Received: 23 October 2015 | Revised: 10 December 2017 | Accepted: 29 December 2017

DOI: 10.1111/isj.12186

RESEARCH ARTICLE

WILEY

Knowledge boundaries in enterprise software platform development: Antecedents and consequences for platform governance

Jens Foerderer[1] | Thomas Kude[2] | Sebastian Walter Schuetz[3] | Armin Heinzl[1]

[1] University of Mannheim, Business School, Mannheim, Germany

[2] ESSEC Business School, Cergy-Pontoise, France

[3] Department of Information Systems, University of Arkansas, Fayetteville, AR, USA

Correspondence
Corrrespondence: J. Foerderer, University of Mannheim, Business School, Schloss, 68131 Mannheim, Germany.
Email: foerderer@uni-mannheim.de

Abstract

The widespread uptake of platform strategies turns many vendors of enterprise software into curators of an ecosystem of firms that collaboratively develop and commercialize a shared technology. As a platform owner's effectiveness in integrating knowledge across ecosystem participants will distinguish it from its competitors, we investigate the management of development-related knowledge across firm boundaries. Our exploratory, multiple-case study of 4 platforms illustrates how “knowledge boundaries” emerge between platform owners and complementors. We observe that knowledge boundaries are influenced by a platform's functional extent, interface design, and evolutionary dynamics, which create differences, dependencies, and novelty of development knowledge, resulting in qualitatively distinct types of knowledge boundaries. To overcome knowledge boundaries, platform owners provide various resources at the boundary, including information portals, documentation, helpdesks, and alignment workshops. We observe that in shaping these resources, platform owners face a trade-off between providing knowledge at the right scope, while allowing for the scalability of knowledge resources for the entire ecosystem. Depending on their scope and scale, we classify knowledge boundary resources as broadcasting, brokering, and bridging, each representing qualitatively distinct patterns in managing knowledge in platform ecosystems. We conclude with implications for researchers and managers.

KEYWORDS

case study, enterprise software, knowledge boundaries, knowledge boundary resources, platform ecosystems, platform governance

1 | INTRODUCTION

In organizing for technological innovation, an enterprise software vendor may choose to "open" its technology and allow third parties to participate in the technology's development and commercialization (Adner & Kapoor, 2010; Teece, 1986; Tiwana, Konsynski, & Bush, 2010). Such a "platform strategy" underlies the success of many of today's biggest vendors of enterprise software. In fact, in 2016, Gartner's 3 leaders in enterprise resource planning—Microsoft, Oracle, and SAP—pursued platform strategies (Gartner, 2015). Platform strategies are risky. If successful, platform owners enjoy enormous returns and competitive positions. If unsuccessful, huge financial losses may be incurred, as is evident from SAP's US$ 3 billion write-off of the platform Business ByDesign (Hesseldahl, 2013).

A substantial part of the risk associated with platform strategies arises from moving the locus of product development from within the firm toward independent third-party firms, so-called complementors (Boudreau, 2010; Boudreau & Lakhani, 2009; Gulati, Puranam, & Tushman, 2012). Complementors develop products and services on top of the platform. Platform owners seek to leverage the expertise of a diverse community of complementors that creatively develop novel capabilities unforeseen in the platform's original design (Nambisan, 2013; Tilson, Lyytinen, & Sørensen, 2010; Tiwana et al., 2010). Essentially, platform strategies require an enterprise software vendor to focus less on managing product and service development within their boundaries, focusing instead on the careful governance of complementors in order to profit from their development outcomes (Bergvall-Kåreborn & Howcroft, 2014; Eaton, Elaluf-Calderwood, Sorenson, & Yoo, 2015; Ghazawneh & Henfridsson, 2013; Iansiti & Levien, 2004).

A crucial issue caused by the shift of innovation outside an enterprise software vendor's boundaries involves the integration of development knowledge across firm boundaries. Given the complex nature of enterprise software, it is anything but trivial for complementors to develop add-ons. Thus, enterprise software vendors face the challenge of furnishing third parties with development knowledge in order to facilitate participation in development and innovation. Such knowledge often encompasses a holistic picture of the offered functionality (software development kits, libraries), system landscapes (eg, data warehouses, business processes), and business functions (eg, finance, human resources). In particular, complementors need to know how to access, combine, and extend platform functionality in order to develop add-on products and services (Bergvall-Kåreborn & Howcroft, 2014; von Hippel & Katz, 2002). In other words, platform strategies inherently impose "knowledge boundaries" between platform owners and complementors. If platform owners do not identify and address these knowledge boundaries effectively, the success of the platform strategy is likely to be endangered. Hence, acquiring platform-specific knowledge is one of the persistent problems for complementors (Bergvall-Kåreborn & Howcroft, 2014) and it has been argued that a platform owner's insufficient provision of knowledge resources is one of the key reasons that platforms fail (Van Alstyne, Parker, & Choudhary, 2016).

We study the antecedents of knowledge boundaries and their consequences for platform management in the context of enterprise software platform ecosystems. Although knowledge management has been subject to prior work on platform governance from the perspectives of boundary resources and boundary spanning (eg, Ghazawneh & Henfridsson, 2013; Huber, Kude, & Dibbern, 2017), platform engineering and design (eg, Baldwin & Woodard, 2008; Ulrich, 1995; West, 2003), and governance strategy (eg, Boudreau & Hagiu, 2009; Cusumano & Gawer, 2002; Shapiro & Varian, 1998), these studies do little to inform why complementors require knowledge and how platform owners convey knowledge to complementors. First, much prior work on platforms is conceptual (eg, Tiwana et al., 2010; Ulrich, 1995) or focused on mobile platforms (eg, Eaton et al., 2015; Ghazawneh & Henfridsson, 2013). Thus, we lack an understanding and related empirical evidence of knowledge management for technologically complex platforms such as enterprise software platforms. Second, although some studies focus on interactions across firm boundaries for the purpose of knowledge management (eg, Sarker, Sarker, Sahaym, & Bjørn-andersen, 2012), these studies have not yet considered the challenges faced by platform owners in managing knowledge boundaries for an unprecedented number of complementor firms (von Hippel & Katz, 2002). In fact, many of the techniques suggested by prior work for managing knowledge across boundaries within firms or between very few firms, such as the joint development of prototypes (Carlile, 2002, 2004) or site visits and joint teams (Carlile & Rebentisch, 2003), are not readily applicable in the one-to-many context of platform ecosystems. Finally, the contributions of work on boundary resources mostly relate to understanding how actions and reactions of platform owners and complementors shape resources at the boundary (Eaton et al., 2015; Huber et al., 2017), rather than what influences gaps in knowledge across the boundary and how platform owners address these gaps by shaping boundary resources and boundary-spanning activities. In sum, although prior work has alluded to knowledge boundaries in platform ecosystems, there is as yet no comprehensive investigation focusing on the context of enterprise software platforms.

To address this gap, we conducted an exploratory multiple-case study of 4 enterprise software platforms, which included 40 interviews with platform managers and the executives of complementor firms. Our analysis was guided by the framework of Carlile (2004), which helped us identify the specific causes and consequences of knowledge boundaries pertaining to enterprise software platforms. We found knowledge boundaries to be contingent on 2 technological properties of the platform—functional extent and interface design. Particularly, we found that changes to the platform's functional extent or interface design broadened knowledge boundaries. We identified various knowledge boundary resources (KBR) that platform owners provisioned in response to broadened boundaries. These include information portals, interface documentations, massive open online courses, communities of practice, helpdesk accounts, and alignment workshops. More importantly, we found that these KBR differ in terms of scope (how much of the gap in knowledge they overcome) and scale (how many complementors they can address). Depending on their scope and scale, we propose 3 broad types of KBR: broadcasting, brokering, and bridging.

Taken together, the findings of this study contribute to our understanding of the challenges enterprise software platform owners face when introducing changes to their platform, and their considerations when choosing to provide broadcasting, brokering, or bridging type KBR to overcome knowledge boundaries.

2 | BACKGROUND

2.1 | Platform governance and knowledge management

Choosing a platform strategy is one popular approach organizations use to develop and commercialize digital products (Teece, 1986), as evidenced by corporations such as Apple or Microsoft pursuing such a strategy for key product lines. *Platforms* are commonly defined as systems that match adopters of the system with firms that provide complements to the system (Parker, Van Alstyne, & Choudhary, 2016). These *complements* are products or services that make the focal product or service more attractive (Rochet & Tirole, 2003; Shapiro & Varian, 1998). Given that platforms are subject to network externalities (Katz & Shapiro, 1994; Parker & Van Alstyne, 2005) and winner-takes-all dynamics (Rochet & Tirole, 2003), platform owners' activities go beyond designing, developing, and distributing the platform itself and also include nontrivial decisions regarding platform governance (Boudreau & Hagiu, 2009; Tiwana, 2013; Tiwana, 2015; Wareham, Fox, & Cano Giner, 2014). Following prior work, we use the term platform governance to refer to the fundamental decisions of platform owners with regards to the ecosystem of complementors (Gawer, 2014; Tiwana, 2013; Tiwana et al., 2010). Platform governance encompasses decisions regarding the ownership of the platform (Boudreau, 2010), platform owner's entry into complementary markets (Gawer & Henderson, 2007), or community-building activities (Cusumano & Gawer, 2002).

One aspect of platform governance pertains to integrating development-related knowledge across firm boundaries (Bergvall-Kåreborn & Howcroft, 2014). In particular, complementors need to know how to access, combine, and extend platform functionality in order to develop add-on products and services (Bergvall-Kåreborn & Howcroft, 2014; von Hippel & Katz, 2002), which encompasses platform functionality (eg, software development kits, libraries), system landscapes (eg, data warehouses, business processes), and business functions (eg, finance, human resources). Hence, successful platform owners govern the integration of knowledge across firm boundaries (Van Alstyne et al., 2016).

Recognizing the importance of providing complementors with requisite knowledge, 3 broad research streams in the related platform literature have touched upon knowledge management in the context of platforms. The first stream relates to platform governance with a focus on interactions at the technological and organizational boundaries between platform owners and complementors (eg, Eaton et al., 2015; Ghazawneh & Henfridsson, 2013; Huber et al., 2017). Work in this stream argues that governance outcomes are the result of complex and dynamic sociotechnical negotiations at the interface between platform owners and complementors. Therefore, platform governance might best be understood by studying the interactions at the boundary between platform owners and complementors in detail (Eaton et al., 2015; Ghazawneh & Henfridsson, 2013; Huber et al., 2017; Sarker et al., 2012). These studies have, for instance, documented that boundary resources (Eaton et al., 2015; Ghazawneh & Henfridsson, 2013) and boundary-spanning activities of platform owners (Huber et al., 2017; Sarker et al., 2012) play important roles in influencing complementor outcomes. For example, prior work studies the broader concept of boundary resources, which are referred to as "the software tools and regulations that serve as the interface for the arm's length relationship between the platform owner and the application developer" (Ghazawneh & Henfridsson, 2013, p. 174), such as application programming interfaces or software development kits (Eaton et al., 2015; Ghazawneh & Henfridsson, 2013). Other studies describe different forms of boundary-spanning activities, which may result in arm's length or in more dyadic relationships between platform owners and complementors (Huber et al., 2017).

