The Power of Propagation: On the Role of Software Operation Knowledge within Software Ecosystems

Henk van der Schuur, Slinger Jansen and Sjaak Brinkkemper
Department of Information and Computing Sciences
Utrecht University
Utrecht, The Netherlands
{h.schuur, s.jansen, s.brinkkemper}@cs.uu.nl

ABSTRACT

Knowledge of in-the-field software operation is still unrecognized as an essential pulse in the veins of software ecosystems. Although software-producing organizations are aware of the ecosystems in which they operate and their relationships with other ecosystem participants, all too often, vendors are unsuccessful in recognizing the potential value and role of such knowledge in their software ecosystems. This paper presents a classification of successful operational software ecosystem practices that may help software-producing organizations to effectively utilize and propagate knowledge of the in-the-field operation of their software, and therewith address challenges that result from ecosystem participation. Analysis of these practices confirms that infrastructures for acquisition, utilization and propagation of such knowledge, allow ecosystem participants to use the 'power of many' in increasing the quality and robustness of their software, and provide them with competitive advantage in terms of software quality, end-user satisfaction, ecosystem stability and ecosystem attractiveness.

Categories and Subject Descriptors

H.1.1 [**Models and Principles**]: Systems and Information Theory; D.2.9 [**Software Engineering**]: Management

General Terms

Economics, Management, Theory, Design

Keywords

software operation knowledge, software ecosystems, knowledge propagation, ecosystem orchestration

1. INTRODUCTION

Software vendors have become networked organizations that are dependent on other software vendors for libraries, components or platforms vital to the correct operation of their software. Due to rapidly changing technology and end-user demands, software vendors have resorted to virtual integration through alliances in or between the software ecosystems in which they operate. While many software vendors thrive and advance because of their participation in software ecosystems, these vendors are simultaneously confronted with several challenges resulting from software ecosystem participation, such as software ecosystem relationship management, ecosystem orchestration strategy development, and ecosystem composition comprehension [12, 5, 2, 1]. Part of the value that is created in software ecosystems is software operation knowledge (SOK): 'Knowledge of in-the-field performance, quality and usage of software, and knowledge of in-the-field end-user software experience feedback' [19]. Vendors use SOK, for instance, to gain insight in the in-the-field behavior of (end-users on) their software, increase software quality by mitigating issues that may surface only after deployment (e.g. compatibility issues, performance problems, etc.) and improve software processes like software maintenance, software product management and customer support [19].

In this paper, we state that SOK forms an essential pulse in the veins of software ecosystems, and serves as a primary enabler for software vendors to thrive and flourish in the ecosystems in which they operate. Software vendors can successfully participate in software ecosystems by addressing challenges resulting from this participation through effective utilization[1] of acquired software operation knowledge, as well as through propagation of such knowledge to other ecosystem participants. To substantiate our position, we report on the operational SOK propagation practices of four software-producing organizations, and show how these organizations have mitigated aforementioned challenges through SOK propagation. Analysis of these practices shows that through effective utilization and propagation of SOK, software quality is increased, relations between vendors are strengthened, and insight into software ecosystem composition is deepened.

This paper continues as follows. In the next section, we elaborate on the concept of SOK propagation. First, we identify two main software ecosystem participant roles and describe SOK propagation characteristics per role. Second,

[1]In this paper, we define 'utilization' of software operation knowledge as the use of such knowledge to support or improve processes, practices or products.

MEDES'11 November 21-24, 2011, San Francisco, USA

SECO Scope Level	SOK Propagation Illustration
Software Ecosystem	Changes to the keystone platform (e.g. in-the-field API behavior) are propagated to all ecosystem participants
Actor	Knowledge of keystone software instability, caused by niche player software, is propagated to that particular niche player

Table 1: SOK propagation illustrated for two SECO scope levels

we describe challenges that result from software ecosystem participation, and illustrate how these challenges can be addressed through SOK propagation. Section 4 presents the identified operational SOK propagation practices of four software-producing organizations (the identification approach is detailed in section 3); analysis of those practices is provided in section 5. Finally, conclusions are presented in section 6.

2. SOK PROPAGATION WITHIN SOFTWARE ECOSYSTEMS

Several different definitions of the term software ecosystem (SECO) exist [16, 3, 14]. In this paper, we use the following definition of a SECO: 'A set of actors functioning as a unit and interacting with a shared market for software and services, together with the relationships among them' [13]. Relationships between SECO actors are frequently underpinned by a common technological platform or market, and operate through exchange of information, resources or artifacts [12, 3]. Participation of actors in software ecosystems can be observed from various points of view. In this paper, we scope participation of software vendors in software ecosystems from two scope levels, both levels corresponding to certain 'objects of study' [13]: software ecosystems as a whole, and software ecosystem actors themselves (see table 1). Companies participating in a software ecosystem (i.e., interacting with a shared market for software and services) can have different roles. Although more roles are identified in literature, the roles of *keystone* (i.e. orchestrator, shaper) and *niche player* (i.e. follower) are recurring, e.g. [13, 9].

Keystone An actor providing a standards or technology platform that forms a foundation for the keystone's ecosystem. The platform defines standards and practices, provides leverage and increases in value and attractiveness with the number of actors. Keystones create and share value (i.e. assets, incentives) within their software ecosystems [10], and aim to improve their ecosystems' quality, stability and productivity by (1) encouraging specialization (i.e. niche creation), (2) cultivating deep understanding of processes and practices and (3) developing and managing feedback loops [11, 4, 6].

