Comparing Platform Core Features with Third-Party Complements. Machine-Learning Evidence from Apple iOS.

André Halckenhäußer
University of Mannheim
halckenhaeusser@uni-mannheim.de

Felix Mann
University of Mannheim
hello@felixmann.de

Jens Foerderer
Technical University of Munich
jens.foerderer@tum.de

Philipp Hoffmann
University of Mannheim
hoffmann@uni-mannheim.de

Abstract

Software-based platforms have become omnipresent both in private and professional contexts. Platform owners constantly invest in platform evolution in that they update the technological core and enrich its feature base. The question arises how such platform core feature changes can be compared with third-party complements. We investigate this question in the context of an exploratory machine-learning based case study on Apple's mobile platform iOS. By analyzing the changes to iOS over time and developing an approach using natural language processing, we are able identify functional overlaps between platform core features and complements. Our results suggest that platform core features are indeed functionally related to those of complementors and that the strategy of releasing novel platform core features changes over time. Besides, our approach enables us to assign platform core features to app categories. The analysis of functional overlaps raises relevant implications for research and practice.

1. Introduction

Platform ecosystems have gained significant economic relevance in recent years. In the context of software-based platforms, competition has become a major topic that materializes not only among platforms [1], but increasingly among contributing participants within a platform market [1, 2, 3, 4]. For example, mobile platforms such as Google Android or Apple iOS have seen an unprecedently rapid expansion, now offering myriads of third-party applications to users via the platform's marketplace. These applications are commonly referred to as complements and are vital to the platform's success [5, 6]. They are provided by external developers on the platform's periphery, which evolves around a core-product maintained by the platform owner [7, 8]. The platform owner invests heavily in updates in order to guarantee the overall stability of the platform as a technological foundation for complement development. [2, 9]. This highlights the established core-periphery argument which indicates that a software-based platform consists of a relatively stable core and a variable periphery [5]. Yet, besides dealing with maintenance and security issues, platform owners integrate novel features into the platform core to address platform evolution and growth over time [5, 9, 10]. Frequent complaints suggest that such features may have been previously provided by third-party complementors, who see their business models severely endangered by the platform owner's decision to enter their market space [e.g., 11]. Thus, the introduction of platform core features may critically affect the complementary market and the complementors' competitive situation. For example, Apple integrated several features such as Flashlight or AirDrop, thereby making the features available to all users and inherently bundling them with the core platform. The introduction of novel features by platform owners may benefit users due to decreased search costs and increased ease of use, yet may have significant impacts on the competitive environment and on complementors' ability to capture value. Following, the introduction of platform core features appears to exceed a platform owner's need to address maintenance and security issues and may target the management and coordination of evolutionary dynamics of platforms and ecosystems. Contrary to the above-mentioned notion of a relatively stable core, it seems that platform owners aim at an evolving platform core by constantly enriching its feature base.

The phenomenon of competition in platform markets has received elevated interest in extant literature on platform governance [3, 4, 12]. Research has provided valuable insights into platform owners' decisions to release first-party content and has studied its consequences on the complementary market [2, 3, 4, 13]. Nevertheless, how competition may emerge in the first place is still rather uncharted [14]. Various claims from practice-related blogs and public press articles as well as burgeoning academic interest in different modes of competitive moves conducted by platform owners

URI: https://hdl.handle.net/10125/80149
978-0-9981331-5-7

highlight the delicacy and relevance of this phenomenon [2, 15, 16]. Within this area, the evolution of the platform core through the introduction of novel features has not found significant attention so far. This is particularly interesting because previous research predominantly studied selected cases of first-party content and has not particularly distinguished among the provision of first-party complements and platform core features [3, 4]. One challenge in studying competitive implications of platform core features on the complementary market lies in the difficulty of comparing platform core changes to the third-party complement market. Anecdotal evidence from blog posts suggest that complementors' business models are affected by platform core features [e.g., 11]. Yet, no systematic approach exists that facilitates a comprehensive analysis of core changes. Therefore, we aim at addressing this gap and a) describe Apple's behavior to implement changes on the iOS platform core, b) develop a method to measure functional proximity of platform core features with third-party apps and c) analyze the relevance and direction of core features vis-à-vis the complementary market.

Against this backdrop, we conduct an exploratory machine-learning based case study on evolutionary changes of Apple iOS and analyze the functional overlap of changes on the platform core with the complements provided by third-party developers. We do this by proposing an approach to measure functional proximity of platform core features with third-party complements based on a combination of existing natural language processing (NLP) approaches. In this way, we aim at providing a first step towards further, in-depth research regarding the strategic role and implications of platform core changes on the complementary market.

This paper yields three main findings. First, we support the notion that platform core features are an inherent part of a platform owner's strategy that raises competitive implications. Second, the implementation of platform core features changes over time. Third, we introduce an actionable approach to compare third-party complements with evolutionary changes of the platform core on a feature level based on NLP approaches. We discuss the contributions of our findings for research on platform governance and platform competition and provide managerial recommendations.

