Association for Information Systems
AIS Electronic Library (AISeL)

UK Academy for Information Systems Conference Proceedings 2011

UK Academy for Information Systems

Spring 4-11-2011

Enhancing Software-As-A-Service With Insufficient Domain Knowledge

Eruani Zainuddin
Queen's University, ezainuddin@business.queensu.ca

Sandy Staples
Queen's University, sstaples@business.queensu.ca

ENHANCING SOFTWARE-AS-A-SERVICE WITH INSUFFICIENT DOMAIN KNOWLEDGE

Eruani Zainuddin
PhD Student, Queen's University, Goodes Hall, 143 Union St., Kingston, ON K7L 3N6, Canada
Email: ezainuddin@business.queensu.ca

D. Sandy Staples
Professor & Distinguished Faculty Fellow of MIS, Queen's University, Goodes Hall, 143 Union St., Kingston, ON K7L 3N6, Canada
Email: sstaples@business.queensu.ca

Abstract

This study addresses the question "How do Software-as-a-Service (SaaS) vendors enhance their software with insufficient domain knowledge?" Results were obtained by analyzing a dataset from a SaaS vendor that provides administrative software to small schools around the world. The dataset includes archived data (email messages, company documents, and Skype messages) and access to the company's online repositories (sales pipeline, client online chats, and engineering repository). We identified three types of domain knowledge that are relevant to SaaS vendors – organization specific, industry-wide, and regional variation. We also generated six propositions explaining how industry-wide and regional variation knowledge influences the SaaS enhancement process, and at which points in the process these two types of domain knowledge come into play. This study refines our current knowledge by highlighting the unfolding stages between insufficient levels of domain knowledge and software enhancement outcomes.

Keywords: Software-as-a-Service, SaaS, software enhancement, domain knowledge, business knowledge, process model

ENHANCING SOFTWARE-AS-A-SERVICE WITH INSUFFICIENT DOMAIN KNOWLEDGE

Abstract

This study addresses the question "How do Software-as-a-Service (SaaS) vendors enhance their software with insufficient domain knowledge?" Results were obtained by analyzing a dataset from a SaaS vendor that provides administrative software to small schools around the world. The dataset includes archived data (email messages, company documents, and Skype messages) and access to the company's online repositories (sales pipeline, client online chats, and engineering repository). We identified three types of domain knowledge that are relevant to SaaS vendors – organization specific, industry-wide, and regional variation. We also generated six propositions explaining how industry-wide and regional variation knowledge influences the SaaS enhancement process, and at which points in the process these two types of domain knowledge come into play. This study refines our current knowledge by highlighting the unfolding stages between insufficient levels of domain knowledge and software enhancement outcomes.

Keywords: Software-as-a-Service, SaaS, software enhancement, domain knowledge, business knowledge, process model

1.0 Introduction

Software-as-a-Service (SaaS) refers to the selling of software that is owned and managed by the vendor, and delivered as a service over the Internet. The SaaS application is based on a single set of common code and data definitions, and distributed in a one-to-many manner to all clients (Xin and Levina 2008). The Gartner Group (2010) estimated that the worldwide SaaS revenue surpassed the projected forecast of $9.2 billion in 2010, up 15.7 percent from the 2009 revenue of $7.5 billion. Meanwhile, another study conducted by the American IDC (2009) research group projected that fifty percent of organizations will use SaaS for business functions that provide strategic advantage to their organizations. This is a major transition since SaaS is currently known to support mostly non-critical business applications (Gartner Group 2006). These statistics imply that the SaaS market is expanding, and SaaS will have stronger impacts on individual organizations.

SaaS offers several benefits to organizations including low upfront costs, faster implementation, flexible subscriptions (clients can subscribe or unsubscribe at any point in time), and continuous software improvements (vendors need to consistently

enhance their products to remain competitive). In addition, SaaS has a more positive impact on the environment. A survey by Symantec Corporation (2009) revealed that organizations consider SaaS as one of the key strategies to achieve sustainable IT goals. There are however, a few negative aspects to SaaS. These include lack of domain knowledge in SaaS vendors, organizations losing control of their own computing and surrendering control to external vendors, as well as security concerns. Despite these concerns, the future prospect for SaaS remains promising.

Due to its relative novelty, there is a paucity of SaaS research especially from the Information Systems (IS) perspective. The available studies focus on distinguishing SaaS from packaged software (Choudhary 2007, Fan et al. 2008), identifying SaaS adoption factors (Benlian et al. 2009, Xin and Levina 2008), comparing different SaaS pricing models (Zheng et al. 2006), and various technical issues such as architecture, scalability, and security (Cusumano 2010, Hudli et al. 2009, Hurkmans 2009, Nitu 2009). Recent call-for-papers for SaaS-related studies (e.g., cloud computing and service science) in peer-reviewed IS journals show that SaaS is slowly gaining traction among IS researchers and more research is needed. Our study addresses this need by specifically examining the software enhancement process in SaaS.

1.1 Software Enhancement in SaaS

One characteristic of SaaS is the focus on software enhancement. Software enhancement (or perfective maintenance) is the process of accommodating new or changed user requirements (Niessink and van Vliet 2000). This type of enhancement involves adding functionalities to the current software. It is usually a continuous process, without an established end date.