Niche player An actor that requires a standard or technology platform provided by a keystone to create business value [13]. A niche player operates in a niche market and prospers through value or knowledge generated by the keystone, other actors or the software ecosystem as a whole [9, 4]. Niche players are invited to participate in software ecosystems by the ecosystems' keystones (for instance by receiving a development program qualification or software compatibility certification, or by having (unrestricted) access to keystone resources), and therewith may become keystone partners on the long term.

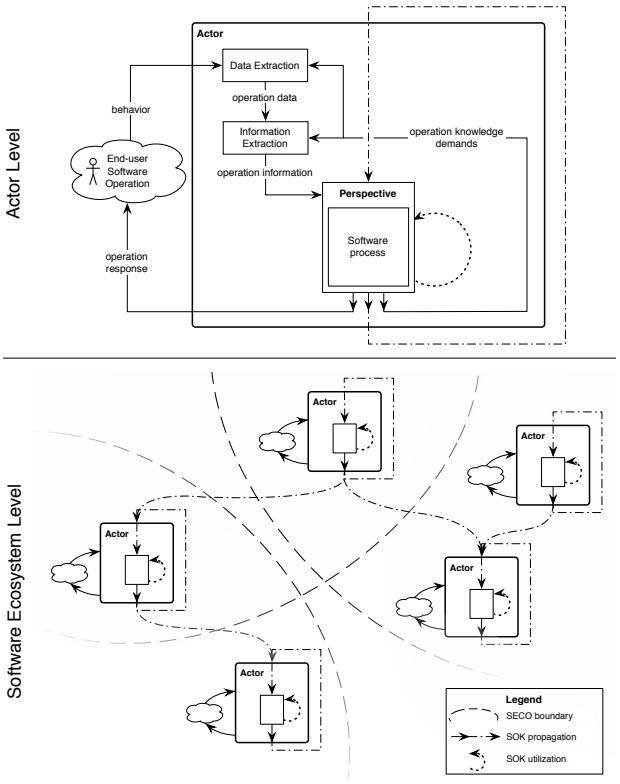

Figure 1: SOK utilization and propagation in software ecosystems

Based on operation knowledge demands, software vendors obtain software operation knowledge and operation information from software operation *data* (as demonstrated in [19, 18, 20]). Figure 1 conceptually visualizes SOK utilization and propagation in software ecosystems. On the actor level, actors (e.g. keystones, niche players) derive software operation knowledge from (behavior of end-users on) software operating in the field and utilize such knowledge within their organizations. On the software ecosystem level, such software operation knowledge is propagated to other actors in the same ecosystem. After application of data mining techniques and data abstraction logic, both based on operation knowledge demands, operation *information* is obtained. This information serves as input for support and improvement of an actor's the products or processes. Interpretation and integration of operation information results in software operation *knowledge*, which is utilized iteratively and continuously to improve an actor's products and processes.

SOK utilization can serve various goals and can thus be viewed from various perspectives, such as a development, company and customer perspective [19]. As a result of their utilization of software operation knowledge, software ecosystem participants may respond by (1) contacting customers

and releasing software patches or updates to eliminate or thoroughly investigate software failures, (2) defining new (or altering existing) operation knowledge demands for more generic or specific operation data acquisition, and (3) propagating (utilized) software operation knowledge to other ecosystem participants, for example to notify partners of unstable or failing operation situations, or provide additional operation environment details to encourage bug elimination and software quality improvement. As detailed in the next section, an ecosystem participant may utilize SOK propagated by other ecosystem participants in addition to SOK it obtains from in-the-field operation of its 'own' software.

2.1 Propagation Characteristics

Propagation of software operation knowledge within software ecosystems encompasses both scope levels in table 1. Although software operation knowledge is propagated by one actor to one or more other actors, the process of propagation itself can be observed from both scope levels: actors as well as software ecosystems as a whole may be influenced by, or accountable for, SOK propagation. SOK propagation characteristics and goals may differ per actor role, however.

Keystone characteristics In terms of SOK propagation, keystones set about establishing a platform for publication and propagation of acquired SOK (e.g. error reports, performance measures, etc.) to niche players in their software ecosystem [9]. Keystones orchestrate which niche players have access to the platform and to which niche players knowledge of keystone software operation is propagated. Keystones are able to propagate SOK through their ecosystem at different levels of granularity, corresponding to the scope levels in table 1. Keystones strive for reinforcement and growth of their software ecosystem. Software operation knowledge can be used to do so. For example, effective propagation of, and response to SOK acquired during operation of a new set of platform APIs can increase the performance and quality of these APIs rapidly, and therewith accelerate their adoption. External actors, potentially participating in rival ecosystems, may be attracted by the stability, feature-richness and 'partner intimacy' of the keystone and its platform.

With over 425,000 software applications, downloaded more than 15 billion times in total in less than 3 years, Apple's AppStore is recognized as the platform that thrives the iPhone software ecosystem. In the first months after the launch of the store, Apple prohibited public discussion about APIs, documentation and sample code. Particularly during these months, incorrect usage of (unofficial) iPhone SDK APIs led to many application crashes (for instance after installation of an operating system upgrade) and much frustration of niche players participating in Apple's iPhone ecosystem. Compatibility and stability issues caused by erroneous API use could have been prevented through SOK analysis and propagation.