The second stream considers knowledge management as a part of platform governance from a design and engineering perspective (Baldwin & Woodard, 2008; Carlile, 2002; Ulrich, 1995; West, 2003). Work in this stream recognizes platforms as a type of product architecture that focuses on the systematic reuse of a shared set of building blocks across different products with the aim of enabling economies of scope (Gawer, 2014). These studies focus on particular architectural aspects of platforms, such as modularity and interface design, and how a careful design of the platform's architecture may influence governance costs (Banker, Davis, & Slaughter, 1998; Brown & Eisenhardt, 1995; Brusoni & Prencipe, 2006; Henderson & Clark, 1990), also in terms of reducing the need to manage knowledge across boundaries (Brusoni & Prencipe, 2006; Furlan, Cabigiosu, & Camuffo, 2014; Henderson & Clark, 1990). These studies suggest that architectural characteristics of the platform technology may influence differences in knowledge between platform owners and complementors.

The third stream includes strategy-related work on optimal forms of platform governance, including questions surrounding knowledge management. The debate in this stream is characterized by 2 paradigms. One paradigm—often advanced in economics-oriented research on platforms—is that platform owners profit most from complementary innovation if their governance stimulates competition among or with complementors (Armstrong, 2006; Rochet & Tirole, 2003; Shapiro & Varian, 1998). The presumption is that competition forces complementors to lower their prices and to continuously differentiate.[1] The opposing paradigm—advanced in management and information systems research on platforms—is that competition has little value in the long run. Instead, platform owners should focus their efforts on cooperation, in terms of investing in long-term "developer relations" (Boudreau & Lakhani, 2009). This argument is based on evidence that competition might crowd out complementors, due to complementors failing to monetize their products (Boudreau, 2012) or because complementors may fear value appropriation by the platform owner (Ceccagnoli, Forman, Huang, & Wu, 2012; Huang, Ceccagnoli, Forman, & Wu, 2013). In light of this discussion, platform owners' investment in overcoming knowledge boundaries represents one form of cooperating with complementors.

Taken together, the platform literature suggests that knowledge boundaries can be overcome with boundary resources and boundary-spanning activities (eg, Eaton et al., 2015; Ghazawneh & Henfridsson, 2013; Huber et al., 2017), that knowledge boundaries are a consequence of platform design (eg, Baldwin & Woodard, 2008; Carlile, 2002; Ulrich, 1995; West, 2003), and that governance decisions pertaining to the platform owner-complementor relationship are of strategic relevance (eg, Boudreau & Hagiu, 2009; Cusumano & Gawer, 2002; Shapiro & Varian, 1998). These findings notwithstanding, it is thus far not clear (1) what influences gaps in knowledge across the boundary and (2) how platform owners overcome these gaps through boundary resources and boundary-spanning activities. The

[1]Shapiro and Varian (1998, p. 279) recommend that platform owners "encourage a […] competitive market for complements," and, if necessary, also actively "compete with your installed base."

lack of research on these 2 questions may be partially explained by the focus of earlier work on consumer platforms, such as Apple's iOS and Google's Android mobile operating systems (eg, Eaton et al., 2015; Ghazawneh & Henfridsson, 2013; Tiwana, 2015). Arguably, enterprise software platforms are complex technological systems that cover a broad range of functionalities, architectures, and business lines, such as customer analytics or large enterprise resource planning systems. This is in contrast to consumer platforms, which naturally cover a narrow set of functionality (eg, Ghazawneh & Henfridsson, 2013). Hence, such technological complexity may have implications for knowledge boundaries that remain unidentified in related literature. Furthermore, enterprise software platforms differ in the variety of platform owner-complementor relationships, and prior work on enterprise software platforms documents complex forms of interactions between platform owners and complementors (Huber et al., 2017; Sarker et al., 2012; Wareham et al., 2014). For example, Sarker et al. (2012) observe different “modes of co-creation” between platform owners and complementors. Wareham et al. (2014) found that platform owners discriminate between complementors, with high performing complementors receiving more extensive support.

As platform strategies have been increasingly adopted in the enterprise software industry to address highly specific customer needs and profit from outside innovation (eg, Ceccagnoli et al., 2012; Huang et al., 2013; Wareham et al., 2014), we deem it important and timely to address the issue of knowledge management in the context of enterprise software platforms. To that end, our study of enterprise software platforms offers unique and valuable insights into the sources and consequences of knowledge boundaries that have remained outside of the scope of extant consumer platform research.

2.2 | Research framework: Knowledge boundaries in Enterprise software platform ecosystems

The problem of integrating knowledge across firm boundaries has been subject to various theoretical frameworks on knowledge management. The presumption that a firm's effectiveness in integrating knowledge will distinguish it from its competitors has been an important insight in explaining organizational outcomes (Grant, 1996). In understanding the problems of knowledge integration, the tacit nature of knowledge (Nonaka, 1994; Polanyi, 1966; Von Krogh et al., 2000), its stickiness (von Hippel & Katz, 2002; von Hippel, 1998), and the difficulty of transferring it (Szulanski, 1996) have emerged as widely accepted explanations.

To guide our subsequent empirical exploration, we adapt Carlile's (2004) framework that has proved to be useful in a similar, yet different context (see Figure 1). The problem of integrating knowledge across boundaries in platform ecosystems is characterized by 2 or more firms seeking to exchange knowledge with each other and encountering difficulties due to varying properties of knowledge at the boundary. The goal of both platform owners and complementors is to integrate knowledge in order to maximize their profits from product development and

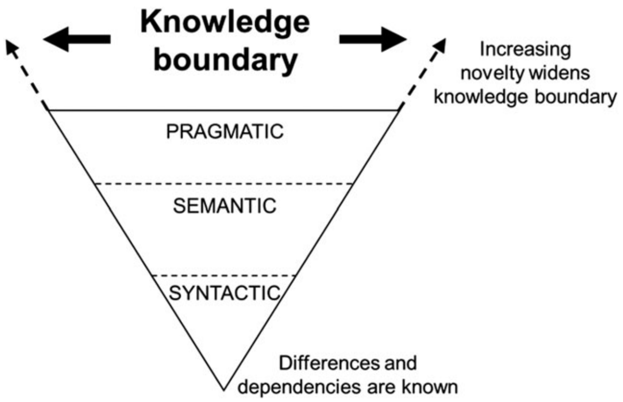
Knowledge boundary
Increasing novelty widens knowledge boundary
PRAGMATIC
SEMANTIC
SYNTACTIC
Differences and dependencies are known

FIGURE 1 Knowledge boundary framework (adapted from Carlile, 2004)

innovation. This problem is subject to Carlile's (2004) framework. In the context of knowledge boundaries within a firm, Carlile (2004) examines the knowledge boundaries between different groups of a firm that are engaged in new product development. The proposed framework qualitatively distinguishes different types of knowledge boundaries and offers guidelines for actors to handle the resulting complexity by borrowing the notion of boundary objects (Star & Griesemer, 1989). This framework suits our exploration because it helps structure the causes of differences in knowledge across firm boundaries, because it structures the set of approaches for overcoming knowledge boundaries. Furthermore, it is related to product development research, which fits with our context of platforms. This framework has been widely adopted in the literature (eg, Kellogg, Orlikowski, & Yates, 2006; Rosenkranz, Vranesic, & Holten, 2014). In the following, we present key aspects of the framework and derive guiding questions for the empirical exploration of knowledge boundaries in enterprise software platform ecosystems.

2.2.1 | Differences, dependencies, and novelty of knowledge at the boundary

The first aspect of Carlile's knowledge boundary framework is that knowledge boundaries emerge from different properties of knowledge that are present at the boundary between platform owners and complementors. Figure 1 illustrates the knowledge boundary schematically. Carlile (2004) describes 3 properties of knowledge at the boundary. *Differences* refer to the degree to which the *domain knowledge* of different actors deviates from each other (Carlile, 2004). Transferred to our context, one example of differences in knowledge is the specificity of knowledge. Whereas some firms have highly specific knowledge about enterprise software and processes in one industry, other firms have accumulated specific knowledge associated with a completely different set of industries and technologies.

Dependencies describe linkages across different domains of knowledge of stakeholders involved in developing the product (Carlile, 2004). For example, complementors may require knowledge on a specific aspect of platform development, such as knowledge about particular software development kits, application programming interfaces, and libraries. Differences and dependencies are not independent from each other because without dependencies, differences would be of no consequence (Carlile & Rebentisch, 2003).

Differences and dependencies of knowledge are particularly consequential when knowledge at the boundary changes (Carlile, 2004). *Novelty* of knowledge describes the recency with which circumstances at the boundary change (Carlile & Rebentisch, 2003), for example, in response to changing customer requirements. To illustrate the role of novelty, consider Figure 1. At the origin, platform owners and complementors share the same knowledge and encounter no difficulties in transferring knowledge, as both platform owners and complementors are aware of all differences and dependencies of the knowledge. As novelty increases, gaps in knowledge that are of syntactic, semantic, and pragmatic nature emerge. *Syntactic* gaps in knowledge are present when actors lack a shared vocabulary and grammar to integrate knowledge. For example, actors may have their own jargon or vocabulary to refer to specific elements of functional or technological knowledge. *Semantic* gaps in knowledge are present when actors share a common syntax, but understand or interpret the meaning of exchanged communication messages differently. Knowledge is often context-specific and experiential so that a message can be exchanged but will not be sufficiently understood by the communication partner (Inkpen & Dinur, 1998; Kellogg et al., 2006; Kogut & Zander, 1992). Consequently, difficulties arise because of differences in meanings, assumptions, and contexts (Carlile, 2002). Shared ontologies may serve as an instrument to reduce semantic gaps between actors. *Pragmatic* gaps in knowledge arise when actors share a common syntax and understanding, but draw different implications or conclusions from it. For example, negative customer feedback on the usability of a complement may trigger different actions on the pragmatic level. While the platform owner may consider to fine-tune its indexing method, the complementor may react to the feedback by improving the graphical user interface. Thus, incongruent actions may happen due to differing beliefs, interests, and strategies related to the integration of knowledge.

Not addressing syntactic, semantic, and pragmatic boundaries may hinder successful product development (Carlile, 2002, 2004; Kellogg et al., 2006). Although Carlile (2004) does not directly refer to knowledge boundaries as syntactic, semantic and pragmatic but rather as differences or gaps in syntactic, semantic and pragmatic knowledge across boundaries, we will denote them in line with prior work that adopted Carlile's (2004) framework (eg, Kellogg et al., 2006, Rosenkranz et al., 2014) as *syntactic, semantic, and pragmatic boundaries* in platform development. We believe this simplification is sound because differences on the respective semiotic level create distinct knowledge barriers which can be directly referred to as syntactic, semantic, or pragmatic boundaries.

