Technology Fragmentation, Platform Investment, and Complementary Innovation[*]

Grace Gu
Carroll School of Management, Boston College, Chestnut Hill, Massachusetts 02467
grace.gu@bc.edu

Zhuoxin Li
Carroll School of Management, Boston College, Chestnut Hill, Massachusetts 02467
zhuoxin.li@bc.edu

[*] We thank Gord Burtch, Rob Fichman, Jerry Kane, Do Yoon Kim, Marios Kokkodis, Sam Ransbotham, Mike Teodorescu, Feng Zhu for helpful feedback. Li acknowledges funding support from National Science Foundation CAREER Award Grant.

Technology Fragmentation, Platform Investment, and Complementary Innovation

Abstract

Complementor innovation is an essential form of value co-creation in open platform ecosystems. However, the increasing platform fragmentation, i.e., users in the ecosystem adopting different versions of the platform technologies, has significantly hindered complementor innovation. For instance, Android devices currently in the market run a dozen old and new versions of the Android operating system, which increases the cost of complementor innovation because app developers must exhaustively test on different versions of the operating system when they release new apps or updates. Reducing platform fragmentation is a complex coordination problem involving several parties in the ecosystem, thus it is unclear whether and how a platform's efforts to fight fragmentation would affect complementor innovation. Focusing on recent efforts by Google to address Android fragmentation, we find that while the platform investment does not immediately reduce Android fragmentation, app developers respond to the platform's action by significantly increasing their innovation efforts in updating apps shortly after platform investment. We find support of two possible explanations of the positive platform investment effect: anticipated lower cost structure and higher platform value. App developers' forward-looking behavior is likely due to developers' anticipation of lowered innovation cost in the future, as the impact of platform investment is greater for developers with more variable cost structure such as a larger and more diverse user base, less development experience and lack of economies of scale (i.e., with a smaller portfolio of apps). The effect is also greater for developers in more competitive markets where gaining user base can be costly due to market competition. Our research highlights the role of platform commitment to improve platform infrastructure in complementor innovation and provides implications for platform investment and intervention.

Keywords: *Multi-sided platforms; fragmentation; innovation; mobile apps; Android*

1. Introduction

Complementor innovation is an integral part of value co-creation on many open platforms (e.g., (Van Alstyne et al. 2016; Boudreau 2010; Ceccagnoli et al. 2012; Gawer and Cusumano 2002; Song et al. 2018; Wen and Zhu 2019). However, increasing platform fragmentation, i.e., divided users in the ecosystem adopting different versions of the platform technologies, is a major hurdle for complementor innovation. For instance, Android devices currently in the market run a dozen old and new versions of the Android operating system, forcing app developers to exhaustively test on different versions of the operating system when they release new apps or updates. Despite the increasing prominence of platform fragmentation and all the hurdles stem from fragmentation, platform owners know little about how to reduce fragmentation and stimulate complementor innovation.

Fragmentation has been a long-lasting issue for many platform ecosystems. For platform ecosystems such as operating systems (e.g., Windows and Android) and enterprise software (e.g., MySQL), old versions of the systems often continue to exist on a large number of devices long after the release of new versions. For instance, Microsoft's Windows XP was first released in 2001 but continued to run in a large number of devices until 2019, ten years after Microsoft introduced Windows 7 and discontinued support for Windows XP in 2009.[1] The Android ecosystem is also heavily fragmented: the one billion Android devices run more than 10 major versions of the Android operating system.[2] The fragmentation problem creates unprecedented challenges for app developers to release new apps or updates because they must exhaustively test updates on different versions of the operating system to ensure compatibility. Therefore, on the Android platform, fragmentation is one of the biggest hurdles for app developers.[3]

Due to the complex coordination problem among different parties, fighting fragmentation is often a long-term commitment and is unlikely to achieve significant effects in the short or medium run. However, although platform infrastructure is unlikely to be immediately improved in the short run, platforms may still be worthwhile to invest in and commit to fighting fragmentation as a first step to stimulate complementor innovation in the multi-party

[1] https://www.extremetech.com/computing/289440-microsoft-xp-is-finally-dead-nearly-18-years-post-launch
[2] https://www.pcmag.com/news/welcome-to-the-fragmentation-party-android-10
[3] https://searchcloudcomputing.techtarget.com/blog/Head-in-the-Clouds-SaaS-PaaS-and-Cloud-Strategy/Android-fragmentation-An-app-developers-worst-nightmare

coordination game (Anderson et al. 2014). Since platform owners do not have full control of the entire ecosystem, they often need to take the lead in growing the ecosystem and incentivize complementor innovation (Wu et al. 2020). For instance, platform owners can invest in first-party innovation or improve platform infrastructure to pave the way for complementors to innovate in the ecosystem (Hagiu and Spulber 2013). Facing increasing fragmentation in their ecosystem, platform owners can also invest in platform infrastructure to reduce fragmentation and signal their commitment to boost complementor innovation. However, the literature has provided little guidance on what platform type of platform investment can be effective, and how such investment drives complementor innovation. Platform investment to reduce fragmentation can drive complementor through different mechanisms. First, reducing fragmentation can help lower the expected cost for complementors to innovate in the ecosystem, especially for small complementors that do not have the resources or economies of scale to develop their own infrastructure to deal with fragmentation. Second, platform investment may boost complementors' expectation on the value of the platform for complementors, which trigger their innovation to create and capture value in the ecosystem. Therefore, this research tests these two possible mechanisms about anticipated cost structure and perceived expected platform value.

To understand whether and how platform's efforts to reduce platform fragmentation affects complementary innovation, our study focuses on the recent efforts by Google to address Android fragmentation. In 2017, Google launched the Treble project, which adds a hardware abstraction layer between the Android Operating System (OS) and the hardware (e.g., chipset). With this vendor interface, the update of Android OS can be done independently by device manufacturers, without having to modify the codes related to hardware such as chips. Google expected the rollout of the Treble project to lower device manufacturers' cost to abandon old Android versions and upgrade to the latest ones, and therefore reduce Android fragmentation. The Treble project is among the first attempts of major platforms to invest in platform infrastructure design to fight fragmentation, which provides an apt empirical setting to study the impact of such platform investment on complementary innovation.

We construct a comprehensive longitudinal dataset about Android fragmentation and app innovation before and after the Treble project. The Android fragmentation data consists of information about the version distribution of the Android operating system, whereas the app activity log records the update history of each app. Our empirical analysis estimates the impact of the Treble project on complementor innovation by comparing two groups of complementors based on the differences in their likelihood of being affected by the Treble project. While most apps are affected by the Treble project, graphic apps that use the OpenGL Library are less affected by the Treble project as the library provides an app-level abstraction layer, which already shields these apps from fragmentation. Also, we leverage the geographical regions in Android fragmentation and estimate the impact of the Treble project by comparing regions of high and low Android fragmentation.

Our findings support the two possible mechanisms of platform investment about *lower anticipated cost structure* and *perceived expected platform value*. Empirical results show that, although the platform investment does not reduce Android fragmentation right away, app developers swiftly respond to the platform's efforts by significantly increasing their innovation efforts. In other words, although platform investment does not significantly improve the platform infrastructure in the short run, the commitment to do so can effectively boost complementary innovation as developers *anticipate* positive changes in the ecosystem (Schatzel and Calantone 2006). This effect is likely due to developers' anticipation to lowered innovation cost in the future, as the impact of platform investment is greater for developers with more variable cost structure such as a larger and more diverse user base, less development experience, and lack of economies of scale (i.e., with a smaller portfolio of apps). The effect is also greater for developers in more competitive markets.

Our research highlights the role of platform commitment to improve platform infrastructure in complementor innovation (Anderson et al. 2014; Hagiu and Spulber 2013). We find evidence that complementors are forward looking—they may enhance their innovation efforts even before the platform investment has materially improved the platform infrastructure. Also, the value of platform investment goes beyond its effects of reducing complementors' cost to innovate. Platforms' commitment to improve its infrastructure can enhance complementors' perceived value of the ecosystem in the long run and change the perceived market dynamics in the ecosystem. This research also provides managerial implications for platform investment and intervention. Some complementors are more responsive than others, depending on various supply-side and demand-side factors (e.g., complementors' cost structure and the competitive dynamics in the submarket). These findings can help platform owners identify which types of complementors are likely to respond to platform investment and design incentives to coordinate their efforts accordingly.

2. Theoretical Background

2.1 Related Literature

Open Platforms and Platform Fragmentation Platform owners can open up their ecosystems to cultivate positive network effects and encourage complementary innovation. For instance, opening up the platform to complementors may encourage broad participation, which can increase the volume, variety, and quality of offerings on the platform (Boudreau 2010, 2012). Alternatively, platform owners may restrict access to their platforms, forgoing some of the benefits of broad participation for better control of the platform (Van Alstyne et al. 2016; Gawer and Cusumano 2002).

Compared to proprietary platforms (e.g., iOS), open platforms (e.g., Android) may not only speed up user adoption but also create the fragmentation problem (Yoon 2014). Prior studies have documented the existence of the fragmentation problem and propose techniques for app developers to combat fragmentation. For example, Han et al. (2012) is among the first to study the compatibility issues in the Android ecosystem and provided evidence of fragmentation. More broadly, fragmentation also exists for technological adoption under open standards (Chen and Forman 2006). Researchers have also proposed techniques to help developers prioritize Android devices for development and testing by mining user reviews and usage data (Khalid et al. 2014; Lu et al. 2016). Fragmentation and the changing Android APIs across versions can affect the quality (e.g., portability and compatibility) and development efforts of Android apps (Linares-Vásquez et al. 2013; McDonnell et al. 2013), as well as the competitive pressure that developers face (Wu et al. 2020). Although these pioneer works have shed light on Android fragmentation and the techniques for developers to deal with fragmentation, little is known about how platform investment can help combat the fragmentation problem. Understanding the role of platform investment is important as most developers still rely on platform-level efforts to combat fragmentation—they do not have the experience and the economies of scale to build their own technology infrastructure to deal with fragmentation.

