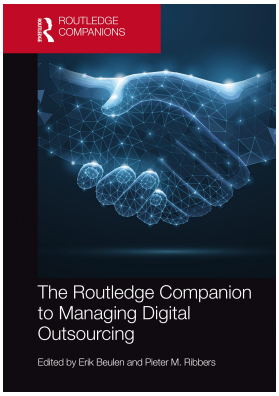

The Routledge Companion to Managing Digital Outsourcing

Erik Beulen, Pieter M. Ribbers

Gravitation of SaaS-centric cloud platforms

Publication details
https://test.routledgehandbooks.com/doi/10.4324/9781351037785-18
André Halckenhaeusser, Armin Heinzl, Kai Spohrer
Published online on: 28 Jul 2020

15

GRAVITATION OF SAAS-CENTRIC CLOUD PLATFORMS

André Halckenhaeusser, Armin Heinzl and Kai Spohrer

15.1 Introduction

In the digital age, innovations in cloud computing are the impetus for the continuing trend to outsource IT resources. This contemporary computing paradigm is considered to be highly promising for sustaining economic and technological advantage due to benefits like reduced upfront investments and high scalability [1,2]. In particular, Software as a Service (SaaS) enjoys constantly growing interest and adoption among enterprise clients [3]. Coupled with a shift from internal to external innovation, currently emerging cloud platforms specifically provide support for the development and distribution of software [4]. Cloud platforms increasingly provide marketplaces to trade and manage SaaS solutions and, thus, become more and more relevant for supporting developers and consumers to connect and to transact with each other [5]. Such cloud platforms create two-sided markets, embracing developers and consumers of SaaS in one ecosystem [4]. We use the term *SaaS-centric cloud platform* to refer to such platforms that focus on the generation, provision and distribution of software services provided by multiple vendors. Providers of such SaaS-centric cloud platforms generate distribution channels, open new sales and marketing opportunities for third-party developers of SaaS solutions and promote collaboration as well as competition among the different ecosystem participants [2,6,7]. Thus, we refer to cloud platforms that provide marketplaces as an integral part of their business model.

Maintaining a flourishing ecosystem of developers and customers is non-trivial [8]. In fact, cloud platforms are constantly exposed to the risk of developers' discontinuing affiliation [9] and customers abandoning the platform due to the dynamics of multi-sided platforms (e.g., [10]). Consequently, some platforms thrive, whereas others remain unsuccessful as they fail to attract or retain developers and customers of SaaS offerings [11]. It is therefore crucial for providers of SaaS-centric cloud platforms to understand what makes platforms gravitate, i.e., what defines a cloud platforms' capability to grow an ecosystem by attracting and retaining developers and customers [12].

Although a considerable body of research has investigated platform ecosystems in the context of software platforms [13–15], research on SaaS-centric cloud platforms is still nascent [4,16]. In particular, it is still unclear which factors determine if SaaS-centric cloud platforms become successful. Prior research on cloud platforms focused primarily on their

ecosystem-based development capabilities and gave only little consideration to the distribution and transaction facilitation capabilities of the business network [17]. The distribution channel, however, constitutes a highly important element of SaaS-centric cloud platforms' business models [6]. Therefore, in line with a call for more context involvement in research [18], we suggest that existing explanations of platform success in general may not suffice to understand why specifically cloud platforms that provide marketplaces to distribute SaaS solutions gravitate.

Extant literature has predominantly adopted two perspectives on platform ecosystems in general [19]. On the one hand, an economics perspective highlights the transaction facilitation capabilities of a platform and focuses on the attraction of participants on both sides of the market (i.e., developers and customers) in order to leverage network effects [20]. This perspective suggests that platforms gravitate if they connect those two sides and thereby overcome an existing transaction problem [19]. On the other hand, a technology-related perspective highlights the innovative capacity of a platform and underlines the importance of its technological architecture [21,22]. This perspective suggests that platforms gravitate if they are technically designed in a way that facilitates third-party innovation and thereby value co-creation [19].

Therefore, platform providers need to manage a shift from developing applications in-house to providing resources that support external innovation [12] and must ensure that participation is beneficial for the external third-party developers [23]. Although these perspectives provide valuable insights into gravitation of platforms in general, they do not sufficiently account for the specifics of SaaS-centric cloud platforms. A more holistic view on SaaS-centric cloud platforms may be necessary to explain why specifically they gravitate.

Against this backdrop, we follow various authors' calls for more research on platform-specific topics (e.g., [24,25]) and examine the factors that drive the gravitation of SaaS-centric cloud platforms. In particular, we aim to answer the question: *How and why do SaaS-centric cloud platforms gravitate?*

To do so, we follow an exploratory qualitative research design and conduct four case studies of SaaS-centric cloud platforms. We employ two theoretical perspectives on platform ecosystems that help us explore factors influencing the gravitation of SaaS-centric cloud platforms [19]. Overall, our study contributes to research on platform ecosystems by identifying catalyzing and inhibiting factors of platform gravitation. In this way, this study provides a starting point for future research on the drivers of gravitation of such platforms. Furthermore, the findings of this study have practical implications for platform providers that aim to maintain a flourishing ecosystem.

15.2 Background and conceptual foundations

cloud computing constitutes a change in IT delivery and in the IT business model [2] by transforming the traditional IT artifact from resources into services [26]. Since its recent emergence, cloud computing is increasingly gaining attention from both researchers and practitioners [26,27]. This recent computing paradigm can be defined as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [28, p. 2].

In cloud computing, there are three service models that differ in their level of abstraction of the provided IT capability and help to categorize service offerings [28,29]: First, *Infrastructure as a Service (IaaS)* focuses on the provision of elementary, on-demand IT-infrastructural resources like storage or processing capabilities. Second, *Platform as a Service*

(PaaS) provides a platform that facilitates deployment and development of cloud-based applications. Third, *SaaS* refers to the provision of complete applications that run on cloud infrastructure and that are accessible through thin interfaces via a network.

In our study, we understand platforms as technologies, products or services which provide a common basis third-party firms can use in order to develop complementary products [30]. These third-party firms that are affiliated to a platform are referred to as complementors and expand the platform's market by providing subsidiary components [30,31]. A platform ecosystem constitutes a loosely coupled inter-firm network that embraces platform provider, complementors and consumers [32,33]. An important characteristic of platforms is the potential existence of network effects [34]. Network effects refer to the phenomenon that a platform's perceived value is contingent upon the number of users adopting it [35].

In the context of cloud computing, the term "PaaS" often highlights the technical aspects of a cloud-based development platform and does not extraordinarily prioritize the idea of business networks and connecting customer segments [17]. We suggest that the elaboration on SaaS-centric cloud platforms requires a more holistic view on the platform and the software services offered. A cloud platform can be defined as a "development and execution environment in which external developers deploy and run their complementary [components and applications]" [4, p. 553]. As stated by Giessmann and Stanoevska [36], cloud platforms are increasingly complemented with marketplace functionality in order to facilitate the trade of services. SaaS-centric cloud platforms offer distribution-channel capabilities, which third parties can leverage to promote and sell SaaS solutions to consumers. These platforms potentially increase market transparency of cloud computing, mitigate risks associated with this computing paradigm and in this way may spur adoption of cloud services especially by small and medium-sized enterprises [1,37].

The existence of a distribution channel that supports developers in transacting with users of SaaS solutions is therefore a central element of SaaS-centric cloud platforms. Distribution channels for SaaS solutions may exhibit distinct foundational specificities and are increasingly emerging in the context of all cloud service models (e.g., SaaS, PaaS, IaaS) [6]. Although the traditional distinction of SaaS, PaaS and IaaS is highly useful, it also has shortcomings. As such, it does not capture the differences in distribution channels but focuses on the service model that is provided. For this reason, we abstract from the traditional categorization of SaaS vs. PaaS vs. IaaS in our investigation of SaaS-centric cloud platforms.

