



Designing a Software Ecosystem for Co-Creation: An Empirical Study of the Platform Infrastructure of an Enterprise Software Channel.

Journal:	<i>17th European Conference on Information Systems</i>
Manuscript ID:	draft
Submission Type:	Research-in-Progress Paper
Keyword:	Software ecosystems, Co-creation, Business innovation, Open Innovation



DESIGNING A SOFTWARE ECOSYSTEM FOR CO-CREATION: AN EMPIRICAL STUDY OF THE PLATFORM INFRASTRUCTURE OF AN ENTERPRISE SOFTWARE CHANNEL.

Abstract

This paper uses an inductive process to formulate a model of platform infrastructure design in technology ecosystems. Our data come from a study of a business software ecosystem consisting of a major, multinational software manufacturer and a system of independent implementation partners and solution developers. We describe the primary platform infrastructure elements which enable value creation in the ecosystem, as well as the party responsible for each element (keystone, partner, or a third Party). We identify alternative structure designs for co-creation between the Software Vendor and the channel Partners. Finally, we explore additional opportunities for co-creation between other members in the ecosystem (Customer-Customer, Partner-Partner, Software Vendor-Customer). This research attempts to address gaps in prior research by focusing on the role of the intermediary in value creation and describing the specific platform infrastructure elements which enable value appropriation in open innovation, emphasizing that supporting these structures need not be the sole responsibility of the core participant. By sharing this responsibility with other participants, opportunities for co-creation of value may be greatly extended.

Keywords: technology ecosystems, channel structure, open innovation, value appropriation, co-creation

1 INTRODUCTION

Recent literature has used the concept of ecosystems to describe the complex interdependencies in various industry sectors. The use of this biological metaphor is an acknowledgement that no one firm can address every customer need, and that the health of each firm in a given sector is dependent on the overall health of the business ecosystem (Iansiti and Levien 2004, Iyer et al. 2006, Adner 2006). Central, core participants in an ecosystem known as keystone players create value by developing “platforms” consisting of services, tools or technologies (Iansiti and Levien 2004). These infrastructure elements have also been called “toolkits” in other research (Jeppesen and Molin 2003, Prugl and Schreier 2006, von Hippel 2001). These toolkits encourage other firms and individuals to create complementary products or add-ons. The Open Innovation literature (Chesbrough 2003, Gassmann 2006) similarly argues that a successful open innovation strategy will include incentives and tools for innovation outside the firm, combined with some mechanism for value appropriation.

One limitation of the prior research is that it generally considers the issues of platform design and value appropriation from the point of view of the keystone, or core participant in the channel. The development of toolkits, for example, is considered the responsibility of the product manufacturer (Jeppesen and Molin 2003; Prugl and Schreier 2006; von Hippel 2001). Likewise, the discussion of value creation and capture by Chesbrough and Appleyard (2007) refers to the ability of a company to extract economic value from their role at the center of an ecosystem through open innovation.

Prahalad and Ramaswamy (2003), in contrast, discuss co-creation experiences, where individuals interact with an “experience environment” composed of a nodal company, suppliers, partners and customer communities in creating value. The authors view innovation from the perspective of the end-user or customer, suggesting that value creation is not company-, product- or even customer-specific, but is rather “the purposeful interaction of the individual consumer with a network of companies and consumer communities that enable personalized experience” (p. 14).

Von Hippel (2001) studied user innovation communities, where consumers adapt an existing product to their specific needs without the need for further intervention from the manufacturer. This idea of community-based innovation has also been studied at length in research on open source communities (for example, Lakhani and Wolf 2003; Hertel et al. 2003; Hars and Ou 2002; von Hippel and von Krogh 2003).

The research cited above has focused on innovation from the perspective of the keystone/core participant in an ecosystem, innovation within user communities, and co-creation between producers of a product or service and the end-user/customer. However, we believe that the role of intermediaries in co-creation has not received sufficient study in prior research, and claim that there are also powerful co-creation opportunities between the keystone player, intermediary participants, and customers. In addition, we assert that there is a need for more empirical analysis of the elements which comprise the specific platform infrastructure in place in an ecosystem, emphasizing that these structures, processes and tools may be managed either by the keystone player, by another participant, or by a combination of ecosystem participants acting in concert. Furthermore, we propose that careful design of these structures and processes enables the co-creation of value among the various participants.

