of the app.

Figure 1: Distribution of downloads



(a) iTunes (b) GP

Comparing the selected and full samples, we observe that the former is made of larger and older apps that are more frequently distributed by multi-app developers, are characterised by lower growth with higher user rating and user rating count, and are more often of the multihoming type (about 10 per cent).

Our data confirms a couple of features that have already been found in the literature (see, among others, Bresnahan et al., 2014): *i*) downloads exhibit an extremely skewed distribution, with top apps accounting for a large fraction of total downloads, and *ii*) large turnover/churn, with few applications which succeed in staying in the top 1,000 list in all the six months of observation. Regarding feature *i*), Figure 1 shows the distribution of downloads for the top 100 apps in Germany for iTunes and Google Play; the two diagrams show the average number of downloads for each decile of the distribution. In the case of Google Play, for instance, the average monthly downloads of apps of the first decile is 436,674, which is twice the average number of monthly downloads of apps in the second decile (214,594) and about eight times more than the downloads of tenth decile (58,690).

As for feature *ii*), Table 2 displays the high level of turnover that characterizes iTunes. The overall number of different apps that we observe during the six-month period in the five countries is 10,986;[13] 18.24% of these apps are observed every month (indicated in the table with "All months"). However, a substantial percentage (about 44%) of apps appear only in one month. The level of turnover is even larger in Google Play (table not reported). For the Android store, the overall number of applications that we observe increases to 13,034 and the share of apps that appears only one month is about 50%.

Figure 2 shows the kernel density of the growth in downloads in the two stores distin-

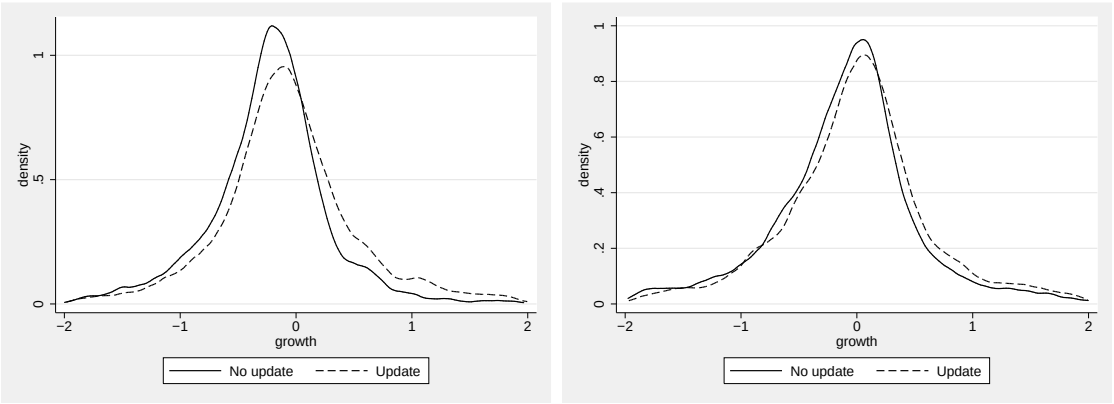
[13]The minimum number of apps we could have observed in our sample is 1,000 (the top 1,000 apps are the same in the five countries during the whole period) while the maximum is 30,000 (the top 1,000 apps change every month and are different in the five countries under observation).

Table 2: Pattern of apps, iTunes, all periods and countries

Freq.	%	Cum.	Pattern
2,004	18.24	18.24	All months
1,098	9.99	28.24	Sept only
1,025	9.33	37.57	Feb only
762	6.94	44.50	Oct only
755	6.87	51.37	Nov only
637	5.80	57.17	Jan only
592	5.39	62.56	Dec only
423	3.85	66.41	Sept & Oct
354	3.22	69.63	Dec, Jan & Feb
3,336	30.37	100.00	(other patterns)
10,986	100.00		N. of different apps

guishing between updated and non-updated apps.[14] Interestingly, in Google Play the two density functions nearly perfectly overlap; this suggests that updated and not updated apps perform very similarly in terms of downloads. On the contrary, in iTunes the density function of non updated apps is more asymmetric to the left and concentrated on negative values of the growth rate of downloads. This is a very preliminary evidence towards a different role played by updates in the two stores. The aim of our theoretical model and of the econometric exercise is to shed some light on this evidence.