In this way, we hope to be better able to capture the particularities of the different distribution channels regarding the provision and distribution of SaaS solutions. Thus, we choose a conceptualization of SaaS-centric cloud platforms that encompasses platforms focusing on SaaS, PaaS, as well as IaaS. Extant SaaS-centric cloud platforms differ considerably regarding the existence of a core product and the provided development capabilities. We use the categorization provided by Giessmann and Legner [4] who distinguish cloud platforms according to the main value proposition provided to technical users (i.e., developers): First, the conceptualization encompasses *application-based* SaaS-centric cloud platforms. These cloud platforms focus on the integration of software components, such as add-ons, into a core SaaS-solution. Providers of this core SaaS solution leverage external innovation by opening their core product to third-party developers and in this way allow for co-creation of value [22]. On this premise, the SaaS product functions as platform for countless extensions developed by third parties which extend the core product's functionality and adjustability. Second, we distinguish *development-focused* SaaS-centric cloud platforms, which focus on the provision of development and deployment capabilities. Platforms of this type, in contrast to the first category, lack an extensible core product and strongly support the application development process of comprehensive

SaaS solutions, which are then distributed via the platforms' marketplace. Third, *distribution-focused* SaaS-centric cloud platforms provide distribution channels and open marketing and sales opportunities to their users as a main value proposition. These cloud platforms exhibit strong user communities and facilitate sharing and trading of ready-to-use cloud software. In this category, cloud services are typically deployed to an external or internal infrastructure and do not require extensive development functionality.

Summarizing, SaaS-centric cloud platforms include distribution-focused, development-focused and application-based cloud platforms that offer distribution-channel functionality and focus on SaaS solutions. Table 15.1 provides an overview of typical SaaS-centric cloud platforms.

Table 15.1 Typical SaaS-centric cloud platforms and respective marketplaces (links accessed August 15, 2017)

Application-based SaaS-centric cloud platforms	
cloud platforms focusing on the provision of a core SaaS product that is open to external developers and that extends its functionality by add-ons	
Atlassian (Marketplace)	https://marketplace.atlassian.com
Intuit QuickBooks (Apps.com)	https://www.apps.com
NetSuite (SuiteApp.com)	http://suiteapp.com
Salesforce Force.com (AppExchange)	https://appexchange.salesforce.com
SAP cloud Platform (SAP AppCenter)	https://www.sapappcenter.com
SugarCRM (SugarExchange)	https://sugarexchange.sugarcrm.com
Zendesk (Apps Marketplace)	https://www.zendesk.com/apps/
Zoho (Zoho Marketplace)	https://marketplace.zoho.com
Development-Focused SaaS-Centric cloud Platforms	
cloud platforms focusing on the provision of a development and execution environment supporting the entire application development process	
Amazon Web Services (AWS Marketplace)	https://aws.amazon.com/marketplace
Google App Engine (G Suite Marketplace)	https://gsuite.google.com/marketplace/
Microsoft Azure (Azure Marketplace)	https://azuremarketplace.microsoft.com/
SoftwareAG (Digital Marketplace)	https://marketplace.softwareag.com/
Distribution-Focused SaaS-Centric cloud Platforms	
cloud platforms focusing on the provision of a distribution channel to offer and trade SaaS solutions to a larger community as main value proposition.	
1und1 cloud Apps	https://hosting.1und1.de/cloud-app-center/cloud-apps
AppDirect Marketplace	https://www.appdirect.com
Bitnami	https://www.bitnami.com
Comcast Business Marketplace	https://cloudsolutions.comcast.com/
Ingram Micro cloud Marketplace	https://de.cloud.im/
Interoute cloudStore	https://cloudstore.interoute.com
Nubocloud	https://www.nubocloud.de
Rackspace Marketplace	https://marketplace.rackspace.com
Singtel myBusiness	https://mybusiness.singtel.com/
Swisscom Business Marketplace	https://businessapps.swisscom.ch
Telekom cloud	https://cloud.telekom.de

We define gravitation of a SaaS-centric cloud platform inspired by the physical notion of gravity as their capability to attract and retain customers on both sides, i.e., developers and users of SaaS services. Being one of the four fundamental forces of physics, gravitation is defined as the mutual attraction of two forces. In astronomy, gravitational interaction is responsible for the fact that planets circulate around the sun and stay in their orbits [38]. Considering what is mentioned above, success of a cloud platform can be measured by its growth, and, thus, highly depends upon the degree to which the potential participants from affiliated sides gravitate toward the platform. This concept therefore alludes to a platform's attractiveness.

15.3 Method

In this study, we follow an exploratory qualitative research approach in order to identify factors influencing platform gravitation. Due to the recency of the object of investigation and consequently the scarcity of studies on the selected cloud platform concept, an exploratory approach was deemed appropriate [39]. In particular, we conducted four case studies of specific SaaS-centric cloud platforms [40]. According to Benbasat et al. [39], case study research is especially helpful for answering "how" and "why" research questions like ours. After having identified a variety of cloud platforms based on a web search and insights from research articles, a framework proposed by Giessmann and Legner [4] helped to categorize the cases (see Table 15.1). In case study research, only a limited number of examples can be examined effectively [40]. Therefore, we selected leading SaaS-centric cloud platforms in order to include only high performing platforms with mature ecosystems in our analysis. We included representative cases from all three categories of SaaS-centric cloud platforms outlined above. The selection of cases across the three categories allowed us to consider distinct platform types and mitigated the risk of setting a narrow focus. While the selected cases Alpha, Beta, Gamma and Delta differ considerably, they all focus on the provision and management of SaaS solutions. We chose representatives of the platform owners as interview partners. Table 15.2 presents the selected SaaS-centric cloud platforms and briefly describes each case.

15.3.1 Data collection

We base our analyses on semi-structured expert interviews with knowledgeable representatives and managers of the platform ecosystems as well as public and internal information from websites, presentations and e-mail conversations. We conducted five semi-structured interviews with these key informants that included questions regarding software, governance, infrastructure criteria as well as the evolution of the platform. The interviews were conducted via telephone and internet calls and lasted between 30 and 70 minutes. Each interview was recorded and transcribed subsequently. We triangulated and extended the qualitative data gained from the interviews by reviewing additional public and internal documents from websites (e.g., company presentation, developer documentation, learning centers), internal presentations and follow-up e-mail conversations with interviewees. The consideration of both interview data and additional documents helped increase data triangulation and validity [41]. Table 15.3 outlines the interview details.

15.3.2 Data analysis

Following Eisenhardt [40], we conducted within-case analyses of all cases individually. These helped us cope with the elevated complexity of the collected data and allowed us to

Table 15.2 Presentation of cases

Case company	*Type [4]*	*Short description*
Alpha	Application-based integration	Alpha is a leading provider of cloud-based enterprise software. Alpha offers a SaaS-solution that has been opened up to facilitate customizability and extensibility by add-ons. These add-ons are built on the cloud platform and are provided by the platform owner and third-party vendors. The platform offers a proprietary marketplace on which add-ons are offered to Alpha's customers.
Beta	Development focused	Beta is a leading software and service provider. The cloud platform offering comprises IaaS, PaaS and SaaS solutions and embraces various integrated services that support the development, deployment and management of cloud applications. The distribution channel connects software vendors and Beta's customers and offers cloud services that run on Beta's infrastructure.
Gamma	Distribution focused	Based on its expertise in software packaging, Gamma focuses on the provision of pre-packaged software services which are easily deployable in the cloud. These services are offered on the proprietary marketplace and on distribution channels of prominent cloud platforms. For application deployment and execution, Gamma developed an automated packaging system.
Delta	Distribution focused	Delta provides a leading platform to support distribution and management of cloud-based services. Besides operating an own distribution channel, Delta offers white-label marketplace platforms. Delta's distribution ecosystem embraces third parties operating a white-label platform as well as developers and customers. The distribution approach of Delta allows developers to reach a high number of potential customers.