The purpose of this paper is to identify and analyze the elements which compose the platform infrastructure of a large software manufacturer, and to explore the ways in which the design of this particular platform enables co-creation between the software manufacturer and other participants, with particular emphasis on a critical intermediary, channel partners. In this study we have adopted the “co-creation” construct because of its emphasis on shared innovation and value creation among the various participants. Co-creation opportunities between other ecosystem participants are also considered. Accordingly, we formulate the following main research questions:

RQ 1 What are the primary platform infrastructure elements required for the coordination of technology ecosystems?

RQ 2 Who is responsible for the primary platform infrastructure elements (keystone, partner, or third party)?

RQ 3 What possible alternative designs might maximize the co-creation of value in the ecosystem?

We answer these questions through an inductive process using data collected in a study of a technology ecosystem in the enterprise software industry.

2 DATA

We conducted 12 interviews during the period of November 2007 to March 2008 with a variety of entities active in the sales channel for the enterprise software suite of a major, multinational software manufacturer (due to non-disclosure agreements in place, the software manufacturer which sits at the core of our study is hereafter referred to as “the Software Vendor”), including representatives from the Software Vendor’s training center and productivity center, as well as partner managers. Software Vendor respondents were selected from a wide enough selection of functional areas within the Software Vendor in order to understand the full breadth of incentives and infrastructure in place. In addition, we interviewed a selection of independent implementation partners. Partners were chosen based on initial recommendations from Software Vendor management, and subsequent recommendations from the interviewees, with the objective of interviewing a representative cross section (size and industry, as well as local and international).

3 PLATFORM INFRASTRUCTURE

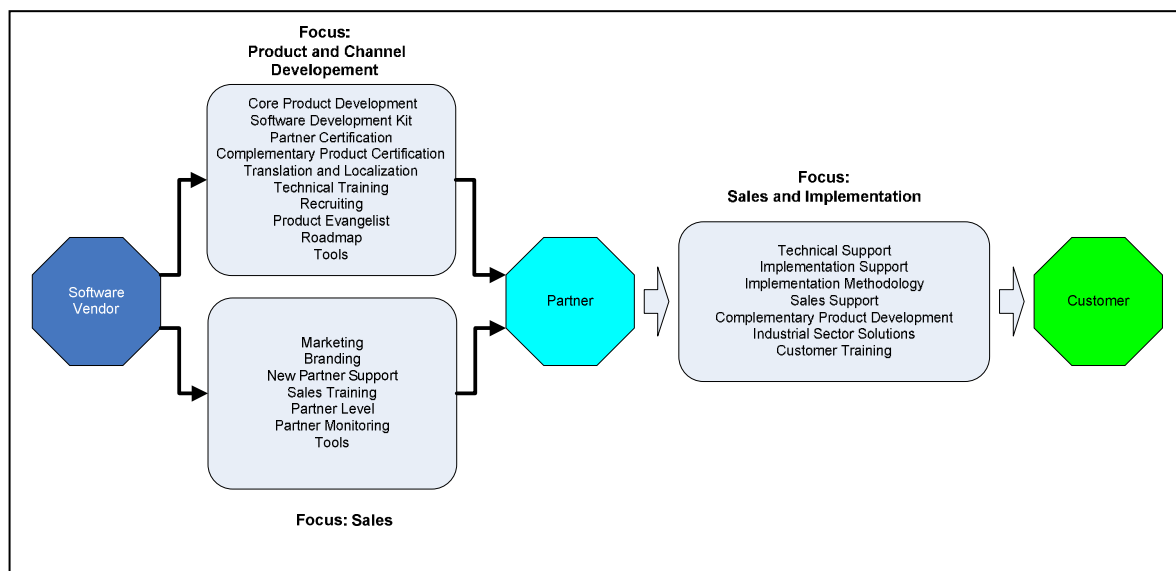


Figure 1. Elements of the platform infrastructure in the software ecosystem

Figure 1 above shows the three main roles in the software ecosystem: the Software Vendor, Partners, and Customers. While we have used the generic term “Partner”, it must be acknowledged that there are a wide variety of companies included in this group who differ on many dimensions such as size; nationality; regional vs. national vs. international focus; and type (independent software vendors, value-added resellers, etc.). In addition, while we believe the list of elements comprising the platform

infrastructure to be comprehensive, it is not meant to be an exhaustive representation of all possible processes, structures and tools in use. Rather, it represents the primary areas necessary to support value creation in the ecosystem, as identified by the participants in our study.