2. Background

2.1. Platforms and platform ecosystems

In this study, we focus on innovation platform ecosystems. Innovation platforms are software-based systems that exhibit a technological core element which external firms can extend by providing complementary products and through which the relationships and interactions between participants on distinct sides are mediated [5, 17]. Examples include mobile platforms such as Apple iOS or Google Android and enterprise software platforms such as Salesforce. Pure transaction platforms differ from innovation platforms in that they lack an extensible core-product and focus on facilitating matchmaking and interaction between different platform sides (e.g., buyers and sellers on eBay) [18]. Innovation platforms exhibit a modular architecture, dividing the system into a core and a periphery [5, 8, 19]. The core is viewed as relatively stable and allows for variability in form of external innovation on the periphery [20].

A platform ecosystem is a socio-technical environment that embraces platform owner, complementors and users [5]. Platform owner and complementors highly depend on each other: Complementors rely on the platform core provided by the platform owner as a technological basis for their complement development. Innovation in the form of complements, however, critically determine the value of a platform and are therefore of utmost importance to the platform owner [4, 5, 21, 22]. In the case of mobile platforms, such complementors are usually third-party developers who leverage development resources provided by the platform owner and offer applications through the platform's marketplace [4, 23].

2.2. Platform governance and competition

Complementors are usually neither bound to the platform by formal contracts nor compensated directly for their development effort [24]. Instead, platform owner and complementors usually cultivate arm's length relationships. In order to guarantee that third-party contributions are in line with the expectations of the platform owner, the latter deliberately coordinates the actors involved in a platform ecosystem [14, 25, 26]. The activities of platform owners to influence behavior and outcomes of third-party complementors is referred to as platform governance [14]. Platform governance is a key concern for platform firms and aims at establishing innovation and coherence concerning complementors in light of behavioral uncertainty [7, 24, 27]. In the context of platform governance, the notion of boundary resources has become central [24]. Platform owners design and employ resources at the boundary to facilitate an effective coordination of complementors. Such boundary resources represent "software tools and regulations that serve as the interface for the arm's-length relationship between the platform owner and the application developer" [24:174]. Boundary resources may be a viable means to manage the existing tension of control and autonomy that platform owners as

coordinating entity are confronted with [7]: On the one hand, boundary resources may secure a platform by regulating (resource) openness and managing privacy [24, 28]. On the other side, boundary resources may highlight development opportunities and stimulate complementary innovation (i.e., resourcing) [24].

A purposeful design and governance of a platform gives rise to environmental dynamics and platform competition that influence the platform's further development [3, 5, 21]. The investigation of competition is of high interest among scholars in the context of platform governance [e.g., 3, 4, 29]. The ways in which competition takes place are multifaceted. For instance, competition exists between platform-based markets as platforms coexist and compete on functionality or pricing [1]. What is more, competition exists within a platform, where it occurs between platform participants. For example, an app market, which is often part of mobile platforms, is a highly competitive environment where apps compete for user downloads [30]. Complementors are not only competing with each other, they also face a risk of experiencing competition with a platform owner: Platform owners frequently decide to enter complementary markets with own competing solutions [21]. Extant work on platform governance increasingly focuses on understanding the phenomenon of such market entries by platform owners [3, 4]. Related research has predominantly investigated consequences of this occurrence on the complementary market and also started to investigate patterns and infer motivations (see [14] and [31] for extensive reviews). In the case of mobile apps, we find several examples of platform owner entry. Apple, for example, releases first-party apps that directly compete with those provided by external complementors [3, 4]. Besides, Apple releases novel features that potentially overlap with features of complements [2, 15]. Such platform design decisions may increase governance costs in that they require complements to renew and update existing knowledge regarding complement development [32].

In the context of this paper, we focus on the practice of platform owners to constantly make changes to the platform core, thereby promoting platform evolution. In particular, we investigate platform core features, which refer to software features that are an integral part of the platform core and exceed maintenance and error handling. A unique characteristic compared to (first-party) apps is that they form a part of the platform core; thus, they are directly shipped with it via global system updates and are non-removable. Platform core features are further directly accessible by platform users (e.g., while using an app), via specific gestures or hardware buttons (e.g., the home button on an Apple iPhone).

3. Method

3.1. Data collection

In this study, we conduct an exploratory, machine-learning based case study in the context of the mobile platform Apple iOS. We choose Apple iOS because it is a leading innovation platform in the mobile context, thus representing a typical case [33]. It exhibits a core product (the operating system) which is complemented by third-party products (apps) and, as such, exhibits intense dynamics between actors involved. Besides, the context offers a great availability of data, which allows a comprehensive and longitudinal analysis of platform core features [33]. We follow an exploratory case-study design because we elaborate on a real-life phenomenon (i.e., the release of platform core features by a platform owner), for which this approach is appropriate [33].

The nature of our research objective requires the collection of data from various sources. Therefore, we collected data in two steps. First, we gathered data on the evolution of Apple iOS. Precisely, we collected data on changes made to the platform on the software-level over time. In line with previous work on software evolution, we relied on software release notes in this step [34, 35, 36, 37]. Thus, we screened publicly available release notes of Apple iOS. Furthermore, we consulted blog and newspaper articles to triangulate the insights on platform core changes. For the final dataset on change descriptions, we subsequently distinguished if the change is related to the "app" or "core" level. We developed and applied decision flowcharts based on ideas of Chapin et al. [38] to categorize changes as either related to maintenance or evolution. Maintenance-related changes address security, performance, or usability issues. In contrast, evolutionary-related changes encompass the release of novel features, application programing interfaces or similar improvements. A set of maintenance and evolutionary keywords based on the works of Yu [36] and Moreno et al. [39] guided our coding process.