Platform Commitment and Investment Commitment to the growth of the ecosystem is an essential governance problem for platforms that exhibit network effects (Hagiu and Spulber

2013). For instance, platforms must periodically decide what level of platform performance to invest in to sustain their user base and encourage complementor innovation (Anderson et al. 2014). Such a platform investment problem is a complex issue due to coordination difficulties among all parties, including the platform owner (e.g., Google), device manufacturers (e.g., Samsung, ZTE, and Xiaomi), and other complementors (e.g., app developers). Value creation on the platforms requires collective efforts from all parties (Ceccagnoli et al. 2012). To break the deadlock in the coordination game, platform owners can seed the markets and commit to exert continuing efforts (Hagiu and Spulber 2013), and therefore set the expectations for complementors to follow up and exert efforts (Hossain and Morgan 2009; Shapiro and Varian 1999). Managing participants' expectations is crucial in multi-sided platforms as such expectations can become self-fulfilling in the presence of network effects (Boudreau 2021; Shapiro and Varian 1999).

2.2 Platform Investment, Complementor Expectation, and Innovation

Platform investment has the potential to reduce fragmentation, and the reduction in fragmentation may then boost complementary innovation. However, due to the complex coordination problem, platform investment is often a long-term commitment and is unlikely to significantly improve the platform infrastructure in the short or medium run. Nevertheless, platform investment can boost complementary innovation in the short run as developers *anticipate* positive changes in the ecosystem (Schatzel and Calantone 2006). For complementors, platform investment can reveal considerable information about the platform owner's strategy and future intentions (Chellappa and Mukherjee 2021; Huang et al. 2018). Platform investment can boost complementor's anticipation of the prospect of the platform ecosystem (Schatzel and Calantone 2006), triggering positive or negative responses from complementors to pursue first-mover advantages (Banbury and Mitchell 1995; Kerin et al. 1992).

Not all complementors may equally respond to platform investment due to their differential beliefs and various supply/demand side factors such as cost structure and market dynamics. Complementors may positively respond to platform investment because of their positive expectations in the value of the ecosystem (Boudreau 2021), i.e., complementors' innovation efforts are guided not only by a platform's value but also by expectations of future value (Shapiro and Varian 1999). Platform investment may change the cost structure of innovations in the ecosystem and disproportionately affect complementors innovation (Anderson et al. 2014; He et al. 2020). We discuss these heterogeneous effects in the rest of this section.

2.2.1 Anticipated Changes in Developers' Cost Structure

Platform investment is expected to lower developers' costs of developing a new app or releasing an update because with reduced fragmentation, developers only deal with a smaller number of versions of the platform technologies (e.g., Android operating system). Developers who anticipate the decrease in future development and maintenance costs may boost their innovation efforts. Platform investment can disproportionately benefit certain types of apps and developers, i.e., developers with apps that attract a more diverse user base (and thus are more subject to fragmentation), with a small portfolio of apps (lack of economies of scale to build their own infrastructure), or with less experience on the platform. These apps and developers may benefit more from platform investment to reduce fragmentation.

Economies of Scale The effects of platform investment are likely to vary across different developers due to their cost structure. Developers' cost structure varies substantially depending on economies of scale. For instance, big developers that offer a large number of apps can afford to build their own costly infrastructure to deal with the fragmentation problem. These big developers likely benefit less from platform investment to reduce platform fragmentation (Wei et al. 2018). Small developers that lack the economies of scale, however, may not afford the high fixed costs to build their own infrastructure and thus may benefit more from reduced platform fragmentation.

Developers' Tenure and Experience Experienced developers are more likely to have accumulated abundant knowledge and have the expertise to build their own infrastructure to deal with fragmentation. New developers on the platform, however, may not have the expertise to cope with platform fragmentation by themselves. The lack of experience and cumulative investment suggests that inexperienced developers are more subjective to platform fragmentation, and thus can be more responsive to platform investment to reduce fragmentation.

Diversity of User Base Popular apps that serve a large and diverse user base are often more affected by platform fragmentation because of the diversity of OS versions installed on the users' devices. For instance, apps in the social media category (e.g., Facebook) are more likely to attract users of high diversity in age, sex, race, and ethnicity. Apps that attract a homogeneous segment of users, however, are less subject to the fragmentation problem as their users tend to be more similar (e.g., holding a similar set of devices of similar OS versions).

2.2.2 Anticipated Changes in Platform Value Creation and Capture

Platform investment has the potential to reshape the ecosystem. In addition to anticipated change in developers' costs of innovation, platform investment can increase developers' expectations on the value of the platform and change the competitive dynamics on the platform. Complementors may respond to platform investment by adjusting their innovation efforts as they anticipate potential changes in market dynamics following the platform investment (Ceccagnoli et al. 2012; Huang et al. 2018).

Reduced fragmentation can increase the competitiveness of the market as it lowers the barrier of entry and the cost of innovation. Therefore, developers in a more competitive market are more likely to increase innovation efforts after platform investment. Being more responsive can help developers gain greater market share and enhance the chance of survival (Banbury and Mitchell 1995). Such a first-mover advantage applies to the Android platform in which users incur moderate switching costs (Kerin et al. 1992).

In the rest of the paper, we provide empirical evidence on the two potential mechanisms discussed above: the effects of platform investment on complementor innovation are moderated by developers' anticipated changes in their cost structure as well as the anticipated changes in platform value creation and capture.

3. Empirical Setting

3.1 Android Fragmentation

To study fragmentation as a key issue that hinders complementor innovation, we choose the setting of Android, the open-sourced mobile operating system developed by Google. Android has the largest market share among all smartphone platforms worldwide and has long suffered from the issue of fragmentation.

The variety of makers, including Original Equipment Manufacturers (OEMs) and device manufacturers, create a substantial fragmentation issue in the Android ecosystem as these manufacturers can freely adopt their preferred version of the operating systems. A large number of newer and older Android OS versions are running in more than one billion devices, which results in a much more substantial fragmentation issue if compared to the iOS platform. For instance, iOS 11 was released in September 2017 and was already running on 65% of iOS devices by January 2018. In contrast, Android Oreo was released in August 2017 but was running on just about 1% of Android devices by the same time (Appendix A.1), whereas early versions were still running in the majority of Android devices: the market share for Nougat released in August 2016 was 29%, Marshmallow released in October 2015 was 28%, Lollipop released in November 2014 was 25%, and the market share for all early versions released prior to 2014 was 17% combined.

The fragmentation issue is due to the complex coordination problem in the Android ecosystem. Device manufactures' slow upgrade to new Android versions is the critical antecedent of Android fragmentation. The fragmentation problem affects every player in the ecosystem. In particular, fragmentation substantially increases the cost for app developers to release new apps or updates because the changes would need to be tested across a variety of devices running different versions of the Android operating systems. The fragmentation problem also creates substantial challenges to deploy security updates, making Android devices vulnerable to security breaches.

3.2 Fighting Fragmentation with Hardware Abstraction Layer: The Treble Project

To fight the fragmentation problem, Google launched the Treble project in 2017 to pave the way for fast Android upgrades. The key idea of the project is to add a *hardware abstraction layer* between the Android OS and the hardware (e.g., chipset). As illustrated in Figure A2 in Appendix A, the decompiling of the Android OS from hardware implementation provides device manufacturers more freedom to choose the most recent Android versions while transferring the responsibilities of hardware implementation to component vendors such as chip manufacturers. With this vendor interface, the update of Android OS can be done independently as device manufacturers do not have to modify the codes related to the hardware. The Treble project has the potential to make it faster, easier, and cheaper for device manufacturers to adopt Android OS updates and get them out to users. The Treble project was completed and rolled out to the market in December 2017.

The Treble project can be considered as Google's commitment to fighting the fragmentation issue in its Android ecosystem. This is among the first attempts of major platforms to invest in the platform infrastructure design to fight fragmentation. The Treble project offers a valuable opportunity to study the impact platform investment on complementor innovation.

Due to the complex coordination problem, the Treble project is unlikely to significantly reduce fragmentation in the short or medium run, suggesting that fighting fragmentation is going to be an enduring process with high uncertainty of eventual success (Dowell and Swaminathan 2006). However, app developers may consider the Treble project as a positive signal, and thus respond to the project despite high uncertainty and long before the project can significantly reduce OS fragmentation. For instance, app developers may boost their innovation efforts responding to the Treble project. With such positive responses, the Treble project's intended effects can be accelerated, reducing the lead time required to achieve its goals.

4. Research Design, Data and Model

4.1 Identifying Treatment Effects based on App-Level Abstraction Layer: OpenGL

To estimate the effect of the Treble project on complementor innovation, we leverage a technical design in the *app-level abstraction layer*, i.e., the Open Graphics Library (OpenGL), which is an industrial standard adopted by most graphic apps with the aim of fixing hardware/OS compatibility issues but at the same time also shields these apps from fragmentation. Apps that use OpenGL are not affected or are less affected by Android fragmentation and thus are not influenced by platform investment to reduce fragmentation (these apps can be considered the "control group"), whereas apps that do not use OpenGL are affected by the Treble project and thus can be considered the "treatment group."

Appendix A.3 provides detailed discussion on what types of apps rely on OpenGL and the release history of the OpenGL Library for Android. OpenGL has been managed by the non-profit technology consortium Khronos Group, which is independent of Android. Also, there was no new release of OpenGL for Android during the period of 2017-2018 covered in this paper (the most recent release was in August 2015). Because the release and update of OpenGL are independent of Google's Android platform, the variations among apps in whether they use OpenGL allow us to estimate the impact of the Treble project by comparing the innovation efforts by the two groups of app developers.

As an alternative empirical design, we also leverage the geographical variations of fragmentation among app developers to identify the treatment and control groups in the analysis.

We estimate the impact of the Treble project by comparing regions of high and low Android fragmentation. Section 4.4 provides the details on how we construct the treatment variable.

In addition, to corroborate the main empirical design, we conduct further analyses comparing the innovation of the same developer or the same app across Android vs. iOS platform, to control for the unobservables between the treatment and control groups (Section 5.2).

4.2 Data Sources

We combine two data sources to construct a comprehensive dataset about Android fragmentation and app innovation. Data on developer/app innovation is collected from the Google Play store, the official marketplace for Android app developers to release/update their apps and for users to download these apps. This longitudinal dataset contains the profile and activity log for all developers and all apps from each developer from 2014 to 2020. The app profile includes the app's release date, category, file size, text description, developer, cumulative downloads, average star rating, number of libraries, etc. The app activity log records the update history of an app, including the timestamp and version number.