Despite the importance of differences, dependencies, and novelty of knowledge for platform development, research on their antecedents is scarce. In fact, Carlile (2002) outlines that the causes of differences, dependencies, and novelty are specific to the product development setting. For example, in Carlile's (2004) context of new product development in automotive companies, differences and dependencies emerged from the particular engineering processes and the involved groups of actors, such as engine design, vehicle styling, and safety. Based on the above review of literature, the unique structuring of enterprise software platform development, in terms of a shared technological enterprise software representing the infrastructure for product development among independent firms, is likely to go beyond prior work on knowledge boundaries in within-firm contexts. While the above reviewed literature on platform engineering and design suggests a considerable influence of architectural characteristics of the platform on knowledge management costs, prior work has not yet explored how architectural characteristics of a platform influence properties of knowledge at the boundary. Therefore, the first goal of our study is to understand the unique antecedents of differences, dependencies, and novelty of knowledge in enterprise software platform development.

Guiding question 1. *What are the antecedents of differences, dependencies, and novelty of knowledge at the boundary between enterprise software platform owners and complementors?*

2.2.2 | Approaches to overcome knowledge boundaries

The second proposition of Carlile's (2004) framework is that syntactic, semantic, and pragmatic boundaries have consequences for the design of objects and activities to overcome these boundaries and enable effective product development outcomes. We refer to objects and activities employed by platform owners in order to overcome knowledge boundaries and enable effective product development outcomes as KBR, both in relation to the work on platform governance from a boundary perspective as well as regarding the notion of boundary objects and boundary spanners (ie, human resources that enable boundary spanning) in the sociological literature (Star & Griesemer, 1989). Boundary objects refer to "artifacts that are plastic enough to adapt to local needs and constraints of the several parties employing them, yet robust enough to maintain a common identity across sites" (Star, 1989, p. 393). Such boundary objects include physical prototypes (Bechky, 2003; Carlile, 2002), communication technology (Kellogg et al., 2006), and reporting forms (Star & Griesemer, 1989). Boundary spanners are human resources employed by an "innovation unit to gather information from and transmit information to several external domains" (Tushman, 1977, p. 587). Examples of boundary spanners are knowledge experts or account managers (Tushman & Scanlan, 1981).

Although Carlile (2002) offers various empirical examples of KBR, such as joint prototyping, the question of how platform owners of enterprise software platforms may overcome knowledge boundaries has not yet been answered. KBR in enterprise software platform development may substantially differ from the within-firm product development context described in Carlile (2004) for 2 reasons. First, platform development takes place across firm boundaries, which confronts platform owners with the difficulty of determining the appropriate scope of knowledge boundaries to be provided at the boundary. For example, Wareham et al. (2014) and Sarker et al. (2012) have documented different types of relationships between platform owners and complementors that are also characterized by different degrees of provided knowledge. Second, platform development is characterized by a large number of firms participating in development (Gawer, 2014; McIntyre & Srinivasan, 2017), which introduces difficulties in scaling KBR for platform owners. For example, Apple's ecosystem currently comprises more than 300 000 app developers, and SAP has more than 10 000 development partners on its enterprise software platforms. Therefore, a reinvestigation of Carlile (2004) with regards to consequences of syntactic, semantic, and pragmatic boundaries for shaping KBR may offer insights into product development that spans an unprecedented number of complementor firms rather than product development within a single firm.

Guiding question 2. *What are the consequences of syntactic, semantic, and pragmatic boundaries for platform governance, in terms of how do enterprise software platform owners overcome these boundaries?*

3 | METHOD

3.1 | Research design

Recognizing the lack of in-depth field studies on knowledge boundaries in platforms on the one hand, and comparative studies on the other hand, we conducted an exploratory qualitative field study of 4 enterprise software platform ecosystems (Eisenhardt, 1989). The enterprise software industry is highly concentrated and dominated by a few players, such as Microsoft, Oracle, Salesforce, and SAP. We selected 2 of the largest firms in this industry, which we refer to as platform owner 1 (PO1) and platform owner 2 (PO2). PO1 offers software solutions for a variety of industries, regions, and customer needs. At the time of our study, PO1 had more than 70 000 employees and revenues exceeding US$20 billion. PO2 covers consumer and business software. PO2 had more than 100 000 employees and generated more than US$80 billion in revenue.

Our aim was to identify platforms of PO1 and PO2 that serve as distinct cases, allowing us to contrast different knowledge boundaries (cf. Fichman, 2004). Thus, we searched for platforms that exhibited sufficient reach to study various complementors as well as a heterogeneous platform technology. Our search was supported by data from http://isvworld.com—a global software industry database providing detailed information on software platforms and complementors. We collected freely available documents and descriptions from http://isvworld.com, including documentations of platform design, training certifications, and presentations. We refer to the selected platforms by their pseudonyms MOBILE, ERP, CRM, and BIGDATA. MOBILE and BIGDATA are platforms of PO1, CRM and ERP are platforms of PO2. Table 1 briefly describes the selected platforms.

3.2 | Data collection

To collect data, we conducted a large-scale interview study with executives of PO1 and PO2, as well as 38 complementors affiliated with MOBILE, ERP, CRM, and BIGDATA, between October 2013 and June 2014. Our interview study began with key executives (vice presidents) of PO1 and PO2 (henceforth referred to as VP PO1 and VP PO2) to gain an in-depth understanding of the research problem and platform contexts. We identified complementors across the 4 platforms MOBILE (n = 11), BIGDATA (n = 11), CRM (n = 8), and ERP (n = 8), through a mix of

TABLE 1 Description of cases

MOBILE
MOBILE is a platform for enterprise applications that connects employees, mobile devices, and business needs. It covers functionality for the exchange of data between traditional back-end systems and multiple MOBILE devices. MOBILE was designed as a layer on top of a number of enterprise software systems and provided a simplified methodology to view and retrieve data from these systems in way that is convenient for handheld devices, including mobile phones and tablets.
ERP
ERP is an enterprise resource planning platform geared toward the core requirements of midsized organizations and subsidiaries of larger enterprises, specifically assisting with the management of finance, supply chain, manufacturing, sales, human resources, and projects. In the context of ERP, vertical and horizontal add-on solutions are distinguished.
CRM
CRM is an integrated customer relationship management platform targeted at small and medium-sized businesses. CRM was established as a platform for increasing the sales and marketing efficiency of companies regardless of size and industry. Central to CRM are add-ons focused on various industries and functional requirements, including business intelligence, social insights, and campaign management.
BIGDATA
BIGDATA is an in-memory, real-time analytics platform and supports advanced algorithmic text mining features, such as unstructured data analysis. Complementors can build persistence or analytic models, server-side or line-of-business logic, or user interfaces for displaying data.

representative sampling based on their size, industry, and location, as well as snowball sampling based on recommendations of our interview partners.[2] We concluded data collection when we reached theoretical saturation on the topics of interest. Table 2 describes our sample of complementors. The complements developed by these firms include specialized geo-information systems for the utility sector, cross-industry HR solutions, tools for back-end integration of global supplier networks, waste management solutions, and applications that support customer contract negotiations.

Table A1 in the Appendix shows the pre-formulated, high-level interview guidelines we used. Two researchers conducted the interviews; one was responsible for guiding the interview, and following the guidelines, and the other one took notes and asked questions based on the respondent's answers (Miles & Huberman, 1994). Interviews were conducted face-to-face or via phone, each involving either the chief executive officers (CEO) or the chief product officers (CPO) of the particular complementor. We chose the CEOs and CPOs of these firms as interview partners because they were responsible for coordinating add-on development. For each platform, we conducted the interviews within a 2-month time frame (Miles & Huberman, 1994). We recorded and transcribed each of the interviews. The interviews lasted about 60 minutes on average and varied from 45 to 150 minutes.

We complemented our interviews with archival data available in the form of internal documents, presentations, and product roadmaps of the particular platforms whenever our informants gave us access to these documents (Eisenhardt, 1989). For MOBILE and BIGDATA, one researcher also attended 2 annual developer conventions in order to get an overview of the relationship between the platform owner and complementors and the general interactions between the platform owner and complementors.[3]

3.3 | Data analysis

We started by openly coding a priori-defined concepts in Carlile's (2004) framework: differences, dependencies, and novelty of knowledge. Table 3 offers definitions of the a priori constructs. Open coding involved attaching descriptive conceptual labels to interview statements and documents, while refining the properties and dimensions of each concept using the constant comparison technique (Charmaz, 2014; Corbin & Strauss, 1990). Analyses related to our guiding questions began with a within-case perspective, and then continued with comparisons of interpretations across cases in order to arrive at a coherent understanding of how differences, dependencies, and novelty of knowledge were manifested. Two authors began coding the interviews independently, but discussed their understanding of the coded passages early on, continuously aligning and refining their understanding of the a priori constructs (Miles & Huberman, 1994).

Once differences, dependencies, and novelty of knowledge were identified, we began to identify patterns in the data regarding our guiding questions. For question 1, we coded data for direct or indirect rationales given by interviewees for the experienced differences, dependencies, or novelty of knowledge. This procedure was based on open and axial coding (Charmaz, 2014, Corbin & Strauss, 1990). We iterated over our data several times with the goal of identifying new patterns, in terms of recurring rationales or emergent constructs that related to differences, dependencies, or novelty. To achieve theoretical distinction between emergent constructs, the coders continuously aligned their understanding of the coded passages and the meanings of the emergent constructs using written down definitions and empirical examples. Table 3 provides the agreed upon definitions. We systematically compared the

[2]For example, one interviewee described the situation of a complementor that had struggled with a new release of the ERP platform because the new release obsoleted parts of the complementor's business. In such cases, we asked interviewees to provide us with the contact data of the particular complementor to allow for further investigations.

[3]We addressed informant bias in various ways. First, we followed interview guidelines that focused on chronologically reflecting objective events and facts of the exchange between platform owners and complementors. Second, we gathered documents from the corresponding organizations and the public media regarding the particular platforms. Third, we collected data in real time and maintained contact with the interviewees for several months after the study. This provided us with a longitudinal perspective on the cases. Finally, we promised confidentiality in order to motivate informants' accuracy.