Niche player characteristics On the short term, niche players strive to create business value through the platform provided by the keystone. For instance, niche players create such value by (1) developing niche software products, (2) realizing efficiencies resulting from the sharing of assets and knowledge by other ecosystem participants, and (3) obtaining access to the keystone's customers [9]. Adequate response to SOK received from keystones, as well as frequent informing of keystones on end-user feedback knowledge (end-user experiences, wishes, etc.) may be rewarded by a keystone with 'niche player partner' qualification (which provides access to a keystone's customer data, for example). While close and trustful relationships and communication with the keystone and other niche players may be advantageous, niche players are prudent and attempt to prevent overdone response (e.g. by sharing specialized knowledge or intellectual property). Niche players seek a SOK propagation response balance: too comprehensive response may reduce a niche player's competitive or knowledgeable advantage, too limited response might not result in quality and robust software and consequent overdue response may result in exclusion from the ecosystem or forfeiture of partnership, which diminishes a niche player's chances to become a keystone itself. Nonetheless, to transform into a keystone actor, niche players can merge forces or gradually adopt keystone skills.

At present, certified niche players participating in the Microsoft Windows ecosystem receive SOK propagated by Microsoft via its Windows Quality Online Services (Winqual) [21, 7]. Niche players can link software operation failures to solutions, which allows directed distribution of software updates and corresponding support web pages to customers. Furthermore, Microsoft propagates knowledge of end-user satisfaction, and software usability and appreciation to niche players through its program for customer experience improvement [17].

2.2 Addressing SECO Challenges through SOK Propagation

Software vendors participating in software ecosystems are faced with new challenges on each software ecosystem scope level [12, 5, 2]. Below, we describe how we envision particular challenges to be addressed through effective utilization and propagation of SOK.

SECO relationship management Evaluation of propagated SOK results in intensification, prolongation or diminishing of relations between software ecosystem participants and their suppliers, buyers and end-users. For example, a keystone may decide to promote a niche player to a (certified) partner when the niche player responds adequately to propagated SOK and its software is robust and of high quality. New relationships are established, for instance, if SOK evaluation shows that a particular niche player's software does not meet quality level agreements and a keystone decides to replace the niche player by a new vendor, or if the keystone platform is to be extended with novel standards or techniques developed by vendors not participating in the keystone's ecosystem.

SECO composition comprehension When SOK is continuously and forthrightly propagated through a software ecosystem, participants gain insight in the structure of and knowledge flows within their ecosystem. As an example, a keystone actor may obtain a comprehensive view of the most-used software add-ons that are operating on top of its extension architecture. Niche players get insight in which ecosystem participants are propagating the greatest amount of SOK.

Software quality management As stated earlier, acquired SOK is used as a basis for monitoring and managing quality of software operation. Also, keystones define quality levels and analyze SOK to determine a niche player's software add-on quality level and optionally propagate SOK to increase add-on compatibility or operation quality. Niche players attempt to acquire certified partnerships by acquiring, monitoring and propagating SOK themselves, as well as by continuously responding adequate to received SOK.

Portfolio and product line planning Ecosystem participants are supported by acquired or received SOK in deciding which new or refined functionality is included in which future versions of their software. SOK analysis may unveil which features are used often, appreciated or wanted most, or cause crashes at the majority of end-users in the field, by which, for instance, release planning activities are supported.

SECO orchestration strategy and policy development Keystones utilize SOK for developing or perfecting their SECO orchestration strategies and policies, to increase attractiveness of their ecosystem. For example, if many software applications based on the keystone's platform are crashing, SOK analysis can be performed to identify most-occurring software failures. Next, based on SOK analysis results, a keystone can (re)define quality level agreements and compatibility certifications to increase the platform's robustness, and therewith increase the attractiveness of the ecosystem compared to rival ecosystems.

3. PRACTICE IDENTIFICATION APPROACH

To investigate the role of SOK in software ecosystems and to empirically evaluate how and to which extent product software vendors have addressed the challenges described in section 2.2, we conducted extensive semi-structured interviews with in total seven employees of four software-producing organizations that are participating in one or more software ecosystems (see table 2). The interviews consisted of 41 questions that were formulated after literature study, divided over five sections[2]. Interview sessions took 1.5 hour on average and were conducted on-site, or via conference calling, over a total period of 35 days. Organizations were contacted from industrial networks. Basis for contacting organizations for interviews was the extent to which an organization utilizes SOK, and propagates SOK within the software ecosystems it operates in.

Interview answers were recorded and used to determine to which extent, and, if applicable, how organizations addressed ecosystem challenges through utilization and propagation of SOK. Having clustered the organizations per challenge they addressed, we subsequently derived aspects that are common to these organizations, and could be generalizable to similar organizations that have addressed the same challenge. Derivation of these aspects (i.e. *Key derivations*) was conducted from the perspective of our position stated in this paper. Key derivations were found during inductive and deductive iterations of interview answers study. The key derivations form the basis for the classification of orchestration and propagation SOK practices in section 5. As the future intensions in section 4.6 are not yet operational, we deliberately omitted those from the classification.