Software enhancement is one of the deciding factors during vendor and product selection in SaaS. Sadegh (2008) highlighted several client expectations for SaaS enhancement, which include monthly enhancement releases, multiple mechanisms to gather client feedback, internal process to incorporate client feedback into product roadmaps, as well as enhancement releases that will not disrupt client operations. Without a high performing software enhancement delivery, SaaS loses its edge over on-premise enterprise software (Choudhary 2007) and over possible competitors.

An essential ingredient to a successful software enhancement delivery is domain knowledge (Kitchenham et al. 1999). Domain knowledge refers to vendors' knowledge of business processes, business rules, policies and procedures, and business objectives of their clients (Tiwana 2009). For example, the domain knowledge for a SaaS vendor offering university admission software includes knowledge of a university's admission process, admission policies, and student selection criteria. Typical SaaS vendors do not have one-to-one and/or prolonged interaction with their clients. As such, SaaS vendors usually do not have a high level of domain knowledge and must operate under insufficient domain knowledge condition. This issue is echoed in a statement by an industry observer, Kevin McCallum (Dye 2008):

> "Elements that are currently missing [with SaaS] are the domain knowledge that a locally sited experienced consultant or reseller can offer, as well as the frequency with which our systems are bespoked to provide absolute fit for that particular business."

We summarize that software enhancement is an important aspect in SaaS and insufficient domain knowledge is a relevant managerial concern.

1.2 Domain Knowledge in Relevant Literature

Domain knowledge has been examined in several different streams of IT research – software development, IT sourcing, and IT-business alignment. A brief summary of the studies investigating domain knowledge under each research stream includes:

- Studies in software development examined the relationship between domain knowledge in software team members and performance (Huckman et al. 2009, Kang et al. 2006, Tesch et al. 2008).
- Studies in IT sourcing examined the relationships between domain knowledge and client-vendor partnerships (Goles 2001, Vlaar et al. 2008), as well as domain knowledge and IT sourcing success (Aubert et al. 2005, Tiwana and Keil 2007).
- Studies in IT-business alignment examined domain knowledge in IT executives, and how domain knowledge influences participation in IT planning and IT-

business executives' partnerships (Bassellier and Benbasat 2004, Bassellier et al. 2003, Kearns and Sabherwal 2006, Reich and Benbasat 2000).

We identify three common findings across previous domain knowledge studies in IT research. First, previous studies have concentrated on investigating the relationship between domain knowledge and performance-related outcome (e.g., efficiency, effectiveness, partnership). Second, domain knowledge is required to effectively execute work activities such as communication and coordination. And three, there is a positive correlation between domain knowledge and performance; higher level domain knowledge implies better performance. We can map the results of previous studies into an input-process-output model as shown in Figure 1.

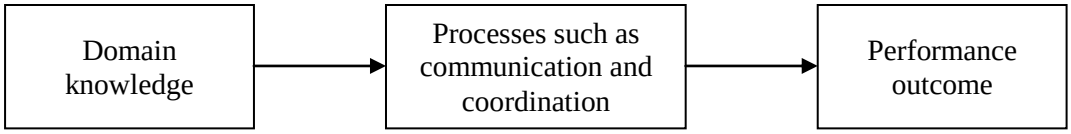

Figure 1. Mapping domain knowledge studies in IT research into an input-process-output model

2.0 Research Question

The available studies in IT research are in agreement that there is a positive relationship between domain knowledge and performance-related outcome. Therefore, we expect SaaS vendors enhancing their products with insufficient domain knowledge to result in a negative outcome. However, this is a rather simplified view of a complex phenomenon. Given that SaaS is a growing and successful practice, there is a need to understand how the process of enhancing SaaS with insufficient domain knowledge unfolds. This research will add to prior research, and address a managerial concern. We put forward the following research question:

> ***"How do Software-as-a-Service (SaaS) vendors enhance with insufficient domain knowledge?"***

It is important to note that the scope for this study is on SaaS vendors supporting enterprise-level software as opposed to individual users. This scope is appropriate after considering the additional complexities involved. SaaS vendors offering

enterprise-level products need to take into account business processes and workflows, organizational level security, as well as multiple users; these issues are not present in software for individual users (Gartner Group 2009). Thus, enhancing SaaS with insufficient domain knowledge is a more germane issue when vendors are offering enterprise-level software products.

3.0 Research Method

Given the current lack of research in this area, our aim is to build a theory explaining how SaaS vendors enhance with insufficient domain knowledge. We used a qualitative data analysis approach to analyze a large dataset from a SaaS vendor. This dataset covered a fifteen-month period from July 2009 until August 2010. Table 1 summarizes the contents of the database.

Category	Type	Details
Archived data	Email messages	A total of 8045 email messages between client and vendor, as well as among internal team members (account managers and engineering team members).
	Company documents	A total of 1911 internal documents containing meeting notes, and sample documents from clients.
	Skype messages	A total of 45232 lines of messages between internal team members.
Online repositories	Sales pipeline	Tracks client-vendor interactions; containing dates, notes, and items/issues communicated.
	Online client chat logs	Contains client inquiries, comments, and questions between clients (users) and account managers.
	Engineering repository	Contains information related to enhancement and fix releases; including source, description, date accepted for implementation, date of release, and engineer(s) assigned.