TABLE 2 Sample of complementors

No.	Platform	Region	Firm Size	Interviewee(s)
1	BIGDATA	APJ	Small	CEO
2	BIGDATA	CEE	Medium	CPO
3	BIGDATA	EMEA	Small	CEO, CPO
4	BIGDATA	EMEA	Small	CPO
5	BIGDATA	EMEA	Small	CEO
6	BIGDATA	EMEA	Small	CEO and CPO
7	BIGDATA	NA	Medium	CEO and CPO
8	BIGDATA	NA	Medium	CEO and CPO
9	BIGDATA	NA	Small	CEO
10	BIGDATA	NA	Small	CPO
11	BIGDATA	NA	Small	CEO and CPO
12	CRM	EMEA	Medium	CEO
13	CRM	EMEA	Medium	CEO
14	CRM	EMEA	Small	CEO/CPO
15	CRM	EMEA	Small	CEO
16	CRM	EMEA	Small	CEO
17	CRM	EMEA	Small	CEO
18	CRM	EMEA	Small	CPO
19	CRM	EMEA	Small	CEO/CPO
20	ERP	EMEA	Large	CEO
21	ERP	EMEA	Medium	CEO/CPO
22	ERP	EMEA	Medium	CEO
23	ERP	EMEA	Medium	CEO
24	ERP	EMEA	Small	CEO
25	ERP	EMEA	Small	CEO/CPO
26	ERP	EMEA	Small	CEO
27	ERP	EMEA	Small	CEO
28	MOBILE	APJ	Medium	CEO
29	MOBILE	APJ	Small	CEO/CPO
30	MOBILE	CEE	Large	CEO and CPO
31	MOBILE	CEE	Medium	CEO/CPO
32	MOBILE	CEE	Small	CEO/CPO
33	MOBILE	EMEA	Large	CEO
34	MOBILE	EMEA	Small	CEO/CPO
35	MOBILE	EMEA	Small	CEO/CPO
36	MOBILE	NA	Medium	CEO and CPO
37	MOBILE	NA	Medium	CEO
38	MOBILE	NA	Small	CPO

Note: The table describes the empirical sample of our interview study. APJ stands for Asia-Pacific, CEE for Central and Eastern Europe, EMEA for Europe, the Middle East and Africa, and NA for North America. CEO stands for Chief Executive Officer and CPO for Chief Product Officer. Firm size is coded based on EU recommendation 2003/361 and staff headcount (small<50, medium<250, otherwise large).

emergent constructs within and across cases using replication logic (Miles & Huberman, 1994). The constructs that emerged in this process are functional extent, interface design, and evolutionary dynamics (see Table 4).

TABLE 3 Definition of a priori and emergent constructs

Construct	Definition
A priori-defined (adapted from Carlile, 2004)	
Differences	The degree to which the domain knowledge of different actors deviates from each other.
Dependencies	The degree to which linkages across different domains of knowledge of stakeholders involved in developing the product exist.
Novelty	The degree to which knowledge changes.
Syntactic boundary	Situations in which actors at the boundary lack a shared terminology (ie, a communication device) for integrating knowledge.
Semantic boundary	Situations in which actors at the boundary lack a shared meaning for integrating knowledge.
Pragmatic boundary	Situations in which actors at the boundary lack shared interests for integrating knowledge.
Emergent	
Knowledge boundary resources	Objects and activities employed by platform owners in order to overcome knowledge boundaries and enable effective product development outcomes.
Scope	The extent of information, skills, and organizational capability provided by knowledge boundary resources to address syntactic, semantic, and pragmatic boundaries.
Scale	The extent to which a knowledge boundary resource can accommodate the demands of complementors.
Functional extent	The degree and depth of core functionality that a platform offers for reuse and recombination.
Interface design	The degree to which a platform implements open or proprietary standards in specifying complementors' interactions with the platform technology.
Evolutionary dynamics	The rate of change with which the platform technology changes.

TABLE 4 Causes of differences, dependencies, and novelty of knowledge at the boundary between platform owners and complementors

Properties of the Knowledge Boundary (Carlile, 2004)	Theoretical Causes of Knowledge Boundary Properties	Empirical Examples
Differences and dependencies in knowledge	Functional extent	● platform for ERP vs platform for mobile applications
	Interface design	● HTML5 (open) vs Silverlight (proprietary) ● OData (open) vs solidDB (proprietary)
Novelty of knowledge	Evolutionary dynamics	● version releases ● acquisitions ● technological change ● integrations ● policy changes ● regulatory requirements

Note: Please refer to Table 3 for the definitions of the constructs.

Regarding guiding question , we followed a 3-step procedure. First, we openly coded data related to Carlile's (2004) theoretical approaches to overcome differences, dependencies, and novelty of knowledge. Table 3 defines the a priori constructs. This process yielded empirical examples of how platform owners sought to overcome knowledge boundaries. Second, we used open and axial coding to identify distinct categories of specific resources provided by the platform owners to overcome knowledge boundaries (Corbin & Strauss, 1990). Over several iterations—under constant comparison within cases and across cases as well as the continuous alignment of coders—scale and scope emerged as key dimensions of KBR (see Table 3), resulting in broadcasting, brokering, and bridging as 3 qualitatively distinct categories of KBR (see Table 5). The last step of our analysis involved exploring which KBR platform owners employed to manage syntactic, semantic, and pragmatic boundaries. To this end, we relied on pattern matching (Miles & Huberman, 1994), ie, we compared the prevalence of a certain type of boundary with the KBR employed by

TABLE 5 Platform owners' approaches to addressing knowledge boundaries

	Broadcasting	Brokering	Bridging
Description	Standardized, based on resources that provide knowledge via transferable objects or through storage in centralized databases. Accessible by complementors without having to interact with the platform owner.	Intermediate type of knowledge boundary resources that provides meta-knowledge in terms of knowing where complementors may obtain help. Implemented via dedicated, semi-formalized interaction. Brokering mediates between platform owner and complementors.	Individualized mode of knowledge boundary resources based on intensive, frequent, and sometimes informal exchanges between platform owners and developers. Organizational members of the platform provider form a direct link with organizational members of complementors.
Scale	Entire ecosystem of complementors	Subset of complementors	Individual complementor
Scope	Factual, generally applicable knowledge, not necessarily problem specific	Meta-knowledge, gives direction on where/how to obtain problem solution	Specific problem-solving capabilities
Empirical examples	● technical documentation ● information portals ● handbooks ● sample code ● Modelling guidelines ● massive online courses ● communities of practice ● hosted developer sandbox	● helpdesks ● account manager	● one-to-one assistance ● technological coaching ● co-innovation activities ● alignment workshops between third-party developers and platform owners

platform owners—both across platforms and within platforms. During this step, we also identified exemplary passages that we use to illustrate the co-occurrence of a boundary and a KBR.

We presented our findings at a practitioner workshop on software platforms in which we received verbal validation for the contextualization of concepts and the identified patterns. The reactions and discussions increased our confidence in the internal and external validity of the insights we had gained. The following section discusses our findings by drawing on detailed empirical evidence. With the exception of patterns that relate to platform owners' management of pragmatic boundaries, the findings we report are based on multiple instances across the 4 studied platforms.

4 | FINDINGS

Tables 4 and 5 summarize the key findings of our study. Table 4 shows our main findings regarding the question of what influences differences, dependencies, and novelty of knowledge at the boundary between platform owners and complementors. We find that differences and dependencies of knowledge are closely connected to properties of the technology at the basis of interfirm collaboration—in particular, the platform's functional extent and interface design. We observe that a platform's functional extent and its interface design are main drivers of differences and dependencies in knowledge at the boundary between platform owners and complementors. Differences and dependencies become most salient when circumstances at the boundary change. We observe such novelty to be driven by the evolutionary dynamics of the platform. Evolutionary dynamics refer to general changes in the technological trajectory of a platform, such as platform updates, technological integrations, or regulatory changes. Evolutionary dynamics create syntactic, semantic, and pragmatic boundaries between platform owners and complementors that require specific approaches in order to be overcome.

Table 5 shows our main findings with regard to the question of how platform owners address syntactic, semantic, and pragmatic boundaries, which emerge from the functional extent, interface design, and evolutionary dynamics of a platform. We observe that platform owners face a trade-off between providing knowledge at the right scope, while allowing for the scalability of knowledge resources for the entire ecosystem. Depending on their scope and scale, we classify KBR as broadcasting, brokering, or bridging, with each category representing qualitatively distinct patterns in managing knowledge in platform ecosystems. We further find that platform owners provide various resources to overcome knowledge boundaries, including information portals, handbooks, documentation, helpdesks, and alignment workshops. Differences and dependencies in knowledge lead to syntactic boundaries between platform owners and complementors, which platform owners address through “broadcasting”—a particular type of KBR characterized by high scalability but limited scope. Evolutionary dynamics lead to semantic and pragmatic boundaries, which platform owners address through the KBR “brokering” and “bridging.” Brokering and bridging represent interactive boundary-spanning activities that platform owners use in addition to broadcasting. Brokering, and particularly bridging, facilitate broader scope but are limited in terms of their scalability.

In what follows, we present our findings with regard to the goals of this study: to observe (1) what influences differences, dependencies, and novelty of knowledge at the boundary between platform owners and complementors, and (2) how platform owners address the resulting syntactic, semantic, and pragmatic boundaries.

4.1 | How do differences, dependencies, and novelty of knowledge emerge at the boundary between platform owners and complementors?

4.1.1 | Differences and dependencies in knowledge

Our analyses substantiated our a priori assertion that knowledge boundaries in the context of platform ecosystems closely relate to properties of the platform's underlying technology. Our findings suggest that 2 specific technological characteristics of platforms influenced differences and dependencies of knowledge between platform owners and complementors—functional extent and interface design. First, differences in knowledge were influenced by a platform's *functional extent*. We refer to functional extent as the degree and depth of core functionality that a platform offers for reuse and recombination. To develop add-on solutions that provide value to clients, complementors required at least a basic understanding of the overall platform and its functionality. Given that complementors were dependent on platform owners to access knowledge of the platform's functionality, greater functional extent was associated with greater differences in knowledge. For example, C5[4] had acquired substantial expertise in the waste management business. To generate customer value from this knowledge, C5 needed to develop a product that tightly interacts with an enterprise software solution (in this case ERP). To do so, C5 required knowledge on using and instantiating platform functionality.

Second, knowledge dependencies were influenced by the *design of platform interfaces*. Platform interfaces differed as to whether they relied on open standards, ie, standards that are publicly available and widely used, or on the platform owner's proprietary standards. These differences in designing interfaces were consequential because interfaces described the rules and procedures delineating how complementors could access platform functionality and data. If interfaces were designed using proprietary specifications, complementors seemed to depend more on the platform owner as the sole locus of development-related knowledge. Whenever complementors sought knowledge, it was necessary to interact with the platform owner. By contrast, when platform owners implemented open standards, complementors could access various third-party resources such as wikis and books. As a consequence, open standards reduced knowledge dependencies between platform owners and complementors.