Figure 2: Download growth density function



(a) iTunes

(b) Google Play

[14]Growth rates are expressed in logarithms and refer to the overall amout of downloads each app obtains in the five countries.

5 The econometric model

We deal with a longitudinal data-set which has a large cross-section of mobile applications (apps), of size J, and limited number of periods and countries, of size T and C, respectively. Following that, asymptotics relies on the apps dimension.

For each store separately, we study whether new updates, captured by a binary variable $\{0, 1\}$, affect the downloads - specifically, the download growth between period t and $t-1$. Furthermore, we are interested in unravelling the determinants of the release of updates. We deal with the possible simultaneity between download growth and change in versions (updates). We denote with g_{jct} the download growth rate for a mobile application $j \in \mathcal{J}_t$ distributed in country $c \in \mathcal{C}$ in the period $t \in \mathcal{T}$. We model the growth rate empirically as a linear dynamic model made of observable and unobservable mobile application characteristics. One key observable characteristic we focus our analysis on, is the apps versioning update. Updates are released simultaneously in all countries and, apart from possible language translations are homogeneous. Because updates are country invariant, we obtain an aggregated measure of the downloads for the list of countries we have data on and compute the growth rate from that aggregate figure. Growth for the app j in period t is calculated as $g_{jt} = \frac{\sum_{c\in\mathcal{C}}(Q_{jct}-Q_{jc,t-1})}{\sum_{c\in\mathcal{C}} Q_{jc,t-1}}$; where we treat the number of downloads (contemporaneous and lagged: Q_{jct} and $Q_{jc,t-1}$) as zero if the app is not present in the top 1000 ranking in the country c, in the relevant period.

We specify both the growth and the update equations as linear autoregressive distributed lag model of order 1. In addition to a set of controls we complement the download growth equation with the contemporaneous impact of updates u_{jt} and supplement the update equation with lagged growth, yielding the system of linear equations

$$\begin{aligned} g_{jt} =& \phi_{11} g_{j,t-1} + \phi_{12} u_{jt} + h_{1t} + \mathbf{x}_{1jt}\boldsymbol{\beta}_1 + \alpha_{1j} + \varepsilon_{1jt} \\ u_{jt} =& \phi_{21} g_{j,t-1} + \phi_{22} u_{j,t-1} + h_{2t} + \mathbf{x}_{2jt}\boldsymbol{\beta}_2 + \alpha_{2j} + \varepsilon_{2jt}, \quad t = 2, \cdots, T. \end{aligned} \tag{4}$$

The choice of modelling lagged growth as determinant of an update is motivated by our theoretical model, where updates are seen as a response to a drop in demand (reduction in download growth). For short panels, as it is the case here, it is common to let the time effect be fixed and hence exclude it from the composite error term, ζ_{kjt}; which is for each equation $k = \{1, 2\}$ the sum of the app unobserved heterogeneity, α_{kj}, and the pure idiosyncratic error term, ε_{kjt}. We assume the idiosyncratic error to be uncorrelated both over time and apps. We account for the time effect on growth with the function h_{kt}. The vector $\mathbf{x}_{kjt}$ include a set of controls. The variable *update* is endogenous and further endogeneity is brought in by the presence of unobserved heterogeneity.

As both the lagged dependent variable and the current value of the update are expected to be correlated with the app unobserved heterogeneity, we follow the dynamic linear panel model literature and take the first difference of both sides of the system of equations (5) to remove inconsistencies of the parameters driven by such correlation, leading to the first

differences econometric system of equations of interest

$$\begin{aligned}\Delta g_{jt} =& \phi_{11}\Delta g_{j,t-1} + \phi_{12}\Delta u_{jt} + \tau_{1t} + \Delta \mathbf{x}_{1jt}\boldsymbol{\beta}_1 + \Delta\varepsilon_{1jt} \\ \Delta u_{jt} =& \phi_{21}\Delta g_{j,t-1} + \phi_{22}\Delta u_{j,t-1} + \tau_{2t} + \Delta \mathbf{x}_{2jt}\boldsymbol{\beta}_2 + \Delta\varepsilon_{2jt}, \quad t = 3, \cdots, T. \end{aligned} \tag{5}$$