Data on Android fragmentation is collected from the version distribution dashboard on Google's Android developer page. The distribution dashboard reported the distribution of Android versions running in all active Android devices that have accessed the Google Play store in the past month. We supplement this platform-level data with more granular country-level version distribution data from AppBrain, an Android app analytics company. AppBrain offers one of the most popular software developer kits (SDK), with tens of thousands of developers worldwide using the AppBrain SDK for their Android app. AppBrain anonymously processes the server traffic of their SDK to generate version distributions of Android devices, globally and by country, from over 100 million monthly unique users.

4.3 Variables and Measurement

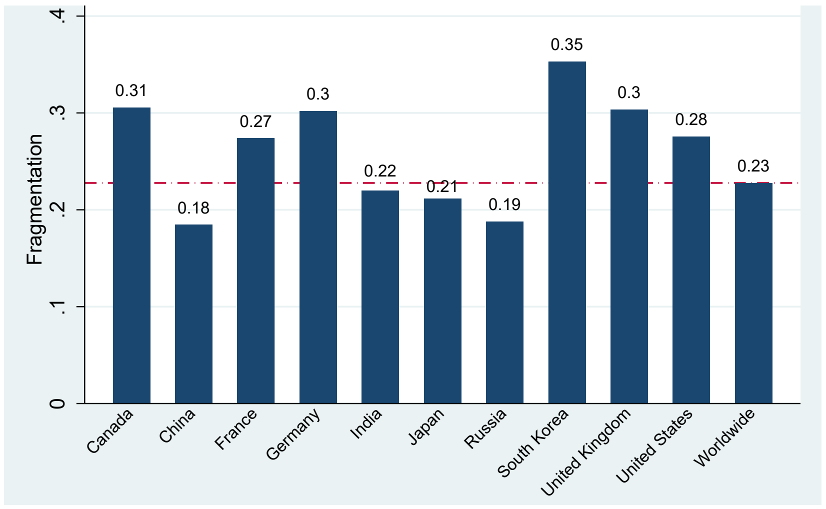
Innovation Efforts As the key outcome measure of the study, a developer's innovation efforts are measured by their app updates (Boudreau 2012; Wen and Zhu 2019). Hence, the key dependent variable for capturing the developer's innovation efforts is *Num_Updates*, which is the total number of updates of an app in a month (Foerderer 2020). We also separate the number of updates into major updates (*Num_Major_Updates*) and minor updates (*Num_Minor_Updates*). Major updates typically refer to substantial changes to an app, such as adding new features and functionalities. Minor updates are small enhancements, such as bug fixes and performance improvement. The widely adopted approach to distinguish major and minor updates is by comparing the current version number to the previous version number (Boudreau 2010; Tiwana 2015; Wen and Zhu 2019). An update is considered major if the leading part of the version number increases to a larger number (e.g., the version number changes from 3.0 to 4.0), whereas the update is considered minor if only the rest of the version number has changed (e.g., the version number changes from 3.0 to 3.1).

Fragmentation The degree of fragmentation of Android operating systems can be measured by the Simpson's Diversity index, which has been widely adopted to quantify the diversity of an ecosystem (Harrison and Klein 2007; Simpson 1949). The index considers the number of Android versions present, as well as the market share of each version. Specifically, the fragmentation index based on the Simpson's index at time *t* is

$$Fragmentation_{rt} = 1 - \sum_{v=0}^{n} p_{rvt}^2,$$

where p_{rvt} is the market share of Android version *v* at time *t* in region *r*. The larger the Simpson's index, the more fragmented the Android platform is. In the OpenGL analysis, we treat the entire Android platform as a region and compute the platform-wise Simpson index. In the geography analysis, we treat each county as a separate region and compute region-specific Simpson index. Figure 1 illustrates the Android fragmentation levels for ten representative countries with the largest number of Android developers and the overall fragmentation level on the entire platform in 2018. We can see rich geographical variations across countries in Android fragmentation.

Figure 1. Geographical Variations in Android Fragmentation



Platform Investment—The Treble Project The Treble project was rolled out in late December 2017. Therefore, we create a binary variable *Post* to capture the timing of the investment, with *Post* equal to 0 for December 2017 and before, and *Post* equal to 1 after.[4]

Moderating Variables The effect of fragmentation on innovation may vary across different developers and apps. For instance, big and experienced developers may be better equipped with knowledge and tool to deal with fragmentation. Big developers with a larger portfolio of apps may also benefit from economies of scale, i.e., they can afford to invest in their own infrastructure to cope with fragmentation. Therefore, big developers may benefit less from platform investment compared to small developers. Hence, we construct the moderating variable *Big_Developer*, a dummy variable that equals 1 if the developer's total number of apps is above the median in the distribution of developer total app counts, and 0 otherwise.

Also, developers with shorter tenure on the platform may be more likely to move early compared to longer tenure developers on the platform, so we construct another variable *Experienced_Developer*, which indicates whether the developer's years on the platform since its first app is above or below the median among all developers.

The characteristics of an app may also influence an app's response to platform investment. For instance, apps of high popularity are more likely to suffer from the fragmentation issue due to their large and diverse user base. Therefore, these apps are more likely to benefit from the Treble project that reduces fragmentation. Hence, we construct another variable *Popular_App*, which equals 1 (or 0) if the app's number of rating counts is above (or below) the median.[5]

Market characteristics, such as market concentration, reflect the competitive environment and may also affect complementors' decision to innovate responding to the Treble project. The competitiveness of the market can be operationalized by the concentration of the market defined by similar apps that support the same Android OS version. We use the widely adopted the Herfindahl–Hirschman Index (HHI)—a smaller number of the index indicates higher market

[4] According to Google researchers' report (Yim et al., 2019), the Treble project was tested out during the release cycle of Android Oreo 8.0 and 8.1. During this cycle, the team completed experimentation with functions such as modularization. We choose the end of this release cycle as the treatment date.

[5] Alternatively, we can measure an app's popularity using the app's user base. Unfortunately, such data is not available (Google only disclosures the bucket of cumulative downloads such as 10,000~50,000 but not the exam number of downloads). As a robustness check, we consider the bucket of the number of downloads of an app as a measure of app popularity.

concentration and thus a more competitive market, whereas a large number of the index indicates lower concentration and thus more a competitive market. We construct the dummy variable *Concentrated_Mkt*, which equals 1 if the HHI for the app's available Android OS segment is above the median among all Android OS versions' markets, and 0 otherwise. As a robustness check, we also construct an alternative HHI measure based on the developers' number of app ratings to proxy for the developers' market share for each Android version. The empirical results remain qualitatively unchanged.

We also create a measure to capture user base diversity. The diversity of the user base determines to what extent fragmentation would affect the complementors. We capture the level of user base diversity by app category using the variable *Diverse_Category*, which equals 1 if the number of various Android versions developed for apps in a category is above the median in the entire distribution of categories, and equals 0 otherwise.

4.4 Empirical Model

To measure the effect of platform investment, we construct two groups of apps: the "control" group of apps that are not affected (or are only slightly affected) by the platform investment to reduce fragmentation and the "treatment" group of apps that are affected by platform investment.

We construct a *Treated* variable, which equals 1 for an app in the "treatment" group and 0 otherwise. We use two different approaches to construct the "control" and "treatment" groups. The first approach leverages app-level variations in their tendency to be affected by the platform investment, whereas the second approach leverages geographical variations in fragmentation.

App-Level Variations in the Use of OpenGL The platform investment does not affect all apps equally. App developers in the graphics/video categories typically use the industry-standard OpenGL to create cross-language, cross-platform applications. For OpenGL analysis, the "control" group consists of apps that leverage OpenGL as an abstraction layer and thus are not (or are only slightly) affected by platform investment to reduce fragmentation, and the "treatment" group consists of apps that do not OpenGL as an abstraction layer and thus are responsive to fragmentation reduction. Matching of apps in these two groups is done at the app level, based on covariates including the app's first release date, app complexity (i.e., the number of libraries used), user ratings (i.e., the average star rating and the number ratings). Appendix B provides the details on how we construct these two groups of apps. For OpenGL analysis, *Treated* equals 1 if an app does not rely on OpenGL and 0 otherwise.

Geographical Variations in Android Fragmentation We also leverage geographical variations in platform fragmentation across locations (i.e., regional markets defined by countries) to identify the differences in the treatment levels. Regional markets with higher fragmentation may benefit more from the platform investment to reduce fragmentation. We therefore construct two groups of projects: the "control" group of app developers that serve regional markets of low fragmentation and the "treatment" group of developers that serve regional markets of high fragmentation. Matching is at the developer level, based on covariates including the developer's tenure on the Android platform, the number of apps offered by the developer, and developer ratings (i.e., average star rating and the total number of ratings across all apps by the developer).

The degree of Android fragmentation is continuous across countries, with the Simpson's index ranging from 0.1 to 0.4. The middle chunk of the distribution is dense, so the 50% cutoff (i.e., the median) would not be suitable for the fragmentation level data. Following prior studies in statistics and econometrics, we equally divide this range into three buckets such that each bucket has the same number of countries (Farronato et al. 2020; Gelman and Park 2009). There, we assign the binary variable *Treated* to be 0 for developers in countries with the fragmentation index from 0.1 to 0.2 (this corresponds to about one-third of developers facing with the least fragmentation), and 1 for developers in countries with fragmentation from 0.3 to 0.4 (this corresponds to about one-third of developers facing the most fragmentation). We conduct robustness checks with alternative cutoffs (0.1~0.25 and 0.25~0.4; 0.1~0.15 and 0.35~0.4). The results remain qualitatively unchanged.