In the following, we illustrate the influence of functional extent and interface design. Regarding functional extent, consider the cases of CRM and MOBILE. Initially, PO2 developed CRM as a stand-alone software, which addressed functionality ranging from customer accounts to sales opportunities, customer leads, as well as collaboration and decision-making support for companies regardless of size and industry. CRM addressed the concerns of producing firms regarding heightened client sensitivity to price, value, and service and helped erode barriers to switching products for

[4]We refer to complementors as C[Number], and to platform owners as PO1 and PO2 (see Table 2). We refer to the interview partners as [Role] C[Number]. Roles are VP (vice president), CEO (chief executive officer), and CPO (chief product owner).

clients. Following a series of technological extensions, PO2 decided to open its platform 8 years prior to our study. Complementors developed add-ons that customized CRM to various industries and functional requirements, including business intelligence, social insights, campaign management, sales, service, marketing, and social intelligence. Complementors regarded CRM as a powerful platform. However, over the years, several acquisitions of competing platforms had “inflated” CRM, transforming it into an extensive enterprise software that was difficult to understand. At CRM, the increased functional extent broadened differences in knowledge between PO2 and complementors.

> “I don't think there exists a single person who knows the entire system in detail. This is because of the many acquisitions in the past that have been integrated into a single system” *(CEO C33)*

> “Back then [CRM] was established as a merged platform from acquisitions of various firms in the [CRM] market that were integrated into one holistic solution. I am completely impressed by its functionality but it is like a jungle when you want to understand how things work.” *(CEO C4)*

By contrast, MOBILE complementors reported less marked knowledge differences. PO1 had introduced MOBILE with the aim of providing a platform for enterprise applications that would connect employees, mobile devices, and business needs. Similar to consumer-oriented mobile platforms, MOBILE complements took the form of “apps.” Strategically, PO1 established MOBILE with the intention of offering complementors the business opportunity of building and commercializing business apps. MOBILE covered functionality for the exchange of data between traditional back-end systems and multiple mobile devices. Compared with CRM, MOBILE offered less functionality, mostly consisting in the mere display of business data on mobile screens, rather than providing substantial business logics or customizations. Some MOBILE complementors reported having simple apps running within just a few days, while some CRM complementors released initial prototypes only several months after joining the platform.

To illustrate this phenomenon, let us consider the examples of BIGDATA and MOBILE. At release time, BIGDATA was an in-memory database and real-time analytics platform. Clients used BIGDATA to maximize the capabilities of current hardware to increase application performance, to reduce cost of ownership, and to enable new scenarios and applications for managing large data sets in ways that were not previously possible. Complementors could build persistence or analytics models, server-side or line-of-business logic, or user interfaces for displaying data. Programming was done in a standard programming language, yet queries had to be made in a proprietary and specifically built language.

> “Yeah, it's definitely costly to build compatibility to [BIGDATA]. We would like to manage the risk, but they [PO1] did not truly open the data specifications.” *(CEO C16)*

> “One thing I would like to add to [BIGDATA] is still missing and that was missing from the start. You know partners put so much time and money and investment into building these products, but there is a complete disconnect between the platform and what you know from other technologies. [...] We keep it going for now, but in the past we struggled.” *(CEO C17)*[5]

To contrast the influence of open interfaces, consider MOBILE. PO1 designed MOBILE interfaces as a layer on top of several enterprise software systems of PO1. MOBILE implemented open standards, including HTML5 and OData. HTML5, for example, is a standard proposed and maintained by the World Wide Web Consortium (W3C) for the design of web pages and interactions. The use of open standards considerably decreased knowledge dependencies. Sometimes complementors already had knowledge on working with these standards from prior projects or previous training. If complementors encountered problems, they could easily access various knowledge sources outside the platform, such as stackoverflow.com—thus decreasing dependence on PO1's knowledge.

[5]Please note that platform owners in our sample referred to complementors as “partners.” Partners are identical to our theoretical definition of complementors in the background.

"We have had historically some bad experiences with things like dashboard design being based on [proprietary technology]. And when you kind of go down that road they [complementors] get locked into a certain plug-in or so. So what really came out of for [MOBILE] that we like so much is the idea of mobility as a platform but then using open source or ideas like open HTML5 standards. [...] We realized that OData is the standard, and when you are working with a standard it is good for learning, because you can find many other people who know the technology." *(VP PO1)*

"For [MOBILE] most interfaces are based on OData and we use a programming language similar to JavaScript, where plenty of information is available. Thus, the complexity to develop is comparably low." *(VP PO1)*

"You know, all these things like capsuled HTML and Gateway. It is simple to understand these things and to get started on [MOBILE] because we knew most of that." *(CEO/CPO C26)*

In sum, our findings suggest that knowledge differences and dependencies were influenced by the technical properties of the platform, in terms of the functional extent of a platform and the design of the technical interfaces of the platform.

4.1.2 | Types of knowledge boundaries: Syntactic, semantic, and pragmatic boundaries

After complementors joined a platform, which introduced the need for development-related knowledge, differences and dependencies of knowledge mattered greatly. In addition to these initial complexities, differences and dependencies became apparent when novelty changed the circumstances at the boundary. We observed that novelty of knowledge at the boundary was primarily influenced by the *evolutionary dynamics* of the platform, in terms of the general changes of the technological trajectory of a platform. PO1 and PO2 added, modified, or removed platform functionality on a regular basis, triggered by competitive pressure, technological advances, and acquisitions. PO1 and PO2 also regularly released platform updates to accommodate security issues or performance problems. Other evolutionary dynamics included broader exogenous factors, such as regulatory requirements with respect to data security or industry-level policy changes, such as differences in waste disposal at the state level.

Evolutionary dynamics made differences and dependencies of knowledge apparent. To illustrate how evolutionary dynamics caused syntactic, semantic, and pragmatic boundaries, let us consider the case of ERP. More than a decade prior to our study, PO2 introduced ERP to address the core requirements of midsized organizations and subsidiaries of larger enterprises—specifically by assisting with the management of finance, supply chain, manufacturing, sales, human resources, and projects. Once a year, PO2 released a new version of its platform. These annual releases contained bug fixes, security improvements, and minor performance enhancements. Although representing comparably minor changes to isolated function libraries, it created differences and dependencies in knowledge. Complementors had to evaluate the changed functionality carefully. In one case, for example, C10's development teams had to evaluate (1) whether deprecated functionality required direct changes to their add-ons, (2) whether deprecated functionality potentially had indirect implications on how the add-on interacted with third-party software such as databases or runtime environments, or (3) whether client-side changes were triggered by the release. By obsoleting parts of the platform technology, the annual updates changed parts of the shared understanding between platform owners and complementors. In this sense, the annual ERP updates created *syntactic* boundaries between PO2 and complementors because they led to differences and dependencies in knowledge at the boundary.

"These releases [...] cost a lot of effort [...] and money of course. Sometimes [they do] more harm than good [...]. When our client calls and their production is down [...] we go there and have to figure out what is going on […] and what has changed." *(CEO C10)*

"One example, [ERP release] back then made some noise for quite some time. Basically, we were not clear about the date when it would be available for us. So there were different dates announced, I think, first it was mid-2012, then it was early 2012—in the end it was released in December 2012. So we —and our customers as well—struggled. And I couldn't find a clear answer to that from [PO2]." *(CEO C10)*

We also observed how evolutionary dynamics created *semantic* boundaries. For example, in 2010, a broader technological shift—cloud computing—triggered a change in the market for enterprise systems. Several competitors had already offered parts of their products as cloud services, forcing PO2 to invest in developing a fundamentally revised version of ERP that customers could run in a private cloud setup. Due to legacy issues with the interfaces of the "Finance" module of ERP, PO2 released an updated specification of how complements had to communicate with the ERP core system. Despite this change being formally announced and documented, complementors said they and their clients had only limited awareness about the connotations and the implications of the change. In addition, PO2 had pushed the changed much faster than usual, which put complementors under pressure to better understand the changes and inform their clients accordingly. Although the change in the finance module had formally been documented by PO2, its connotations and implications were much more context specific, and therefore tacit. For example, the change to the "Finance" module required substantial changes for C12, a company that produced a niche add-on for clients with regulatory requirements necessitating a customized side-process for keeping and reporting their ledgers. The impact of the interface change for C12 was not clear to PO2 because it would have required PO2 to consider a particular context of a complementor to understand and precisely communicate the consequences. The change in the interface triggered context-specific implications that were less codifiable and therefore not explicitly articulated. As a consequence, PO2 and C12 interpreted the change differently, creating a semantic boundary.

> "One of our products targets customized ledger processes for firms subject to [German regulation]. The regulation affects less than 50 companies. When we contacted [PO2] in this matter, they were not even aware of this issue. It was simply not on the radar of [PO2]." *(CEO C12)*

> "I think where everyone struggled [...] is, you know, targeting where to go [to] get the information you need in a very exact way." *(CEO C12)*

Finally, we observed that evolutionary dynamics caused *pragmatic* boundaries—conflicts of interest and strategic misalignment in particular. A recurring cause of pragmatic boundaries involved PO1's and PO2's decisions to integrate functionality into their platforms that overlapped with or obsoleted the products of some complementors. Integrations created conflicts of interest as complementors feared that clients would always choose the platform owner's own solutions over a third-party solution if both were comparable regarding features or price. Such discrepancies directly affected many complementors, yet also had indirect spillovers on the behavior of other complementors in terms of holding back further investments into their products. Some of our interviewees suggested that reliable knowledge concerning the platform's long-term roadmap and strategy would have prevented such consequences.

> "I think it's very important to give more information to complementors about what functionality is planned for [ERP]. I've just found out the next version of [ERP] will have a function that can copy and paste data between Excel and the grids in [ERP transactions]. This is exactly what one of my add-ons does. I know that there will always be a risk of this happening but with better information on the planned functionality for [ERP] things like this can mostly be avoided." *(CEO C5)*

> "And we think that development companies or small-sized companies are actually acting faster than the large organizations. What we expect from [PO2], and what is critical for us: we would like to receive some kind of information up-front, so that we are not pushed by them. [...] That [PO2] helps us to keep spaces" *(CEO C7)*

In sum, the evolutionary dynamics of the platform—triggered, for instance, by updates of the platform technology, regulatory changes, or changes in the strategic intent of platform owners—caused changed circumstances at the knowledge boundary between platform owners and complementors. Evolutionary dynamics significantly disrupted development by creating qualitatively distinct types of boundaries: syntactic, semantic, and pragmatic.

4.2 | How do platform owners overcome syntactic, semantic, and pragmatic knowledge boundaries?

4.2.1 | Scope and scale of knowledge boundary resources

One of the notable issues that emerged from our analyses is that the decision-making of PO1 and PO2 was influenced by 2 considerations, which we refer to in the following as scope and scale. One consideration of PO1 and PO2 was to provide complementors with the "right" *scope* of knowledge, in terms of the extent of information, skills, and organizational capability provided by platform owners to address syntactic, semantic, and pragmatic boundaries. Designing KBR of appropriate scope constituted a considerable aspect of work in the management of PO1 and PO2. Frequent discussions in the program management of PO1 for instance focused on whether BIGDATA complementors should receive free annual classroom training sessions in key aspects of the technology to provide them with an extensive scope of knowledge. Another consideration of PO1 and PO2 was the *scale* of KBR, in terms of the number of complementors that an approach in managing the knowledge boundary was capable of addressing. Although PO1 and PO2 strived to maximize the scope of their KBR, they were often limited by considerations of scale. For example, toward the end of our study, the MOBILE ecosystem encompassed 426 complementors, creating substantial challenges for PO1 in disseminating knowledge to this number of complementors. Although PO1 maintained a physical training center for complementors, the offered classroom sessions were not readily available for all MOBILE complementors, due simply to insufficient scalability in terms of instructors, seats, and content provision.