Though in the structural system of equations (5) we have eliminated the unobserved heterogeneity and the aforementioned issue of correlation, still the system cannot be estimated consistently by OLS. We also deal with other sources of correlation. In the first equation we cope with the correlation between $g_{j,t-1}$ and $\varepsilon_{1j,t-1}$ in addition to that between the update variable and download growth, which causes $E(\Delta u_{jt}, \Delta\varepsilon_{1jt}) \neq 0$. Similarly in the second equation we tackle the endogeneity between $\Delta\varepsilon_{2jt}$ and $\Delta u_{j,t-1}$, in addition to the endogeneity between $\Delta\varepsilon_{2jt}$ and $\Delta g_{j,t-1}$.

In the next subsection we discuss the instruments that we use to correct for such multifaceted endogeneity.

5.1 Instruments

To identify the download growth dynamics and the effect of an update on growth, along with the update dynamics and the impact of lagged growth on updates - the first and second equation in (5) - we select a set of instruments that are *strong*, i.e. explain the endogenous variables in each equation, but do not explain the dependent variables directly, if not via the other explanatory variables that determine the dependent variable, that is, are also *valid*. To select the instruments that satisfy validity and strength we exploit the time series dimension of the panel and the multiproduct position of a sizeable number of developers.

Underneath we describe the main successful instruments and refer the complete list of instruments, supported by relevant tests on the validity and strength, to the footnote of the results table.

- We instrument $\Delta g_{j,t-1}$ with $g_{j,t-2}$, as suggested in Anderson and Hsiao (1981), in addition to the first difference of average lagged growth of the other $J_{f,t-1} - 1$ apps distributed in top positions (top-1000 apps) and in at least one of the countries of investigation by the developer f. The instrument for the first difference of lagged growth of app j is calculated as $\Delta\frac{\sum_{l\in(\mathcal{J}_{f,t-1}-j)} g_{l,t-1}}{J_{f,t-1}-1}$. For cases where the developer has marketed only one app in top positions in at least one country ($J_{f,t-1} = 1$), the instrument takes value zero.

- We instrument the update variable, Δu_{jt}, with the second difference of average updates (as the first difference raised issues of validity) of the other $J_{ft} - 1$ apps distributed in top positions (top-1000 apps) and in at least one of the countries of investigation by the developer f. The formula for this instrument is $\Delta^2\frac{\sum_{l\in(\mathcal{J}_{ft}-j)} u_{lt}}{J_{ft}-1}$. Here, the variable takes the mid-point value between zero and one, which is one-half, if the app is the only top-app distributed by the developer in at least one of the relevant countries ($J_{ft} = 1$).

Additional instruments that we use are: the first difference of lagged age of the version, the first difference of lagged number of versions, and the first difference of the average age of the version.

This logic of instrumentation is extended to the endogenous variables of the update equation. As stated earlier we document the full list of instruments in the footnote of the results table.

In the next section we discuss the main results.

6 Results

Columns (1) and (2) of Table 3 show the estimates of the first equation of the system (5) for iTunes and Google Play, respectively. For iTunes, the third column highlights the estimates when distinguishing between major and minor updates. The estimates are presented in first differences, hence time invariant variables (most of the App Annie variables that we have collected) are not identified. From the table it follows that:

1. The past performance of the app, measured in terms of the growth rate of downloads during the previous month, has a negative and statistically significant effect on the current rate of growth. This result seems to suggest that downloads follow a cyclical pattern with alternating periods of growth and downturn;

2. Consistent with Conjecture 3, we find that the release of an update has different effects on iTunes and Google Play. On iTunes the growth rate of downloads is positively affected by the variable *update*, meaning that the release of an update boosts downloads; quantitatively we find that having released an update increases the rate of growth of downloads by about 36%. By contrast, the publication of an update on Google Play does not have a significant impact. This evidence might seem quite surprising, but it can be interpreted on the basis of our theoretical model. Corollary 1 suggests that developers might be willing to release low quality updates in order to stimulate the buzz surrounding the app; this strategy can be implemented only on Google Play, where developers are not subject to any screening concerning the quality of their applications. From these considerations the non significance of the *update* coefficient for Google Play might be due to the fact that developers release both high and low quality updates and, on average, they do not significantly impact on downloads. On the contrary, the strict quality check on iTunes ensures that only high quality updates get to be published in the store, thus explaining the positive impact of the variable *update*.