Table 1. Summary of dataset contents

The SaaS vendor (referred to as GlobalSchool) provides administrative software for small schools (i.e., schools with less than four hundred students). GlobalSchool was

originally a technology consulting and services company, but moved into the SaaS business model in late 2008. About eighty percent of its clients are located in North America, with others are located in other parts of the world.

GlobalSchool runs a highly distributed operation. Its employees are located in three different countries – Malaysia, United States, and Canada. To support this highly distributed operation, employees stay connected through emails and online conferences. The clients' primary contact persons are the account managers. Account managers handle all client inquiries, and forward client requests to the engineering team members. The engineering team members are responsible to implement enhancement requests. Throughout the fifteen-month period examined, the number of account managers varied from two to seven people, while the number of engineering team members varied from two to five people.

3.1 Data Analysis

Data analysis was done in four major steps:

Step 1: Tracing the growth of the software by examining the engineering repository
We began by examining the engineering repository, and classifying each release as either a fix (i.e., correcting defects) or an enhancement. We identified 200 fixes, and 139 enhancements altogether. We further grouped the enhancements based on modules. Each module contains a group of functionality that supports a specific organizational workflow. For example, the mass parent messaging module in GlobalSchool's software allows school administrators to send messages to a large number of parents and guardians simultaneously. This module contains functions that enable school administrators to send emails, voice messages, and text messages to parents and guardians. The modules and number of enhancements attached to them (described in brackets) are as follows: mass parent messaging (eight), parent portal (four), homework (four), report card (eighteen), discipline tracking (two), sports tracking (one), teacher tracking (one), student tracking (four), extra-curricular activities (one), fee tracking (eight), admissions (one), attendance (nineteen), enrolment (four), subscription (nine), and sign-in (one).

Step 2: Creating a history/trail for each enhancement by consolidating information from all sources

The main purpose of this step is to understand the important incidences that occurred within each enhancement release. We systematically conducted the following: (1) keyword searches on the archived data, sales pipeline, and online client chat logs based on enhancement name, module name, enhancement number, and people involved; (2) ordered the related information chronologically; and (3) identified whether the vendor experienced insufficient domain knowledge for a particular enhancement.

Step 3: Patterning the events for the enhancements with insufficient domain knowledge

We examined in detail the enhancements with insufficient domain knowledge. We diagrammed each enhancement using the event-state network outlined by Miles and Huberman (1994). We were able to detect key events as well as their sequences using this method.

Step 4: Comparing across cases, and grouping cases that have similar sequence of events and generating propositions

We compared the event-state network charts created in Step 3. We marked sequences of events that are similar across different network charts. We then generated the relevant propositions.

4.0 Findings

In this section, we explain our findings. First, we describe the general SaaS enhancement process that we found by outlining the sequence of activities (Figure 2). Second, we present the types of domain knowledge that emerged from our dataset (Table 2). Third, we provide a process model describing the stages that SaaS vendors go through when enhancing their products under the insufficient domain knowledge condition. We also offer a set of propositions (and example evidence) explaining how insufficiency in industry-wide and regional variation knowledge influence the events and actions within a particular stage. Last, we indicate where the process for

enhancing while having insufficient domain knowledge fits into the general SaaS enhancement process (Figure 3).

4.1 SaaS Enhancement Process from the Managerial Perspective

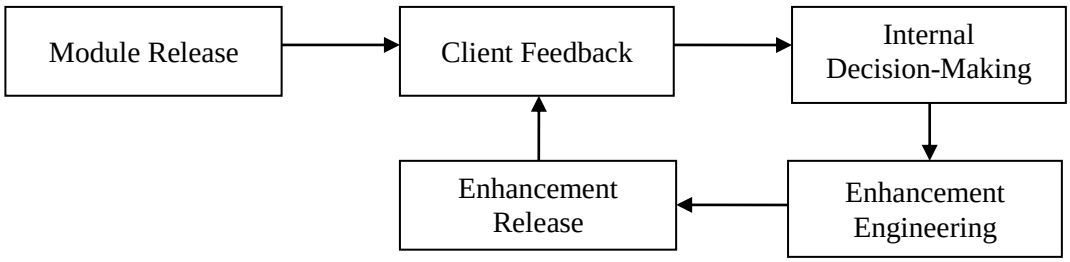

Figure 2. General SaaS enhancement process model from the managerial perspective

Figure 2 represents the overall activities observed in the SaaS enhancement process. This model focuses on enhancements of ***existing modules*** (i.e., adding new functionalities to current modules) as opposed to enhancements of ***new modules*** (i.e., developing new modules). The SaaS enhancement process is initiated when a module is released or made available to clients. Once a module is available to clients, the clients are encouraged to give feedback. Consistent with April et al.'s (2005) classification, we organized client feedback into three categories: (1) requests for assistance or additional information (i.e., operational support), (2) error notices (i.e., correction issues), or (3) suggestions for improvements. We observe from our data that the first feedback category necessitated software corrections, but not software enhancements. The second feedback category necessitated either software corrections or enhancements. Meanwhile, the third feedback category generally leads to software enhancements. Thus, our data suggest that SaaS vendors will consolidate and use feedback from the second and third categories as input for the next stage.