Table 15.3 Interview details

Case company	*Interviewee position*	*Date*	*Length of interview*	*Additional material*
Alpha	Vice President of Customer Success (A#1)	07/2017	70 minutes	Internal marketing presentation, online documentation
Beta	Program Manager (B#1)	08/2017	55 minutes	Website, documentation
Gamma	Sales Representative (C#1)	08/2017	30 minutes	Website, online documentation, e-mail, business overview, white paper
Delta	cloud Expert (D#1)	08/2017	35 minutes	Website, online documentation, e-mails
	Key Account Manager (D#2)	08/2017	50 minutes	

Table 15.4 Taxonomy and selected coding examples

Taxonomy

Dimension	*Concepts*	*Coding examples*
cloud Platform Main Value Proposition	Distinction between application-based integration, development focus and distribution focus	"Every app is integrated into the core product. The ISV cannot run independently, the core product and platform is essentially required." (A#1)
Software Criteria	Modularity • App-decoupling • Interface standardization	"[Add-ons] are highly decoupled. […] Everything that has been built upon the platform […] has to be robust for this process." (A#1)
	Integration • Platform-app integration (e.g., Single-Sign-On) • Cross-app integration	"[Cross-application integration] is possible. It depends upon the channels that the respective ISV opens. Software can be provided completely encapsulated or open to other solutions." (D#2)
	Development support • Boundary resources (e.g., SDK, API, app reviews)	"We offer our customers SDKs that support the development of applications on Java, Python, Ruby, PHP […] It is very diverse." (B#2)
	Technical openness of platform technology and SaaS (e.g., open-source)	"We are increasingly offering open-source and strongly evolving into an open platform. I think this is one of Delta's most significant." (D#2)
Underlying Infrastructure	Infrastructure Transparency • Selection (fixed/selectable) • Location Distribution • Ownership (proprietary/external)	"You can freely choose the data-center region [your software will be deployed on]." (D#2) "We partly operate the infrastructure, but we also build on external data-centers [depending on the region]." (A#1)
Governance Criteria	Control mechanisms	"We have a […] security review every application placed on our marketplace has to pass [..] It is important for us that they conform to our security standards […]" (A#1)
Pricing Policies/ Revenue Model	Access/usage pricing Revenue sharing model	"these software providers have actually paid us to put their applications in our library." (B#1)
Marketplace Functionality	Portfolio • Number of applications • Specificity of offerings	"The apps that we onboard to our platform are very selected, […] we are working with around 300 different applications. […] they are very specific." (C#1)
	Domain Specificity	"We want to offer a high spectrum of applications to our customers [..] I can't even tell you how many apps we are offering on our marketplace right now, but it is a very high number." (D#2)
	Support of transaction phases	"We do not carry out billing on the platform. The ISV is responsible for that." (A#1) "The contract is closed directly over the platform and also embraces important elements like support." (D#2)

understand the specifics of the different cases. Subsequently, we employed a between-case analysis in order to identify commonalities and differences and to derive factors that influence the gravitation of SaaS-centric cloud platforms.

Interview transcripts were coded using the qualitative data analysis software NVivo 10, which helped to reduce complexity. In order to effectively conduct the data analysis, we synthesized relevant concepts from the extant literature on platform ecosystems, cloud platforms and marketplaces and derived a taxonomy that served as our analytical structure. In total, we included six dimensions in the taxonomy. This taxonomy allowed us to compare the selected cases and helped avoiding shortcomings in case study research approaches [40]. We relied on a priori codes on the basis of the dimensions and concepts proposed in our taxonomy. The taxonomy is presented in Table 15.4. Selected coding examples are added in order to facilitate the understanding of the concepts and the coding process.

15.4 Results

15.4 1 Within-case analysis

Alpha. The proprietary distribution channel embraces more than 3,000 applications that complement Alpha's core SaaS, which is a leading enterprise software system. Applications are highly decoupled from the platform. Both end consumers and developers appreciate and require the platform's rigor as far as stability and compatibility of the platform is concerned. Although the platform owner conducts several updates of the platform and its functionality, third-party solutions are broadly independent upon these changes. Furthermore, add-ons do not require specific interfaces in order to communicate with standard objects provided by the platform. This is due to the fact that extensions run in the application base of the platform and use the platform's functionality directly. The extensions are decoupled from the platform, yet rely on it due to their thorough integration with the platform and a lack of internal interfaces. The software services run natively on the platform and are generally integrated into the core product, avoiding any technical integration efforts between platform and add-ons. The platform thereby facilitates automatic deployment and provides Single Sign-On (SSO) of add-ons offered on the marketplace. Due to the fact that the communication between software services and the platform is realized via sharing one data model, no integration effort is necessary to facilitate the communication between different extensions.

Alpha is currently introducing a new component-model service, allowing customers to assemble components offered on the marketplace and thereby to create new combined services. Alpha attaches the developer community a high strategic importance (A#1: "Around 70% of the platform's load is Custom-Development while only 30% can be associated with the standard objects we provide with our SaaS offerings"). Developers appreciate the ease of getting started and the support and learning opportunities the platform provides. To this end, Alpha also provides comprehensive information and educational documentation and further conducts various programs and competitions with complementors. Alpha operates own infrastructure and leverages external infrastructure in smaller markets due to scalability reasons. Users are able to select a specific data center region that best fits their performance and compliance requirements.

The platform owner frequently conducts updates, and potentially integrates new functionality that was formerly exclusively provided by the solution of an independent software vendor (ISV). Furthermore, the platform owner occasionally offers own add-ons that are similar to the scope of application of ISVs' solutions and extend the functionality and

attractiveness of the core product to end user. Alpha therefore possibly enters into a complementor's market by publishing a proprietary solution providing the same functionality of an existing ISV's offering or by integrating existing functionality into the platform. Especially in the latter case, the platform owner entry threatens the ISV's position.

As far as control mechanisms are concerned, Alpha is not very restrictive regarding content and functionality. However, security and reliability of the platform is highly important to Alpha. Therefore, a formal application is obligatory prior to participating in the marketplace. Alpha further conducts comprehensive security reviews on a recurring basis.

Alpha charges customers for using the SaaS product and developers for using its development platform. Moreover, developers are charged a fee for placing their applications on the marketplace and they share a certain percentage of their net revenues with Alpha.

The lacking focus on a specific business size regarding the services on the marketplace is criticized by Alpha since larger customers might not trust the reliability and support capabilities of products of small ISVs. As stated by the interviewee, the global orientation of the marketplace requires ISVs to provide a sufficient scaling of their solutions. Besides, Alpha aims at providing a comprehensive portfolio across industries and actively attracts significant developers that cater important business processes not supported hitherto by the ecosystem. In the past, the active attraction of significant software vendors has been an important factor for growth. While the marketplace supports contracting and deployment of services, Alpha does not carry out billing in order to concede ISVs certain autonomy. After having deployed an add-on, which is usually provided as a trial version first, the ISV asks customers for their payment details to gain full access. According to the interviewee, the easy deployment and integration of add-ons into the core product leads to high uncertainties regarding support and maintenance responsibilities of providers. "If they [customers] have installed 10 ISV applications, they also have to maintain them, enter into contracts with vendors and train skills" (A#1).

Beta. Beta attaches high importance to the robustness of its platform and the services offered. Services placed on the marketplace predominantly run in virtual machines and include a pre-configured environment for application execution. In this way, the dependencies between the platform and applications as well as across applications are minimized. Beta provides various components to facilitate integration of software services with the platform and provides SSO. Furthermore, the marketplace supports automated provision and deployment of software services. Cross-application integration is feasible and depends upon the way an ISV opens the respective software service. Third-party contributions on the marketplace are regarded as highly important for Beta. The platform provides strong and manifold development services that simplify access to the platform and the development of services, e.g., specific software development kits (SDK) focusing on various programming languages. Furthermore, Beta provides considerable educational resources. Beta operates a variety of own data centers worldwide. Customers may select a specific region for the deployment of software services in order to meet compliance and legal requirements or performance needs. Although data centers are proprietarily owned, Beta reacted on regulatory requirements in specific regions by implementing innovative models that exhibit different data-ownership structures. Beta controls its distribution channel and the offerings placed by ISVs. The input control embraces a formal application process. Furthermore, software services are reviewed thoroughly prior to publication. Regarding the revenue model, Beta keeps a percentage of revenues generated on the marketplace. Beta's portfolio provides a variety of different software services. While some applications clearly address developers, many pre-packaged images of software services cater specific business needs. Furthermore, Beta selects software

services purposefully according to their relevance in order to provide a widespread portfolio. The marketplace thereby exhibits a low domain specificity. The trend of increasingly focusing on open-source offerings is regarded as one of Beta's "most significant factors of success" (B#2). As far as transaction on the marketplace is concerned, customers may adjust the selected service to own requirements and budget limitations. A formal contract is closed between the ISV and the customer. Payment and billing is then handled by the platform owner in case this is desired by the respective ISV. Finally, contracted software services are automatically deployed, a virtual machine is created and standard or custom configurations are set.