Second, to compare platform core features with third-party complements, our study requires data on the complementary market of Apple iOS. Thus, we obtained monthly data on the complementary market (such as app name, release date, app description) of Apple iOS from 2012 to 2020 from a leading app analytics provider. We exclusively focus on apps that (1) are available for Apple's iPhone, (2) are listed in English language and (3) provide a description that exceeds 200 characters. Furthermore, we exclude apps from the category "Games". This is due to the fact that this category is not compatible with the nature of platform core features. Besides, none of the studied changes touches this category throughout the observation period.

3.2. Data analysis and measurement approach

To measure the functional proximity of evolutionary changes of the platform core and features of third-party complements, we designed a measurement method leveraging six NLP approaches. The designed measurement method is based on a textual comparison of app descriptions of complements and the descriptions of platform core features. This is due to the assumption that complementors record the purpose and at least the most essential features of their applications offered on the app market in their respective descriptions [40].

The development of the measurement method was informed by a set of NLP techniques and methods from extant literature and is based on the following requirements related to textual comparisons: The proposed method is supposed to (1) determine the similarity of two texts with a resulting numerical value, (2) focus on semantic relations between texts, (3) be invariant to the length and structure of provided texts and (4) be unsupervised, with no need for labeled data.

Based on these requirements, three different NLP techniques and methods were selected: Word embeddings with Soft Cosine Similarity evaluation, the Universal Sentence Encoder and a simple keyword search.

The first method for identifying affected third-party apps uses Soft Cosine Similarity based on applying word embeddings and a subsequent calculation of the similarity of the resulting vector representations. Word embeddings are word representations as multi-dimensional vectors where similar words tend to have similar representation vectors [41]. The comparison of platform core features and app descriptions relies on the semantic similarity of textual descriptions based on Soft Cosine Similarity. While the more simple Cosine Similarity measure for determining the similarity of two descriptions is limited due to the loss of information by using averaged word vectors for the whole document [42], Soft Cosine Similarity also considers the similarity of individual words (features) [43]. We used pre-trained word embedding models based on the GloVe approach developed by Pennington et al. [44] in three variants for our approach. The GloVe model is a successor of the work by Mikolov et al. [41] with improved performance on word similarity tasks [44]. We implemented the first two variants on a pre-trained GloVe word vector model based on Wikipedia and Twitter, respectively, with no stemming (i.e., reducing words to their root form) as preprocessing steps and added a Twitter-based GloVe model with stemming as a third variant.

Our second method uses the Universal Sentence Encoder incorporating models based on transformer learning to construct sentence embeddings [45]. Given a string in English, the models are able to deliver an embedded vector representation for a whole document that is fixed in length independently from the length of the input string [45]. This feature allows us to make a textual comparison by using the models to construct an embedded vector representation of the platform core features and app descriptions, respectively.

We implemented two variants of the Universal Sentence Encoder, which differ in the use of the pre-trained models. The first variant ("v1") is based on the first published version of the Universal Sentence Encoder model. This model is based on a deep averaging network (DAN), which produces the embeddings for the input texts based on the average of the respective word and bi-gram embeddings [45]. The second variant ("v2") uses the fourth version of the deep averaging network encoder proposed by the Universal Sentence Encoder project team [45].

Additionally, we implemented a simple keyword search as a third method to take advantage of an intuitive concept and support the other two methods, ending up in total with six implemented variants. We introduce a scoring system to summarize the identification results of all the variants. In detail, the number of occurrences of each app is summed over all measurement approaches. The resulting sum is the individual score per app.

First, the six implemented approaches were provided with the appropriate input, such as the keywords or app descriptions and the descriptions of the platform core features. Apart from the keyword approach, where a simple match was sufficient, we set the threshold for the probability value for the similarity greater than 0.5 to identify an affected app. Since we conducted an unsupervised learning approach comparable with a binary classification (1 if an app is affected by a core feature, and 0 otherwise) we followed the default threshold of 0.5 for binary classification.

Afterwards, we manually reviewed a set of 22,776 identified third-party applications to check if a certain feature is indeed provided by the respective app. The applications in this set were detected by at least three of the implemented approaches (see scoring system). Table 1 provides a summary of the identification results for the individually implemented approaches, including the respective precision benchmark. The precision indicates the fraction of truly identified applications within the 22,776 retrieved ones by the implemented approaches. Since we worked with unsupervised approaches on a large-scale dataset, we were only able to provide the precision (False Positive Rate) of the approaches but not the recall (True Positive Rate).

In terms of precision, the keyword search and the second version of the Universal Sentence Encoder are found to be the most effective approaches. However, compared to the other techniques, these two approaches

identified a much smaller number of apps whose features indeed overlap with the identified platform core features. The total number of identified apps shows that the first version of the Universal Sentence Encoder and the approach based on Soft Cosine Similarity with stemming preprocessing are relatively insensitive.