The next stage, which is the internal decision-making stage, is a complex and critical part of the SaaS enhancement process model. In this stage, SaaS vendors develop the roadmap for their products; deciding on whether to accept or reject numerous enhancement suggestions. As suggested by prior research, vendors must incorporate different factors such as resource constraints, technical compatibility, and alignment with strategic goals in finalizing their enhancement decisions (Bennett 1996, Kitchenham et al. 1999). We found examples where suggestions led to enhancements and when they did not, as explained in section 4.3. Once a decision is made to pursue

an idea, the decisions are then forwarded to engineering for further action, and subsequently, new enhancements are released to clients. As in module releases, enhancement releases may also initiate client feedback, and instigate more enhancement work. As explained in section 4.3, we found that main challenges created by insufficient domain knowledge were related to this area of the enhancement process (i.e., interpreting client feedback and potentially acting on it).

The process model described in Figure 2 could be construed as a reactive approach to software enhancement. It appears that SaaS vendors simply wait for client feedback before improving their products. On the contrary, we found that SaaS vendors can be proactive in soliciting client feedback. GlobalSchool for example, aggressively sought client feedback. Almost all correspondences to its clients included a request for feedback, enhancement release notices included a "let us know what you think" statement, and furthermore, its agents routinely made calls to clients with whom they have strong relationships to ask for suggestions. Rather than implying a reactive approach to software enhancement on the part of SaaS vendors, the client feedback stage reflects the principle of incorporating client input into a product roadmap. SaaS vendors may take a more proactive approach in acquiring client feedback, by directly contacting and asking clients to share their knowledge and ideas.

4.2 Types of Domain Knowledge

Three types of domain knowledge – ***organization specific***, ***industry-wide*** and ***regional variation*** – emerged from our dataset. We introduce the definition for each type of domain knowledge in Table 2. We found instances where gaps in industry-wide knowledge and gaps in regional variation knowledge, identified via client feedback, led to product enhancements. We discuss the series of events and actions that take place from receiving client feedback until the related enhancement release in section 4.3.

Interestingly, we did not find any examples of gaps in organization specific knowledge leading to product enhancements. Perhaps this is not surprising since SaaS vendors do not develop custom products for specific clients, and therefore, gaps in organization specific domain knowledge do not come into play.

Type	Definition [Knowledge…]
Organization specific	…that is exclusive to individual organizations. For example, organizational jargons and acronyms, or internal workflows that differ significantly from those typically observed in the industry at large.
Industry-wide	…that is common across organizations within the same industry. For example, common jargons and acronyms within a particular industry, or typical internal workflows.
Regional variation	…that highlights the differences within an industry based on regional or country locations. For example, schools outside of North America require more detailed disciplinary actions tracking (i.e., tracking for drills, detentions, warnings received) compared to schools in North America (i.e., tracking for occurrences of rules violations without any specific details noted).

Table 2. Areas of domain knowledge emerging from the dataset

4.3 SaaS Enhancement under the Insufficient Domain Knowledge Condition

SaaS enhancement under the insufficient domain knowledge condition commences when vendors receive client feedback. Client feedback signals gaps or errors in vendors' products, and vendors begin to recognize insufficiency in their knowledge bases. Next, vendors identify the relevant knowledge for enhancements. Client feedback is sometimes misinterpreted, delaying the implementation of potentially strategic functions. Finally, vendors develop certain strategies to enable them to enhance while having insufficient domain knowledge. We provide further details on the enhancement stages along with a set of propositions below.

Stage 1: Receiving client feedback

SaaS vendors focus on serving the mass market. Hence, their main principle is to implement functions that are useful to a large number of schools. Insufficient industry-wide knowledge often causes vendors to release incomplete modules. Functions that are needed by a large number of organizations are not implemented because vendors are unaware. Vendors will receive multiple client feedback alerting them of such gaps. For example, GlobalSchool released their attendance module without the subject-based attendance function (i.e., a function that allows several

teachers to take student attendance several times throughout the day). They were unaware of the need for subject-based attendance until they received several requests from their clients. Part of the evidence for this example can be found in the following online conversation between an account manager (AC) and the engineering lead (ETL) below:

ETL: And which schools use subject-based?

AC: School X

AC: School Y

AC: Maybe School Z, in the future

ETL: Interesting...

The analysis of the data yielded four other instances where there was an apparent lack of industry-wide knowledge with similar consequence. We therefore offer the following proposition:

> ***Proposition 1a:*** The vendor's insufficient industry-wide knowledge may cause a gap in software functionality; when this occurs, the vendor will likely receive multiple client feedback identifying the gap in its software product.

SaaS vendors also serve the worldwide market. Insufficient regional variation knowledge often causes vendors to implement functions that work only for clients in certain regions (or countries). However, the same functions do not perform as expected and appear as errors for those in other regions. This is due to the variations within the same industry across regions; without sufficient regional variation knowledge, vendors are unable to implement functions that accommodate these variations. Vendors will subsequently receive client feedback informing them of such issues. For example, GlobalSchool received error reports from several of their UK-based clients after the report card module was initially released. These error reports were related to grade calculations; GlobalSchool's software allows for bonus marks and grades to exceed 100 percent. This policy is common for schools in North America, but not for schools outside of North America. The analysis of the data yielded four other instances where there was an apparent lack of regional variation knowledge with similar consequence. Therefore, we posit that:

> ***Proposition 1b:*** The vendor's insufficient regional variation knowledge may cause the software to not perform as expected (an error); when this occurs, the

vendor will likely receive client feedback identifying the error in their software product.