[2]All questions are available at
http://people.cs.uu.nl/schuurhw/sokpropagation

4. IDENTIFIED SOK PROPAGATION PRACTICES

After having introduced the four vendors, we describe for each challenge presented in section 2.2 how two or more vendors address these challenges through SOK utilization and propagation. Note that for reasons of objective comparability, company names as well as names of software products and techniques have been anonymized.

OSComp OSComp currently employs 80,000 people of which one-third is performing software engineering activities. The ecosystems around OSComp's operating system, office and software development products are only a few examples of ecosystems in which OSComp fulfills a keystone role. Also, the company is a niche player actor in Apple Mac OS X and iPhone software ecosystems. OSComp provides technologies to enable niche players to develop desktop applications, web applications and extensions. The company acquires knowledge of its in-the-field software operation by means of its Error Reporting (ER) technology, for acquisition of crash and failure data (excluding operation environment data), and the Analysis Component (AC) technology for collecting software operation data (including operation environment data) of particular customers.

NetComp NetComp currently employs 67,000 people, of which one-third are software engineers developing communication, collaboration and operating system software at 24 locations world-wide. The company is the keystone in the software ecosystem around its Net software that is running on the vast majority of NetComp's hardware. Niche players in this ecosystem are partners of NetComp, and contribute to new releases of the Net software. With keystones as Nokia and OSComp, NetComp considers itself a niche player in the ecosystem of collaboration and conferencing software: the company develops plugins for end device software platforms to enable collaboration and conferencing on the corresponding devices. Knowledge of operation of NetComp software on idem hardware is acquired by NetComp only with accordance of its customers.

ERPComp ERPComp is a global ERP software vendor with 2,100 employees (250 software engineers) that fulfills different roles in several ecosystems. Its main product portfolio consists of an offline desktop application, ERPComp ERPOffline, and an online, service-based web application called ERPComp ERPOnline. With both applications based on OSComp technology, ERPComp is in the top 50–100 of OSComp's global independent software vendors team and is therefore a partner niche player in OSComp ecosystems. ERPComp participates as a keystone in its dealer and partner networks: the vendor has initiated technology platforms with SDKs or only APIs to enable dealers to develop niche add-ons for its ERPOffline product, or to allow tighter integration and collaboration between its ERPOnline product and niche players like banks or governmental tax institutes.

CADIntComp CADIntComp is an organization that develops a Computer Aided Design (CAD) platform, and is

	OSComp	NetComp	ERPComp	CADIntComp
SECO platform	OS-, office- and development platforms	Net	ERPOffline, ERPOnline	CadInt
Platform technology	.NET	Various	.NET	.NET
SOK propagation infrastructure	Quality service, error reporting	Net hardware	OSComp Quality service	Error reporting, member portal
SECO participation incentive	Logo program	Partner alliance program	Developer key	Consortium membership
Number of employees	80,000	67,000	2,100	25[a]
Number of interviewees (role; avg. years of IT employment)	1 (principal researcher; 28)	1 (CIO Europe; 23)	3 (technology research director, product line manager, principal research engineer; 15)	2 (development director, quality assurance manager; 24.5)

[a]Excluding employees of consortium members

Table 2: SECO characteristics of the four software-producing organizations researched

owned and governed by its members. Members of CADIntComp integrate the main technology of this platform, CADInt, in their own software products and customize it to serve niche engineering markets. In turn, CADIntComp uses .NET technology to build CADInt. The CADIntComp consortium consists of about 50 members from 38 countries, mostly software vendors, which form the niche players in the software ecosystem of which CADIntComp is the keystone. Niche players in the CADInt ecosystem may be keystones in other software ecosystems. For example, one of CADIntComp members integrates the CADInt technology in its CAD product software for building industry. Simultaneously, the vendor has built a platform for manufacturers of engineering materials to provide and publish CAD software drawing symbols that are used in its software for drawing these materials. Two types of niche players participate in the CADInt ecosystem: commercial members (which have full source code access and may contribute to the platform) and API members (which only have access to APIs).

4.1 SECO Relationship Management

OSComp Niche players can enter most of OSComp's software ecosystems by producing software for the corresponding platforms. The company attempts to intensify relationships with niche players by stimulating them to become (certified) partners through certification and compatibility programs. For example, by means of the Quality service that is part of OSComp's Logo Program, the vendor verifies compatible operation of hardware and software of certified niche players with most recent versions of its own software (e.g., the operating system software). Also, the company propagates knowledge of software operation failures to niche players certified for program participation.

NetComp Companies are allowed to participate in NetComp's ecosystems through the NetComp partner program, where the volume of the potential niche player as well as partnership capacity available at NetComp determine wether or not a niche player is recognized as an official partner. Knowledge of in-the-field operation of niche player software is used by NetComp to promote or acquire niche players. Also, relationships with niche players are diminished or discontinued when acquired software operation data indicates defective niche player software. Strategic alliances are formed with corporations like IBM and AT&T, players with which NetComp interchanges engineers and to which NetComp discloses virtually all of its intellectual property.

ERPComp Niche players are invited to ERPComp's ecosystems by obtaining an SDK license (which, for instance, allows developers to add new functionality and extend data models) or registering for an API developer key (which enables developers only to import data from and export data to the ERPComp software). ERPComp stimulates its niche players to develop software for distinct niche markets, or to collaborate with other niche players by developing a joint add-on (for instance when too many players are developing equivalent add-ons for one and the same niche market).