We divided the platform elements which coordinate the relationship between the Software Vendor and the Partner into two areas: focus on product and channel development, and focus on sales. The platform elements which coordinate the relationship between the Partner and the Customer are focused mainly on sales and implementation. The individual elements are described below.

3.1 Focus: Product and channel development

Core Product Development: The Software Vendor is responsible for the development of the core enterprise software suite, as well as the software development kit. The Partner is responsible for configuring and customizing the core software for individual Customers. Revenue from sales of software licenses is shared between Partners and the Software Vendor.

Software Development Kit: The Software Vendor is responsible for maintaining the SDK, which contains the development tools for customizing the software and for creating complementary products. The SDK includes information such as the software architecture, entity model, security model, etc.

Partner Certification: In order to represent the Software Vendor's products, Partners must have certain certifications (technological, functional, product, etc.), which are handled by training centers run by third parties.

Complementary Product Certification: Product certification is the responsibility of third parties with the direction of the Software Vendor. By certifying their complementary products, Partners receive the "seal of approval" from the Software Vendor which helps differentiate the Partner in relation to its competitors.

Translation and Localization: Adapting the core software to local languages and regulatory requirements is the responsibility of the Software Vendor. However, Partners often make additional modifications to address perceived gaps in the localized solutions.

Technical Training: Technical training is the responsibility of external training centers with the certification of the Software Vendor. The Software Vendor also implements training programs in conjunction with universities.

Recruiting: Recruiting is the responsibility of the Partner, though the Software Vendor influences the supply of qualified consultants through its university programs.

Roadmap: The Software Vendor maintains the product roadmap, which keeps the partners informed as to the timing of future updates and upgrades, as well as the specific changes to the software included in the updates/upgrades. This is a critical area since Partners are responsible for maintaining compatibility between their complementary products and the core software.

Product Evangelist: Product evangelists are employed by the Software Vendor to promote the core product to Partners and Customers, helping them understand the product roadmap including the benefits of new product features.

Product and Channel Development Tools: To support the relationship between the Software Vendor and the Partners, the Software Vendor maintains an extranet which provides up-to-date information on the various platform elements such as the product roadmap, certification requirements, software development kits, the availability of complementary products, etc.

3.2 Focus: Sales (relationship between Software Vendor and Partner)

Marketing: While the Software Vendor and Partners engage in their own marketing campaigns, there are some co-marketing efforts where the cost is shared between the parties.

Branding: The Software Vendor is responsible for developing and maintaining the brand of the core software products. Partners are responsible for developing and maintaining their own company and product brands.

New Partner Support: Third parties, with the support of the Software Vendor, maintain external centers to help train new, generally smaller, partners in marketing, sales, and product implementation issues. These centers are evaluated based on their ability to impact the sales growth of new partners.

Sales Training: Third parties are responsible for sales training with the support of the Software Vendor.

Partner Level: There are three partner levels which roughly relate to partner size. Higher levels include higher levels of technical support, training, marketing and sales support, use of Software Vendor logos, and access to international projects.

Partner Monitoring: Partners are primarily monitored via the Software Vendor's extranet, where the Partners report their license sales. There are also third party centers for comparing a Partner's performance with channel benchmarks.

Sales Tools: The Software Vendor's extranet provides tools to report sales and to maintain information on a Partner's sales pipeline. There are also requests for proposal for collaboration opportunities between international partners.

3.3 Focus: Sales and implementation (relationship between Partner and Customer)

Technical Support: Technical support to the Customer is the responsibility of the Partner, however the Software Vendor may provide additional support depending on the Partner Level.

Sales and Implementation Support: On most projects, the Partner retains sole responsibility for selling and implementing the Software Vendor's solutions. However, on larger implementation projects the Software Vendor gives additional, direct support to the Partners in areas such as sales, technical support and project management. On very large projects the Software Vendor takes on the role of primary contractor and subcontracts areas of the project to Partners.