3. Both on iTunes as well as on Google Play, the growth rate of downloads increases with the number of other applications by the same developer listed in the top 1,000 (the number of other applications is measured at the country, month and store level). This result reveals the presence of complementarities among apps (that developers might exploit through a "branding effect" or cross-advertising).

Table 3: First difference growth and update equations†

	Growth equation gdl_t			Update equation u_t			
	iTunes	GP	iTunes maj-min	iTunes	GP	iTunes major	iTunes minor
	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Lag growth (gdl_{t-1})	-0.026^a	-0.009^a	-0.028^a	-0.042^a	0.0004	-0.007^c	-0.032^b
	(0.008)	(0.001)	(0.008)	(0.015)	0.0003	(0.004)	(0.015)
Update (u_t)††	0.362^a	0.281		0.056^b	0.219^a		
	(0.059)	(0.186)		(0.027)	(0.037)		
Major update (u_{1t})††			0.621^b			0.048^b	-0.347^a
			(0.266)			(0.023)	(0.070)
Minor update (u_{2t})††			0.316^a			-0.0002	0.110^a
			(0.065)			(0.0099)	(0.030)
In-app	-0.462	-0.657^a	-0.455	0.698^a	0.317^a	0.234^b	0.525^a
	(0.287)	(0.286)	(0.285)	(0.089)	(0.125)	(0.110)	(0.163)
Free	1.687^a		1.681^a	-0.034		-0.008	-0.0002
	(0.497)		(0.504)	(0.124)		(0.023)	(0.113)
Number of apps by developer	0.078^a	0.031^b	0.083^a	0.011^a	-0.002	-0.001	0.012^a
	(0.014)	(0.014)	(0.014)	(0.003)	(0.002)	(0.001)	(0.003)
Lag age (minor) version				0.134^a	0.236^a	-0.006^b	0.166^a
				(0.012)	(0.017)	(0.003)	(0.015)
Lag age major version						0.046^a	-0.028^a
						(0.004)	(0.004)
Lag average age (minor) version				-1.282^a	-0.008	-0.105^b	-1.224^a
				(0.155)	(0.007)	(0.046)	(0.161)
Lag average age major version						-0.248^a	-0.682^a
						(0.070)	(0.176)
Tests of hypothesis							
F-stat lagged growth (gdl_{t-1})	2075.3^a	1.4e+05^a	1686.14^a	30.99^a	1.8e+05^a	22.73^a	22.73^a
F-stat Update (u_t)††	177.5^a	48.57^a		598.09^a	377.85^a		
F-stat Major update (u_{1t})††			6.45^a			483.5^a	483.5^a
F-stat Minor update (u_{2t})††			114.64^a			426.7^a	426.7^a
Under identif. F-stat	636.2^a	312.2^a	99.83^a	69.75^a	639.6^a	76.32^a	76.32^a
Weak identif. F-stat	407.5^a	98.71^a	27.14^a	26.45^a	348.0^a	48.02^a	48.01^a
Over identif. J-test	3.803	6.687	9.814	1.635	4.926	1.303	3.610
Observations	3,660	2,956	3,556	3,660	2,956	3,530	3,530
Uncentered R-squared	0.058	0.019	0.064	0.139	0.066	0.266	0.140

†Notes: all regressions are in first difference and include period dummy variables. Standard errors are in parenthesis and clustered by app. Superscripts a, b, c indicate parameters or test which are significant at 1%, 5% and 10%, respectively. ††The variable in the update equation is lagged. Instruments: 1) Multi-app developers instruments (first difference of the lagged variables: gdl (columns 1-4), Dver (columns 1 and 2), and Dminver and Dmajver (columns 3 and 4)); 2) Two-period lags for: gdl (columns 1-4), Dver (columns 1 and 2), and Dminver and Dmajver (columns 3 and 4).