> "The program [MOBILE] is indeed a volume program, which means that it has very strong standardized processes and we cannot regard the needs of each individual partner. So that is certainly one of the essential features that the partners exhibit, that they can orientate themselves within the program and are capable of working within such a standard approach." *(VP PO1)*

> "You could say that we try to coordinate partners by providing certain assets to them and by making their usage as easy as possible. [...] However, this is always limited considering the number of partners that we have to manage." *(VP PO2)*

In sum, we observed that platform owners' decisions regarding shaping knowledge boundaries was influenced by a trade-off between providing knowledge at the right scope, attempting to overcome the knowledge boundary itself, and providing knowledge at the right scale, in terms of how many distinct complementors a KBR was able to address.

4.2.2 | Categories of knowledge boundary resources

Next, we want to offer a closer look at the actual KBR implemented by PO1 and PO2. PO1 and PO2 employed various approaches of conveying knowledge to complementors, including information portals, interface documentations, massive open online courses, communities of practice, helpdesk services, and alignment workshops. Depending on their scope and scale, we broadly categorized KBR as broadcasting, brokering, and bridging (see Table 3),[6] which we describe below. Subsequently, we explain the circumstances at the knowledge boundary that led PO1 and PO2 to implement them.

We refer to KBR that are highly standardized, technology-based, and scalable to the collective of complementors as *broadcasting*. Broadcasting provides standardized, formalized knowledge that complementors can access without having to interact with the platform owner. Broadcasting includes boundary objects such as guidelines, handbooks, programming tutorials, information portals, and technical documentation. Broadcasting resources encompass a known

[6]Please note that our claim is not that these types are independent of each other but rather that they represent qualitatively distinct approaches through which platform owners provide complementors with knowledge.

response to anticipative, recurring, and well-understood developer requirements in the form of predefined, fixed, and objective facts. Broadcasting provides knowledge to complementors in terms of transferable objects or through storage in centralized databases.

Next, we refer to KBR that are semiformal, dedicated, and interaction-based structures as *brokering*. Brokering includes, for example, help desks, account managers, or face-to-face or phone resources that provide technical information about aspects of platform development. Thus, brokering includes boundary objects, but also boundary-spanning activities, given that personal interaction between organizational members of platform owner and complementor constitutes a key aspect of brokering. Despite such personal interaction, we observed brokering to be strongly formalized. Platform owners used brokering to disperse recently updated knowledge. Brokering was limited in its scale, particularly by the availability of skilled resources. For example, PO2 provided ERP complementors with account managers, which covered 5 to 20 complementors each.

Finally, we refer to KBR that are based on ongoing, frequent interactions between experts of the platform owner and complementors as *bridging*. Embedded, personal linkages between organizational members of platform owners and complementors, ie, boundary-spanning activities, are characteristic of bridging, which also limits the scale to individual exchanges. Platform owners in our sample used bridging to provide complementors with problem-solving capabilities and context-specific knowledge. Bridging includes alignment workshops, one-to-one assistance, technological coaching, and coinnovation projects. For example, on an annual basis, PO1 held a developer convention with selected complementors of BIGDATA to exchange technological aspects of the platform and its future directions.

4.2.3 | Platform owners' use of knowledge boundary resources

Although we do not intend to derive propositions on the effectiveness of broadcasting, brokering, and bridging in overcoming different types of boundaries, we illustrate PO1's and PO2's use of different KBR in the presence of syntactic, semantic, and pragmatic boundaries. In our case studies, we found evidence that platform owners accommodated syntactic boundaries by using broadcasting resources. On MOBILE, for example, PO1 relied on the "learning hub," an information portal publishing publicly available development-related information that was available 24/7. Along with new releases of MOBILE, PO1 broadcasted information on changes to the platform via the learning hub and provided detailed change logs and tutorials making new behaviours of the platform apparent. For PO1, broadcasting resembled an appropriate approach because it allowed for the transfer of knowledge at an appropriate scope while preserving scalability for the large number of complementors on the platform.

> "Documentation is the most valuable resource for developers. We occasionally underestimate that. We always have the mindset that we need to provide courses or webinars to explain everything. However, that is not what all developers prefer. They learn the technologies by searching for the needed information." *(VP PO1)*

Broadcasting appeared limited in addressing semantic boundaries. Consider again the case of ERP and, more particularly, the update of the API for the "Finance" module. Before the update, PO2 had relied almost exclusively on broadcasting to convey knowledge following platform updates. Broadcasting resources were guidelines that documented technical specifications and interfaces for third parties. The reference and modelling guidelines describing how to implement database queries had grown to several hundred pages of instructions over time. Complementors noted that technical documentation and handbooks were crammed with specifications. After the update, complementors seemed to struggle with these guidelines. Many of the available guidelines were outdated. PO1 promised new ones, yet even after publication, guidelines were inconsistent. According to complementors, a "single source of truth" was missing.

> "So, we wanted some level of stability before we made that [development] decision. I think there was a period of stability for a good six to eight months during which we did a lot of development work. But

> between then and now we've seen that there has been another change on the software's architecture. In the past years, [PO2] has been extremely innovative and their list of acquisitions has multiplied development speed by ten compared to earlier years, which is one of the issues we raised." *(CEO C5)*

> "The new program documentation, it's very lengthy. [...] So, with every new version it takes us about two weeks to review that internally. It's so extensive. We had to get help from all sides to understand and review that." *(CEO/CPO C8)*

The update in the "Finance" module created semantic boundaries. First, this required PO2 to reestablish a "common lexicon" (function libraries, API documentation, guidelines) for communicating with complementors. Second, it required PO2 to translate the changes to reestablish a common understanding. PO2 recognized these challenges. Throughout the migration period, PO2 significantly invested in brokering resources. PO2 set up dedicated human resources responsible for handling third parties. PO2 referred to these capacities as "account managers." Oftentimes, account managers were full-time employees that stood in loose but regular contact with a small set of complementors. Account managers were trained in handling technological requests of complementors. If complementors required knowledge that exceeded their capacities, account managers forwarded the requests to program managers or the internal development teams of PO2. In addition, PO2 established helpdesks capable of clarifying technical issues for complementors. Helpdesks provided on-site face-to-face and phone resources dedicated to addressing technological requests in a formalized manner. PO2 offered complementors a few free helpdesk inquiries when they signed up for the platform. Our interviews with the complementors strengthened our impression that helpdesk interactions tend to be brief, formalized, and instrumental in nature. Although helpdesks and account managers provided mostly codified information, the semiformal interactions helped complementors narrow down the information that was relevant for their immediate context.

> "We established a hybrid program. We have account managers […] responsible for a certain number of developers in the respective regions. They work as an extension and point developers toward the direction were they find detailed information and work efficiently as a first anchor point." *(VP PO2)*

> "It's quite pleasant if you have a personal contact. If you have trouble with something, you can call her and she takes care of it. She also escalates our requests to the program management, if things need to go fast." *(CEO C6)*

Evidence of knowledge management in the case of pragmatic boundaries was comparably scarce. Yet, in the few instances when pragmatic boundaries occurred, PO1 and PO2 seemed to invest in bridging resources. One such episode occurred in the case of BIGDATA. Technological developments and customer requirements had forced PO1 to integrate new features into the platform itself, which had previously been under the purview of complementor C17. The integration created conflicts of interest between PO1 and C17. PO1 recognized this conflict, and, with the goal of maximizing the outcomes both for complementors and itself, implemented resources to communicate potential conflicts ex ante. For example, PO1 provided selected complementors with an 18 to 24-month roadmap on planned changes to the platform. In addition, PO1 invited BIGDATA complementors to its annual "developer summit," during which PO1 presented recent advances of the platform technology. Interviewees reported that such events gave frequent and informal access to knowledge, ultimately allowing complementors to anticipate changes to the platform early, allowing them to integrate potential consequences into add-on development. In addition, employees at PO1 accumulated knowledge about the particular focus of complementors over time, which allowed them to better contextualize knowledge.

> "My colleague is six to eight times a year at [PO1 headquarters]. This makes the exchange very intensive and this is also very good […]. We have been doing this for a long time and this has proven successful for not running into the danger of being too much surprised by the actions of [PO1]." *(CPO C17)*

> "On Monday before the developer conference, we all came to the headquarters and there was a sort of a meeting, where [PO1 manager] was there, [BIGDATA head of development] was there, a number of people were there and presenting information […]. You know, the sort of secrets of the conference and sort of what helped us." *(CEO C30)*

> "Sometimes there is also strong technological support. We recently had a performance issue, as the add-on was not running fast enough. This had to do with the access to the database and [PO1] sent a software specialist that supported us intensively so that the performance was as high as possible. You find such opportunities on [BIGDATA] but surely not on all other platforms." *(CEO C16)*

> "[We engaged in closer interaction because] we do not want to earn money from the partners, but together with the partners." *(VP PO1)*

In sum, we observed that the considerations of platform owners regarding the scope and scale of knowledge resources at the boundary emerged in 3 qualitatively distinct types of KBR—broadcasting, brokering, and bridging. Platform owners used these types of KBR to address syntactic, semantic, and pragmatic boundaries.

5 | DISCUSSION

5.1 | Main findings

The goals of this study were (Guiding Question 1) to understand what influences differences, dependencies, and novelty of knowledge at the boundary between enterprise software platform owners and complementors and (Guiding Question) how platform owners overcome the resulting syntactic, semantic, and pragmatic boundaries. We investigate these questions by conducting a multiple-case study of 4 enterprise software platforms involving 40 interviews and secondary data.

Our study yields 3 major findings. First, we find that characteristics of the technical design of a platform, in terms of its functional extent and interface design, create differences and dependencies in knowledge between platform owners and complementors. Differences and dependencies in knowledge become apparent once the evolutionary dynamics of the platform alter the conditions at the boundary—eg, because platform owners decide to modify parts of the platform or due to regulatory changes—thus creating syntactic, semantic, and pragmatic knowledge boundaries.

Second, we observed that platform owners address knowledge boundaries by providing different kinds of KBR. The provision of these resources is guided by the question of how to provide complementors with knowledge at the right scope and scale. Depending on their scope and scale, we classified the observed KBR as broadcasting, brokering, or bridging.