Table 1. Identification results for each implemented measurement approach independently

Implemented approach	Apps identified	Affected by a core feature	Precision
Simple Keyword Search	6,359	1,340	0.174
Soft Cosine Similarity Wiki NoStem	147,922	5,484	0.036
Soft Cosine Similarity Twitter NoStem	145,852	4,927	0.033
Soft Cosine Similarity Twitter Stem	343,384	5,763	0.017
Universal Sentence Encoder - v1	335,789	4,147	0.012
Universal Sentence Encoder - v2	10,904	1,043	0.087

This suggests that for the task of identifying affected apps among descriptions, these two approaches have rather generated generally higher values for the similarity. Thus, the set of returned apps is much larger, but only a similar number of relevant apps are included compared to the other approaches. Due to the high number of false positives, these two approaches are the least precise, shown by the calculated precision value. In summary, none of the implemented NLP approaches delivers satisfactory results as a standalone solution based on data input. Nevertheless, the total number of identified third-party apps that overlap with features of the platform core is promising.

By applying a scoring system, we combined the results of the individual approaches: An identified third-party app received a score between 0 and 6, depending on how many of the six approaches identified it to be affected by platform core features. In other words, our approach builds a logical conjunction of the outputs of the six isolated approaches. We came up with the idea of a similarity scoring system by comparing the precision metrics of a combined approach in contrast to the single approaches (Table 1). The precision metrics are increasing significantly when combining the output of multiple approaches, enhancing the overall reliability of our identification results.

Table 2. Identification results for the combination of approaches in a scoring system

Scoring system (combining all approaches)	Apps identified	Affected by a core feature	Precision
Apps with score = 3	15,458	2,225	0,144
Apps with score = 4	5,627	2,630	0.467
Apps with score = 5	1,467	1,137	0.775
Apps with score = 6	224	211	0.942
Apps with score >= 3	22,776	6,203	0.272

Based on the scoring results, we performed an additional evaluation. A precision of 0.272 was achieved for apps with a score of at least 3, improving the overall precision in comparison to the stand-alone approaches. Focusing on higher scores even increased the precision up to 0.942 (score = 6). Further details are illustrated in Table 2. These results show that once an app has been identified as affected by multiple of the implemented approaches based on its textual description, it is much more likely that the app actually shares the identified features with the platform core.

4. Results

4.1. Evidence on platform evolution of Apple iOS

Apple's proprietary mobile operating system was released in 2007, together with the first generation of the mobile device iPhone. Since then, Apple has continuously released and documented updates to iOS. In this study, we cover the time period from June 2007 to April 2020 resulting in 140 released iOS versions clustered within 13 major versions. Apple uses integer values to refer to major version updates (e.g., iOS 11.0), and cascaded fractional digits values to refer to minor platform changes. Our data shows that the average time span between version releases alternates strongly throughout the observed period, with up to an average of 80 days between two releases in one year. However, from 2013 on, these average values settle at less than 40 days. Thus, Apple increased the number of annual releases and reduced the intervals between versions since 2013.

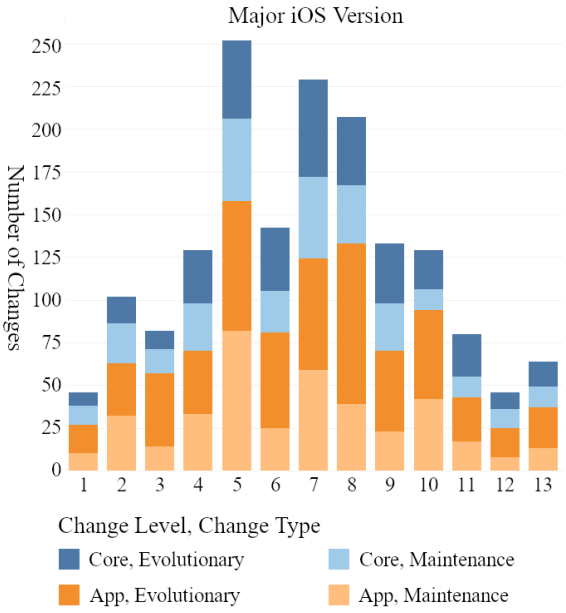

Figure 1. Number of changes to iOS per major version, categorized among change level and type

The analysis of release notes identified 1,641 changes that were documented for the 140 versions in the period under review. The changes can be allocated to a total of 91 distinct areas of iOS (core features and first-party apps). We categorized those changes among the app and core level and accounted for their change type (maintenance or evolutionary). Our results emphasize that most of the releases focus on maintenance changes, while the minority of changes involve novel features. This observed proportion has increased in recent years. Releases of iOS are found to be successive to each other in most cases. Apart from some overlaps after a new version release, previous major versions usually do not receive further updates after a new major version has been introduced.

As illustrated in Figure 1, the majority of changes introduced via major versions target the app level. While iOS evolves with each major release on both the core and app level, we observe a focus on the introduction and maintenance of first-party apps that are shipped with iOS. Platform core features and first-party apps are primarily introduced via major version releases. In contrast, minor version number changes predominantly focused on maintenance changes. This update behavior was observed throughout the entire observation period. Therefore, our analysis shows that the evolution of the iOS platform core reveals recognizable patterns and suggests a certain predictability regarding the release time of platform core changes.