Stage 2: Identifying relevant knowledge

When SaaS vendors receive feedback from their clients, the vendors must be able to identify industry-wide knowledge. Industry-wide knowledge enables vendors to enhance their products for a large number of organizations. Unfortunately, industry-wide knowledge is sometimes misinterpreted as organization specific. SaaS vendors will only be aware of this faulty interpretation after a certain period of time has elapsed, and they have accumulated more client feedback. Vendors face the risk of delayed implementation of potentially strategic enhancement. For example, GlobalSchool received the following request: "Our school combines home schooling into our model… [Do you] have something where teachers can type up the assignments per class and parents [to] be able to view them?" This request was not considered until we made an inquiry to GlobalSchool, and pointed out that there is another similar request in their database from a different school. GlobalSchool team admitted that the first request was ignored since it came from a specific type of school: a school that combines home schooling model with on campus attendance. As such, it was misinterpreted as organization specific as opposed to industry-wide. The related enhancement was scheduled for implementation after our inquiry. The analysis of the data yielded two other instances where there was a misinterpretation of industry-wide knowledge as organization specific, and resulting in implementation delays. We summarize this situation into the following proposition:

> ***Proposition 2a:*** When a client shares knowledge that appears to be organization specific, the vendor tends to disregard this knowledge. This will delay the implementation of a potentially strategic enhancement if the vendor's initial assessment is faulty.

Even when SaaS vendors are able to accurately identify industry-wide knowledge, they need to be able to accurately attach relative value to the knowledge. Vendors have to consider their resource constraints, especially in terms of labour and time. Hence, SaaS enhancements give priority to functions related to high demand (or highly utilized) modules. When vendors discover industry-wide knowledge that they see related to low demand (or underutilized) modules, they often disregard the

discovery. This becomes an issue when a module that is low in demand at present, becomes more in demand in the future. If such a shift occurs, vendors face the risk of delayed implementation of potentially strategic enhancements. For example, requests were made for the ability to "text message [to] the teachers" and "send billing info to parents via SMS." These requests were not considered until we made an inquiry to GlobalSchool. GlobalSchool team admitted that the requests were ignored since they are related to an underutilized module – teacher tracking. After considering that the teacher tracking module was becoming more utilized over the past few months, the GlobalSchool team decided to implement these enhancement requests after our inquiry. The analysis of the data yielded another instance where there was a similar miscalculation on GlobalSchool's part. We therefore offer the following proposition:

> ***Proposition 2b:*** When a client shares knowledge that appears to be industry-wide but related to low demand modules, the vendor tends to disregard the knowledge. This will delay the implementation of a potentially strategic enhancement if the module receives high demand in the future.

Stage 3: Enhancing strategies

SaaS vendors develop strategies enabling them to enhance while under the insufficient domain knowledge condition. These strategies include collaborating with their clients and/or implementing configuration mechanisms that are internally supported. Before proceeding further, we would like to clarify the concept of configuration mechanisms in SaaS. Configuration mechanisms in SaaS provide options for clients while still maintaining a single code base for the software (Nitu 2009). These mechanisms allow flexibilities in user interface, workflow, data, and access control. Some examples include: enabling clients to add user-defined columns to tables, and enabling clients to define and set their own security privileges. SaaS vendors can either support the configuration mechanisms internally (i.e., vendors configure for clients), or provide it as self-service (i.e., clients configure on their own). Vendors prefer and aim for self-service configuration mechanisms to lower costs.

When there is insufficient level of domain knowledge, SaaS vendors will approach enhancements in two different ways. The first approach is employed when vendors face insufficient industry-wide knowledge. When such situation occurs, vendors will enhance by collaborating with select clients. Vendors and clients will work together

much like in traditional outsourcing projects, but with no formal contract or governance. Vendors also tend to implement configuration mechanisms that are internally supported. Our data suggest that vendors make this decision because they do not want "to commit or make public something uncertain," which will likely require additional changes in the future. These two strategies – collaborating and internally-supported configuration mechanisms – are not mutually exclusive. Often, they are jointly utilized for a particular enhancement. GlobalSchool for example, collaborated with one of their clients to implement the subject-based attendance function. In addition, the company will provide support to their clients in configuring the subject-based attendance workflow. The analysis of the data yielded five instances where GlobalSchool collaborated with clients and/or provided internally-supported configuration mechanisms when enhancing with insufficient industry-wide knowledge. Thus, we posit that:

> ***Proposition 3a:*** When the vendor decides to implement an enhancement with insufficient industry-wide knowledge, the vendor will be compelled to either (i) collaborate with select clients, (ii) add internally-supported configuration mechanisms (as opposed to self-service), or (iii) both.