The platform investment is exogenous to the *Treated* variable as either app-level variations or geographical variations are independent of the Treble project. We estimate the main effect of the platform investment using the following linear model:

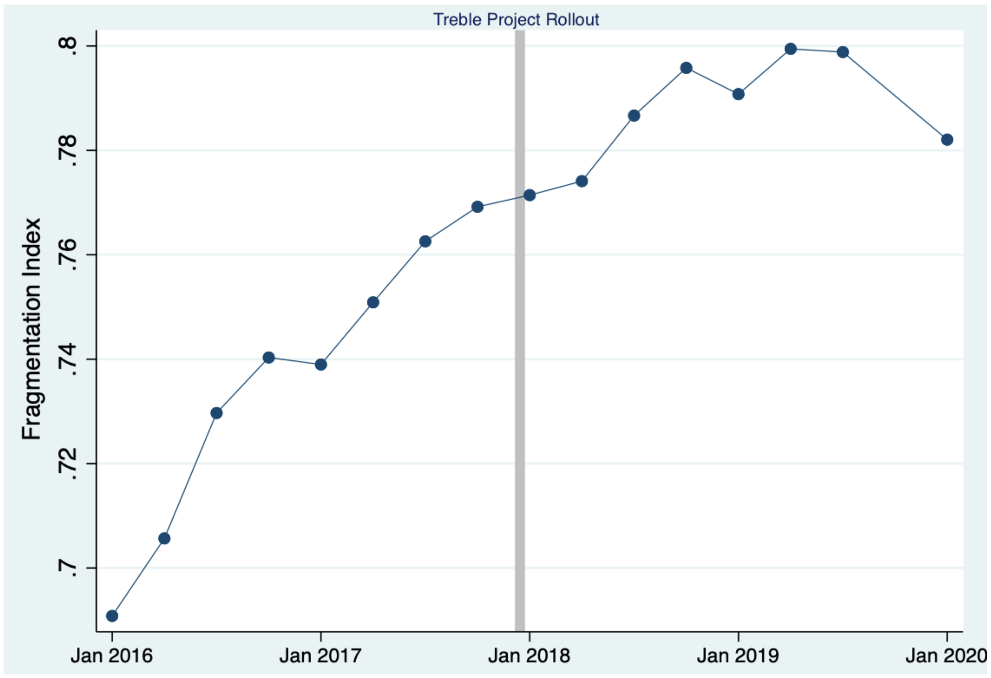
$$Log(Num\_Updates)_{ij} = \beta_0 + \beta_1 Treated_i + \beta_2 Post_t + \beta_3 Treated_i \times Post_t + \varepsilon_{it}. \quad (1)$$

The unit of analysis is at the app-month level. The linear model is appealing for its interpretability. For app i in month t, the estimated coefficient for the interaction term, β_3, captures the differences in the impacts of the platform investment on app innovation, which can be interpreted as the percentage change in update frequency. Since the dependent variable (i.e., the number of updates) is an integer variable, we also use count models as robustness checks (estimates from the count models cannot be interpreted as straightforward as the linear model).

4.5 Descriptive Evidence of Complementor Anticipation

We first investigate whether the Treble project reduces fragmentation in the Android ecosystem in the short run. Figure 2 illustrates the platform-wise fragmentation of Android, from January 2016 (two years prior to the Treble project) to January 2020 (two years after the Treble project). As we expected, the Treble project did *not* mitigate platform fragmentation until the middle of 2019, possibly due to the slow response from device manufacturers.[6]

Figure 2. Android Fragmentation over Time



However, despite the lack of short-term changes in fragmentation, the Treble project has an immediate effect on app developers' innovation efforts. As we proposed in the theoretical development (Section 2), platform investment such as the Treble project signals the platform's commitment to reducing fragmentation, which boosts complementors' confidence in exerting their own efforts. Developers may anticipate lowered cost to develop and maintain their apps and the changing competitive dynamics in the ecosystem. Hence, we may still observe changes in the

[6] Compared to app developers, device manufacturers face more constraints in new product releases. Popular brands often follow a regular timeline of annual product releases. For example, to release a new product around September and October.

developers' innovation efforts shortly after the Treble project's rollout. The matching algorithms in Appendix B construct a balanced sample between the treatment and control group observations. For the OpenGL analysis, the affected and matched apps are balanced across a set of covariates (Table B1 in Appendix B) in the 6 months before the Treble project takes effect. For the geography analysis, the affected and matched developers are also balanced on a set of developer characteristics (Table B2 in Appendix B).

Based on the matched pairs, our data sample contains app-month level observations for the OpenGL analysis, and developer-month level observations for the geography analysis. Table 1 reports the summary statistics.

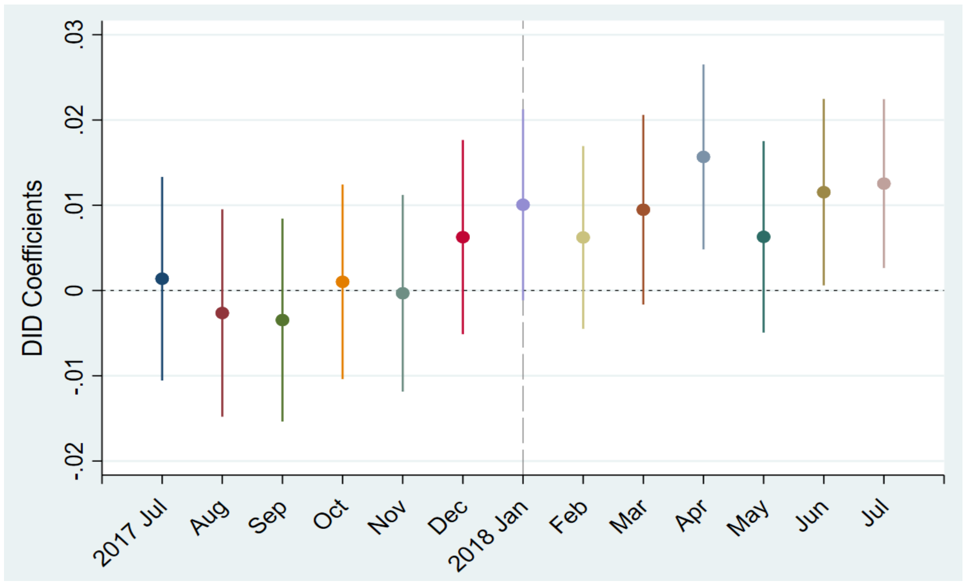
Table 1. Summary Statistics

Variable	Mean	Std. dev.	Min	Max
For OpenGL Analysis:				
Treated	0.501	0.500	0	1
Post	0.539	0.499	0	1
Log(Num_Updates)	0.143	0.317	0	2.639
Log(Num_Major_Updates)	0.024	0.129	0	1.609
Log(Num_Minor_Updates)	0.121	0.296	0	2.639
Big_Developer	0.625	0.484	0	1
Experienced_Developer	0.598	0.490	0	1
Popular_App	0.441	0.497	0	1
Concentrated_Mkt	0.442	0.497	0	1
For Geography Analysis				
Treated	0.495	0.500	0	1
Post	0.557	0.497	0	1
Log(Num_Updates)	0.165	0.340	0	2.197
Log(Num_Major_Updates)	0.029	0.141	0	1.609
Log(Num_Minor_Updates)	0.140	0.317	0	2.197
Big_Developer	0.499	0.500	0	1
Experienced_Developer	0.361	0.480	0	1
Popular_App	0.475	0.499	0	1
Concentrated_Mkt	0.440	0.496	0	1

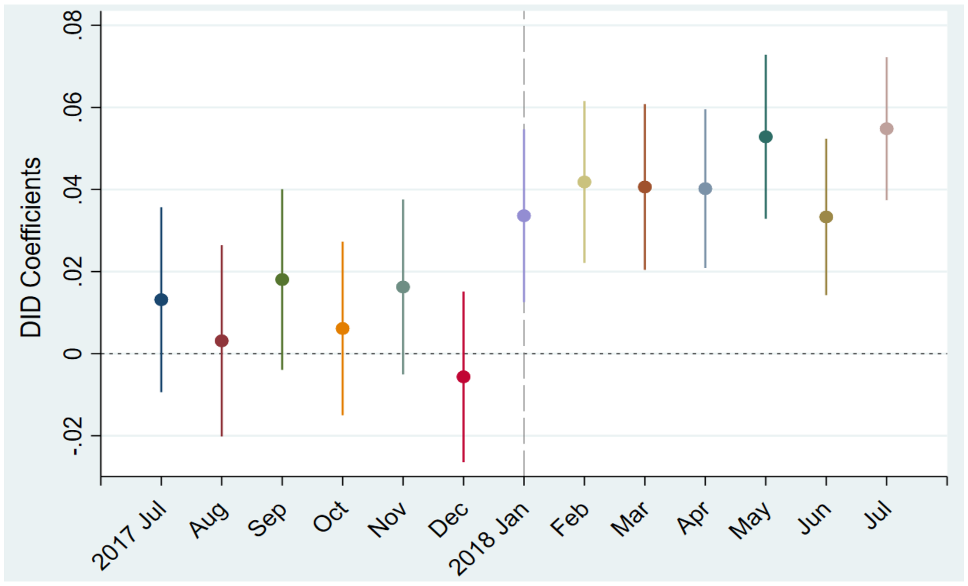
Notes. The unit of analysis is at the app-month level. Number of observations in this table is 158,255 for the OpenGL analysis, and 54,531 for the geography analysis.

Based on the matched groups, as a preliminary step, we first examine the changes in app update frequency by comparing the monthly number of updates over time for the treated and matched groups. Figure 3 shows that, in the pre-treatment period, there are no substantial differences in update frequency between these two groups of apps. After the Treble project rolls out, however, our OpenGL analysis (Figure 3a) and geography analysis (Figure 3b) both show clear trends that app developers introduce more updates.

Figure 3. Coefficients over Months with OpenGL as Treatment



(a) Based on OpenGL Analysis



(b) Based on Geography Analysis

These increases in developer's innovation efforts shortly after Treble project's rollout is a contrast to the previous evidence in Figure 3, which illustrates that Treble project does not immediate reduce fragmentation in the Android platform in the short run. In fact, the app developers' responses to the Treble project in the short run is *not* much driven by the actual reduction in fragmentation, but possibly more by app developers' anticipation of future changes in the Android ecosystem after the Treble project. In other words, app developers are forward looking; the Treble project boosts app developers' confidence in innovation, and hence we observe more efforts devoted to app updates on the complementor's side that follow the platform investment. In the next section, we report the precise estimates of these effects using the difference-in-differences (DID) models.

5. Empirical Results

5.1 Evidence of Complementor Innovation Increase After Platform Investment

Table 2 reports the main effects estimation based on Equation (1). We can see that in the time period following the platform's investment to reduce fragmentation, app developers significantly increase their innovation efforts. The results are robust in both the OpenGL analysis and geography analysis, where we observe that developers update their apps 1% to 3.4% more frequently following the platform investment.