Gamma. The software services that are offered on Gamma's marketplace can be deployed to various infrastructure providers who are partnering with the case company. Once an infrastructure provider has been selected, the chosen SaaS services from the marketplace are deployed automatically. Afterward, services run highly decoupled from Gamma's platform. Nevertheless, the software services are not integrated with the platform after setup. Gamma counts on external applications and provides an easy-to-use process in order to onboard applications to its platform. This process of easily accessing and deploying services was explicitly demanded by Gamma's customers. The application selection is principally conducted by Gamma albeit partnering infrastructure providers and commercial ISVs may also influence it. Regarding technical openness, a focus on open-source software is apparent. Gamma offers an unprecedented flexibility regarding infrastructure. While Gamma does not operate own data centers, the services offered can be deployed to the partnering infrastructure providers immediately. The company realizes input control and examines applications with respect to their fit with the packaging and deployment technology and regarding additional components necessary for deployment. The onboarding process takes between one and two months and includes an extensive feasibility study. Gamma intents to keep it as simple as possible. Software services are selected purposefully based on the perceived value they bring to Gamma and partnering infrastructure providers. Due to the packaging process, Gamma controls the cloud images and the update process in order to ensure provision of secure applications. Gamma does not charge consumers for using its platform. In contrast, it charges developers for using its packaging services to prepare services for deployment and imposes a fee for placing an application on the marketplace. Furthermore, infrastructure providers pay Gamma commission fees for facilitating simple deployment of the offered services on their respective infrastructure. Thus, billing and payment handling is neither supported nor necessary. Fees related to the necessary infrastructure are handled by the respective IaaS provider. Gamma's portfolio embraces less than 200 offerings. In its portfolio, various offerings covering the same scope of application exist. For example, in the category of CRM, the marketplace lists more than ten different CRM solutions. Apart from its focus on open-source software, Gamma's portfolio exhibits low specificity regarding domain and applications.

Delta. Delta's approach facilitates high distribution channel capabilities. On the one hand, developers may publish applications on Delta's marketplace. On the other hand, once integrated with Delta, developers have the opportunity of placing their solution on a variety of third-party marketplaces associated with Delta. Delta provides open API in order to integrate external software services with the platform. The software services that Delta manages are decoupled from the platform. Due to the fact that the communication between platform and applications is clearly defined and kept simple, the platform and subsystems are fairly independent. The platform further supports SSO via OpenID as well as automated deployment for integrated applications. Delta provides a service for searching application-specific

data and finding information stored in different cloud applications. However, the platform does not explicitly realize cross-application integration. "It is a future desire for anyone in the cloud space. That is definitively something that we will be able to offer [in the future]" (D#1). In order to support ISVs during the processes of onboarding and integrating applications, Delta provides a development environment. Besides providing integration assistance, the developers may manage product offerings (e.g., pricing and discounts), marketing and promotional activities, payments, orders, invoices and customers. Delta exerts input and output control by requesting a formal application and conducting an application review. Applications are thereby comprehensively evaluated regarding usability and service standards. A distribution agreement contract precedes the application review process.

Furthermore, once approved, each change made on an application necessarily needs to be approved before coming into effect. The onboarding process is conducted free of charge. Prior to the application integration and the publication of a software service, a revenue sharing model is negotiated and stipulated in a formal contract between Delta and ISVs. Besides, third-party marketplace operators are charged for using Delta's white-label marketplace platform. While Delta's portfolio embraces purposefully selected free and commercial applications, a focus on open-source software is not evident. Free offerings usually refer to trial versions that exhibit certain limitations, e.g., regarding the number of users, transactions or storage. Delta focuses neither on specific industries nor on a specific business size, albeit a slight focus on small and medium-sized enterprises becomes prevalent. Integrated applications can be contracted from the software vendor and are provisioned immediately. Besides the placement and distribution of software services, Delta manages payment and billing.

15.4.2 Between-case analysis

Based on the within-case analysis, a comparison of the single cases led us to understand factors that catalyze or inhibit gravitation of SaaS-centric cloud platforms. First, we report on the catalyzing factors. Subsequently, the identified inhibiting factors will be presented. Table 15.5 contrasts the results of the individual case analyses.

15.4.2.1 Catalyzing factors

Ease of Access and Use (1). It is noticeable that all cases aim at providing a high ease of access and use for both developers and customers. While ease of use predominantly refers to the way a user may interact with the platform, ease of access addresses the degree to which developers are supported and promoted in participating in the platform ecosystem and respectively the obstacles they might experience.

Integration, regarded as an important concept in the proposed taxonomy, applies to communication between platform and applications as well as across applications [13]. All cases provide SSO, which enable customers to access software services from different complementors easily. A tight integration of applications with the platform further facilitates management of contracted services in a central place. For instance, Delta provides a dashboard where users may contract and access services as well as modify subscriptions. Furthermore, all cases support various phases of a market transaction on the marketplaces. We use the phases defined in Menychtas et al. [42] and distinguish negotiation, contracting and settlement of a market transaction in our study. In the negotiation phase, service customization and price building take place. The contracting phase comprises the closing of a legally binding contract between provider and consumer. During the settlement phase, the services are delivered (delivery) and

Table 15.5 Comparison of cases

Dimension	*Concept*	*Cases*			
		Alpha	*Beta*	*Gamma*	*Delta*
MVP	Main value proposition	Application-based integration	Development focus	Distribution focus	Distribution focus
Software Criteria	Modularity	Medium (coupled with platform, no use of interfaces)	High (executed in sandboxes/ virtual machines, highly decoupled)	High (apps are encapsulated, highly decoupled)	High (highly decoupled, standardized interfaces)
	Platform-app integration	Fully integrated	Apps run on or deploy to Beta, automated deployment, single sign-on	Deployment, server-management (reboot, shutdown, delete), single sign-on	Deployment, single sign-on, management and distribution of applications
	Cross-app integration	Fully integrated	Feasible, implementation necessary, supported by integration services.	Not feasible	Not feasible (albeit cross-application data searching service)
	Development Support	High. Development portal, various programs and competitions, learning center	High. Developer community, developer tools (IDE, SDKs), online documentation and resources	Medium. App packaging, online community, onboarding support	Medium. Onboarding, application review, development environment
	Technical openness	Closed, private marketplace available. Offerings: mainly closed source	Closed, private marketplace available. Offerings: open and closed source	Technology: closed source offerings: mainly open source	White-label marketplace platforms. Offerings: mainly closed source

Underlying Infrastructure	Infrastructure Transparency	User-selectable (region, hardware) worldwide distributed proprietary and external data centers	User-selectable (region, hardware) worldwide distributed proprietary data centers	User selectable (provider, region, hardware) worldwide distributed external data centers	Fixed (handled by SaaS vendor) external data centers
Governance	Control	Input: formal application Output: security reviews	Input: formal application Output: app nomination, app review	Input: formal application Output: feasibility study, application review and packaging	Input: formal application, contract Output: application review
Pricing / Revenue Policies	Pricing	Customer: license needed Developer: review fee (setup and annual)	Customer: no fee Developer: no onboarding fee	Customer: no fee Developer: publication fee (commercial software)	Customer: no fee Developer: no fee
	Revenue model	Revenue share: certain percentage	Revenue share: certain percentage	Revenue model: IaaS commission	Revenue share: upon negotiation
Marketplace Functionality	Portfolio	> 3,000 add-ons and components low app specificity	> 5,000 offerings low app specificity	> 150 pre-packaged offerings low app specificity	> 300 integrated apps low app specificity
	Domain specificity	Low (generic focus)	Low (generic focus)	Low (generic focus)	Low (generic, slightly SME focused)
	Support of transaction phases	No payment handling	All transaction phases supported	No payment handling	All transaction phases supported

charged (payment). In the analyzed cases, software services are easily executable in a cloud environment due to the abstraction of the service setup and deployment process. This clearly eases the settlement phase, due to reduced complexity regarding deployment and delivery on the marketplace and is suggested to promote a high user acceptance and satisfaction. Besides, all cases support third parties in their development work. Development support refers to the provision of resources and assistance to complementors that facilitate platform-specific development and therefore includes the concept of boundary resources [12]. For instance, SDKs, application programming interfaces (APIs) and onboarding processes may be part of a platform owner's support strategy [43].

Simplifying the access of developers may foster third-party participation on the platform. First, a straightforward onboarding and integration was found to be catalyzing. All examined cases exhibit and highlight their onboarding and platform-application integration support. Gamma, for instance, reduces the efforts of external developers to place solutions on the marketplace by carrying out software packaging and onboarding. Second, Alpha and Beta clearly focus on support of external contribution and innovation and clearly highlight their strategic significance for success. Both platforms provide considerable educational support like learning opportunities and comprehensive documentations. Alpha further conducts programs and competitions in order to spur innovation. Thus, the way a platform provider treats the developer community highly influences developers' participation decision and thereby the success of the platform. Consequently, simplifying the processes of provisioning, deployment and execution of software services and also supporting third parties in developing and distributing their solutions has been identified to catalyze a platform's gravitation.