The second approach is employed when vendors face insufficient regional variation knowledge. SaaS vendors accommodate regional variation by implementing configuration mechanisms. Since adding a configuration mechanism implies adding a new functionality to the current software, implementing a configuration mechanism is a software enhancement activity. When vendors face insufficient regional variation knowledge, they tend to implement internally-supported configuration mechanisms. Our data suggest similar reasons behind this strategy as discussed previously. Vendors do not want "to commit" or "make public" changes that will require re-work in the future. For example, GlobalSchool implemented an internally-supported configuration mechanism for report card data because they are uncertain of how schools in different regions would want to format and display the related data. The analysis of the data yielded six instances where GlobalSchool provided internally-supported configuration mechanisms when enhancing with insufficient regional variation knowledge. We therefore offer the following proposition:

> ***Proposition 3b:*** The vendor will likely resolve the issue caused by regional variation difference by adding a configuration mechanism (an enhancement).

When the vendor decides to implement a configuration mechanism with insufficient regional variation knowledge, the vendor will be compelled to add an internally-supported configuration mechanism (as opposed to self-service)

We observe that the process for enhancing while having insufficient domain knowledge maps onto the general SaaS enhancement process. The mapping is as follows: (1) Stage 1 occurs between module release and client feedback, or enhancement release and client feedback; (2) Stage 2 occurs during client feedback, internal decision-making and enhancement engineering activities; and (3) Stage 3 occurs between enhancement engineering and enhancement release. Figure 3 visually indicates where the two processes fit.

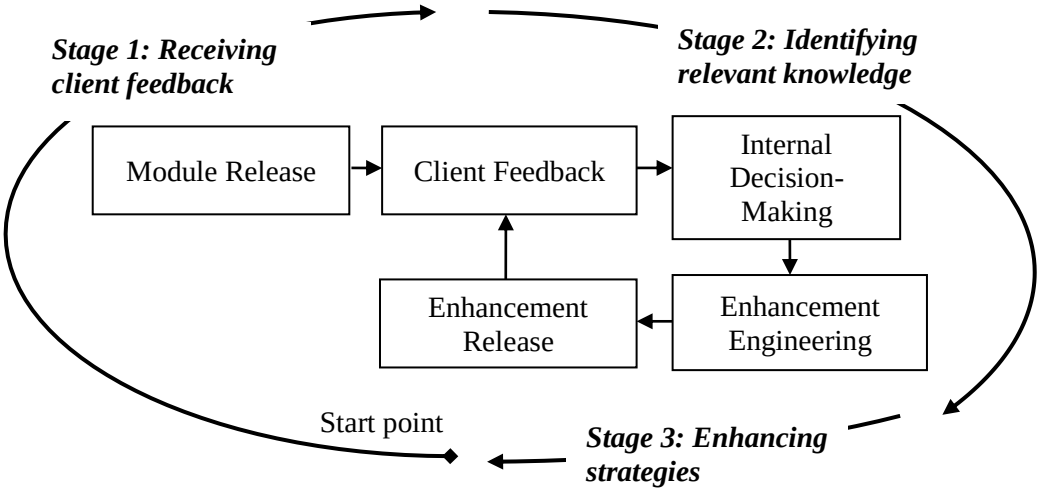

Figure 3. Mapping the Enhancing SaaS with Insufficient Domain Knowledge Process onto the General SaaS Enhancement Process

5.0 Discussion and Conclusion

Our objective in this study is to address the question, "how do SaaS vendors enhance with insufficient domain knowledge?" We studied the enhancement process in GlobalSchool, a SaaS vendor providing administrative software for small schools. Based on our analysis of GlobalSchool's archival data, we put forward two process models. The first process model describes the sequence of activities SaaS vendors engage in to enhance their software (Figure 2), while the second process model describes the stages that SaaS vendors undertake to enhance while having insufficient domain knowledge. We also show how the two models fit together (Figure 3). The fitting enables us to see the connection between the enhancing activities in the first

model and the stages in the second model. We believe that our findings offer important theoretical and managerial contributions. In this section we outline these contributions, asses the limitations, and consider avenues for future research.

First, we build on prior research by examining the process that SaaS vendors undertake when they need to enhance while having insufficient domain knowledge. Prior research has shown that providing software services to organizations is a knowledge-intensive process, in which knowledge is an important requirement (Rivero et al. 2009). However, to the best of our knowledge, this line of questioning has not yet been addressed in current literature. We identified three types of relevant domain knowledge, and determined industry-wide and regional variation knowledge influences SaaS enhancement. We stated our findings in six testable propositions. Propositions 1a and 1b draw attention to how insufficient industry-wide and regional variation knowledge trigger an enhancement process. Propositions 2a and 2b highlight the challenges that SaaS vendors face when attempting to enhance under insufficient domain knowledge. Lastly, Proposition 3a and 3b highlight the enhancing strategies utilized by SaaS vendors. Therefore, our study extends current knowledge by explaining how SaaS vendors are able to enhance despite not have a sufficient level of knowledge.

And second, we discovered several activities that SaaS vendors and clients carry out. As a service business model, SaaS involves the process of co-creation of value. The process of co-creation of value requires both vendors and clients to assume certain responsibilities and activities for each service request (Alter 2010). Our study brings forward knowledge management activities that SaaS vendors undertake during the process of value co-creation. Propositions 2a and 2b show that client feedback trigger knowledge acquisition (i.e., a process in which knowledge flows from an entity's environment and to one that assimilates it within the entity for subsequent use (Holsapple and Joshi 2004)) and knowledge selection (i.e., a process in which knowledge is identified for subsequent use (Holsapple and Joshi 2004)) in SaaS vendors. Meanwhile, Propositions 3a and 3b show that SaaS vendors also engage in knowledge sharing activities (i.e., knowledge flow processes in general (Hosapple and Joshi 2004)). In addition, we discovered that SaaS vendors and clients may engage in collaboration; implying that relationships between SaaS vendors and clients may

intensify. Our findings are consistent with the concept of value co-creation in the service environment as explained by Pralahad and Ramaswamy (2004), in which one of the activities in value co-creation is vendor-client collaboration.