Key derivations Keystones provide different incentives for stimulating niche players to participate in their software ecosystem, and therewith initiate SOK acquisition from niche player software. Selection of incentives is based on the role and behavior of niche players as well as on the keystone's capacity and resources available for supporting the niche player. Keystones attempt to initiate niche player relationships via developer keys, after which relations are strengthened through certification programs and (private) SDK access. SOK resulting from *monitoring and verification of software operation compatibility* is used as input for relationship management. Ultimately, keystones merge niche player partnerships or form strategic alliances with niche players.

4.2 SECO Composition Comprehension

OSComp According to OSComp, software ecosystems compositions are continuously changing. Therefore, effective SOK acquisition is practically the only way for the vendor to gain and maintain insight in the ecosystems in which it is operating. For example, if a niche player joins the OS ecosystem and releases a new hardware driver, OSComp can be aware of it in a matter of days. Also, received SOK provides OSComp with insight in the actual distribution of (particular versions of) its software over its software ecosystems.

NetComp NetComp is legally obliged to register what are the physical operation locations of its network products, so that it can prove that its hardware is not placed illegally at certain locations or in particular countries. In addition to knowledge of *how* Net is operating in the field (e.g. hardware uptime, number of unexpected shutdowns, request/response failure rates, etc.), knowledge of *where* Net hardware is currently operating, is part of Net operation knowledge. Hence, SOK is vital to NedComp for monitoring the geographical location of its hardware, and therewith to NetComp's comprehension of its Net ecosystem.

Key derivations Keystones utilize SOK to gain and maintain insight in their ecosystems' compositions. SOK is not only used to *monitor ecosystem entrance and exit of niche players and their software*, but also to keep track of the physical location of the hardware on which the software is operating. Vendors may be (legally, or otherwise) constrained to register such ecosystem composition information.

4.3 Software Quality Management

OSComp OSComp's ER service has helped to improve the quality of many niche player device drivers, which caused a majority of its OS crashes in the past, and alleviated the demanding challenge of isolating obscure Heisenbugs (bugs that disappear or alter when an attempt is made to study it) [8]. According to OSComp, it is impossible to extensively test all unique devices, let alone unique device configurations. As an attempt to nonetheless maintain and manage the quality of its software, the company uses its ER service for error report diagnosis, statistics-based debugging and automatic direction of end-users to solutions of corrected software operation failures. Since 2009, more than 700 ecosystem participants with roughly 7000 applications were using the ER technology, of which 175 participated with registering almost 1500 solutions to hardware- or software operation failures.

NetComp Knowledge of in-the-field Net operation is used to improve the quality of NetComp's software. When operation data indicate software failures, regression analysts are asked to determine the 'real', underlying cause of the failure by mining acquired operation data, after which the bug that causes the failure is solved and the hardware involved is remotely updated with the most recent version of the software. With respect to software quality management, NetComp's goal is to be the first to identify and address failures of its software.

ERPComp ERPComp has certified its ERPOffline product for OSComp's Logo Program, and SOK originating from in-the-field ERPOffline operation is propagated to ERPComp by means of OSComp's Quality service. Since error reports created with this service provide ERPComp insufficient data to identify the exact cause of software failure, ERPComp has developed an analysis tool that end-users can download to determine the exact problem source. When ERPOffline operation failure is caused by a niche player add-on, the niche player is contacted by ERPComp to ensure the issue is solved adequately. Concerning operation of ERPOnline, ERPComp acquires and monitors performance, quality, usage and feedback data during operation of the ERPOnline web application to quickly respond to performance issues and answer usage trend questions (e.g. regarding Internet Explorer 6 usage). The vendor is able to identify niche players responsible for malformed API requests or responses, by filtering operation data on the corresponding developer key. ERPComp propagates SOK to partner niche players, for instance to evaluate and improve operation and compatibility of couplings between its ERPOnline application and partner services.

CADIntComp Within CADIntComp's CADInt ecosystem, SOK propagation contributes to improvement of software quality and reproduction of rare bugs. CADIntComp has built a member portal which is vital to the propagation of CADInt operation knowledge through its ecosystem. Apart from patches, language translations and feature requests, members submit crash logs containing a mini memory dump, call stack as well as other software operation environment details to the portal. Crash logs are mined and analyzed, after which CADIntComp quality assurance employees are notified if critical information has been submitted.

Key derivations Vendors use SOK to manage quality of all kinds of software. It is clear that adequate SOK propagation is considered particularly advantageous in *determining the root cause of in-the-field software failures* that do not occur at the software vendor site, or in isolating Heisenbugs [8]. The more vendors are able to acquire and effectively utilize knowledge of software operation, the more these vendors are successful in managing quality of their software.

4.4 Portfolio and Product Line Planning

OSComp Within OSComp, SOK is used in every phase of the software life cycle. Roadmaps of new software products are based on the usage and quality of the most recently released software product as well as SOK acquired from the last product. For example, a new user interface introduced by OSComp in its office software is largely based on SOK (particularly, software usage data) that is acquired during operation of earlier versions of the software. Also, SOK is used to determine the schedule and composition of new software releases: inter alia crash data and end-user feedback are used to determine the impact of particular in-the-field software operation failures, and therewith decide on the priority of including a fix to these failures in a next release of the software.