Over the entire study period from 2007 to 2020, we observed the release of 113 relevant core features on iOS based on an analysis of 354 evolutionary core-level iOS changes. 43 of these features were assigned a high potential impact on third-party complements. The potential impact was considered to be medium for 16 features and low for 54 features. These core features were assigned to 18 different feature areas. Most of the features were traced back to the areas “Siri”, “iCloud”, “Spotlight”, “Accessibility”, “General” and “Keyboard”. Besides, some features led to the introduction of novel feature areas such as “Printing”, “Control Center”, “Parental Controls”, or “Screenshot” at the time of their release.

Our analysis further shows that Apple introduced several novel platform core features to iOS every year. Figure 2 illustrates the number of released platform core features and their assumed impact on the complementary market. While the number of released platform core features does not vary considerably across years, the number of features that exhibit a high impact on the app market is rising. This finding is remarkable in light of the finding that the overall number of changes has decreased in recent years. The number of platform core features released per year seems to be relatively unaffected by this overall decrease.

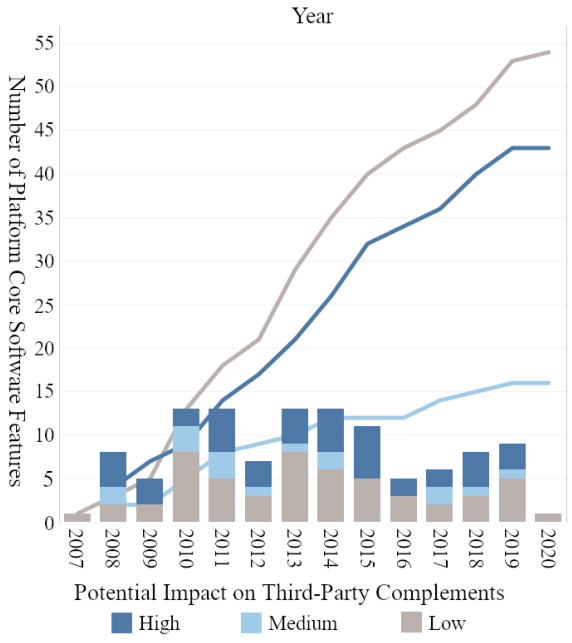

Figure 2. Released iOS platform core software features per year

The number of platform core feature releases in the early years of the iOS platform (i.e., before 2012) is higher compared to recent years (i.e., after 2015), with a higher percentage of features coded as exhibiting a potentially high impact on third-party complements. We further find that most platform core features (46 in

particular), are introduced within the first release of a major version update (e.g., iOS 12.0). Minor version updates, in contrast, predominantly convey a small number of new features over the last updates.

4.2. Platform core features vis-à-vis the third-party complement market

In this section, we present the results of applying the measurement method developed in the context of this study. In particular, we report on the relevance and direction of platform core features released by Apple vis-à-vis the third-party complement market. The developed measurement method and an additional manual evaluation (see chapter 3) yielded a total of 6,203 affected third-party complements for 29 highly relevant platform core features released between 2012 and 2020 (14 have been released before 2012). Most identified apps were affected by the platform core features "Internet radio via Siri" (2,198 identified affected third-party apps), "Flight status information via Siri" (901), "Credentials and account info manager" (372), "Emergency handling" (345) and "Flashlight" (282). We further find that the number of affected apps nearly doubled in the three most recent years of the observed time period. This is interesting given that the number of introduced relevant platform core features decreased in more recent years, yet this analysis does not control for the total number of apps and number of apps per category. The categories "Music", "Entertainment" and "Utilities" show the highest numbers of apps affected by platform core features. Nevertheless, the number of platform core features assignable to app categories differs slightly from this observation. The categories most platform core features are assigned to are "Productivity", "Utilities", "Entertainment" and "Lifestyle". The mentioned categories show similarities to more than 20 platform core features each.

Besides, we assess the functional overlap of third-party complements with platform core features on the publisher-level. The identified set of third-party complements are found to be distributed among 3,726 publishers. Over 616 publishers operated at least one app that shows a functional overlap with platform core features. 80 publishers are identified to have operated more than two affected apps. These affected publishers mainly operate apps in the categories "Productivity", "Utilities" and "News". In summary, the majority of the identified app publishers have been affected once by novel platform core features to iOS. Few publishers have been identified to operate several hundred affected apps, yet exhibit a functional overlap with only a single platform core feature. A closer analysis revealed that the apps of these publishers are remarkably similar regarding their provided features and functionalities. Apps that allow users to check the status of a flight and receive flight information may serve as an example. We identified one publisher who operate more than 350 such apps, each offering similar features as individually branded app for a specific airport. The apps of this particular publisher are therefore considerably similar for a large number of airports worldwide. We observed a similar phenomenon for publishers with affected apps by the platform core features "Internet radio via Siri" and "Selected Information via Spotlight". In total, almost 800 of the identified third-party applications are operated by five publishers.

5. Discussion

The aim of this paper was to describe the behavior of platform owners to implement changes on the platform core and to analyze the relevance and direction of platform core features vis-à-vis the complementary market. To this end, we conducted a qualitative machine-learning based case study on Apple iOS.