Security (2). All examined cases highlight the importance of security regarding both the offered services and the underlying technology. In our work, security refers to the appropriateness and trustworthiness of software services as well as to the reliability and robustness of the platform technology. On the one hand, the analyzed cases exert control on third-party contributors and their software services. We distinguish input control, which makes reference to vertical openness to external developers and respectively to existing entrance barriers [44]. Output control refers to the proposal of favorable and necessary criteria concerning complementors' outputs and scrutinizing complementors' solutions prior to publication by implementing review processes [13]. All cases restrict access to their ecosystem by requesting a formal application of third-party firms. The implementation of output control mechanisms ensures the fit of applications regarding security and quality standards. Gamma automated the update process of both software and additional components and thereby guarantees that offered software services are permanently up-to-date and secure.

On the other hand, reliability and robustness of the underlying platform technology are found to be highly important to maintain developers' and customers' trust in the respective platform. The concept of modularity refers to the extent to which a platform's ecosystem depends upon changes. It can be accomplished by reducing interdependencies (app-decoupling) and defining the way the platform and applications interact (interfaces) [13]. In all cases, the software services are highly decoupled, which minimizes the risk of necessary investments in software updates due to possible changes on the platform. Especially for Alpha and Beta, the robustness and continuity of the platform are considered to be highly important success factors. Consequently, customers rely on the security regarding both the underlying platform technology and software services, thus representing a catalyzing factor.

Portfolio Significance (3). The analysis revealed that all cases explicitly curate a significant portfolio. We understand portfolio significance as the degree to which offerings have been purposefully selected. A significant portfolio thereby refers to a catalogue of service offerings

on the platform's marketplace that is appropriate and relevant to the focused customer segments. Thus, this concept may allude to the quantity, quality and specificity of the offerings and further relates to control mechanisms. On the one hand, high portfolio significance may be achieved by purposefully selecting and actively searching for suitable software services with regard to core processes of addressed industries, as in the case of Alpha. Similarly, Gamma selects the applications according to their highest perceived value. By exerting input and output control, all cases limit the existing portfolio at least to a certain degree. Moreover, offering open-source software services has been found to be especially beneficial due to their usefulness and cost-effectiveness, as the case of Gamma illustrates. Gamma explicitly focuses on open-source software and highlights the elevated demand for such solutions. Similarly, Beta increasingly focuses on offering significant open-source solutions as well. Thus, this factor further addresses the concept of technical openness of the software services. On the other hand, the participation of large and significant ISVs can highly impact the platform's appeal on both software vendors and customers. In the case of Alpha, the attraction of significant software vendors profoundly affected the evolutionary trajectory of the platform. Due to their prominence, many other participants joined the ecosystem as well. Consequently, the provision and maintenance of a significant portfolio has been identified as a catalyzing factor of gravitation.

Infrastructural Flexibility and Transparency (4). It is conspicuous that the majority of examined cases grant customers high degrees of flexibility regarding the underlying infrastructure on which the software services run. Major issues influencing the adoption of cloud computing are related to security, trust, privacy as well as legal aspects. These issues are highly interdependent with the cloud infrastructure [45]. Due to the layered architecture of cloud computing, the underlying network structure is usually abstracted from the user of a specific service model, which may lead to a lack of transparency. Gamma provides an elevated flexibility across significant providers due to the maintenance of partnerships with a variety of IaaS providers. A customer may choose between various IaaS providers and may further select concrete data center regions. In the cases of Alpha and Beta, customers and developers may select the desired region of the assigned data centers. In the special case of Beta, tailored regional offerings address extraordinary regulatory requirements (e.g., legal issues, compliance). Enabling customers to select certain elements of the underlying infrastructure may further increase service transparency. Due to this selection, a user is potentially aware of the provider and location of the underlying infrastructure. Consequently, the facilitation of infrastructural flexibility and correspondingly transparency has been found to be an important factor regarding the platform's gravitation (Table 15.6).

Table 15.6 Catalyzing factors and exemplary characteristics

Catalyzing factors	
Ease of access and use	• Simplifying provisioning, deployment and execution of services • Support of development and onboarding processes
Security	• Selection and trustworthiness of complementors and services • Reliability and continuity of the platform
Portfolio significance	• Appropriate selection of services (growing relevance of open source) • Attraction of large developers
Infrastructural flexibility and transparency	• Addressing regulatory requirements • Enabling customers to select elements of the infrastructure (e.g., location)

15.4.2.2. Inhibiting factors

Intra-Platform Competition (1). This factor focuses on the degree to which developers compete with other ecosystem participants, including the platform owner. On the one hand, all examined cases provide various services that cover similar scopes of application. Although all cases control both the input and the output, developers and platform owners are allowed to publish software services that are similar to those which already exist. The openness regarding the scope of application leads to an increase in the quantity of the application portfolio and a possible decrease in the specificity of the offered services. This, in turn, might have an impact on the mode of competition prevailing in the platform ecosystem [15]. For example, Gamma's application catalogue embraces 12 different CRM solutions. Similarly, Delta lists 138 results in the category of CRM. On the other hand, especially in application-based SaaS-centric cloud platforms, a potential threat of the platform owner entering a complementor's market is evident. Especially for Alpha, this represents an inhibiting factor since third-party developers might be put in a disadvantage. Consequently, based on the findings we suggest that an increased intra-platform competition represents an inhibiting factor.

Contractual Complexity (2). Contractual complexity refers to the degree of complexity of the contractual commitment negotiated between ecosystem participants. In some settings, customers may be exposed to an increased contractual complexity as a result of having purchased various applications on the marketplace. The majority of the examined cases support the immediate contracting and deployment of software services. Thereby, customers usually enter into a contract with the respective software vendor who publishes a solution on the platform's marketplace. Furthermore, platform owners engage in rather arm's length relationships with third-party developers, and these complementors therefore often manage their own contracts with customers. Especially in the case of Alpha, an uncertainty regarding support responsibilities arises. This is due to the degree of integration of the offered add-ons into the core product, which is characteristic of application-based SaaS-centric cloud platforms. In the event of errors or failures, customers struggle to determine the cause and the ISV in charge of the problem. Alpha reports that the resulting complexity presents a considerable challenge for its customers. While in the other examples the origin of an error is likely to be more easily identifiable, customers still have to deal with a variety of different contracts. Our analysis reveals that the contractual complexity could be confronted by maintaining overall agreements embracing all developers who provide software services to a specific customer. In such settings, platform owners would engage in high-level relationships with developers, i.e., they would provide the customer with one specific contract that envelops all service providers related to that customer.

This would require different contractual constructs that govern responsibilities, duties and error handling. Consequently, the potentially complex contractual situation a customer is confronted with on a platform's marketplace represents an inhibiting factor.

Lack of Customer Centricity (3). The concept of customer centricity is based on the idea of positioning customers at the center of focus of a company and assumes that an organization needs to focus on and especially listen to the needs of the customers it directly addresses [46]. In this context, the degree to which a platform addresses specific industries can be understood as domain specificity. All examined cases exhibit a low domain specificity since they do not focus on selected industry branches in particular. Furthermore, all cases exhibit a global orientation and embrace participants of all business sizes in their ecosystems. Therefore, the customer focus of the analyzed platforms seems to be rather unspecific. In the case of Alpha, the lack of a regional focus as well as a lack of a focus on a specific business size might produce

Table 15.7 Inhibiting factors and exemplary characteristics

Inhibiting factors	
Intra-platform competition	• High number of competing services covering a similar scope of application • Threat of a platform owner entering the market
Contractual complexity	• Customers enter in a variety of contracts due to immediate provisioning • Uncertain support responsibilities in the case of errors
Lack of customer centricity	• Support deficiencies due to a lacking focus on specific domains or regions • Reduced trust in scaling capabilities due to differences in business size

a reluctant behavior of customers. Similarly, considerable differences in business sizes between vendor and customer might further reduce the trust in a vendor's capability to provide global support. As the interview with Alpha reveals, especially larger companies might be influenced in their decision to contract services from smaller ISVs. For example, a larger company might lack trust in the ISV's capabilities to provide sufficient scaling and support of its service. A customer-centric orientation regarding developers has been therefore identified to be beneficial. According to the interviewee from Alpha, specific addressed industries might benefit from a closer collaboration between platform owner and developers. Consequently, an insufficient customer centricity represents an inhibiting factor (Table 15.7).