Table 2. The Effect of Platform Investment on Innovation

Model	(1)	(2)
	OpenGL Analysis	
Dependent Variable	Log(Num_Updates)	Log(Num_Updates)
Treated	0.0004	0.0004
	(0.002)	(0.053)
Post	-0.023***	-0.043***
	(0.002)	(0.001)
Treated × Post	0.010***	0.010***
	(0.003)	(0.002)
Observations	158,255	158,255
R-squared	0.001	0.001
Month FE	No	Yes

(a) Based on OpenGL Analysis

Model	(1)	(2)
	Geography Analysis	
Dependent Variable	Log(Num_Updates)	Log(Num_Updates)
Treated	0.008*	0.008*
	(0.005)	(0.004)
Post	-0.037***	-0.064***
	(0.004)	(0.002)
Treated × Post	0.034***	0.034***
	(0.006)	(0.004)
Observations	54,531	54,531
R-squared	0.003	0.003
Month FE	No	Yes

(b) Based on Geography Analysis

Notes. The unit of analysis is at the app-month level. Number of observations in this table is 158,255 for the OpenGL analysis, and 54,531 for the geography analysis. Variables are defined in Section 4.3. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

5.2 Additional Evidence: Comparing Innovation on Android vs. iOS Platforms

To corroborate the main effects, we also examine the effect of The Treble project on complementary innovation by comparing app updates of the same developer or the same app between the Android vs. iOS platforms. By comparing the same developer across platforms, we identify a treatment and control "twin pair" where the developer's innovation on the iOS platform serves as the counterfactual estimate to its innovation on the Android platform, except for that Project Treble did not take place in the counterfactual scenario. This is to minimize the potential confounding effects caused by unobservables between the treatment and control groups.

Specifically, we identify 7,079 matched developers on both Android and iOS platforms, with 10,710 matched apps that have update records on both Android and iOS platforms. We compare these apps' updates between the two platforms before vs. after the Treble Project, both at the developer level and at the app level.

Table 3 Panel (a) reports the main effects estimation by comparing the same apps across platforms, and Table 3 Panel (b) reports the main effects estimation by comparing the same developers across platforms, based on Equation (1). As in previous main analyses, we can see that in the time period following the platform's investment to reduce fragmentation, app

developers significantly increase their innovation efforts on Android compared to that on the iOS platform. The results are robust in both the app-level analysis and the developer-level analysis, where we observe that developers update their apps about 3.8% to 5.1% more frequently following the platform investment.

Table 3. The Effect of Platform Investment on Innovation (Android vs. iOS)

Model	(1)	(2)
	App Level Analysis	
Dependent Variable	Log(Num_Updates)	Log(Num_Updates)
Treated	-0.028***	-0.028***
	(0.002)	(0.002)
Post	-0.010***	-0.045***
	(0.003)	(0.002)
Treated × Post	0.038***	0.038***
	(0.003)	(0.002)
Observations	239,168	239,168
R-squared	0.001	0.001
App FE	No	Yes

(a) Comparing same apps across platforms

Model	(1)	(2)
	Developer Level Analysis	
Dependent Variable	Log(Num_Updates)	Log(Num_Updates)
Treated	-0.040***	-0.040***
	(0.004)	(0.003)
Post	-0.021***	-0.049***
	(0.004)	(0.003)
Treated × Post	0.051***	0.051***
	(0.005)	(0.003)
Observations	158,604	158,604
R-squared	0.001	0.003
Developer FE	No	Yes

(b) Comparing same developers across platforms

Notes. Robust standard errors in parentheses.
*Significant at 10%; ** at 5%; *** at 1%.

5.3 Heterogeneous Effects in Complementor Innovation

Why do we observe an increase in complementor innovation when the Treble project does not immediately reduce fragmentation in the Android platform? Following our theoretical development in Section 2, we test whether the cost structure of the complementors and market competitiveness can explain the differences in complementors' choice to innovate early after the platform's investment. In the rest of this paper, we report the empirical results from the OpenGL analysis in the main text and put the findings from the geography analysis in Appendix C (the findings are qualitatively the same as those from the OpenGL analysis).

Size of a Developer's App Portfolio (Economies of Scale). First, the size of a developer's app portfolio indicates the developer's cost of developing apps and can affect their decision to increase innovation efforts following platform investment. Results in Table 4 Column (1) show that developers who have developed more apps in the past (developers offering a larger portfolio of apps) tend to increase their innovation less compared to smaller developers after the platform investment. This result could be due to economies of scale. Specifically, big developers can afford to build their own infrastructure to deal with the fragmentation problem themselves (testing environment for multiple OS versions and devices, etc.). Such infrastructure can be used across multiple apps and lower the cost of innovation, which small developers are unable to do. Therefore, small developers benefit more from the platform investment potentially due to their lack of the internal infrastructure, thus the smaller developers are more likely to take advantage of the platform's investment to reduce fragmentation compared to larger developers.

Developer's Tenure and Experience. Similarly, more experienced developers may also have better infrastructure to lower development costs compared to new developers, whereas new developers may likely anticipate to take advantage of the reduce in fragmentation to lower cost. Table 4 Column (2) reports the estimated coefficients for the heterogeneous effects regarding a developer's tenure on the platform. New developers can be more responsive and are more likely to leverage the Treble project to improve their apps after the platform investment. Experienced developers already develop the internal capability to deal with fragmentation, and thus are less responsive to platform investment to reduce fragmentation.

App Popularity and Diversity of User Base. At the app level, developing a popular app can be costly as it needs to cater to a larger and more diverse user base than a less popular app. This may also affect the developer's decision to increase innovation with anticipated fragmentation reduction. We can see from Table 4 Column (3) that popular apps have a greater magnitude of update increase compared to less popular apps when fragmentation is reduced. As expected, the plausible explanation is that apps of high popularity are more likely to suffer from the fragmentation issue due to their large user base and the variety of users they serve. Therefore, these apps are more likely to benefit from the platform investment to reduce fragmentation, thus the developers are more likely to decide to move early and increase innovation efforts when they anticipate a lower cost in future.

Market Competitiveness. Lastly, developers in a more competitive market may tend to innovate more than those in less competitive markets, as it may be more costly for them to attract the users who will soon update to newer versions of the Android platform. Table 4 Column (4) shows that in Android markets with greater concentration, i.e., less competition, developers are less likely to increase their innovation compared to those in more competitive markets after the Treble project. This result could be due to developer incentives to gain network effects. Reduced fragmentation enables users to update their Android OS systems which opens up a new user segment to complementors. When competition exists in a market subject to network effects, complementors move early to attract the user base in the initial stage of competition to gain network effects. Therefore, platform investment to reduce fragmentation incentivizes complementors to increase innovation.

In sum, the forward-looking increase in complementor innovation that we find in the main effect is likely due to developers' anticipation of lowered innovation cost in the future, as the impact of platform investment is greater for developers with more variable cost structure such as a larger and more diverse user base, less development experience, and lack of economies of scale (i.e., with a smaller portfolio of apps). The effect is also greater for developers in more competitive markets where gaining user base can be costly due to market competition.

Table 4. Heterogeneity in Innovation Efforts

Model	(1)	(2)	(3)	(4)
Dependent Variable	Log(Num_Updates)			
Treated	-0.001	0.003	0.005*	-0.004
	(0.005)	(0.005)	(0.003)	(0.004)
Post	-0.027***	-0.043***	-0.020***	-0.027***
	(0.004)	(0.004)	(0.002)	(0.003)

Treated × Post	0.019*** (0.006)	0.017*** (0.006)	0.003 (0.003)	0.017*** (0.005)
Big_Developer	-0.117*** (0.004)			
Treated × Big_Developer	0.005 (0.005)			
Post × Big_Developer	0.005 (0.005)			
Treated × Post × Big_Developer	-0.014** (0.007)			
Experienced_Developer		-0.098*** (0.004)		
Treated × Experienced_Developer		-0.002 (0.005)		
Post × Experienced_Developer		0.029*** (0.005)		
Treated × Post × Experienced_Developer		-0.012* (0.007)		
Popular_App			0.135*** (0.004)	
Treated × Popular_App			-0.011** (0.005)	
Post × Popular_App			-0.008* (0.005)	
Treated × Post × Popular_App			0.014** (0.007)	
Concentrated_Mkt				-0.061*** (0.003)
Treated × Concentrated_Mkt				0.007 (0.005)
Post × Concentrated_Mkt				0.008* (0.004)
Treated × Post × Concentrated_Mkt				-0.016** (0.006)
Observations	158,255	158,255	158,255	158,255
R-squared	0.032	0.019	0.042	0.009

Notes. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

6. Robustness Checks

We conduct several robustness checks to ensure that the findings are robust across alternative measures of innovations by app developers. We also rule out alternative explanations.

Major vs. Minor Updates Repeating the main analysis using the dependent variables of the apps' major vs. minor updates, results in Table 5 show that the affected developers tend to make both more major and more minor innovations after fragmentation is reduced.

Table 5. The Effect of Platform Investment on Major vs. Minor Innovations

Model	(1)	(2)	(3)	(4)
	OpenGL Analysis			
Dependent Variable	Log(Num_Major _Updates)	Log(Num_Major _Updates)	Log(Num_Minor _Updates)	Log(Num_Minor _Updates)
Treated	0.0004	0.0004	-0.0004	-0.0004
	(0.0011)	(0.0009)	(0.0022)	(0.0015)
Post	-0.0141***	-0.0180***	-0.0111***	-0.0284***
	(0.0009)	(0.0007)	(0.0021)	(0.0011)
Treated × Post	0.0026**	0.0026*	0.0079***	0.0079***
	(0.0013)	(0.0014)	(0.0030)	(0.0022)
Observations	158,255	158,255	158,255	158,255
R-squared	0.0025	0.0025	0.0002	0.0002
Month FE	No	Yes	No	Yes

Notes. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

New OS Releases One might be concerned that the release of a new OS in 2017 that accompanies the rollout of the Treble project may confound our estimated main effects. Since the new OS release affects apps in both the treatment and control groups, its effect should have been controlled for in the DID analysis if the effect is similar across the two groups.