We highlight three main findings. First, we provide evidence that the release of platform core features is indeed an inherent part of a platform owner's strategy. Our identification results show that for almost every relevant feature, multiple third-party apps existed on the App Store at the time of its release. This observation indicates an elevated supply of apps for the specific features and suggests that complementors have served the market prior to the feature releases.

Second, we find evidence that the use of platform core features changes over time. The analysis of the release activity over the years since the platform's inception in 2007 revealed a considerable increase in the number of additional minor version changes per major version. For example, iOS major versions 1.0 to 4.0 comprised approximately five sub-versions (e.g., 4.1.; 4.2.), whereas later major versions encompass up to 15 such sub-versions. Our results also suggest that the number of affected apps nearly doubled in the last three years of the observed time period. This is particularly interesting in light of the observation that the number of relevant platform core features decreased in recent years. Most importantly, we identify a higher number of platform core features in the platform's first years compared to later years.

Third, we provide an actionable approach to compare third-party complements with evolutionary changes of the platform core on a feature level based on the computation of textual similarity. We implement and evaluate different NLP approaches based on the work and ideas of Cer et al. [45], Mikolov et al. [41], Pennington et al. [44] and Sidorov et al. [43] as well as a supplementary keyword search. In particular, we

propose a scoring approach that combines various existing and proven techniques. The implementation of this combination yields a significant increase with regards to precision values. Besides providing a means to compare platform core features with the complementary market, the measurement approach further enables us to assign platform core features to app categories, facilitating a category-centric analysis of evolutionary changes of a platform.

We want to highlight the following three contributions to research on platform governance and competition in platform ecosystems. First, we contribute to extant work on platform governance in general by providing evidence that supports the reality of platform core features as relevant strategic move by platform owners. Following extant research, consequences of this phenomenon may be contingent on the respective degree of interface openness and align with the two processes "securing" and "resourcing" of the boundary resources model [24, 46]. A commoditization of features that are solely provided to be used by customers might refer to a closing strategy of the platform owner in that it secures the platform from undesired contributions. In contrast, platform core features may also represent novel interfaces that third parties use and connect complements to. Under these circumstances, the release of platform core features might represent an act of resourcing by evolving the interfaces offered for complement development [47]. In acknowledging a purposeful design and release of functional core changes, our findings challenge the established core-periphery argument that regards the platform core as relatively stable [5, 8]. Our study shows that platform core features indeed compete with complements and suggests that innovation also takes place on the platform core [20, 48]. Hence, our findings support the notion of platforms as evolving organizations [20]. Future research is required to understand under which conditions the platform core should remain stable and when variability is beneficial.

Second, we contribute to extant work on platform governance that has investigated the phenomenon of platform owner entry by studying how competition emerges between platform owner and complementors [2, 3, 4]. We conduct a systematic analysis of the relevance and functional overlap of platform core features with existing third-party complementors. Our findings support recent work and suggest that functional updates to the platform core represent a strategic option of platform owners to induce competition [2, 15].

The measurement approach may help future research in further analyzing idiosyncrasies of different modes of platform owner entry. In this way, we contribute to work that studied platform core features [2, 15], complement acquisitions [48], and the release of first-party apps [4, 49], highlighting the potential influence and relevance of single features introduced by extending the underlying software system in the analysis of competition within platform ecosystems.

It is yet to investigate how the threat of facing competition with a platform owner via platform core features compares with the threat posed by first-party apps and how core changes influence the complementary market. A promising avenue for future research may be to build on this observation and analyze the particular competitive effect of platform core features on the complementary market. The developed measurement approach can support research regarding how the threat of facing platform core features materializes in other platform ecosystems.

Third, we highlight differences in the release behavior of platform owners that may point to latent patterns of active participation of platform owners in the complementary market [12, 17]. On the one hand, we emphasize the importance of analyzing platform core features over time for this might inform the discussion on motivations of platform owners to enter complementary markets. On the other hand, the increase in affected apps in recent years may also be interpreted as an indication of the popularity of the respective platform core feature. Bender et al. [15] deliver a supportive finding: According to their interviews with app developers, popular complements are more likely to be integrated into the platform core.

These circumstances call for further investigation of how the platforms' release behavior affects user experience and if such behavior has any appreciable effects on complementors. Extant work has highlighted the relevance of developer conferences for platform governance [50]. Future work may explore their role in addressing potential negative consequences caused by platform core features, given that they are frequently preannounced during such events.

Future work may also elaborate on the influence of feature breadth of complements on the effect of competitive first-party content. In doing so, future work could emphasize the perspective of complementors and customers more intensively and inform work on complementor responses and coping strategies to overcome and survive competitive conflicts with a platform owner [e.g., 12, 15, 51].

These findings have the following managerial implications. On the one hand, the release patterns and update behavior observed introduce a somehow predictable course regarding the introduction of platform core features. While the actual moves and their purposes remain difficult to predict, the timing of such moves in relation to the platform core can be narrowed down to the release cycles. Complementors on mobile platforms should therefore pay special attention to the

annual major releases, since it is mainly through these releases that novel platform core features are introduced. Therefore, complementors are well advised to prepare for the yearly occurring major versions, which usually introduce multiple novel features to the platform core. As those introductions may contain competitive moves that potentially jeopardize the complementors' business model, complementors should actively monitor the announced changes and adjust their development strategy accordingly. What is more, complementors should also be alert to evolutionary changes implemented on competing platforms. On the other hand, platform core features are often related to the categories "utility" and "productivity". Therefore, third-party applications and related complementors from these categories are particularly likely to be confronted with a potential market entry of the platform owner. Besides, platform core features affect a vast number of third-party apps in entertainment, music and travel categories, whereby these features are less category-specific and affect several categories simultaneously. This urges complementors to strongly monitor adjacent categories with regards to potential core feature overlaps.