15.4.2.3 Platform type as a moderating factor

Although our exploration yielded catalyzing and inhibiting factors that affect platform gravitation, not all of these effects were equally strong in all cases. A closer examination of the differences showed that several effects differed in strength depending on the platform type on which they were observed. Our analysis showed no differences regarding the factors "ease of access and use," "security and robustness" (catalyzing factors) and "lack of customer centricity" (inhibiting factor) across the cases. In the following, we elaborate on the rest. Figure 15.1 summarizes the moderating effects.

Portfolio Significance. The distribution channels of all examined cases limit the portfolio to a certain degree and in this way ascribe certain importance to a selected portfolio. Nevertheless, in the cases of application-based and distribution-focused platforms, providers especially emphasized the strategic importance of a highly curated service portfolio for success. The curation of a significant portfolio allows these platforms to provide a relevant and high-quality service catalogue that specifically caters to the platforms' customers and addresses all relevant business processes. In contrast, development-focused platforms, such as Beta, do not extraordinarily focus on the provision of a tailored portfolio. This platform type puts less emphasis on portfolio curation because its customers are more widespread and it strongly targets developers. Particularly, developers may prefer a wide and diverse repository of services which they can orchestrate and use within their development work. Besides, the marketplace facilitates them to easily distribute their specific services to users. The comprehensive coverage of a single domain with few specific services may be less relevant to them. Therefore, we argue that the impact of portfolio significance is especially high in application-based and distribution-focused platforms but less pronounced in development-focused platforms.

Infrastructural Flexibility and Transparency. As outlined, development- and distribution-focused platforms highlight the infrastructural flexibility granted to customers. Although

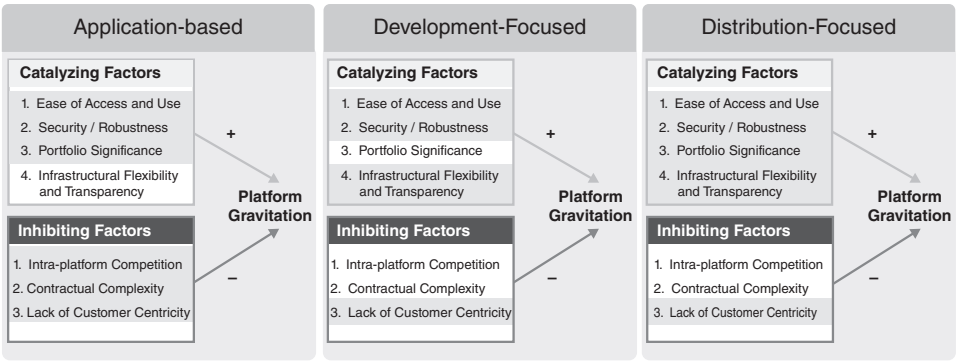

Figure 15.1 Moderating effects of platform type (highlighted factors have a high impact on platform gravitation in the respective type)

application-based platforms also list their services transparently, our analysis suggests that the impact of this factor is lower in application-based platforms. While customers of application-based platforms may select the data-center region the core application is deployed to, additionally contracted services directly run in the customer's application base. Thus, no further specification of the infrastructure is necessary. Customers of application-based platforms may be more interested in a neat integration of contracted services with the core application rather than their physical location. Decisions about infrastructural aspects are therefore more readily delegated to the provider of this core application, who abstracts the underlying infrastructure from customers in order to provide a more integrated experience with the core application. Thus, we argue that this factor is particularly relevant in development- and distribution-focused platforms but less so in application-based platforms.

Intra-Platform Competition. We further identify that the impact of intra-platform competition is contingent upon the service integration. Our results show that the threat of a platform owner entering the market is especially strong if services are highly integrated with the platform's core. In application-based platforms such as Alpha, external developers are exposed to challenging situations if the platform owner integrates functionality into its core application that was formerly exclusively provided by third parties. Competing complementors must therefore constantly fear that the platform includes the service of one of the competitors in the core application thereby making the complementors' service effectively unnecessary. Consequently, third-party service providers on such platforms tend to refrain from making large investments or access many such competitive settings at the same time if the platform owner does not mitigate this fear. Therefore, we argue that this factor has an especially high impact in application-based platforms.

Contractual Complexity. As mentioned earlier, customers engage in a variety of individual contracts with the service providers on SaaS-centric cloud platforms. Due to the straightforward transaction-phase support of the marketplaces, such contracts are quickly negotiated and closed. In the event of errors, however, customers are unable to easily identify the responsible service provider if the contracted service is highly integrated with a platform's core, as it is the case in application-based platforms. When a tightly integrated complementary service causes errors, customers may actually be unable to identify which service is faulty. Consequently, they face problems in finding the right contracted service provider to address with a maintenance request or compensation claims. What is more, contractually agreed

service levels may differ across the core application and complementary services and therefore pose a major challenge for customers in enforcing a satisfactory overall service level of the integrated set of platform core and complementary services. Owners of application-based platforms can therefore highly increase their platform's attractiveness to customers if they can reduce this contractual complexity and offer customers a sole interface and contract for maintenance requests and related issues across all services, not only regarding the platform core. In distribution-focused and development-focused platforms, services run more isolated from each other and from the platform. Consequently, such problems occur less frequently. Contractual complexity therefore has a particularly strong impact on platform gravitation in application-based platforms.

15.5 Discussion

The overarching objective of this study is to explore the factors that influence gravitation in SaaS-centric cloud platforms, a specific type of cloud platform that exhibits capabilities to distribute and manage SaaS solutions. Gravitation refers to a platform's capability to attract and retain participants in its ecosystem. Our study therefore identified important factors that influence the success of SaaS-centric cloud platforms. Specifically, we have elaborated four catalyzing factors, which are (1) ease of access and use, (2) portfolio significance, (3) security and (4) infrastructural flexibility and transparency. Furthermore, we discovered three inhibiting effects to the gravitation of SaaS-centric cloud platforms, namely (1) intra-platform competition, (2) contractual complexity and (3) lack of customer centricity. Moreover, we show that the factors are contingent upon the respective platform type.

15.5.1 Implications for research

The identification of catalyzing and inhibiting factors of platform gravitation provides insights into beneficial settings of SaaS-centric cloud platforms. Our research contributes to the emergent literature on cloud platform ecosystems in several ways.

First, this study identifies a need to focus on alternative contractual configurations that may be beneficial for all ecosystem participants to reduce contractual complexity. As stated by Wareham et al. [15], platform ecosystems are exposed to the risk that direct responsibilities of complementors might be difficult to determine. Contractual agreements of ecosystem participants have been taken into consideration in the extant literature, highlighting their potential to facilitate value co-creation in platform ecosystems [22]. This study contributes to this existing research by suggesting that maintaining rather high-level relationships might be a viable means to counter the contractual complexity customers are exposed to. Our results suggest that high-level contracts between complementors and platform owners that provide a single, unified contract to customers may be beneficial for customers especially in the case of application-based SaaS-centric cloud platforms. This could clarify responsibilities and duties of complementors. Future research should elaborate on concrete contractual configurations and possible arrangements. However, this would mean a relationship-specific investment for platform owners. Prior work suggests that platform owners seek to balance arm's length coordination that reduces investments in single relationships and allows for leveraging synergies with the freedom they provide to ecosystem participants that allows them to co-create value [14]. Future work may therefore need to investigate how platform owners can achieve a balance between reducing the contractual complexity for customers and still keeping relationship-specific investments moderate.

Second, our findings suggest that the benefits of narrow customer foci need to be illuminated for SaaS-centric cloud platform ecosystems, especially from an economics perspective. By drawing on the concept of customer centricity, we identify a lack of a specific customer focus on specific industries, regions and business sizes as inhibiting gravitation. This finding is in line with the recommendation of a vertical focus that could provide domain-specific knowledge necessary for certain industries [5,47]. Our findings are thereby also consistent with research that identifies inequalities with respect to business size to be considerably inhibiting the co-creation of value in inter-organizational settings [22,23]. Specifically, this study reveals that differences in business size further affect the relationship between complementors and consumers.