Considering the situation if the new OS release has different effects on each group, we conduct a robustness check by excluding from our sample any apps that ever upgraded to the new OS release during the observed time period and re-run the main analysis. From Table 6, we can see that all main findings continue to hold: both the signs and magnitudes of the parameter estimates are comparable to those of the base models in Table 2.

Table 6. Parameter Estimates Excluding New OS Releases

Model	(1)	(2)
	OpenGL Analysis	
Dependent Variable	Log(Num_Updates)	Log(Num_Updates)
Treated	0.0002	0.0002
	(0.002)	(0.053)

Post	-0.023***	-0.043***
	(0.002)	(0.01)
Treated × Post	0.010***	0.010***
	(0.003)	(0.002)
Observations	157,931	157,931
R-squared	0.001	0.001
Month FE	No	Yes

Notes. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

Count Model Specification To further corroborate the main results, because our outcome variable, the number of app updates, is a count variable with nonnegative values, instead of taking the natural logarithm of the dependent variable to correct for the skewness as we did for the main analysis, an alternative model specification is to employ a negative binomial regression which accounts for nonnegative over-dispersed count data to test the robustness of the effects. Table 7 presents the estimated results using the alternative model specification of negative binomial regression models. Using the same main analysis samples, all the main findings continue to hold.

Table 7. Parameter Estimates with Count Models

Model	(1)	(2)
	OpenGL Analysis	
Dependent Variable	Num_Updates	Num_Updates
Treated	-0.006	0.006
	(0.017)	(0.017)
Post	-0.186***	-0.207***
	(0.017)	(0.029)
Treated × Post	0.078***	0.076***
	(0.024)	(0.023)
Observations	158,255	158,255
R-squared	0.001	0.001
Month FE	No	Yes

Notes. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

Alternative Measures To test the robustness of our moderating effects, we also calculate alternative measures to conduct the analyses. The empirical results (Table 8) remain qualitatively unchanged compared to those in Table 4.

First, for *app popularity*, alternatively, we can measure an app's popularity using the app's user base. Unfortunately, such data is not available (Google only discloses the bucket of cumulative downloads such as 10,000~50,000 but not the exact number of downloads). As a robustness check, we consider the bucket of cumulative downloads of an app as a measure of app popularity. Our results remain qualitatively unchanged. Second, when measuring *market concentration*, instead of computing HHI based on developers' number of apps for each minimum supported Android version, we also construct an alternative HHI based on the developers' number of app ratings to proxy for the developers' market share for each Android version. Using this alternative measure for market concentration, we find the effects continue to hold. Lastly, we find that, in app categories where the user base is more diverse, developers are more likely to increase their innovation following the completion of the Treble project.

Table 8. Heterogeneity in Innovation Efforts with Alternative Measures

Model	(1)	(2)	(3)
Dependent Variable		Log(Num_Updates)	
Treated	0.004	-0.004	-0.005
	(0.004)	(0.004)	(0.005)
Post	-0.013***	-0.027***	-0.016***
	(0.003)	(0.003)	(0.005)
Treated × Post	0.002	0.017***	0.001
	(0.005)	(0.005)	(0.006)
Popular_App2	0.120***		
	(0.003)		
Treated × Popular_App2	-0.007		
	(0.005)		
Post × Popular_App2	-0.013***		
	(0.004)		
Treated × Post × Popular_App2	0.010*		
	(0.006)		
Concentrated_Mkt2		-0.061***	
		(0.003)	
Treated × Concentrated_Mkt2		0.007	
		(0.005)	
Post × Concentrated_Mkt2		0.008*	
		(0.004)	
Treated × Post × Concentrated_Mkt2		-0.016**	
		(0.006)	
Diverse_Category			-0.038***
			(0.004)

Treated × Diverse_Category			0.0002
			(0.006)
Post × Diverse_Category			-0.010*
			(0.006)
Treated × Post × Diverse_Category			0.015**
			(0.007)
Observations	158,255	158,255	158,255
R-squared	0.018	0.009	0.004

Notes. The sample is the same as the main analysis in Table 3 based on OpenGL analysis. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

7. Discussion and Conclusions

This research provides insights into the impact of platform investment to reduce fragmentation on complementary innovation. Due to the complex coordination issue among all parties, platform investment is often a long-term commitment and is unlikely to significantly change the ecosystem in the short run. However, we find evidence of that app developers are forward looking. Developers may enhance their innovation efforts even before the platform investment has reduced fragmentation and their cost to innovate. The platform owner's commitment to investment in the platform can enhance developers' perceived value of the ecosystem in the long run and change the market dynamics in the ecosystem. This research contributes to the literature by highlighting the broader impact of platform investment on complementary innovation.

7.1 Theoretical Implications

This research provides theoretical insights into how platform investment to improve platform infrastructure influences complementary innovation. Such an investment often would not be able to significantly reshape the platform infrastructure immediately (e.g., reducing platform fragmentation) due to the complex coordination issues involving different types of participants (e.g., device manufacturers and app developers). However, this research shows that some complementors may quickly respond to platform investment as they anticipate possible changes in cost structure and market dynamics in the long run, despite the substantial uncertainty about whether the platform investment will eventually improve the platform infrastructure. This research provides empirical evidence on the existence of such complementor behavior and how it differs across apps and developers of different ages, economies of scale, user base diversity, and the competitive environments the complementors operate in. This research highlights the role of complementors' anticipation in their innovation efforts in platform ecosystems.

7.2 Practical Implications

This research informs platform participants about the short-/medium-term effects of platform investment to fight fragmentation. We find that such efforts may not reduce fragmentation in the short run due to the complex coordination problem among all parties involved in the process. However, platform owners should not be discouraged by the enduring fragmentation problem in the short run after their investment. App developers do perceive such platform investment as valuable and increase their innovation efforts following the platform investment. Ignoring such positive responses from app developers may lead to a problematic conclusion that platform investment would not create a positive effect.

Platform investment to reduce fragmentation has heterogeneous effects on different apps and developers. Our findings have implications for targeted interventions. The effects are stronger for apps with a larger and more diverse user base and apps in a more competitive category, and for less experienced developers and developers without economies of scale (i.e., with a smaller portfolio of apps). Platform owners can reach out to these apps and developers as they are more responsive to platform investment.

7.3 Future Research

This research focuses on the short-/medium-term effects of platform investment on app developers' innovation. We do not consider the long-term effects of platform investments because of the technical challenges in measuring such effects. In the long run, device manufacturers may also respond to platform investment. Measuring the responses by device manufacturers can be very challenging because device manufacturers are constrained by other factors when they release new devices. For instance, device manufacturers have their own product development cycles, and thus we may not observe manufacturer responses in the middle of the product release cycle.

References

Van Alstyne, M. W., Parker, G. G., and Choudary, S. P. 2016. "Pipelines, Platforms, and the New Rules of Strategy," *Harvard Business Review* (94:4), pp. 54–62.

Anderson, E. G., Parker, G. G., and Tan, B. 2014. "Platform Performance Investment in the Presence of Network Externalities," *Information Systems Research* (25:1), pp. 152–172.

Banbury, C. M., and Mitchell, W. 1995. "The Effect of Introducing Important Incremental Innovations on Market Share and Business Survival," *Strategic Management Journal* (16:S1), pp. 161–182.

Boudreau, K. 2010. "Open Platform Strategies and Innovation: Granting Access vs. Devolving Control," *Management Science* (56:10), pp. 1849–1872.

Boudreau, K. J. 2012. "Let a Thousand Flowers Bloom? An Early Look at Large Numbers of Software App Developers and Patterns of Innovation," *Organization Science* (23:5), pp. 1409–1427.

Boudreau, K. J. 2021. "Promoting Platform Takeoff and Self-Fulfilling Expectations: Field Experimental Evidence," *Management Science*.

Ceccagnoli, M., Forman, C., Huang, P., and Wu, D. J. 2012. "Cocreation of Value in a Platform Ecosystem: The Case of Enterprise Software," *MIS Quarterly* (36:1), pp. 263–290.

Chellappa, R. K., and Mukherjee, R. 2021. "Platform Preannouncement Strategies: The Strategic Role of Information in Two-Sided Markets Competition," *Management Science* (67:3), pp. 1527–1545.

Chen, P., and Forman, C. 2006. "Can Vendors Influence Switching Costs and Compatibility in an Environment with Open Standards? " *MIS Quarterly* 30, pp. 541–562.

Dowell, G., and Swaminathan, A. 2006. "Entry Timing, Exploration, and Firm Survival in the Early US Bicycle Industry," *Strategic Management Journal* (27:12), pp. 1159–1182.

Farronato, C., Fong, J., and Fradkin, A. 2020. "Dog Eat Dog: Measuring Network Effects Using a Digital Platform Merger," *NBER Working Paper* (28047).

Foerderer, J. 2020. "Interfirm Exchange and Innovation in Platform Ecosystems: Evidence from Apple's Worldwide Developers Conference," *Management Science* (66:10), pp. 4772–4787.

Gawer, A., and Cusumano, M. A. 2002. *Platform Leadership: How Intel, Microsoft, and Cisco Drive Industry Innovation*, (Vol. 5), Harvard Business School Press Boston, MA.

Gelman, A., and Park, D. K. 2009. "Splitting a Predictor at the Upper Quarter or Third and the Lower Quarter or Third," *The American Statistician* (63:1), pp. 1–8.

Hagiu, A., and Spulber, D. 2013. "First-Party Content and Coordination in Two-Sided Markets," *Management Science* (59:4), pp. 933–949.

Han, D., Zhang, C., Fan, X., Hindle, A., Wong, K., and Stroulia, E. 2012. "Understanding Android Fragmentation with Topic Analysis of Vendor-Specific Bugs," in *2012 19th Working Conference on Reverse Engineering*, pp. 83–92.

Harrison, D. A., and Klein, K. J. 2007. "What's the Difference? Diversity Constructs as Separation, Variety, or Disparity in Organizations," *Academy of Management Review* (32:4), pp. 1199–1228.

He, S., Peng, J., Xu, L., Li, J., and Dai, B. 2020. "The Impact of Platform Owner's Entry on Third-Party Stores," *Information Systems Research* (31:4), pp. 1467–1484.

Hossain, T., and Morgan, J. 2009. "The Quest for QWERTY," *American Economic Review*

(99:2), pp. 435–440.