Implementation Methodology: The Partner may use no methodology, the Partner's own methodology, or the methodology developed by the Software Vendor. Consistent use of implementation methodology is critical as a way to reduce project time, increase project success, and reduce total cost of ownership.

Complementary Product Development: Partners may create their own complementary products for Customer needs which are not adequately addressed in the core product. There is a high degree of variation in size and scope of these products. The Partner owns the software licenses for its complementary products, and is responsible for technical support, Customer training, and for maintaining compatibility with future versions of the core software.

Industrial Sector Solutions: While these are also a type of complementary product, they represent much more comprehensive solutions that address a given industrial sector. These may be for a specific geographic region and a highly-focused industry niche, or they may have broader application. Larger partners may develop solutions that become part of the core software product.

Customer Training: While the Software Vendor is responsible for creating the software manuals for the core product, the high level of customization and the large number of complementary products

result in vastly heterogeneous implementations. Customer training is therefore ultimately the responsibility of the Partners.

RESPONSIBILITIES

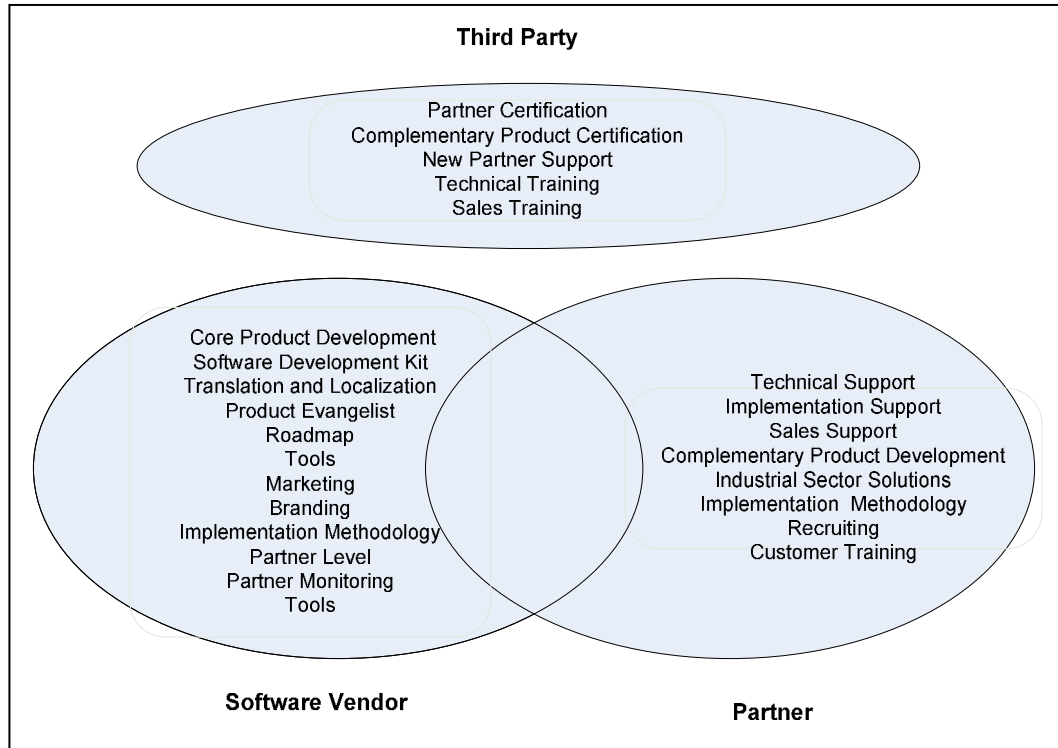


Figure 2. Responsibilities for processes, structures and tools in the software ecosystem.

In Figure 2 we identify the party responsible for each of the platform infrastructure elements. While the majority of the elements are maintained by the Software Vendor, the Partner is also responsible for a significant number of important areas. In addition, there are third parties responsible for areas such as Partner certification and complementary product certification, and new Partner support and training. Several Partners raised concerns as to whether externalizing these elements adds value to the ecosystem. The main issue raised is that these services are essential for the proper functioning of the business, and that allowing third parties to charge for these services may create a conflict of interest. The situation is exacerbated by the fact that there is usually only one firm offering a given service. In a situation with more competition, market forces would drive firms to keep prices reasonable and quality high.