Research on platform ecosystems argues from an economics perspective that platforms should refrain from a narrow focus in order to leverage strong network effects [48]. Research that is in favor of a limited focus mainly elaborates on the tension between quality and quantity of offered services (e.g., [20,49]). However, another perspective suggests that a narrow focus could provide domain-specific knowledge necessary for certain industries [5,47]. Especially in the context of SaaS-centric cloud platforms, the examination of such domain-specific settings has received little attention in research so far [50]. In this study, we provide support for the perspective arguing that a narrower focus on specific customer groups can mitigate restraining reactions of consumers. Future research should therefore investigate how to balance these two strategies and identify contingency factors to understand the effectiveness of these strategies in different circumstances.

Third, the elaboration on the factor ease of access and use unveils promising contemporary themes considering developments regarding integration and onboarding. Our findings suggest that cross-application integration represents a promising concept affecting ease of use and thereby platform gravitation. This is in line with related work that predicts that the demand for cloud-based integration is likely to increase due to the ongoing IT outsourcing (e.g., [51]). With regard to the case examples, avoiding isolated software services might become increasingly relevant. As stated by [52], this type of integration is likely to play an important role in the trajectory of cloud computing. As our findings show, the emergence of mashup tools might further increase in relevance. As such, mashup tools facilitate modeling of software services on the platform without programming and thereby increase ease of access and use [6]. This may represent a considerable means in order to promote innovation on the platform allowing users to develop add-ons specifically tailored to their business needs without coding. This field may therefore constitute a promising area for future design science research.

Fourth, we identify moderating effects of the platform type on the impact of the catalyzing and inhibiting factors. In this context, our study contributes to research on intra-platform competition. This phenomenon has been examined from different theoretical perspectives suggesting positive (e.g., [53]) and negative effects (e.g., [10]) on a platform ecosystem. Although platform settings are increasingly characterized by competition [21], especially the literature on multi-sided markets suggests that excessive competition represents a source of market failure [10]. Extant literature proposes secondary beneficial mechanisms of a platform owner's entry that may foster innovation activities [53]. The findings of our study suggest an inhibiting effect that is contingent upon the integration of competing software services into the platform core, e.g., if the platform owner integrates functionality of complementors' services into the platform core. Future research is needed to shed light on these dynamics and elaborate on potential contingencies.

15.5.2 Implications for practice

Since the success of a platform ecosystem strongly depends on its size [11], gravitation of a platform is highly important. Our findings may therefore provide several valuable implications for practice. First, it seems to be beneficial to increase the ease of access and use of a SaaS-centric cloud platform and thereby to consider all sides. In this context, the integration of software solutions plays an important role. Due to the importance of developers and customers in platform ecosystems, facilitating their access to and improving their experience on the platform may sustain competitive advantage. Second, this study highlights the importance of side-specific requirements of both developers (e.g., degree of development support required) and customers (e.g., desired portfolio, infrastructural transparency, compliance and legal issues). We recommend that practice takes this into consideration. Third, the degree of intra-platform competition might allude to an inherent challenge. As outlined above, a tension between control and diversity is existent in platform settings. Therefore, the situation of developers and the strategic focus of the platform owner should be continuously scrutinized. Fourth, we identify an inhibiting effect of contractual complexity, which underlines the need to focus on ownership structures and contractual configurations that may be beneficial for all ecosystem participants. Finally, by ensuring security and reliability regarding the technological foundation and the software services, platform owners might increase the participants' trust in their platform. Taking into account these recommendations may be beneficial for platform owners and may help SaaS-centric cloud platforms to gravitate.

15.5.3 Limitations

This study offers a few limitations. The first limitation concerns the applied sampling strategy. The selection of cases according to category resulted in relatively divergent cases. Although this allowed us to analyze very different strategies and to thereby learn from extremes, it may also limit the generalizability of our findings. In future studies, the focus might be set on specific categories in order to better understand within-group commonalities and differences. A second limitation is the number of interviews conducted per case. The selected interview partners were highly suitable for our study due to their sophisticated knowledge regarding both operational and strategic platform decisions. Nevertheless, the involvement of more than one representative of each platform could allow data triangulation. Integrating the perspectives of customers and complementors in the case analysis could further increase data validity and provide a more comprehensive view. Thus, the consideration of other ecosystem participants in future studies might reveal further valuable insights. Last, especially development-focused cloud platforms sometimes provide marketplaces for reusable components to support the development on the respective platforms. Examples of these distribution channels, which exhibit a focus on developers, are Engine Yard Add-Ons (https://addons.engineyard.com, accessed on August 15, 2017) and Heroku Elements (https://elements.heroku.com, accessed on August 15, 2017). Due to our focus on platforms that facilitate distribution and management of SaaS solutions, these examples have been neglected. Focusing on the specificities of these distribution channels might provide a fruitful avenue for future studies.

15.6 Conclusion

In today's digital age, the outsourcing of IT resources in the context of cloud computing is considered to be highly promising and has seen a strong proliferation. In this study, we

examined factors influencing the gravitation of SaaS-centric cloud platforms, i.e., their capability to attract and retain ecosystem participants. Based on four case studies, we identified four catalyzing factors (ease of access and use, security, portfolio significance, infrastructural flexibility and transparency) as well as three inhibiting factors (intra-platform competition, contractual complexity, lack of customer centricity) for SaaS-centric cloud platform gravitation. Our findings represent a starting point to support the academic conversation on cloud platforms, which are subject to constant change and development. Due to the ongoing trend of implementing domain-specific individual solutions instead of adopting one holistic software system, SaaS-centric cloud platforms might offer viable avenues to confront the increasing complexity of heterogeneous IT landscapes [54]. The demand of integration in such a heterogeneous software environment poses a significant contemporary challenge that still needs to be overcome. In fact, it is likely to constitute the key challenge for cloud platforms in our digital age.

References

1. KPMG and Bitkom. (2015). https://www.bitkom.org/sites/default/files/file/import/cloud-Monitor-2015-KPMG-Bitkom-Research.pdf
2. Yang, H., Tate, M.: A Descriptive Literature Review and Classification of cloud Computing Research. Communications of the Association of Information Systems. 31, 2, 35–60 (2012).
3. Ma, D., Seidemann, A.: Analyzing Software as a Service with Per-Transaction Charges. Information Systems Research. 26, 2, 360–378 (2015).
4. Giessmann, A., Legner, C.: Designing Business Models for cloud Platforms. Information Systems Journal. 26, 5, 551–579 (2016).
5. Hahn, C., Röhrer, D., Zarnekow, R.: A Value Proposition Oriented Typology of Electronic Marketplaces for B2B SaaS Applications. In: AMCIS 2015 Proceedings (2015).
6. Giessmann, A., Stanoevska-Slabeva, K.: Business Models of Platform as a Service (PaaS) Providers: Current State and Future Directions. Journal of Information Technology Theory and Application. 13, 4, 31–55 (2012).
7. Aydin, M.N., Perdahci, N.Z., Odevci, B.: cloud-Based Development Environments: PaaS. In: Murugesan, S., Bojanova, I. (eds.) Encyclopedia of cloud Computing, pp. 62–69. John Wiley & Sons. Ltd. Publishing, Chichester (2016).
8. Dellermann, D., Jud, C., Reck, F.: Understanding Platform Loyalty in the cloud: A Configurational View on ISV's Costs and Benefits. In: Proceedings der 13. Internationale Tagung Wirtschaftsinformatik (WI), pp. 514–528, Siegen (2017).
9. Tiwana, A.: Evolutionary Competition in Platform Ecosystems. Information Systems Research, 26, 2, 266–281 (2015).
10. Hagiu, A.: Strategic Decisions for Multisided Platforms. MIT Sloan Management Review, 55, 2, 71–80 (2014).
11. Schreieck, M., Wiesche, M., Krcmar, H.: Design and Governance of Platform Ecosystems – Key Concepts and Issues for Future Research. In: ECIS 2016 Research Papers (2016).
12. Ghazawneh, A., Henfridsson, O.: Balancing Platform Control and External Contribution in Third-Party Development: The Boundary Resources Model. Information Systems Journal. 23, 2, 173–192 (2013).
13. Tiwana, A., Konsynski, B., Bush, A.A.: Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics. Information Systems Research. 21, 4, 675–-687 (2010).
14. Huber, T.L., Kude, T., Dibbern, J.: Governance Practices in Platform Ecosystems: Navigating Tensions between Cocreated Value and Governance Costs. Information Systems Research. 28, 3, 563–584 (2017).
15. Wareham, J., Fox, P.B., Cano Giner, J.L.: Technology Ecosystem Governance. Organization Science. 25, 4, 1195–1215 (2014).
16. Hahn, C., Huntgeburth, J., Winkler, T.J., Zarnekow, R.: Business and IT Capabilities for cloud Platform Success. In: Proceedings of the 37th International Conference on Information Systems, pp. 1–20 (2016).