4. Downloads of Google Play apps are negatively affected by the presence of the in-app purchase option, while they are not affected on iTunes. Possibly, the first result can be interpreted as the classical negative effect of price on demand. This observation is reinforced by the fact that almost all Google Play apps in our sample are free and the in-app purchase option represents the only form of pricing.

5. On iTunes, free apps are characterized by a larger growth rate of downloads.[15]

The estimates of the second equation of the system (5), about the determinants of updates, are documented in columns (4) and (5) of Table 3. We have chosen to model the binary variable *update* as a linear probability model, facing the limitations of the methodology, but aware that the issues related to linear probability models are less severe than imposing a parametric assumption that may not hold, and lead to a misspecified model. The theoretical model developed in Section 3 suggests that the main determinant of the developer's decision to release a new version of their app is the past performance, here measured in terms of growth in downloads, gdl_{t-1}.

We find that the past performance does affect (mildly) the decision to release an update only in iTunes. In this platform, the coefficient of the variable gdl_{t-1} is negative and weakly statistically significant, meaning that, other things equal, a poor performance of the app in the previous period makes it more likely for the developer to release an update. This amounts to saying that our data supports Conjecture 1, with app update being an effective strategy to react to downturns in downloads. The same does not occur when considering Google Play where past performance does not have an impact on the decision to release an update. This difference between iTunes and Google Play can, again, be reinterpreted on the basis of their different way of governing the release of updates. As mentioned, iTunes apps must pass a quite severe quality control and this limits the freedom of developers to publish updates. As a consequence, developers who have written an update and are ready to publish it on the store may decide to delay publication until when they really need it, when the performance of the app is poor and triggers a response to counter the drop in downloads. On the contrary, in Google Play developers can publish apps any time they want; at the very moment an update is ready, they can make it available for download with a simple click of the mouse. In this environment, updates are continuously published, thus diluting the impact of past performance on the decision to update.

From columns (4) and (5) other (minor) observations follow:

1. On both platforms, apps that have been updated in the previous period are more likely to be updated today. This suggests a form of persistency in the decision to release new versions of the apps;

2. Apps with in-app purchases are more likely to be updated. This evidence is consistent with Conjecture 4 and occurs on both platforms. As discussed above, we interpret this result on the basis that developers of apps with in-app purchases have more to gain from stimulating buzz and improving visibility.

[15]The dummy variable $free$ is omitted for Google Play, provided that nearly all apps are free of charge.

3. The number of other apps by the same developer listed in the top 1,000 during the same month and in the same country positively affects the likelihood to update apps in iTunes while it is not statistically significant on Google Play. This partially confirms Conjecture 2. As discussed above, estimates of the growth equation of system (5) suggest the presence of complementarities among apps, both on iTunes and Google Play. According to Conjecture 2 this implies that the probability of updating an app should increase with the number of other apps of the same developer.[16]

Concluding this section, it is important to discuss an implicit assumption we have made when computing the estimates of the second equation in (5). In testing Conjecture 1, we have considered the past performance experienced by the app in the five countries included in our sample (measured in terms of the growth rate during the previous months); hence, we have implicitly assumed that those five are the reference countries for the developers' decisions. However, it might be the case that developers base their strategies by looking at the performance in a wider set of countries or in a set of countries different from that composing our sample.[17] As a matter of fact, when published in the store an update becomes available in all countries worldwide; hence, the decision to release a new version of the software might be based on the world-wide performance of the app. Alternatively, a developer might decide to update the app on the basis of the growth rate experienced in a set of countries different from our ones (e.g. a US-based developer might take her decisions by looking at the performance in the USA or in North America). In both cases, by considering France, Germany, Italy, Spain and the UK we would not control for the right set of countries determining the decision about whether to make an update.