6. Conclusion

Platform ecosystems are becoming increasingly important as they have become omnipresent nowadays. This study has provided evidence that the evolution of a platform is intertwined with competition in that changes of a platform core may introduce features that inherently overlap with features provided by third-party complementors. In particular, we analyze changes conducted by Apple on its mobile platform iOS and identified 29 potentially competing platform core features that are published between 2012 and 2020. The proposed measurement approach identifies various third-party apps that are directly related to platform core features.

The growing amount of power and influence of platform owners and the inherent interdependencies among ecosystem participants clearly emphasize the necessity of an ongoing discourse on the proper governance of platform ecosystems, as well as competitive implications of governance decisions. By focusing on the integration of features into the platform core as potentially inducing competition with complementors as a particularly subtle but effective strategic move, we aim to provide a fruitful basis for subsequent discussions on the phenomenon of competition within research on platform governance.

7. References

[1] Eisenmann, T., G. Parker, and M. Van Alstyne, "Platform envelopment", *Strategic management journal 32*(12), 2011, pp. 1270–1285.

[2] Bender, B., and N. Gronau, "Coring on Digital Platforms-Fundamentals and Examples from the Mobile Device Sector.", *38th International Conference on Information Systems*, (2017).

[3] Foerderer, J., T. Kude, S. Mithas, and A. Heinzl, "Does platform owner's entry crowd out innovation? Evidence from Google photos", *Information Systems Research 29*(2), 2018, pp. 444–460.

[4] Wen, W., and F. Zhu, "Threat of platform-owner entry and complementor responses: Evidence from the mobile app market", *Strategic Management Journal 40*(9), 2019, pp. 1336–1367.

[5] Tiwana, A., B. Konsynski, and A.A. Bush, "Research commentary—Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics", *Information systems research 21*(4), 2010, pp. 675–687.

[6] MindSea, "28 Mobile App Statistics To Know In 2020", 2020. https://mindsea.com/app-stats/

[7] Wareham, J., P.B. Fox, and J.L. Cano Giner, "Technology ecosystem governance", *Organization science 25*(4), 2014, pp. 1195–1215.

[8] Baldwin, C.Y., and C.J. Woodard, "The architecture of platforms: A unified view", *Platforms, markets and innovation 32*, 2009, pp. 19–44.

[9] Eaton, B., S. Elaluf-Calderwood, C. Sørensen, and Y. Yoo, "Distributed tuning of boundary resources", *MIS quarterly 39*(1), 2015, pp. 217–244.

[10] Ojala, A., and K. Lyytinen, "Competition logics during digital platform evolution", 2018.

[11] Smith, D., "Inevitable Sherlocking", 2017.

[12] Zhu, F., and Q. Liu, "Competing with complementors: An empirical look at Amazon.com", *Strategic management journal 39*(10), 2018, pp. 2618–2642.

[13] Hagiu, A., and D. Spulber, "First-party content and coordination in two-sided markets", *Management Science 59*(4), 2013, pp. 933–949.

[14] Halckenhaeusser, A., J. Foerderer, and A. Heinzl, "Platform governance mechanisms: an integrated literature review and research directions", *European Conference on Information Systems*, AISeL (2020).

[15] Bender, B., C. Thim, and F. Linke, "Platform Coring in the Browser Domain-An Exploratory Study", *40th International Conference on Information Systems*, (2019).

[16] Nicas, J., "Apple Cracks Down on Apps That Fight iPhone Addiction", *The New York Times*, 2019. https://www.nytimes.com/2019/04/27/technology/apple-screen-time-trackers.html

[17] Rietveld, J., M.A. Schilling, and C. Bellavitis, "Platform strategy: Managing ecosystem value through selective promotion of complements", *Organization Science 30*(6), 2019, pp. 1232–1251.

[18] Cusumano, M.A., A. Gawer, and D.B. Yoffie, *The business of platforms: Strategy in the age of digital competition, innovation, and power*, Harper Business