There are several important practical implications for this study. First, we see the importance of having mechanisms to gain client feedback for SaaS vendors and investing in giving feedback for SaaS clients. Second, SaaS vendors need to ensure that they are sensitive to feedback, and create ways to share the feedback among internal team members (i.e., important to have a good knowledge management system). And last, SaaS vendors need to create appropriate strategies to handle insufficient domain knowledge problem. It is not possible for vendors to possess complete domain knowledge for the organizations they are supporting. SaaS vendors might consider a few of GlobalSchool's enhancing strategies such as collaborating with clients and offering vendor-supported configuration mechanisms.

Two of the primary limitations of this study include having a single organization as the source of data, and using a single person to interpret the dataset. As for the first limitation, we admit that having only a single organization to be analyzed makes this study vulnerable to the idiosyncrasies of the said organization. Nevertheless, we would like to highlight that we are generalizing to theory as opposed to population. In addition, being able to analyze such a comprehensive dataset is very rare and valuable (Scacchi 2001); it allowed us to develop the process models in our findings. As for the second limitation, we believe that one of the benefits of a single interpreter is consistency in the analysis (Cramton 2001). We are however, in the stage of getting feedback on our analysis from GlobalSchool (i.e., member review) to increase the trustworthiness of our analysis. Thus far, we have received feedback on propositions 2a and 2b. We will make the necessary revisions to our analysis as we receive more feedback from GlobalSchool.

Overall, our research shows that SaaS enhancement is an incremental and iterative development process. The software evolves through client feedback, and thus, clients play important roles in determining the future direction of the software. SaaS vendors do not necessarily have to have high levels of domain knowledge before releasing their modules (i.e., SaaS vendors do not necessarily have to release "complete"

modules). Instead, SaaS vendors may release modules based on their current levels of domain knowledge, and enhance as needed afterwards. Future researchers are encouraged to test our findings by examining various types of SaaS vendors, and further explore related research avenues.

References

April, A., Hayes, J.H., Abran, A. And Dumke, R. (2005). Software Maintenance Maturity Model (SM^{mm}): The Software Maintenance Process Model. *Journal of Software Maintenance and Evolution: Research and Practice*. 17(3). 197-223.

Alter, S. (2010). Service System Fundamentals: Work System, Value Chain, and Life Cycle. *IBM Systems Journal.* 47(1). 71–85.

Aubert, B.A., Patry, M. and Rivard, S. (2005). A Framework for Information Technology Outsourcing Risk Management. *The DATA BASE for Advances in Information Systems.* 36(4). 9-28.

Bassellier, G. and Benbasat, I. (2004). Business Competence of Information Technology Professionals: Conceptual Development and Influence on IT-Business Partnerships. *MIS Quarterly.* 28(4). 673-694.

Bassellier, G., Benbasat, I. and Reich, B.H. (2003). The Influence of Business Managers' IT Competence on Championing IT. *Information Systems Research.* 14(4). 317-336.

Benlian, A., Hess, T. and Buxmann, P. (2009). Drivers of SaaS-Adoption – An Empirical Study of Different Application Types. *Business & Information Systems Engineering.* 5. 357-369.

Bennett, K. (1996). Software Evolution: Past, Present and Future. *Information and Software Technology*. 38(11). 673-680.

Choudhary, V. (2007). Comparison of Software Quality under Perpetual Licensing and Software as a Service. *Journal of Management Information Systems.* 24(2). 141-165.

Cramton, C.D. (2001). The Mutual Knowledge Problem and Its Consequences for Dispersed Collaboration. *Organization Science.* 12(3). 346-371.

Cusumano, M. (2010). Cloud Computing and SaaS as New Computing Platforms. *Communications of the ACM*. 53(4). 27-29

Dye, M. (2008). SaaS – A Risky Business? *Business Systems News & Analysis for Finance and IT Professionals*. Retrieved on November 15th, 2010 from: http://www.fsn.co.uk/channel_sme_accounting/saas_a_risky_business

Fan, M., Kumar, S. and Whinston, A.B. (2009). Short-Term and Long-Term Competition between Providers of Shrink-Wrap Software and Software as a Service. *European Journal of Operational Research.* 196. 661-671.

Gartner Group. (2006). *SaaS Delivery Challenges On-Premise Software*. Retrieved January 15th, 2010 from: http://www.gartner.com/it/page.jsp?id=496886

Gartner Group. 2006. *SaaS Delivery Challenges On-Premise Software*. Retrieved January 15th, 2010 from: http://www.gartner.com/it/page.jsp?id=496886

Gartner Group. (2010). *Forecast Analysis: Software as a Service, Worldwide, 2009-2014, Update*. Retrieved January 7th, 2011 from: http://www.gartner.com/resId=1468024.

Goles, T. (2001). The Impact of the Client/Vendor Relationship on Outsourcing Success. *Unpublished PhD Dissertation*, University of Houston, Texas, USA.