In order to tackle this issue, we employ a useful information provided in the Priori dataset, namely whether an app is local or not. An app is defined as local in a given country when at least 40% of its all time downloads occurs in that country. For local apps the performance in the five countries composing our sample is very likely to represent the basis developers use in order to take their decisions. Therefore, we re-estimated the second equation of the system (5) by restricting the sample only to local apps. The results of this estimation are given in Table 4 and largely confirm our previous findings.

6.1 Zooming-in on iTunes: Major *vs* minor updates

For iTunes apps we collected additional information about the type of updates by distinguishing major updates (i.e. significant changes in functionalities) from minor ones (i.e. bugs fixing and minor changes to the software code). This allows us to investigate more closely the versioning strategy followed by developers. We have re-estimated the two equations of the system (Equation 5) by considering *update major* and *update minor* separately.[18] The

[16]The theoretical model only takes into account complementarity/subsititutabilty looking at the demand side, but complementarity/substitutability can also emerge on the production side. For example, a developer may re-use the code written for another app (complementarity).

[17]By contrast, it is quite natural to estimate the growth equation using the total downloads.

[18]Notice that the second equation of the system must be estimated separately for minor and major updates.

Table 4: First difference update equation for local apps†

	Update equation u_t			
	iTunes	GP	iTunes major	iTunes minor
	(1)	(2)	(3)	(4)
Lag growth (gdl_{t-1})	-0.035^b	-0.009	0.004	-0.037^b
	(0.015)	0.017	(0.005)	(0.015)
Update (u_{t-1})	0.107^b	0.224^a		
	(0.044)	(0.056)		
Major update ($u_{1,t-1}$)			0.043	-0.519^a
			(0.045)	(0.123)
Minor update ($u_{2,t-1}$)			0.003	0.200^a
			(0.014)	(0.047)
In-app	0.785^a	0.445	0.122	0.570^b
	(0.109)	(0.275)	(0.139)	(0.270)
Number of apps	0.021	-0.008	0.004	0.025^c
by developer	(0.014)	(0.011)	(0.005)	(0.015)
Lag age			0.037^a	-0.031^a
major version			(0.004)	(0.006)
Lag age	0.108^a	0.208^a	-0.005	0.158^a
(minor) version	(0.015)	(0.026)	(0.005)	(0.021)
Lag average age			-0.145	-1.261^a
major version			(0.128)	(0.330)
Lag average age	-1.781^a	-0.026	-0.005	-1.631^a
(minor) version	(0.302)	(0.026)	(0.005)	(0.295)
	Tests of hypothesis			
F-stat lagged growth (gdl_{t-1})	907.9^a	183.0^a	607.7^a	607.7^a
Update (u_{t-1})	274.1^a	187.9^a		
Major update ($u_{1,t-1}$)			228.5^a	228.5^a
Minor update ($u_{2,t-1}$)			177.1^a	177.1^a
Under identif. p-value	285.4^a	273.6^a	282.0^a	282.0^a
Weak identif. p-value	337.5^a	200.0^a	187.9^a	187.9^a
Over identif. p-value	4.845^c	1.478	4.773	1.051
Observations	1,233	1,068	1,194	1,194
Uncentered R-squared	0.139	0.045	0.255	0.106

† all regressions are in first difference and include time period dummy variables. Standard errors are in parenthesis and clustered by app. Superscripts a, b, c indicate parameters or test statistics which are significant at 1%, 5% and 10%, respectively. Instruments: 1) Multi-app developers instruments (first difference of the lagged variables: gdl (columns 1-4), Dver (columns 1 and 2), and Dminver and Dmajver (columns 3 and 4)); 2) Two-period lags for: gdl (columns 1-4), Dver (columns 1 and 2), and Dminver and Dmajver (columns 3 and 4).

results are displayed in columns (3) and (6-7) of Table 3.

Estimates of the growth equation confirm the results obtained in the general estimation, where we have not distinguished between major and minor updates (column (1) of Table 3). Interestingly, major updates have stronger impact on the growth rate of downloads than minor ones (the coefficient of *update major* is nearly twice as large as the coefficient of *update minor*). This evidence suggests that adding new and significant functionalities to an application is more effective in stimulating downloads than bug fixing and minor adjustments.