New York, 2019.
[19] Schilling, M.A., "Toward a general modular systems theory and its application to interfirm product modularity", *Academy of management review 25*(2), 2000, pp. 312–334.
[20] Gawer, A., "Bridging differing perspectives on technological platforms: Toward an integrative framework", *Research policy 43*(7), 2014, pp. 1239–1249.
[21] Halckenhaeusser, A., J. Foerderer, and A. Heinzl, "Wolf in a Sheep's Clothing: When Do Complementors Face Competition With Platform Owners?", *41th International Conference on Information Systems*, (2020).
[22] Rochet, J.-C., and J. Tirole, "Platform competition in two-sided markets", *Journal of the european economic association 1*(4), 2003, pp. 990–1029.
[23] Eisenmann, T.R., G. Parker, and M. Van Alstyne, "Opening platforms: how, when and why?", *Platforms, markets and innovation 6*, 2009, pp. 131–162.
[24] Ghazawneh, A., and O. Henfridsson, "Balancing platform control and external contribution in third-party development: the boundary resources model", *Information systems journal 23*(2), 2013, pp. 173–192.
[25] Boudreau, K., "Open platform strategies and innovation: Granting access vs. devolving control", *Management science 56*(10), 2010, pp. 1849–1872.
[26] Evans, D.S., and R. Schmalensee, *Matchmakers: The new economics of multisided platforms*, Harvard Business Review Press, 2016.
[27] Tiwana, A., *Platform ecosystems: Aligning architecture, governance, and strategy*, Newnes, 2013.
[28] Karhu, K., R. Gustafsson, and K. Lyytinen, "Exploiting and defending open digital platforms with boundary resources: Android's five platform forks", *Information Systems Research 29*(2), 2018, pp. 479–497.
[29] Tiwana, A., "Evolutionary competition in platform ecosystems", *Information Systems Research 26*(2), 2015, pp. 266–281.
[30] Lim, S.L., P.J. Bentley, and F. Ishikawa, "The effects of developer dynamics on fitness in an evolutionary ecosystem model of the App Store", *IEEE Transactions on Evolutionary Computation 20*(4), 2016, pp. 529–545.
[31] Zhu, F., "Friends or foes? Examining platform owners' entry into complementors' spaces", *Journal of Economics & Management Strategy 28*(1), 2019, pp. 23–28.
[32] Foerderer, J., T. Kude, S.W. Schuetz, and A. Heinzl, "Knowledge boundaries in enterprise software platform development: Antecedents and consequences for platform governance", *Information Systems Journal 29*(1), 2019, pp. 119–144.
[33] Yin, R.K., *Case study research: Design and methods.*, Sage, Thousand Oaks, CA, 2009.
[34] Abebe, S.L., N. Ali, and A.E. Hassan, "An empirical study of software release notes", *Empirical Software Engineering 21*(3), 2016, pp. 1107–1142.
[35] German, D.M., "Using software trails to reconstruct the evolution of software", *Journal of Software Maintenance and Evolution: Research and Practice 16*(6), 2004, pp. 367–384.
[36] Yu, L., "Mining change logs and release notes to understand software maintenance and evolution", *CLEI Electron Journal 12*(2), 2009, pp. 1–10.
[37] Kemerer, C.F., and S. Slaughter, "An empirical approach to studying software evolution", *IEEE transactions on software engineering 25*(4), 1999, pp. 493–509.
[38] Chapin, N., J.E. Hale, K.M. Khan, J.F. Ramil, and W. Tan, "Types of software evolution and software maintenance", *Journal of software maintenance and evolution: Research and Practice 13*(1), 2001, pp. 3–30.
[39] Moreno, L., G. Bavota, M. Di Penta, R. Oliveto, A. Marcus, and G. Canfora, "ARENA: an approach for the automated generation of release notes", *IEEE Transactions on Software Engineering 43*(2), 2017, pp. 106–127.
[40] Harman, M., Y. Jia, and Y. Zhang, "App store mining and analysis: MSR for app stores", *IEEE International Working Conference on Mining Software Repositories*, 2012, pp. 108–111.
[41] Mikolov, T., K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space", *International Conference on Learning Representations*, (2013).
[42] Hassan, B., S.E. Abdelrahman, R. Bahgat, and I. Farag, "UESTS: An unsupervised ensemble semantic textual similarity method", *IEEE Access 7*, 2019, pp. 85462–85482.
[43] Sidorov, G., A. Gelbukh, H. Gómez-Adorno, and D. Pinto, "Soft similarity and soft cosine measure: Similarity of features in vector space model", *Computación y Sistemas 18*(3), 2014, pp. 491–504.
[44] Pennington, J., R. Socher, and C.D. Manning, "Glove: Global vectors for word representation", *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, (2014), 1532–1543.
[45] Cer, D., Y. Yang, S. Kong, et al., "Universal sentence encoder", *arXiv preprint arXiv:1803.11175*, 2018.
[46] Parker, G., and M. Van Alstyne, "Innovation, openness, and platform control", *Management Science 64*(7), 2018, pp. 3015–3032.
[47] Gawer, A., "Digital platforms' boundaries: The interplay of firm scope, platform sides, and digital interfaces", *Long Range Planning*, 2020, pp. 102045.
[48] Toppenberg, G., S. Henningsson, and B. Eaton, "Reinventing the platform core through acquisition: A case study", *2016 49th Hawaii International Conference on System Sciences (HICSS)*, IEEE (2016), 4634–4643.
[49] Li, Z., and A. Agarwal, "Platform integration and demand spillovers in complementary markets: Evidence from Facebook's integration of Instagram", *Management Science 63*(10), 2017, pp. 3438–3458.
[50] Foerderer, J., "Interfirm Exchange and Innovation in Platform Ecosystems: Evidence from Apple's Worldwide Developers Conference", *Management Science 66*(10), 2020, pp. 4772–4787.
[51] Jiang, B., K. Jerath, and K. Srinivasan, "Firm strategies in the 'mid tail' of platform-based retailing", *Marketing Science 30*(5), 2011, pp. 757–775.