Finally, while the effectiveness of the provided KBR was not at the center of our investigation, we observed that the reliance of platform owners on broadcasting, brokering, and bridging depended on the knowledge boundary. Differences and dependencies in knowledge lead to syntactic boundaries between platform owners and complementors, which platform owners address through broadcasting—a particular type of KBR characterized by high scalability but limited scope. Evolutionary dynamics lead to semantic and pragmatic boundaries, which platform owners address through brokering and bridging KBR that complement boundary objects with more personal boundary-spanning activities.

5.2 | Theoretical contributions

Our study makes 3 main contributions. First, our work contributes to literature on knowledge management in platform ecosystems, particularly in the context of enterprise software platforms. Prior work has documented the importance of knowledge management as an instrument of platform governance from various perspectives, including boundary resources and spanning (eg, Eaton et al., 2015; Ghazawneh & Henfridsson, 2013; Huber et al., 2017), platform engineering and design (eg, Baldwin & Woodard, 2008; Carlile, 2002; Ulrich, 1995; West, 2003), and governance strategy (eg, Boudreau & Hagiu, 2009; Cusumano & Gawer, 2002; Shapiro & Varian, 1998). Yet, a comprehensive picture of knowledge management in platform ecosystems did not yet exist. Our study is among the first to offer a thorough examination of the interactions at the knowledge boundary between platform owners and complementors. We contextualize Carlile's (2004) framework on knowledge boundaries for platform development (Davison & Martinsons, 2016; Johns, 2006). Our study sheds light on the causes of differences, dependencies, and novelty of knowledge at the boundary. In our context, we observed that these different types of boundaries also exist when product development is organized as a platform ecosystem. In this context, differences and dependencies appeared to derive from the technological design of the platform rather than from organizational relationships between employees and departments. In addition, we describe in detail the various approaches platform owners take in order to overcome knowledge boundaries by shaping resources at the boundary. Platform owners' considerations in providing KBR involved a trade-off in the scope and scale that platform owners used to shape the resources. Our description of scale as a property of KBR adds a further facet to Carlile's (2004) framework—namely, the notion that handling knowledge boundaries is not only a question of the right scope but also the right scale. This appears to be particularly relevant in contexts where an unprecedented number of organizations work together, as in our study. This overview extends our understanding of a core problem in platform management—a contribution researchers have recently called for (Bergvall-Kåreborn & Howcroft, 2014; Van Alstyne et al., 2016).

Second, we contribute to work that studies platform governance decisions from a design and engineering perspective. Although properties of platform architectures have been examined previously (Baldwin & Woodard, 2008; Carlile, 2002; Ulrich, 1995; West, 2003), few, if any, studies have assessed implications for the knowledge boundary between platform owners and complementors. Our findings add to these studies by showing how decisions regarding platform design can create knowledge boundaries. As outlined above, platform design decisions can be consequential. If not accounted for by the provision of KBR, platform owners risk losing out on complementary innovation. By outlining the theoretical causes underlying differences, dependencies, and novelty of knowledge boundaries, we also respond to work calling for an understanding of the influence of platform technology on decisions regarding platform management (Bergvall-Kåreborn & Howcroft, 2014; Tiwana et al., 2010).

Finally, our findings contribute to the ongoing debate on the optimal form of platform governance. Work in this stream engages with the question of whether platform owners should organize complementors by means of competition or by means of cooperation (Armstrong, 2006; Boudreau & Lakhani, 2009; Huang et al., 2013; Shapiro & Varian, 1998). Our study illustrates that platform owners may need to invest in cooperation in contexts of increasing knowledge boundaries. As our findings show, platform owners faced syntactic, semantic, and pragmatic knowledge boundaries with increasing evolutionary dynamics of the platform, which required them to invest in brokering and bridging in order to address them. Therefore, our findings add to the ongoing discussion about optimal forms of platform governance by illustrating an important contingency that may require platform owners to consider investing in cooperation: knowledge boundaries.

5.3 | Research implications

Our findings have at least 3 implications for research. First, the finding that the technical design of a platform creates differences and dependencies in knowledge at the boundary between platform owners and complementors implies that decisions regarding platform design and governance are closely linked to each other. Therefore, a complex technical design of the platform may require substantial costs of governance in later stages of the platform by creating knowledge boundaries for complementors. Even in later stages of the platform lifecycle, such governance costs may be incurred when platform owners conduct changes to the platform. During episodes of evolutionary dynamics, development seems to become particularly difficult for complementors, because parts of their existing knowledge may be rendered obsolete, resulting in semantic and pragmatic boundaries and the need to reacquire development-related knowledge. Whether and how platform owners can address such ripple effects of technological decisions is unclear, because platform owners are subject to macro changes in technologies, or must constantly evaluate whether to invest in new versions of their platform (Anderson, Parker, & Tan, 2014; Katz & Shapiro, 1994).

Second, the finding that platform owners provide different categories of KBR depending on issues of scope and scale implies that the provision of KBR goes beyond pure cost considerations. One of the key challenges for platform owners is to attract a critical mass of clients and complementors to benefit from 2-sided network effects (eg, Parker & Van Alstyne, 2005). If platform owners predominantly use KBR that are not scalable to the ecosystem level, then they may fail to achieve this critical mass of complementors. This may point to a key difference between consumer platforms, such as Apple's iPhone, and more complex enterprise software platforms. Although our study's focus was not on the success or survival of platform ecosystems, it may provide the foundation for future research on the complex interplay between technological characteristics, knowledge boundaries, KBR, and 2-sided network effects.

Finally, the finding that platform owners vary in their provision of KBR lays the groundwork for research on the effectiveness of KBR. While it was beyond the goal of our study to identify the effectiveness of particular types of KBR in addressing particular knowledge boundaries, our findings provide several qualitative insights. Our findings suggest that depending on the knowledge boundary between platform owners and complementors, broadcasting may become ineffective in enabling successful add-on development. This contrasts with work promoting a competitive governance of complementors, and would instead advocate investing in cooperative relationships with complementors. In our study, we observed that platform management is not solely a matter of providing complementors with access to a joint technological infrastructure, but rather requires investments in cooperative relationships with complementors. As bridging represents a costly activity, our findings emphasize that platform owners should decide on strategic innovation partners with which they engage closely.

5.4 | Practical contributions and managerial implications

Our findings also have important managerial implications. First, our findings suggest that platform owners may consider changes to their platform as a trade-off: changes may offer a competitive edge but may widen the knowledge gap for complementors if not aligned with investments in KBR. This discussion is particularly interesting in light of the current trend toward cloud-based software solutions. A key aspect of cloud-based software development is that the former infrequent but comprehensive update releases are substituted by quick releases of updated microservices —a development culture often referred to as DevOps (Roche, 2013). In particular, cloud-based enterprise software will likely show a high degree of evolutionary dynamics when small and quick releases accumulate. The trade-off related to evolutionary dynamics points to the crucial role of functional extent and interface design. Prior studies have touched on the decisive role of interfaces for the management of platforms, but have thus far not considered their crucial role for adequately equipping third-party developers with knowledge (Baldwin & Clark, 1997; Ghazawneh & Henfridsson, 2013; Kallinikos, Aaltonen, & Marton, 2013). Conclusions from our insights on knowledge boundaries would suggest that owners of platforms with high evolutionary dynamics (such as cloud platforms) may need to limit the functional extent of their platforms and rely on open interface standards to reduce complexity at the boundary and cushion the potential adverse effects of evolutionary dynamics for complementors. In sum, our analyses underscore propositions in prior research of a close linkage between platform technology and its management (Tiwana et al., 2010; Yoo, Henfridsson, & Lyytinen, 2010).

Second, by providing a classification of 3 broad types of KBR, we equip managers with a tool for evaluating their approaches in providing third-party developers with development-related knowledge. In particular, our findings provide practitioners with guidance in sensing knowledge requirements of complementors and offer advice on designing KBR that address these requirements. Foremost, our results suggest that the design and management of platforms represent distinct dimensions that require a careful balance. Moreover, platform owners may find the categorization of KBR into broadcasting, brokering, and bridging helpful in assessing and evaluating the status quo of knowledge dissemination to complementors in their platform ecosystems.

6 | CONCLUSION

We studied the management of development-related knowledge across firm boundaries in enterprise software platform development. Our exploratory, multiple-case study of 4 enterprise software platforms illustrates that knowledge management is a nontrivial challenge of platform governance. Technical characteristics of the platform—in particular its functional extent, interface design, and evolutionary dynamics, seem to create differences, dependencies, and novelty of knowledge, resulting in different types of knowledge boundaries between platform owners and complementors. To mitigate knowledge boundaries, platform owners provide various resources, including information portals, documentation, helpdesks, and alignment workshops. The provision of these resources is subject to a trade-off between providing knowledge in the right scope, while allowing for the scalability of knowledge resources for the entire ecosystem. We classify KBR depending on their scope and scale as broadcasting, brokering, and bridging, each representing qualitatively distinct patterns in managing knowledge in platform development.

ACKNOWLEDGEMENTS

We thank Jens Dibbern, Hartmuth Hoehle, Kristina Meier, Michael Roemer, Tracy Sykes, and Viswanath Venkatesh for their helpful advice and comments. We appreciate comments by seminar participants in Mannheim, as well as participants of the European Workshop on Software Ecosystems 2014, the International Conference on Information Systems 2014, the Koenig Research Colloquium 2015, and the Academy of Management Annual Meeting 2015. This research has received financial support from the Graduate School of Economic and Social Sciences, University of Mannheim.

ORCID

Jens Foerderer http://orcid.org/0000-0002-3090-4559
Thomas Kude http://orcid.org/0000-0002-9891-0517

REFERENCES

Adner, R., & Kapoor, R. (2010). Value creation in innovation ecosystems: How the structure of technological interdependence affects firm performance in new technology generations. *Strategic Management Journal*, *31*(3), 306–333.

Anderson, E. G., Parker, G. G., & Tan, B. (2014). Platform performance investment in the presence of network externalities. *Information Systems Research*, *25*(1), 152–172.

Armstrong, M. (2006). Competition in two-sided markets. *The Rand Journal of Economics*, *37*(3), 668–691.

Baldwin, C. Y., & Clark, K. B. (1997). Managing in an age of modularity. *Harvard Business Review*, *75*(5), 84–93.

Baldwin, C. Y., & Woodard, C. J. (2008). The architecture of platforms: A unified view. Working Papers—Harvard Business School Division of Research, pp. 1–31.

Banker, R. D., Davis, G. B., & Slaughter, S. A. (1998). Software development practices, software complexity, and software maintenance performance: A field study. *Management Science*, *44*(4), 433–450.

Bechky, B. A. (2003). Sharing meaning across occupational communities: The transformation of understanding on a production floor. *Organization Science*, *14*(3), 312–330.

Bergvall-Kåreborn, B., & Howcroft, D. (2014). Persistent problems and practices in information systems development: A study of mobile applications development and distribution. *Information Systems Journal*, *24*(5), 425–444.

Boudreau, K. (2010). Open platform strategies and innovation: Granting access vs. devolving control. *Management Science*, *56*(10), 1849–1872.