Most importantly, there are no areas where co-creation exists as the norm between the various parties. Therefore, there is a severe lack of mechanisms to ensure that core product strategy and operations are in line with the evolving market, and that overall ecosystem value is maximized. In Table 1 below, we analyze the primary platform infrastructure elements and make recommendations for increasing the co-creation of value.

Infrastructure Element	Responsibility	Advantages	Disadvantages	Co-creation Possibilities
Core Product Development	Software Vendor	Economies of scale, standardization.	Limited to the Software Vendor's view of the market, but Partners are closer to Customer needs.	Provide a mechanism for Partners and Customers to report bugs and request new product features.
Software Development Kit	Software Vendor	Economies of scale, technical product knowledge, standardization.	Difficult to know which features to add to help Partners to increase their productivity	Establish a feedback loop with Partners.
Translation and Localization	Software Vendor	Standardization.	Limited knowledge of local business practices.	Allow Partners to define the requirements of the localizations.
Product Evangelist	Software Vendor	Technical knowledge of products	Focus on technical aspects not "business solutions"	Collaborating with Partners and end Customers could identify new applications for the software.
Roadmap	Software Vendor	Control over product development	Limited to the Software Vendor's view of the market, but Partners are closer to Customer needs.	Provide a mechanism for Partners and Customers to report bugs and request new product features.
Product and Channel Development Tools	Software Vendor	Consistent platform for distributing and collecting information.	Communication is generally broadcast in nature from the Software Vendor to the Partners.	The Software Vendor could maintain the platform while the content could be a collaborative effort between the Software Vendor, Partners and Customers.
Marketing	Software Vendor	Economies of scale, standardization.	Limited ability to target specific Customer segments.	Increase emphasis on co-marketing opportunities between Software Vendor and Partners.
Branding	Software Vendor	Economies of scale, standardization, global brand.	Different markets have varying levels of familiarity with the products.	Increase emphasis on co-branding opportunities between Software Vendor and Partners in local markets and in certain sectors.
Implementation Methodology	Software Vendor	Technical knowledge of products, standardization.	Currently, Partners either use no methodology, their own, or the Software Vendor's.	Developing a unified methodology with the input of Partners and making its use mandatory will increase standardization and result in lower total cost of ownership.
Partner Level	Software Vendor	Establish different levels of support to the Partners.	Partner levels are established based solely on sales quantities.	Adding additional criteria for achieving higher partners could align incentives with important objectives such as quality software development, implementation success,

				lowering total cost of ownership, etc.
Partner Monitoring	Software Vendor	Quality assurance	Focused on one-way reporting from the Partner to the Software Vendor.	Share the results with partners so they can benchmark themselves against their peers.
Sales Tools	Software Vendor	Consistent platform for distributing and collecting information.	Currently limited to reporting of individual partners' license sales and pipeline.	Provide a better mechanism for collaboration between Partners in terms of sharing knowledge, finding complementary products, finding partnering opportunities on projects, etc.
Technical Support	Partner	Partner has technical knowledge of a specific Customer implementation.	Lack of standardization and technical knowledge.	Partners work closely with Customers to help ensure successful adoption.
Sales and Implementation Support	Partner	Partner has sales and implementation knowledge of a specific Customer project.	If partner does not have a good implementation methodology then the risk of failure increases.	In some cases, like large implementations, the Software Vendor can offer large-scale technical and project management expertise.
Complementary Product Development	Partner	Knowledge of specific Customer needs.	Lack of economies of scale, difficult to keep compatibility with core product.	Share development with Software Vendor and/or other Partners.
Industrial Sector Solutions	Partner	Knowledge of specific sector.	Lack of economies of scale, difficult to keep compatibility with core product. Also, there could be significant national differences in certain sectors.	Share development with Software Vendor and/or other Partners.
Recruiting	Partner	Knowledge of specific human resource needs.	Lack of qualified personnel limits channel growth.	Software Vendor could increase collaborations with universities and third-party training facilities.
Customer Training	Partner	Knowledge of the specific Customer implementation.	Lack of standardization and technical knowledge.	Create Customer training materials which allow for Partners to make changes based on specific Customer implementations.
Partner Certification	3 rd Party	Independent evaluation, broad view of the market	Difficult to transfer best practices to the Software Vendor and to other Partners.	There could be better knowledge sharing between the third parties, Partners and the Software Vendor.
Complementary Product Certification	3 rd Party	Independent evaluation, broad view of the market	Difficult to transfer best practices to the Software Vendor and to other Partners.	There could be better knowledge sharing between the third parties, Partners and the Software Vendor.