NetComp Operation knowledge of NetComp software operating on set-top boxes in the field is acquired to measure feature adoption and therewith determine cost-effectiveness of the boxes. In other words, if operation data analysis indicates that certain features of the set-top box software are rarely used by end-users, NetComp considers the software to offer too much (or too specific) functionality. The company then attempts to build a new version of the box operated by the same software minus the rarely-used features, for a lower price.

ERPComp While ERPComp mainly uses acquired or received SOK in release management and product line planning processes to decide which software mutations (updates with bug fixes or new functionality) affect most customers positively, the vendor foresees SOK to be increasingly used in optimization of internal business processes. Based on SOK analysis, the vendor witnessed a significant change in its average end-user's work-life balance. ERPComp could use this (location-dependent) knowledge not only as input for portfolio planning processes, but also for staffing its support departments, for example.

CADIntComp According to CADIntComp, propagation of SOK through their CADInt ecosystem supports realistic planning of new (versions of) CADInt releases: if crash log mining indicates weak areas in CADInt software, CADIntComp focuses on strengthening these areas when determining specifications for the upcoming release. Code reposito-

ries, build- and test results, roadmaps and design specifications as well as information resulting from crash log mining in the form of operation reports, allow members to get insight in build status, development progress and trends concerning in-the-field software operation quality and performance.

Key derivations SOK propagated to software ecosystem keystones is utilized by these participants to determine the composition of new versions of their software, and to schedule release of these versions. In the long term, software vendors use received SOK to *optimize the cost-effectiveness of their software products* by aligning product functionality with actual feature adoption, as well as internal business processes (e.g. support department staffing).

4.5 SECO Orchestration Strategy and Policy Development

OSComp Apart from software quality, OSComp uses its Quality service to monitor the satisfaction of its partners as well as the satisfaction of its partners' end customers. This allows the company to recognize and orchestrate areas of improvement across an ecosystem. If such areas are identified, OSComp assists involved niche players in realizing improvement of software quality or partner satisfaction, and therewith increase stability and attractiveness of its ecosystems. In addition, OSComp uses its certification programs as a way to control the level of collaboration with niche players. If a particular participant appears to distort ecosystem stability, OSComp can diminish the player's ecosystem participation, for example by limiting its platform privileges.

NetComp One of NetComp's strategies to orchestrate its software ecosystems is to adapt its partner alliance program specifically to the needs of particular partners. Based on a partner's software operation data history, NetComp creates partner-specific incentives and policies. Also, the company involves partners of certain alliances in upscaling the operation speed and scale of its ecosystem. By prospecting potential access to its intellectual property to partners, NetComp stimulates partners to build robust software and behave well in its ecosystems.

Key derivations The way SOK is used by keystones to orchestrate their software ecosystem is influenced by the trust between these partners. With a relative lack of trust, keystones concentrate their incentives on niche player participation. With trustful relationships, keystones adapt their incentives and policies per partner, to *maximize each partner's contribution to ecosystem stability and attractiveness.*

4.6 Future Intentions

OSComp According to OSComp, one of the biggest challenges in software development is to understand the actual intent of end-users during software operation. While software crashes that are a direct result of an end-user action obviously are considered as unwanted behavior, it is often unclear what was an end-user's actual intent when a window, waiting for response, was closed. Also, the company expects software development processes to further evolve around the question 'What is the end-user doing?'. To an increasing extent, operation failures are no longer constrained to one machine, one piece of software or one software vendor. End-users increasingly interpret operation failures as a failure of the 'total thing', i.e., the computing experience as a whole. One of OSComp's future challenges is to get a more thorough insight in the cause and frequency of software that is not fulfilling end-user expectations, based on SOK that is acquired by or propagated to partners participating in its software ecosystems.

NetComp For NetComp, challenges lie in responding adequately to acquired or received software operation knowledge. The relatively uncomplicated software operating on set-top boxes is automatically updated when, based on software operation data, a bug in the software has been identified and fixed. Simultaneously, operation data acquired from NetComp's Net software running on mission-critical instances of its most advanced enterprise hardware, require highly sophisticated response to administrators of the hardware. Therefore, NetComp is investigating ways to not only use the 'power of many' in terms of SOK acquisition and utilization, but also let its employees respond adequately to received SOK. The company foresees a transition to software that is based on a platform of which is formally proven that it is stable and correct; software functionality that is not part of the platform is completely modularized. According to NetComp, such a software architecture will alleviate bug localization, module dependency determination and run-time software component updating processes.

ERPComp In the context of software ecosystems, ERPComp envisages to provide its partner niche players with real-time add-on operation knowledge by means of a central add-on operation dashboard. According to the vendor, adequate dashboard use would result in more intimate collaboration with its partner niche players, and ultimately result in continuous improvement of software operation and software functionality.

CADIntComp While SOK propagation helps CADIntComp in obtaining insight in the capabilities of a niche player in terms of improving CADInt technology, as well as strengthening its relationship with particular niche players, CADIntComp intends to acquire more knowledge of how CADInt is used and appreciated in the field in future. Therefore, it attempts to further intensify relationships with its niche players, since they are in closer contact with actual end-users. CADIntComp may introduce a 'social requirements engineering system', in which the priority of niche players' feature requests is based on a niche player's participation in the CADInt ecosystem, and agreement with the feature request by other niche players.