Huang, P., Tafti, A., and Mithas, S. 2018. "Platform Sponsor Investments and User Contributions in Knowledge Communities:: The Role of Knowledge Seeding," *MIS Quarterly* (42:1), pp. 213–240.

Kerin, R. A., Varadarajan, P. R., and Peterson, R. A. 1992. "First-Mover Advantage: A Synthesis, Conceptual Framework, and Research Propositions," *Journal of Marketing* (56:4), pp. 33–52.

Khalid, H., Nagappan, M., Shihab, E., and Hassan, A. E. 2014. "Prioritizing the Devices to Test Your App on: A Case Study of Android Game Apps," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pp. 610–620.

Linares-Vásquez, M., Bavota, G., Bernal-Cárdenas, C., Di Penta, M., Oliveto, R., and Poshyvanyk, D. 2013. "Api Change and Fault Proneness: A Threat to the Success of Android Apps," in *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, pp. 477–487.

Lu, X., Liu, X., Li, H., Xie, T., Mei, Q., Hao, D., Huang, G., and Feng, F. 2016. "PRADA: Prioritizing Android Devices for Apps by Mining Large-Scale Usage Data," in *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*, pp. 3–13.

McDonnell, T., Ray, B., and Kim, M. 2013. "An Empirical Study of Api Stability and Adoption in the Android Ecosystem," in *2013 IEEE International Conference on Software Maintenance*, pp. 70–79.

Schatzel, K., and Calantone, R. 2006. "Creating Market Anticipation: An Exploratory Examination of the Effect of Preannouncement Behavior on a New Product's Launch," *Journal of the Academy of Marketing Science* (34:3), pp. 357–366.

Shapiro, C., and Varian, H. R. 1999. "The Art of Standards Wars," *California Management Review* (41:2), pp. 8–32.

Simpson, E. H. 1949. "Measurement of Diversity," *Nature* (163:4148), p. 688.

Song, P., Xue, L., Rai, A., and Zhang, C. 2018. "The Ecosystem of Software Platform: A Study of Asymmetric Cross-Side Network Effects and Platform Governance," *MIS Quarterly* (42:1), pp. 121–142.

Tiwana, A. 2015. "Evolutionary Competition in Platform Ecosystems," *Information Systems Research* (26:2), pp. 266–281.

Wei, L., Liu, Y., Cheung, S.-C., Huang, H., Lu, X., and Liu, X. 2018. "Understanding and Detecting Fragmentation-Induced Compatibility Issues for Android Apps," *IEEE Transactions on Software Engineering* (46:11), pp. 1176–1199.

Wen, W., and Zhu, F. 2019. "Threat of Platform-Owner Entry and Complementor Responses: Evidence from the Mobile App Market," *Strategic Management Journal* (40:9), pp. 1336–1367.

Wu, X., Kumar, S., and Pang, M.-S. 2020. "Tackling Android Fragmentation: Mobile Apps' Dilemma and the Platform's Strategies," *International Conference on Information Systems*.

Yoon, I. 2014. "Platform Policy and Its Effect on Diffusion: The Case Study of Android and IOS," Massachusetts Institute of Technology.

Appendix A. Industry Background

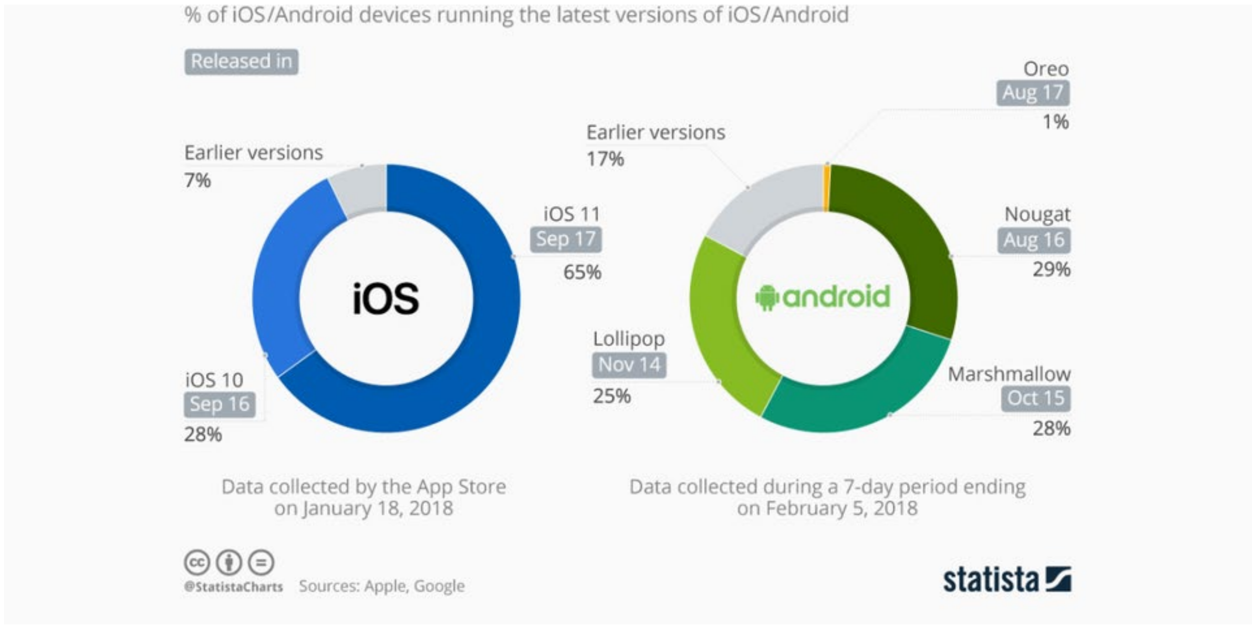
A.1. Android Version History and Fragmentation

Table A1. Android Version History

Name	Version Number	Release Date	API Level
No official codename	1.0	September 23, 2008	1
No official codename	1.1	February 9, 2009	2
Cupcake	1.5	April 27, 2009	3
Donut	1.6	September 15, 2009	4
Eclair	2.0 – 2.1	October 26, 2009	5 – 7
Froyo	2.2 – 2.2.3	May 20, 2010	8
Gingerbread	2.3 – 2.3.7	December 6, 2010	9 – 10
Honeycomb	3.0 – 3.2.6	February 22, 2011	11 – 13
Ice Cream Sandwich	4.0 – 4.0.4	October 18, 2011	14 – 15
Jelly Bean	4.1 – 4.3.1	July 9, 2012	16 – 18
KitKat	4.4 – 4.4.4	October 31, 2013	19 – 20
Lollipop	5.0 – 5.1.1	November 12, 2014	21 – 22
Marshmallow	6.0 – 6.0.1	October 5, 2015	23
Nougat	7.0 – 7.1.2	August 22, 2016	24 – 25
Oreo	8.0 – 8.1	August 21, 2017	26 – 27
Pie	9	August 6, 2018	28
Android 10	10	September 3, 2019	29
Android 11	11	September 8, 2020	30
Cupcake	1.5	April 27, 2009	1

Source. Wikipedia and manually verified by search of news articles.
https://en.wikipedia.org/wiki/Android_version_history

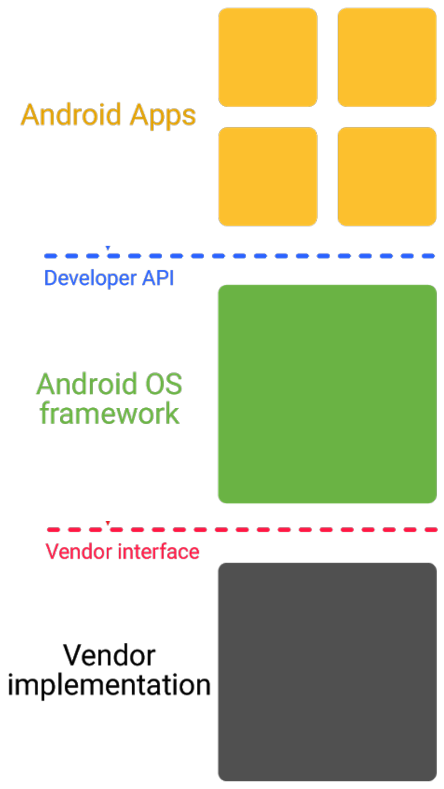
Figure A1. Comparison of Fragmentation: iOS vs. Android



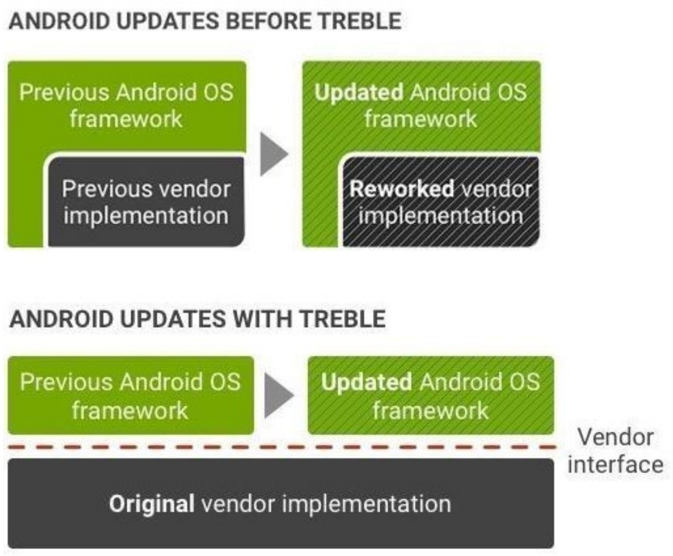
Source. Forbes: https://www.forbes.com/sites/ianmorris/2018/04/13/android-is-still-failing-where-apples-ios-is-winning

A.2. The Treble Project

Figure A2. Comparison of Fragmentation: iOS vs. Android



(a) New Vendor Interface between Android OS and Hardware


ANDROID UPDATES BEFORE TREBLE
Previous Android OS framework
Previous vendor implementation
Updated Android OS framework
Reworked vendor implementation
ANDROID UPDATES WITH TREBLE
Previous Android OS framework
Updated Android OS framework
Vendor interface
Original vendor implementation

(b) Process of Android Updates before and after Treble

A.3. OpenGL Standard

Apps with graphic elements such as video games and graphic design apps are sensitive to hardware (e.g., PC, video game console, or mobile devices) and software (e.g., different operating systems) specification, which leads to compatibility issues arising from different Android versions. To address such compatibility issues, the industry has developed OpenGL, which is a cross-language, cross-platform API for rendering 2D and 3D vector graphics. Specifically, OpenGL can be considered as an abstraction level between an app and the hardware/software that supports the app.