Boudreau, K., & Lakhani, K. (2009). How to manage outside innovation. *MIT Sloan Management Review*, *50*(4), 69–76.

Boudreau, K. J. (2012). Let a thousand flowers bloom? An early look at large numbers of software app developers and patterns of innovation. *Organization Science*, *23*(5), 1409–1427.

Boudreau, K. J., & Hagiu, A. (2009). Platform rules: Multi-sided platforms as regulators. In A. Gawer (Ed.), *Platforms, markets and innovation* (pp. 163–191). London: Edward Elgar Publishing.

Brown, S. L., & Eisenhardt, K. M. (1995). Product development: Past research, present findings, and future directions. *Academy of Management Review*, *20*(2), 343–378.

Brusoni, S., & Prencipe, A. (2006). Making design rules: A multidomain perspective. *Organization Science*, *17*(2), 179–189.

Carlile, P. R. (2002). A pragmatic view of knowledge and boundaries: Boundary objects in new product development. *Organization Science*, *13*(4), 442–455.

Carlile, P. R. (2004). Transferring, translating, and transforming: An integrative framework for managing knowledge across boundaries. *Organization Science*, *15*(5), 555–568.

Carlile, P. R., & Rebentisch, E. S. (2003). Into the black box: The knowledge transformation cycle. *Management Science*, *49*(9), 1180–1195.

Ceccagnoli, M., Forman, C., Huang, P., & Wu, D. J. (2012). Cocreation of value in a platform ecosystem: The case of Enterprise software. *MIS Quarterly*, *36*(1), 263–290.

Charmaz, K. (2014). *Constructing grounded theory*. London, GB: Sage.

Corbin, J. M., & Strauss, A. (1990). Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative Sociology*, *13*(1), 3–21.

Cusumano, M. A., & Gawer, A. (2002). The elements of platform leadership. *MIT Sloan Management Review*, *43*(3), 51–58.

Davison, R. M., & Martinsons, M. G. (2016). Context is king! Considering particularism in research design and reporting. *Journal of Information Technology*, *31*(3), 241–249.

Eaton, B., Elaluf-Calderwood, S., Sorenson, C., & Yoo, Y. (2015). Distributed tuning of boundary resources the case of Apple's iOS service system. *MIS Quarterly*, *39*(1), 217–243.

Eisenhardt, K. M. (1989). Building theories from case study research. *Academy of Management Review*, *14*(4), 532–550.

Fichman, R. G. (2004). Going beyond the dominant paradigm for information technology innovation research: Emerging concepts and methods. *Journal of the Association for Information Systems*, *5*(8), 314–355.

Furlan, A., Cabigiosu, A., & Camuffo, A. (2014). When the mirror gets misted up: Modularity and technological change. *Strategic Management Journal*, *35*(6), 789–807.

Gartner. (2015). Magic quadrant for single-instance ERP for product-centric midmarket companies. in Gartner Retrieved 21. August, 2017, from https://www.gartner.com/doc/3177020.

Gawer, A. (2014). Bridging differing perspectives on technological platforms: Toward an integrative framework. *Research Policy*, *43*(7), 1239–1249.

Gawer, A., & Henderson, R. (2007). Platform owner entry and innovation in complementary markets: Evidence from Intel. *Journal of Economics & Management Strategy*, *16*(1), 1–34.

Ghazawneh, A., & Henfridsson, O. (2013). Balancing platform control and external contribution in third-party development: The boundary resources model. *Information Systems Journal*, *23*(2), 173–192.

Grant, R. M. (1996). Toward a knowledge-based theory of the firm. *Strategic Management Journal*, *17*(Winter), 109–122.

Gulati, R., Puranam, P., & Tushman, M. (2012). Meta-organization design: Rethinking design in interorganizational and community contexts. *Strategic Management Journal*, *33*(6), 571–586.

Henderson, R. M., & Clark, K. B. (1990). Architectural innovation: The reconfiguration of existing product technologies and the failure of established firms. *Administrative Science Quarterly*, *35*(1), 9–30.

Hesseldahl, A. (2013). SAP cutting back on development of business ByDesign. Retrieved 3. January, 2017, from http://allthingsd.com/20131019/sap-cutting-back-on-development-of-business-bydesign/.

Huang, P., Ceccagnoli, M., Forman, C., & Wu, D. J. (2013). Appropriability mechanisms and the platform partnership decision: Evidence from Enterprise software. *Management Science*, *59*(1), 102–121.

Huber, T., Kude, T., & Dibbern, J. (2017). Governance practices in platform ecosystems: Navigating tensions between co-created value and governance costs. *Information Systems Research.*, *28*, 563–584.

Iansiti, M., & Levien, R. (2004). Strategy as ecology. *Harvard Business Review*, *82*(3), 68–78.

Inkpen, A. C., & Dinur, A. (1998). Knowledge management processes and international joint ventures. *Organization Science*, *9*(4), 454–468.

Johns, G. (2006). The essential impact of context on organizational behavior. *Academy of Management Review*, *31*(2), 386–408.

Kallinikos, J., Aaltonen, A., & Marton, A. (2013). The ambivalent ontology of digital artifacts. *MIS Quarterly*, *37*(2), 357–370.

Katz, M. L., & Shapiro, C. (1994). Systems competition and network effects. *Journal of Economic Perspectives*, *8*(2), 93–115.

Kellogg, K. C., Orlikowski, W. J., & Yates, J. (2006). Life in the trading zone: Structuring coordination across boundaries in postbureaucratic organizations. *Organization Science*, *17*(1), 22–44.

Kogut, B., & Zander, U. (1992). Knowledge of the firm, combinative capabilities, and the replication of technology. *Organization Science*, *3*(3), 383–397.

McIntyre, D. P., & Srinivasan, A. (2017). Networks, platforms, and strategy: Emerging views and next steps. *Strategic Management Journal*, *38*(1), 141–160.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook* (2nd ed.). Newbury Park, CA: Sage.

Nambisan, S. (2013). Information technology and product/service innovation: A brief assessment and some suggestions for future research. *Journal of the Association for Information Systems*, *14*(4), 215–226.

Nonaka, I. (1994). A dynamic theory of organizational knowledge creation. *Organization Science*, *5*(1), 14–37.

Parker, G., Van Alstyne, M., & Choudhary, P. (2016). *Platform revolution: How networked markets are transforming the economy—and how you can make them work for you*. Boston, MA: W. W. Norton & Company.

Parker, G. G., & Van Alstyne, M. W. (2005). Two-sided network effects: A theory of information product design. *Management Science*, *51*(10), 1494–1504.

Polanyi, M. (1966). *The tacit dimension*. New York: Doubleday & Co.

Roche, J. (2013). Adopting DevOps practices in quality assurance. *Communications of the ACM*, *56*(11), 38–43.

Rochet, J.-C., & Tirole, J. (2003). Platform competition in two-sided markets. *Journal of the European Economic Association*, *1*(4), 990–1029.

Rosenkranz, C., Vranesic, H., & Holten, R. (2014). Boundary interactions and motors of change in requirements elicitation: a dynamic perspective on knowledge sharing. *Journal of the Association for Information Systems*, *15*(6), 306.

Sarker, S., Sarker, S., Sahaym, A., & Bjørn-andersen, N. (2012). Exploring value cocreation in relationship between an ERP vendor and its partners: A revelatory case study. *MIS Quarterly*, *36*(1), 317–338.

Shapiro, C., & Varian, H. R. (1998). *Information rules: A strategic guide to the network economy*. Cambridge, MA: Harvard Business Press.

Star, S. L. (1989). In L. Gasser, & M. N. Huhns (Eds.), *The structure of Ill-structured solutions: Boundary objects and heterogeneous distributed problem solving* (pp. 37–54). London: Pitman.

Star, S. L., & Griesemer, J. R. (1989). Institutional ecology, translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. *Social Studies of Science*, *19*(3), 387–420.

Szulanski, G. (1996). Exploring internal stickiness: Impediments to the transfer of best practice within the firm. *Strategic Management Journal*, *17*(S2), 27–43.

Teece, D. J. (1986). Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy. *Research Policy*, *15*(6), 285–305.

Tilson, D., Lyytinen, K., & Sørensen, C. (2010). Research commentary—digital infrastructures: The missing IS research agenda. *Information Systems Research*, *21*(4), 748–759.

Tiwana, A. (2013). *Platform Ecosystems: Aligning Architecture, Governance, and Strategy*. Burlington, MA: Morgan Kaufmann.

Tiwana, A. (2015). Evolutionary competition in platform ecosystems. *Information Systems Research*, *26*(2), 266–281.

Tiwana, A., Konsynski, B., & Bush, A. A. (2010). Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics. *Information Systems Research*, *21*(4), 675–687.

Tushman, M. L. (1977). Special boundary roles in the innovation process. *Administrative Science Quarterly*, *22*(4), 587–605.

Tushman, M. L., & Scanlan, T. J. (1981). Boundary spanning individuals: Their role in information transfer and their antecedents. *Academy of Management Journal*, *24*(2), 289–305.

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research Policy*, *24*(3), 419–440.

Van Alstyne, M., Parker, G., & Choudhary, S. P. (2016). *6 reasons platforms fail*. Harvard Business Review Blog.

Von Hippel, E. (1998). Economics of product development by users: The impact of "sticky" local information. *Management science*, *44*(5), 629–644.

von Hippel, E., & Katz, R. (2002). Shifting innovation to users via toolkits. *Management Science*, *48*(7), 821–833.

Von Krogh, G., Ichijo, K., & Nonaka, I. (2000). *Enabling knowledge creation: How to unlock the mystery of tacit knowledge and release the power of innovation*. Oxford: Oxford University Press.

Wareham, J., Fox, P. B., & Cano Giner, J. L. (2014). Technology ecosystem governance. *Organization Science*, *25*(4), 1195–1215.

West, J. (2003). How open is open enough?: Melding proprietary and open source platform strategies. *Research Policy*, *32*(7), 1259–1285.

Yoo, Y., Henfridsson, O., & Lyytinen, K. (2010). Research commentary—the new organizing logic of digital innovation: An agenda for information systems research. *Information Systems Research*, *21*(4), 724–735.

How to cite this article: Foerderer J, Kude T, Schuetz SW, Heinzl A. Knowledge boundaries in enterprise software platform development: Antecedents and consequences for platform governance. *Info Systems J*. 2018;1–26. https://doi.org/10.1111/isj.12186

APPENDIX A.

TABLE A1 Interview guideline

Information about company and interviewee ● information about the company, in particular prior relationship to the platform owner and historical development. ● position within the company, role with regards to the relationship to platform owner.
Complementor support on the platform ● description of how development-related knowledge is obtained. ● how is information exchanged between platform owner and your company?
Development ● process and complexities in add-on development for the platform. ● architectural characteristics of the platform, in particular description of core functionality. ● changes to the platform, unanticipated events.