New Partner Support	3 rd Party	Offers a competency that the Software Vendor lacks.	Difficult to transfer best practices to the Software Vendor and to other Partners.	By taking on this role, established Partners could create relationships with new Partners, and possibly find opportunities for joint business development.
Partner Training	3 rd Party	Independent evaluation, broad view of the market	Difficult to find third parties with knowledge of local markets and specific industry sectors.	By taking on this role, established Partners would be required to increase their own level of knowledge, and the trainees would benefit from access to Partners with extensive field experience. The Software Vendor could offer Partners other incentives such as additional sales and marketing support, technical support, etc.

Table 1. Advantages and disadvantages of the current design as well as co-creation possibilities.

In the table above, the most common advantages offered by the Software Vendor include economies of scale, technical product knowledge, standardization, and quality assurance. The Partners offer practical knowledge gained through implementation, knowledge of specific local markets and industry sectors, and direct contact with Customers which allow them to anticipate changes in the market and in Customer needs. Third parties offer independent evaluation and a broader view of the market obtained from working with a wide range of partners and solutions.

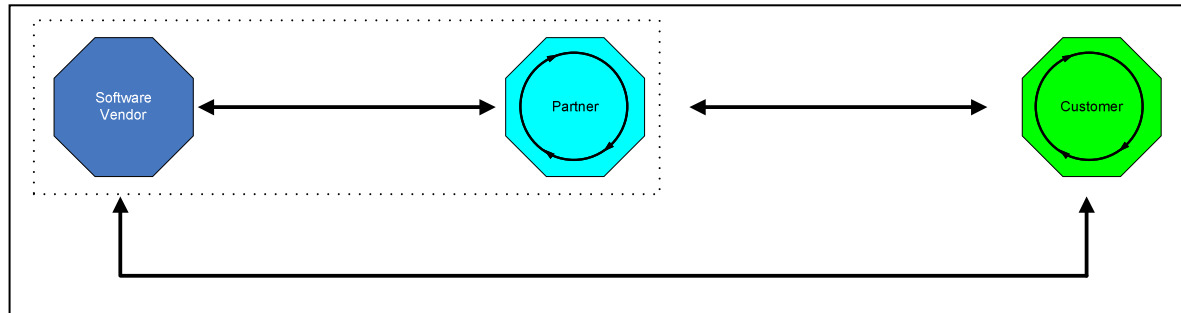
In the areas for which the Software Vendor is responsible, providing additional mechanisms to incorporate Partner input would improve the applicability to specific local markets and industry sectors, and make the core products more responsive to changes in the market. For example, in the areas of “core product development” and the “roadmap”, incorporating a mechanism for Partners and Customers to provide input in terms of errors and new feature requests would improve the Software Vendor’s ability to synchronize its efforts with changes in the market. In the area of “translation and localization”, it is extremely difficult for a software vendor to properly address language and regulatory requirements in every region. Currently, the Software Vendor produces its own local versions, and then individual Partners make further changes to fill gaps not addressed in the official version. The result is several localized versions of the software which are incompatible. Our recommendation based on respondent input is to create a feedback loop between the official localized version and input from the Partners in each region. That way, any further changes made by Partners can be incorporated into the official version.

In areas for which the Partner is responsible, benefits accrue due to the Partner’s knowledge of a specific Customer segment, geographic region and specific Customer businesses. However, a more collaborative environment could help incorporate the knowledge from other Partners and from global best practices. For example, complementary products and industry sector solutions are developed by individual Partners. As a result, only very large partners can afford to create these solutions, and the inclusion of only one Partner’s perspective severely limits the flexibility and scalability of the final product. A more collaborative development environment would allow smaller partners to contribute their knowledge in targeted areas, and would improve the quality of the final product.