Key derivations On both SECO scope levels, there are several areas in which keystones plan to improve. First, keystone vendors intend to increase effort to more precisely understand their end-users' actual behavior and intentions during software operation. Also, while SOK is contributive to understanding and responding to end-users, it is a challenge to *determine to which extent response to acquired SOK is experienced as adequate by the end-users.* Third, keystones envisage to propagate acquired SOK realtime to partners, providing them with realtime software operation graphs and statistics, and involve these partners in decision making, for instance concerning requirement prioritization.

		SECO Life Cycle Phase: Creation	SECO Life Cycle Phase: Continuation
SECO Orchestration Focus	**Actor Level**	Trust building (4.1) • Certification programs • Developer keys	Trust strengthening (4.5) • Advanced certification programs • Private SDK access
	Ecosystem Level	Niche player participation (4.1) • Generic participation incentives • Niche teams	Partner contribution (4.5) • Partner-specific incentives • Partner teams
SOK Propagation Focus	**Actor Level**	Software quality management (4.3) • Operation failure cause identification • Heisenbug isolation	Portfolio and product line planning (4.4) • Release composition and scheduling • Cost-effectiveness optimization
	Ecosystem Level	Relationship management (4.1) • Software operation monitoring • Software compatibility verification	Ecosystem composition comprehension (4.2) • Niche player entrance and exit monitoring • Hardware location tracking

Table 3: Classification of identified orchestration and propagation practices. Numbers between parentheses refer to the respective sections presenting the corresponding interview results

5. ANALYSIS OF SOK PROPAGATION PRACTICES

After interpretation, contextualization and abstraction of both the orchestration and propagation practices, as well as the key derivations described in section 4, a classification along two dimensions (SECO life cycle phase and scope level) was established in accordance with the principles for interpretive field research of Klein and Meyers [15] (see table 3). As reflected by the interview results, challenges resulting from software ecosystem participation are not addressed in a predetermined, specific order. Simultaneously, it appears that during the software ecosystem life cycle, focus of SECO orchestration practices (of keystones) and SOK propagation practices (of ecosystem participants in general) changes — both on actor level and ecosystem level.

When keystones begin to orchestrate their software ecosystem, for instance, their orchestration strategy and incentives are focused on building trustful relationships with other software-producing organizations (actor level) and stimulating them to participate as a niche player in their software ecosystem (ecosystem level). Keystones then propagate SOK to address the challenges of relationship management (e.g. through software operation monitoring and compatibility verification) and software quality management (e.g. through utilization of SOK for identifying the root cause of software operation failures, or Heisenbug isolation). In the longer term, keystones attempt to sustain orchestration of their software ecosystem.

On the longer term, orchestration of keystones is focused on strengthening the trust a niche player has in the ecosystem and its keystone (actor level) and on the actual contribution of partners to the ecosystem (ecosystem level). While ecosystem actors continue to be focused on software quality management, SOK is propagated for planning software portfolios and software product lines (e.g. through alignment of product functionality with actual feature adoption), and to maintain insight in the ecosystem's composition (e.g. through monitoring ecosystem entrance and exit).

Based on the identified practices, we particularly consider the challenges described in sections 4.1–4.4 (i.e. relationship management, ecosystem composition comprehension, software quality management, portfolio and product line planning) addressed through utilization and propagation of SOK. We consider the challenge described in section 4.5 (i.e. orchestration strategy and policy development) to a lesser extent supported by the identified practices, since we found limited concrete proof for the addressing of this challenge through actual utilization and propagation of SOK.

While challenges resulting from software ecosystem participation are addressed through effective utilization and propagation of SOK, additional challenges emerge. For example, keystones should be prepared to cope with niche player distrust resulting from SOK utilization (e.g., when SOK propagated to a keystone indicates frequent unstableness of certain niche player software). Second, vendors that acquire, utilize and propagate SOK to understand and serve their end-users, may find it challenging to determine how and when they respond in such a manner, that the end-user feels understood and served well. Third, concerning SOK acquisition, utilization and propagation processes, software-producing organizations will have to find an appropriate balance between realizing adequate customer intimacy and protecting the same customer's privacy.

6. CONCLUSIONS AND FUTURE WORK

While software vendors are aware of the software ecosystems in which they operate and their relationships with other ecosystem participants, vendors all too often are unsuccessful in recognizing the potential value and role of software operation knowledge (SOK) in software ecosystems.

In this paper, we state that SOK forms an essential pulse in the veins of software ecosystems, and serves as a primary enabler for software vendors to thrive and flourish in the ecosystems in which they operate. We substantiate our position by reporting on the operational SOK utilization and propagation practices of four software-producing organizations, which are successfully operating in software ecosystems and address challenges resulting from participation in these ecosystems through utilization and propagation of SOK. With in total four of five challenges addressed by these organizations, we found substantial support for our position.