Holsapple, C.K. and Joshi, K.D. (2004). A Formal Knowledge Management Ontology: Conduct, Activities, Resources, and Influences. *Journal of the American Society for Information Science and Technology*. 55(7). 593-612.

Hudli, A.V., Shivaradhya, B. and Hudli, R.V. (2009). *Level-4 SaaS Applications for Healthcare Industry,* in Proceedings of the 2nd Bangalore Annual Compute Conference (COMPUTE), ACM, Bangalore, India.

Huckman, R.S., Staats, B.R. and Upton, D.M. (2009). Team Familiarity, Role Experience and Performance: Evidence from Indian Software Services. *Management Science.* 55(1). 85-100.

Hurkmans, T. (2009). *Zero Downtime for Multi Tenant SaaS Systems*, in Proceedings of the 2009 ESEC/FSE Workshop on Software Integration and Evolution, ACM, Amsterdam, The Netherlands.

IDC. (2009). *Economic Crisis Response: Worldwide Software as a Service Forecast.* Retrieved February 5th, 2010 from: http://www.idc.com/getdoc.jsp?containerId=prUS21641409

Kang, H., Yang, H. and Rowley, C. (2006). Factors in Team Effectiveness: Cognitive and Demographic Similarities of Software Development Team Members. *Human Relations.* 59(12). 1681-1710.

Kitchenham, B.A., Travassos, G.H., von Mayrhauser, A., Niessink, F., Schneidewind, N.F., Singer, J., Takada, S., Vehvilainen, R. and Yang, H. (1999). Towards an Ontology of Software Maintenance. *Journal of Software Maintenance, Research, and Practice.* 11(6). 365-389.

Kearns, G.S. and Sabherwal, R. (2006). Strategic Alignment Between Business and Information Technology: A Knowledge-Based View of Behaviors, Outcome, and Consequences. *Journal of Management Information Systems.* 23(3). 129-162.

Miles, M.B. and Huberman, A.M. (1994). *An Expanded Sourcebook: Qualitative Data Anaysis*, 2nd Edition, Sage Publications, California, USA.

Niessink, F. And van Vliet, H. (2000). Software Maintenance from a Service Perspective. *Journal of Software Maintenance, Research, and Practice*. 12(2). 103-120.

Nitu (2009). *Configurability in SaaS (Software as a Service) Applications*, in Proceeding of the 2nd Annual Conference on India Software Engineering Conference (ISEC), ACM, Pune, India.

Prahalad, C.K. and Ramaswamy, V. (2004). Co-creating Unique Value with Customers. *Strategy & Leadership*. 32(3). 4 – 9.

Reich, B.H. and Benbasat, I. (2004). Factors that Influence the Social Dimension in Alignment between Business and IT Objectives. *MIS Quarterly.* 24(1). 81-113.

Rivero Y., Sampieri, M., Ferruzca, M., Fernandez, J., Monguet, J., Munoz, J.L., Villalobos, H., and Blanco, B. (2009). A SaaS-based framework to support the management and deploy of web applications for exchanging information and sharing knowledge. In *Proc. of the International Conference on Engineering and Meta-Engineering*.

Rottman, J.W. and Lacity, M.C. (2004). Twenty Practices for Offshore Outsourcing. *MIS Quarterly Executive.* 3(3). 117-130.

Sadegh, B. (2008). *10 Questions to Ask a Potential SaaS Vendor*. Retrieved on December 15th, 2010, from: http://www.ebizq.net/topics/saas/features/10361.html?page=3

Scacchi, W. (2001). Process Models in Software Engineering, in J.J. Marciniak (editor), *Encyclopedia of Software Engineering*, 2nd Edition, John Wiley and Sons, New York, USA.

Symantec Corporation. (2009). *Symantec 2009 Worldwide Green IT Report.* Retrieved April 21, 2010 from: http://www.symantec.com/content/en/us/about/media/GreenIT09_Report.pdf

Tesch, D., Sobol, M.G., Klein, G. and Jiang, J.J. (2009). User and Developer Common Knowledge: Effect on Success of IS Development. *International Journal of Project Management.* 27(7). 657-664.

Tiwana, A. (2009). Governance-Knowledge Fit in Systems Development Projects. *Information Systems Research.* 20(2). 180-197.

Tiwana, A. and Keil, M. (2007). Does Peripheral Knowledge Complement Control? An Empirical Test in Technology Outsourcing Alliances. *Strategic Management Journal.* 28(6). 623 – 634.

Vlaar, P.W.L., van Fenema, P.C. and Tiwari, V. (2008). Cocreating Understanding and Value in Distributed Work: How Members of Onsite and Offshore Information Systems Development Vendor Teams Give, Make, Demand and Break Sense. *MIS Quarterly*. 32(2). 227-255.

Xin, M. and Levina, N. (2008). *Software-as-a Service Model: Elaborating Client-Side Adoption Factors*, in Proceedings of the 29th International Conference on Information Systems (ICIS), AIS, Paris, France.

Zheng, Y., Cao, R., Sun, W., Zhang, K., Jiang, Z., Tsai, W.T., Chung, J.Y. and Younas, M. (2006*). Practical Application of FDC in Software Service Pricing*, in Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE), IEEE, Shanghai, China.