While third parties provide the benefits of independent evaluation and a broad view of the market gleaned from contact with many types of partners and products, externalizing these services restricts knowledge transfer between the external parties and the Software Vendor and Partners. Furthermore, partners raised the concern that giving these third parties a profit motive may reduce their ability to

remain independent and objective. We were not able to reach any concrete conclusions regarding third parties, but have highlighted this as an important area for future study.

4 CO-CREATION OPPORTUNITIES



The arrows in the figure above indicate the possible co-creation dyads among the three primary ecosystem participants identified in our study. In our analysis thus far we have emphasized the opportunities for co-creation between the Software Vendor and the Partners, but there are also the following possibilities for co-creation: Partners-Customers, Partners-Partners, Customers-Customers; and Software Vendor-Customers. Specific examples of these types of co-creation in the context of our data are discussed below. Please note that these examples are illustrative and in the present model the construct of “value” is not elaborated upon.

4.1 Co-creation between Partners

As described above, a more collaborative environment between Partners would allow multiple partners with experience in a particular industry segment, for example, to combine their efforts to create an industry sector solution with greater flexibility and economies of scale than either could develop individually. Another example is when a software bug appears partners could share information about the bug along with possible solutions. This kind of group effort would reduce the number of errors and improve the overall quality of the software.

Co-creation between partners with complementary expertise is an even more powerful possibility. For example, a Partner with expertise in a particular industry segment may turn to another Partner for their expertise in a “horizontal” application such as tax reporting, or they may wish to expand internationally and find a Partner in another country who could help localize their vertical application.

4.2 Co-creation between Partners and Customers

Partners could collaborate with end Customers to develop more standardized vertical solutions for an industry. Currently, Partners usually create their vertical solutions based on their experience with one or, at best, a few Customers operating in that industry segment. The Partners claim that their vertical is based on industry standards, but it is often actually based on a simple, small sample. While Partners already have direct contact with Customers and do engage in co-creation through activities such as product customization and business process redesign, these are generally project based and one-to-one, i.e. one Partner to one Customer. Tools which enable many-to-many engagements between Partners and Customers would surface more co-creation opportunities.

4.3 Co-creation between Customers

While there are certainly some areas where Customers might be reluctant to work together for fear of losing competitive advantage, they could, for example, share best practices in areas such as partner

selection, implementation processes, and ways to minimize total cost of ownership. They may also help each other discover new ways to configure the product which avoid costly customization, as well as ways to alter their business processes to maximize product benefits.

4.4 Co-creation between the Software Vendor and Customers

The more the Software Vendor opens up their platform to input from Partners and Customers, the better they will be able to align the development of the core software products with Customers' "real" needs, both from a technical as well as functional perspective. For example, we previously discussed creating a mechanism for allowing Customers to provide direct feedback on features which they would like to see in future versions of the software. In addition, the software evangelists should consider their role as not just disseminating information on new software features, but as collecting information from Customers and Partners as to the real-world applications of these features.

5 FUTURE RESEARCH

This research represents a study of one ecosystem with a particular company at the core and for a particular range of products. In addition, the majority of the data was taken from a specific geographic region (Europe), and the primary focus is on the relationship between the Software Vendor and the channel Partners. We plan to collect additional, empirical data to analyze co-creation opportunities between Partners and other Partners, between Partners and Customers, between Customers and between the Software Vendor and Customers. Future research will also validate the model by collecting data from additional geographies within the Software Vendor ecosystem. The findings and consequent model will be enriched and validated by applying it to ecosystems with different characteristics.