Based on analysis of the identified practices, we conclude that software ecosystem keystones should continuously invest in infrastructures for SOK acquisition, utilization and propagation to (allow niche players to) effectively utilize the 'power of many' in obtaining knowledge of in-the-field software and end-user behavior, and therewith in increasing the quality and robustness of their software. In the long term, successful SOK utilization and propagation provide keystone software vendors with competitive advantage in terms of ecosystem stability, ecosystem attractiveness, software quality and end-user satisfaction. Ecosystem participants should aim for stable and robust relationships: such relationships contribute to a stable and attractive software ecosystem. Therefore, mutual trust between software ecosystem participants is essential for effective SOK propagation within software ecosystems: distrustful ecosystem participants diminish or even discontinue propagation of their SOK to partners, which may cause these partners to focus on participation in alternative, rival ecosystems.

Future work includes quantification and qualification of SOK propagation in software ecosystems. For example, metrics for measuring the effectiveness of SOK propagation and the contribution of SOK propagation to the vitality of software ecosystems are to be developed.

7. ACKNOWLEDGMENTS

We would like to thank all interviewees of OSComp, NetComp, ERPComp and CADIntComp for their contributions. Also, we would like to thank Gijs Willem Sloof and Ronald Dähne for their cooperation.

8. REFERENCES

[1] J. v. Angeren, J. Kabbedijk, S. Jansen, and K. M. Popp. A Survey of Associate Models used within Large Software Ecosystems. In *IWSECO '11: Proceedings of the 3rd International Workshop on Software Ecosystems*, pages 1–13, 2011.

[2] J. Bosch. From Software Product Lines to Software Ecosystems. In *Proceedings of the 13th International Conference on Software Product Lines (SPLC)*. Springer LNCS, 2009.

[3] J. Bosch and P. Bosch-Sijtsema. From integration to composition: On the impact of software product lines, global development and ecosystems. *Journal of Systems and Software*, 83(1):67–76, 2010.

[4] J. S. Brown, S. Durchslag, and J. Hagel. Loosening up: How process networks unlock the power of specialization. *The McKinsey Quarterly: Risk and Resilience*, (2):59–69, September 2002.

[5] B. Farbey and A. Finkelstein. Exploiting software supply chain business architecture: a research agenda. In *1st Workshop on Economics-Driven Software Engineering Research (EDSER-1), 21st International Conference on Software Engineering*. Electronic, 1999.

[6] A. G. Gawer and M. A. Cusumano. *Platform Leadership*. Harvard Business School Press, Boston, MA, USA, 2002.

[7] K. Glerum, K. Kinshumann, S. Greenberg, G. Aul, V. Orgovan, G. Nichols, D. Grant, G. Loihle, and G. C. Hunt. Debugging in the (Very) Large: Ten Years of Implementation and Experience. In *SOSP'09: Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles*, pages 103–116. ACM, 2009.

[8] J. Gray. Why Do Computers Stop and What Can Be Done About It? In *Symposium on Reliability in Distributed Software and Database Systems*, pages 3–12, 1986.

[9] R. Häcki and J. Lighton. The future of the networked company. *The McKinsey Quarterly*, (3):26–39, 2001.

[10] M. Iansiti and R. Levien. Strategy as Ecology. *Harvard Business Review*, 82:68–78, March 2004.

[11] M. Iansiti and R. Levien. *The Keystone Advantage: What the New Dynamics of Business Ecosystems Mean for Strategy, Innovation, and Sustainability*. Harvard Business School Press, August 2004.

[12] S. Jansen, A. Finkelstein, and S. Brinkkemper. A Sense of Community: A Research Agenda for Software Ecosystems. In *ICSE Companion*, pages 187–190, 2009.

[13] S. Jansen, A. Finkelstein, and S. Brinkkemper. Business Network Management as a Survival Strategy: A Tale of Two Software Ecosystems. In *IWSECO'09: Proceedings of the First International Workshop on Software Ecosystems, co-located with the 11th International Conference on Software Reuse*, pages 34–48, 2009.

[14] H.-B. Kittlaus and P. N. Clough. *Software Product Management and Pricing: Key Success Factors for Software Organizations*. Springer Publishing Company, Incorporated, 2009.

[15] H. K. Klein and M. D. Myers. A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems. *MIS Quarterly*, 23:67–93, March 1999.

[16] D. G. Messerschmitt and C. Szyperski. *Software Ecosystem: Understanding an Indispensable Technology and Industry*. MIT Press, 2003.

[17] Microsoft Customer Experience Improvement Program. http://www.microsoft.com/products/ceip/.

[18] H. van der Schuur, S. Jansen, and S. Brinkkemper. Becoming Responsive to Service Usage and Performance Changes by Applying Service Feedback Metrics to Software Maintenance. In *23rd IEEE/ACM International Conference on Automated Software Engineering - Workshop Proceedings (ASE Workshops 2008)*, pages 53–62. IEEE Computer Society, 2008.

[19] H. van der Schuur, S. Jansen, and S. Brinkkemper. A Reference Framework for Utilization of Software Operation Knowledge. In *SEAA'10: 36th EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 245–254. IEEE Computer Society, 2010.

[20] H. van der Schuur, S. Jansen, and S. Brinkkemper. Reducing Maintenance Effort through Software Operation Knowledge: An Eclectic Empirical Evaluation. 2011. . Accepted for publication in the proceedings of the 15th European Conference on Software Maintenance and Reengineering (CSMR 2011).

[21] Windows Quality Online Services. https://winqual.microsoft.com/.