As far as the second equation of the system is concerned, the results obtained in the general estimation are confirmed, particularly with respect to minor updates. Although statistically significant, the coefficient of the lagged growth rate has an extremely small magnitude in the regression on the determinants of major updates (columns 6). On top of this, the coefficient of the variable *number of apps by developer* is not significant anymore. These findings reveal that the strategic use of updates described by the theoretical model is particularly suited to minor updates. This result can be easily interpreted if one considers the different nature of major and minor updates; while minor updates can be developed with a certain ease, major ones require much more development effort and time. This implies that only minor updates can be used strategically in reaction to poor performances or to exploit cross-app effects. On the contrary, major updates are inherently less suitable for these purposes.

7 Conclusions

App developers frequently release new versions of their mobile applications. In this study, we have found evidence that such updates have a strategic flavour as they are used by developers as a tool to increase the buzz surrounding their apps, in an attempt to improve users' engagement, and increase or maintain high app visibility. Our empirical analysis has shown that updates play a quite different role in stimulating downloads on iTunes and Google Play stores. On iTunes, updates trigger further growth in the number of downloads; by contrast, on Google Play their effect is not significant. This result is consistent with the prediction of our theoretical model which suggests that the lack of quality control by Google Play can lead to "excessive updating": developers release both high and low quality updates, which, on average, do not impact on downloads. Another interesting result that we have come across is that multi-app developers can exploit the complementarity among their apps.

The study has also investigated the determinants of updates. In line with the predictions of our theoretical model, we find that in iTunes developers are more likely to release an update when their app experienced a decline in the performance. On the contrary, in Google Play, the past performance of the app has no impact on the decision to release a new version of the software. Again, we interpret these opposite results obtained for iTunes and Google Play on the basis of the different way of governing the release of updates in the two stores. Moreover, we have also found a certain degree of persistence in the update behavior. In addition, our estimates have shown that apps with the in-app purchase option are more frequently updated.

The article has also provided insights on the comparison between major and minor updates occurring on iTunes. The main finding is that major updates have a stronger impact on growth than minor ones. We interpret this result as evidence that adding significant new functionalities to an application contributes to growth in downloads more than routine-based bug fixing or minor improvements. We also find that a poor past performance of the app increases the chances that the developer releases a minor update but has no impact on major updates. We interpret this result as evidence that only minor updates are used as strategic response to poor past performance.

We view our results as a first step at understanding the strategic use of updates by app developers. One aspect that deserves to be better addressed is the direct link between app updates and app visibility. Should more detailed data on user reviews and on the flow of users become available, one can build a comprehensive visibility index to investigate further on this interesting issue.

References

Bresnahan, T., Davis, J., and Ying, P.-L. (2014). Economic value creation in mobile applications. Forthcoming in The Changing Frontier: Rethinking Science and Innovation Policy, Jaffe A. and B. Jones eds, University of Chicago Press.

Carare, O. (2012). The impact of bestseller rank on demand: Evidence from the app market. *International Economic Review*, 53(3):717–742.

Datta, D. and Sangaralingam, K. (2013). Do app launch times impact their subsequent commercial success? 2013 International Conference on Cloud Computing and Big Data.

Garg, R. and Telang, R. (2014). Estimating app demand from publicly available data. School of Information Systems and Management, Heinz College Carnegie Mellon University.

Ghose, A. and Han, S. P. (2014). Estimanting demand for mobile applications in the new economy. Forthcoming in Management Science.

Greenbaum, J. (2005). This is just in: Consumers hate their software vendors. *Intelligent Enterprise, San Mateo*, 8(10) 14.

Ifrach, B. and Johari, R. (2014). The impact of visibility on demand in the market for mobile apps. Available at SSRN: http://ssrn.com/abstract=2444542.

Priori (2014). Global insights report - Android platform & iOS platform. February 2014.

Sankaranarayanan, R. (2007). Innovation and the durable goods monopolist: The optimality of frequent new-version releases. *Marketing Science*, 26(6):774–791.