6 CONCLUSIONS

Number	Question	Data Result
RQ 1	What are the primary platform infrastructure elements required for the coordination of technology ecosystems?	Figure 1 presents a comprehensive list of elements which comprise the infrastructure platform in the ecosystem. These elements are categorized based on whether they support the relationship between the Software Vendor and Partner (further divided between "Product and Channel Development" focus and "Sales" focus), or the Partner and Customer (classified as focused on "Sales and Implementation". Each element is described
RQ 2	Who is responsible for the primary platform infrastructure elements (keystone, partner, or third party)?	Based on the data, these elements are classified by responsible party in Figure 2, including elements for which responsibility is shared between Software Vendor and Partners.
RQ 3	What possible alternative designs might maximize the co-creation of value in the ecosystem?	In Table 1, we present the advantages and disadvantages for the way each element is managed in our study. We then describe opportunities for co-creation for each element. We conclude that the Software Vendor offers economies of scale, technical product knowledge, standardization, and quality assurance. The Partners offer practical knowledge gained through implementation, knowledge of specific local markets and industry sectors, and direct contact with Customers which allow them to anticipate changes in the market and in Customer needs. Third parties offer independent evaluation and a broader view of the market obtained from working with a wide range of Partners and solutions. By sharing the responsibility for maintaining certain elements, the ecosystem may

		be able to extract greater value by combining the benefits offered by each party individually.
--	--	--

Table 2. Research questions

The research questions identified in this study and the results for each based on the empirical data collected are described in Table 2. Our research presents an ongoing study of a particular ecosystem which includes the Software Vendor, and an ecosystem of independent Partners, third parties and Customers. We employ data collected from this research to identify the primary platform infrastructure elements in a software ecosystem, and to explore the ways in which the specific design of these elements can increase opportunities for co-creation. Our findings suggest that a more collaborative platform structure based on information and communication technologies can enable co-creation opportunities between the Software Vendor and Partners; between Partners; between Customers; and between the Software Vendor and Customers. Proper design of the platform infrastructure can balance the benefits of size and technical expertise brought by the Software Vendor with the market knowledge of the Partners and the specific business expertise of the Customer. While the analysis of “value” in our analysis has been primarily qualitative, an important objective of future research is to identify ways in which co-creation opportunities can reduce total cost of ownership to the end Customer.

7 REFERENCES

- Adner, R. (2006). Match your Innovation Strategy to your Innovation Ecosystem. *Harvard Business Review*, April 2006.
- Chesbrough, H.W. (2003). *Open Innovation: The New imperative for Creating and Profiting from Technology*. Boston, Harvard Business School Press, 2003.
- Chesbrough, H.W. and M. Appleyard (2007). *Open Innovation and Strategy*. *California Management Review*, (50:1), 2007, pp. 57-76.
- Gassmann, O. (2006). Opening up the Innovation Process: Towards an Agenda. *R&D Management*, (36:3), June 2006, pp. 223-226.
- Hars, A. and Ou, S. (2002). Working for free? Motivations for participating in open-source projects. *International Journal of Electronic Commerce*, 6 (3), 25-39.
- Hertel, B., Niedner, S. and Herrmann, S. (2003). Motivation of software developers in open source projects: an Internet-based survey of contributors to the Linux kernel. *Research Policy*, 32 (7), 1159-177.
- von Hippel, E. (2001). Perspective: user toolkits for innovation. *Journal of Product Innovation Management*, 18, 4, 247–257.
- von Hippel, E. and von Krogh, G. (2003). Open source software and the ‘private collective’ innovation model. *Organization Science*, 14(2), 209-223.
- Iansiti, M and R. Levien (2004). Strategy as Ecology. *Harvard Business Review*, (82:3), 2004, pp. 1-11.
- Iyer, B., Chi-Hyon, L. and Venkatraman, N. (2006). Managing in a ‘small world ecosystem’: Lessons from the software sector. *California Management Review* (48:3), Spring 2006, pp. 28-47.
- Jeppesen, L. B., M. J. Molin (2003). Consumers as co-developers: Learning and innovation outside the firm. *Tech.Anal.Strategic Management* (15:3), 2003, pp. 363–384.
- Lakhani, K. and Wolf, R. G. (2003). Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects. MIT Sloan Working Paper No. 4425-03, URL: <http://freesoftware.mit.edu/papers/lakhaniwolf.pdf>.
- Prahalad, C.K. and Ramaswamy, V. (2003). The new frontier of experience innovation. *MIT Sloan Management Review*. Summer 2003.
- Prugl, R. and M. Schreier. (2006). Learning from Leading-Edge Customers at The Sims: Opening up the Innovation Process using Toolkits. *R&D Management* (36:3), June 2006, pp. 237-250.