17. Hahn, C.: Digitalisierung der IT-Industrie mit cloud Plattformen – Implikationen für Entwickler und Anwender. HMD Praxis der Wirtschaftsinformatik. 53, 594–606 (2016).
18. Johns, G.: The Essential Impact of Context on Organizational Behavior. Academy of Management Review. 31, 2, 386–408 (2006).
19. Gawer, A.: Bridging Differing Perspectives on Technological Platforms: Toward an Integrative Framework. Research Policy. 43, 1239–1249 (2014).
20. Boudreau, K.J.: Let a Thousand Flowers Bloom? An Earlier Look at Large Numbers of Software App Developers and Patterns of Innovation. Organization Science. 23, 5, 1409–1427 (2012).
21. Ceccagnoli, M., Forman, C., Huang, P., Wu, D.J.: Cocreation of Value in a Platform Ecosystem: The Case of Enterprise Software. MIS Quarterly. 36, 1, 263–290 (2012).
22. Sarker, S., Sarker, S., Sahaym, A., Bjørn-Andersen, N.: Exploring Value Cocreation in Relationship Between an ERP Vendor and Its Partners: A Revelatory Case Study. MIS Quarterly. 36, 1, 317–338 (2012).
23. Kude, T., Dibbern, J., Heinzl, A.: Why Do Complementors Participate? An Analysis of Partnership Networks in the Enterprise Software Industry. IEEE Transactions on Engineering Management. 59, 2, 250–265 (2012).
24. Ghazawneh, A., Henfridsson, O.: A Paradigmatic Analysis of Digital Application Marketplaces. Journal of Information Technology. 30, 3, 198–208 (2015).
25. Yoo, Y., Henfridsson, O., Lyytinen, K.: The New Organizing Logic of Digital Innovation: An Agenda for Information Systems Research. Information Systems Research. 21, 4, 724–735 (2010).
26. Wang, N., Liang, H., Jia, Y., Ge, S., Xue, Y., Wang, Z.: cloud Computing Research in the IS Discipline: A Citation/Co-Citation Analysis. Decision Support Systems. 86, 35–47 (2016).
27. Kappelman, L., Nguyen, Q., McLean, E., Maurer, C., Johnson, V., Snyder, M., Torres, R.: The 2016 SIM IT Issues and Trends Study. MIS Quarterly Executive. 16, 1, 47–80 (2017).
28. Mell, P., T. Grance, T.: The NIST Definition of cloud Computing. Special Publication 800-145. National Institute of Standards and Technology, Gaithersburg, MD (2011).
29. Voorsluys, W., Broberg, J., Buyya, R.: Introduction to cloud Computing. In: Buyya, R., Broberg, J., Goscinski, A.M. (eds.) Hoboken. Cloud Computing Principles and Paradigms, pp. 3–42. John Wiley & Sons, Hoboken, NJ (2011).
30. Gawer, A., Cusumano, M.A.: Industry Platforms and Ecosystem Innovation. Journal of Product Innovation Management. 31, 3, 417–433 (2014).
31. Cusumano, M.A., Gawer A.: The Elements of Platform Leadership. MIT Sloan Management Review. 43, 3, 50–58 (2002).
32. Hilkert, D., Benlian, A., Hess, T.: Perceived Software Platform Openness: The Scale and Its Impact on Developer Satisfaction. In: Proceedings of the 32th International Conference on Information Systems, pp. 1–20 (2011).
33. Hurni, T., Huber, T.L.: The Interplay of Power and Trust in Platform Ecosystems of the Enterprise Application Software Industry. In: ECIS 2014 Proceedings, pp. 1–15 (2014).
34. Katz, M.L., Shapiro, C.: Systems Competition and Network Effects. Journal of Economic Perspectives. 8, 2, 93–115 (1994).
35. Evans, P.C., Gawer, A.: The Rise of the Platform Enterprise: A Global Survey. The Emerging Platform Economy Series No. 1. The Center for Global Enterprise. (2016). https://www.thecge.net/app/uploads/2016/01/PDF-WEB-Platform-Survey_01_12.pdf
36. Giessmann, A., Stanoevska, K.: Platform as a Service – A Conjoint Study on Consumer's Preferences. In: Proceedings of the 33th International Conference on Information Systems, pp. 1–20 (2012).
37. Floerecke, S., Lehner, F.: cloud Computing Ecosystem Model: Refinement and Evaluation. In ECIS 2016 Proceedings, pp. 1–14 (2016).
38. Tipler, P.A., Mosca, G., Wagner, J.: Physik: für Wissenschaftler und Ingenieure. Springer Spektrum, Heidelberg (2015).
39. Benbasat, I., Goldstein, D.K., Mead, M.: The Case Research Strategy in Studies of Information Systems. MIS Quarterly. 11, 3, 369–386 (1987).
40. Eisenhardt, K.M.: Building Theories from Case Study Research. The Academy of Management Review. 14, 4, 532–550 (1989).
41. Yin, R.K.: Case Study Research: Design and Methods. Sage, Thousand Oaks, CA (2003).
42. Menychtas, A., García-Gómez, S., Giessmann, A., Gatzioura, A., Stanoevska, K., Vogel, J., Moulos, V.: A Marketplace Framework for Trading cloud-Based Services. In: Vanmechelen, K.,

Altmann, J., Rana, O.F. (eds.) Economics of Grids, clouds, Systems, and Services. GECON 2011. Lecture Notes in Computer Science, vol. 7150, pp. 76–89. Springer, Heidelberg (2012).
43. Eaton, B., Elaluf-Calderwood, S., Sørensen, C., Yoo, Y.: Distributed Tuning of Boundary Resources: The Case of Apple's iOS Service System. MIS Quarterly. 39, 1, 217–243 (2015).
44. Benlian, A., Hilkert, D., Hess, T.: How Open Is This Platform? The Meaning and Measurement of Platform Openness from the Complementors' Perspective. Journal of Information Technology. 30, 3, 209–228 (2015).
45. Buyya, R., Pandey, S., Vecchiola, C.: cloudbus Toolkit for Market-Oriented cloud Computing. In: cloudCom 2009: Proceedings of the 1st International Conference on cloud Computing (2009).
46. Parniangtong, S.: Competitive Advantage of Customer Centricity. Springer Nature, Singapore (2017).
47. Hompel, M. ten, Rehof, J., Wolf, O.: cloud Computing for Logistics. Springer International Publishing, Basel (2015).
48. Rochet, J.C., Tirole, J.: Platform Competition in Two-Sided Markets. Journal of the European Economic Association. 1, 4, 990–1029 (2003).
49. Hagiu, A.: Quantity vs. Quality and Exclusion by Two-Sided Platforms. Harvard Business School Strategy Unit Working Paper No. 09-094, pp. 1–26 (2009).
50. Daniluk, D., Holtkamp, B.: Logistics Mall – A cloud Platform for Logistics. In: Hompel, M. ten, Rehof, J., Wolf, O. (eds.) cloud Computing for Logistics, pp. 13–27. Springer International Publishing, Cham (2015).
51. Ebert, N., Schlatter, U.: cloud-basierte Integration. Informatik-Spektrum. 40, 3, 278–282 (2017).
52. Fortiş, T.F., Munteanu, V.I., Negru, V.: Towards a Service Friendly cloud Ecosystem. In: 11th International Symposium on Parallel and Distributed Computing (ISPDC), pp. 172–179 (2012).
53. Foerderer, J., Kude, T., Mithas, S., Heinzl, A.: Does Platform Owner's Entry Crowd Out Innovation? Evidence from Google Photos. Information Systems Research. 29, 2, 444–460 (2018).
54. Hentschel, R., Leyh, C.: cloud Computing: Gestern, heute, morgen. HMD Praxis der Wirtschaftsinformatik. 53, 5, 563–579 (2016).