While the use of the OpenGL Library helps handle the issue of compatibility arising from different Android versions, it also makes fragmentation of the Android platform less of a concern to the apps that use OpenGL. From a technical perspective, unlike the Treble project which mainly affects the hardware abstraction layer of the ecosystem, OpenGL mainly affects the app-level abstraction layer. Therefore, with OpenGL as an additional abstraction layer, apps rely more on the consistency and updates of the OpenGL Library rather than the changes in hardware layer fragmentation of the Android ecosystem.

Since 2006, OpenGL has been managed by the non-profit technology consortium Khronos Group, which is separate from Android. Due to this reason, the release and update of OpenGL are independent of Google's Android platform, including the Treble project. During the rollout of the Treble project, the OpenGL Library remained consistent and did not release any updates. Therefore, apps that have already leveraged OpenGL (e.g., graphic related apps) are less affected by the rollout of the Treble project, whereas apps that do not use OpenGL (e.g., non-graphic related apps) are more affected by the Treble project.

The use of OpenGL provides us with differences in the level of "treatment" from fragmentation, i.e., how much an app is affected by fragmentation or the Treble project. Specifically, this distinction enables us to measure the effect of the Treble project by comparing two groups of apps: apps that use OpenGL and thus are not (or only slightly) affected by the Treble project (the "control" group) and those that do not use OpenGL and thus are affected by the Treble project (the "treatment" group), constructed under a matching approach.

Android includes support for high performance 2D and 3D graphics with the Open Graphics Library (OpenGL), specifically, the OpenGL ES API. OpenGL is a cross-platform graphics API that specifies a standard software interface for 3D graphics processing hardware. OpenGL ES is a flavor of the OpenGL specification intended for embedded devices. Android supports several versions of the OpenGL ES API.

Table A2. OpenGL ES Version History

Version Number	Release Date	Support of Android Versions
1.0 and 1.1	July, 2003	Android 1.0 and higher
2.0	March, 2007	Android 2.2 (API level 8 and higher)
3.0	August, 2012	Android 4.3 (API level 18 and higher)
3.1	March, 2014	Android 5.0 (API level 21 and higher)
3.2	August 2015	Android 6.0 (API level 23 and higher)

Source. Android Developer Site: https://developer.android.com/guide/topics/graphics/opengl

Appendix B. Construction of Treated and Matched Group

B.1. Identify Affected Apps and Matched Apps with OpenGL

OpenGL ES (OpenGL for embedded systems) is a library intensively used by graphic related Android apps. To identify apps that leverage OpenGL and those that do not, we conduct code analysis and text analysis of app description. First, we look into the libraries used by an app and determine if the app uses the OpenGL library. If yes, we consider the app as OpenGL supported. Second, we analyze the text description of the app and determine if the description contains "OpenGL" related keywords. These related keywords are compiled from the OpenGL API documentation.[7] Specifically, we use the Python Natural Language Processing toolkit (NLTK) to identify the top 20 keywords in the documentation. After further stemming these keywords, we end up with 17 unique keywords: "graphic photo visual video media map 2d 3d cad game motion model animat simulat render design virtual".

We then apply similar text analysis to identify if an app's description contains some of the OpenGL related keywords to determine if the app is associated with OpenGL. We try different thresholds on the number of matched keywords and the empirical results are robustness across various thresholds.

[7] https://www.opengl.org//documentation/

B.2. Balance Checks

Table B1. Balance Check of the Treated and Matched Developers' Characteristics

Variables	Treated		Matched		Paired t-test
	Mean	Standard error	Mean	Standard error	t-stats
Developer Years on Android	6.072	0.052	6.067	0.051	0.072
Developer App Count	4.482	0.145	4.449	0.127	0.172
Developer Average Rating	3.967	0.016	3.941	0.016	1.144
Developer Rating Count	6.992	0.045	6.902	0.046	1.391

Notes. The unit of analysis is each developer in the treated and control countries. The number of observations in each group is 1,146. Variables are calculated based on the developers' characteristics before January 2018 when the Treble project was initiated. Developer Rating Count is calculated as the logarithm of the developer's total rating count plus 1. None of the above paired t-test results is significant.

Table B2. Balance Check of the Treated and Matched Apps' Characteristics

Variables	Treated		Matched		Paired t-test
	Mean	Standard error	Mean	Standard error	t-stats
App Years on Android	5.740	0.021	5.718	0.021	0.742
Number of Libraries	9.627	0.098	9.597	0.099	0.218
App Average Rating	3.910	0.008	3.908	0.007	0.196
App Rating Count	5.162	0.019	5.162	0.020	0.001

Notes. The unit of analysis is each app in the treated and control group. The number of observations in each group is 6,702. Variables are calculated based on the apps' characteristics before January 2018 when the Treble project was initiated. *App Rating Count* is calculated as the logarithm of the app's total rating count plus 1. None of the above paired t-test results is significant.

Appendix C. Results from Geography Analysis

Table C1. Heterogeneity in Innovation Efforts

Model	(1)	(2)	(3)	(4)
Dependent Variable		Log(Num_Updates)		
Treated	0.009	0.021***	0.007	-0.0002
	(0.007)	(0.006)	(0.005)	(0.007)
Post	-0.050***	-0.053***	-0.023***	-0.049***
	(0.007)	(0.005)	(0.004)	(0.006)
Treated x Post	0.047***	0.039***	0.020***	0.048***
	(0.010)	(0.008)	(0.006)	(0.008)
Big_Developer	-0.131***			
	(0.006)			
Treated x Big_Developer	-0.004			
	(0.009)			
Post x Big_Developer	0.022***			
	(0.008)			
Treated x Post x Big_Developer	-0.030***			
	(0.012)			
Experienced_Developer		-0.100***		
		(0.006)		
Treated x Experienced_Developer		-0.024***		
		(0.009)		
Post x Experienced_Developer		0.035***		
		(0.008)		
Treated x Post x Experienced_Developer		-0.019*		
		(0.011)		
Popular_App			0.153***	
			(0.006)	
Treated x Popular_App			-0.0001	
			(0.009)	
Post x Popular_App			-0.024***	
			(0.008)	
Treated x Post x Popular_App			0.025**	
			(0.012)	
Concentrated_Mkt				-0.068***
				(0.006)
Treated x Concentrated_Mkt				0.010
				(0.009)
Post x Concentrated_Mkt				0.022***
				(0.008)
Treated x Post x Concentrated_Mkt				-0.031***
				(0.012)
Observations	54,531	54,531	54,531	54,531
R-squared	0.039	0.023	0.049	0.011

Notes. The sample is the same as in Table 3 based on the geography analysis. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

Table C2. Heterogeneity in Innovation Efforts with Alternative Measures

Model	(1)	(2)	(3)
Dependent Variable	Log(Num_Updates)		
Treated	0.017** (0.006)	-0.0002 (0.007)	-0.002 (0.006)
Post	-0.016** (0.006)	-0.049*** (0.006)	-0.036*** (0.006)
Treated × Post	0.019* (0.010)	0.048*** (0.008)	0.034*** (0.008)
Popular_App2	0.133*** (0.006)		
Treated × Popular_App2	-0.009 (0.009)		
Post × Popular_App2	-0.024*** (0.008)		
Treated × Post × Popular_App2	0.017 (0.012)		
Concentrated_Mkt2		-0.068*** (0.006)	
Treated × Concentrated_Mkt2		0.010 (0.009)	
Post × Concentrated_Mkt2		0.022*** (0.008)	
Treated × Post × Concentrated_Mkt2		-0.031*** (0.012)	
Diverse_Category			-0.060*** (0.006)
Treated × Diverse_Category			0.020** (0.009)
Post × Diverse_Category			-0.001 (0.008)
Treated × Post × Diverse_Category			0.001 (0.012)
Observations	54,531	54,531	54,531
R-squared	0.018	0.011	0.004

Notes. The sample is the same as the main analysis in Table 3 based on the geography analysis. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

Table C3. The Effect of Platform Investment on Major vs. Minor Innovations

Model	(1)	(2)	(3)	(4)
	Geography Analysis			
Dependent Variable	Log(Num_Major _Updates)	Log(Num_Major _Updates)	Log(Num_Minor _Updates)	Log(Num_Minor _Updates)
Treated	-0.0032	-0.0032	0.0114***	0.0114***
	(0.0020)	(0.0025)	(0.0042)	(0.0029)
Post	-0.0154***	-0.0237***	-0.0236***	-0.0439***
	(0.0017)	(0.0014)	(0.0037)	(0.0015)
Treated × Post	0.0099***	0.0099***	0.0250***	0.0250***
	(0.0025)	(0.0027)	(0.0055)	(0.0030)
Observations	54,531	54,531	54,531	54,531
R-squared	0.0017	0.0017	0.0023	0.0019
Month FE	No	Yes	No	Yes

Notes. Robust standard errors in parentheses. *Significant at 10%; ** at 5%; *** at 1%.

Table C4. Parameter Estimates Excluding New OS Releases

Model	(1)	(2)
	Geography Analysis	
Dependent Variable	Log(Num_Updates)	Log(Num_Updates)
Treated	0.009**	0.009**
	(0.005)	(0.004)
Post	-0.037***	-0.063***
	(0.004)	(0.002)
Treated x Post	0.034***	0.034***
	(0.006)	(0.004)
Observations	54,449	54,449
R-squared	0.003	0.003
Month FE	No	Yes

Notes. Robust standard errors in parentheses.
*Significant at 10%; ** at 5%; *** at 1%.

Table C5. Parameter Estimates with Count Models

Model	(1)	(2)
	Geography Analysis	
Dependent Variable	Num_Updates	Num_Updates
Treated	0.051*	0.050*
	(0.028)	(0.027)
Post	-0.263***	-0.322***
	(0.028)	(0.047)
Treated × Post	0.230***	0.236***
	(0.038)	(0.037)
Observations	54,531	54,531
R-squared	0.002	0.002
Month FE	No	Yes

Notes. Robust standard errors in parentheses.
*Significant at 10%; ** at 5%; *** at 1%.