4-15-2014

Software Service Innovation: An Action Research into Release Cycle Management

Neda Barqawi
GSU

Software Service Innovation: An Action Research into Release Cycle Management

By

Neda A. Barqawi

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree

Of

Executive Doctorate in Business

In the Robinson College of Business

Of

Georgia State University

GEORGIA STATE UNIVERSITY

ROBINSON COLLEGE OF BUSINESS

2014

Neda A. Barqawi

J. Mack Robinson College of Business
Georgia State University
Robinson College of Business
35 Broad Street NW
Atlanta, GA 30303

Dr. Lars Mathiassen

J. Mack Robinson College of Business
Georgia State University
Robinson College of Business
35 Broad Street NW
Atlanta, GA 30303

ACCEPTANCE

This dissertation was prepared under the direction of the *Neda A. Barqawi* Dissertation Committee. It has been approved and accepted by all members of that committee, and it has been accepted in partial fulfillment of the requirements for the degree of Doctoral of Philosophy in Business Administration in the J. Mack Robinson College of Business of Georgia State University.

H. Fenwick Huss

Dean, J. Mack Robinson College of Business

DISSERTATION COMMITTEE:

Dr. Lars Mathiassen (Chair)

Dr. Wesley Johnston

Dr. Balasubramaniam Ramesh

TABLE OF CONTENTS

List of Tables viii

List of Figures ix

Abbreviations and Definitions x

ABSTRACT xi

1.0 Introduction 1

2.0 Area of Concern 4

2.1 Software as a Service 4

2.2 Release Cycle Management 6

3.0 Theoretical Framework 8

3.1 Service Science 8

3.2 Service-dominant Logic 10

4.0 Research Methodology 16

5.0 Problem-Solving Cycle 17

5.1 Initiation phase 18

5.2 Diagnostic phase 18

5.3 Establishment phase 21

5.4 Acting phase 27

5.5 Learning Phase 30

6.0 Research Cycle 33

6.1 Data Collection 34

6.2 Data Analysis 35

7.0 Improving SaaS Releases at Software Inc. 39

7.1 Value Proposition 39

7.2 Service Dominance 41
7.3 Value Co-Creation 46
7.4 Service Delivery Process 50
8.0 Discussion 57
8.1 Software Service Innovation at Software Inc. 57
8.2 S-D Logic Perspective on SaaS 62
8.3 Grounded SaaS Delivery Model 66
9.0 Conclusion 72
Appendix A: Shared Platform Document 77
A1.0 Problem Setting 78
A1.1 Recently Acquired Software 78
A1.2 Complexity of Service Delivery 79
A1.3 New Engineering and Product Management Team 79
A1.4 Low Software Engineering Process Maturity 80
A1.5 Actors 80
A2.0 Release Cycle Management 82
A3.0 Research Methodology 87
A3.1 Engaged Scholarship 87
A3.2 Action Research Design 89
A4.0 Problem-Solving Cycle 92
A4.1 Initiation Phase 93
A4.2 Diagnostic Phase 94
A4.3 Establishment Phase 100
A4.4 Acting Phase 102

A4.5 Learning Phase 103

A5.0 Research Cycle 106

A5.1 Data Collection 107

A5.2 Data Analysis 108

A6.0 Principles of Canonical Action Research 110

Appendix A References 118

Appendix B: Secure-on-Request Improvement Projects – Status Report 122

Appendix C: Secure-on-Request Processes Assessment and Improvement Options 124

Appendix D: Secure-on-Request New Release Cycle Model 131

Appendix E: Customer Advisory Board - Meeting Items 132

References 134

LIST OF TABLES

Table 1.0 - 1 Research Design – Style Composition 3

Table 3.2 - 1 S-D Logic Concepts and Their Transition.. 11

Table 3.2 - 2 Core Foundational Premises of S-D Logic... 12

Table 5.0 - 1 Problem Solving Time Line at Software, Inc. 18

Table 5.2 - 1 Service Delivery Practice-Based Assessment - Diagnostic Phase 20

Table 5.3 - 1 Project Schedule 21

Table 5.3 - 2 Improvements in Customer Relationship Project.. 22

Table 5.3 - 3 Conceptual Definition of Six SaaS-Qual Factors... 24

Table 5.3 - 4 Improvements in Requirements and Quality Project.. 25

Table 5.3 - 5 Improvements in Release Cycle Project.. 26

Table 5.4 - 1 Improvements in Customer Relationship - Acting Phase...................................... 28

Table 5.4 - 2 Improvements in Requirements and Quality - Acting Phase 29

Table 5.4 - 3 Improvements in Release Cycle - Acting Phase.. 29

Table 5.5 - 1 Service Delivery Practice-Based Assessments – Learning Phase........................ 31

Table 6.1 - 1 Primary and Secondary Data Sources 35

Table 8.3 - 1 Grounded Process Model for Value Co-Creation in SaaS 69

Table 8.3 - 2 Service Innovation Principles in SaaS Environments 71

LIST OF FIGURES

Figure 5.0 - 1 IDEAL Model 17

Figure 5.2 - 1 *Secure-on-Request* Service Blueprint at Software, Inc. 21

Figure 5.5 - 1 *Secure-on-Request* Reported Issues - Learning Phase 32

Figure 6.2 - 1 Data Analysis Activities 36

Figure 8.3 - 1 Grounded Process Model for Value Co-Creation in SaaS 68

ABBREVIATIONS AND DEFINITIONS

AM	Account Manager
CAR	Canonical Action Research
CPR	Collaborative Practice Research
G-D	Logic- Goods-Dominant Logic
OPS	Operation Services
PM	Product Manager
SaaS	Software-as-a-Service
S-D	Service-Dominant
TAM	Technical Account Manager
QA	Quality Assurance

ABSTRACT

SOFTWARE SERVICE INNOVATION:

AN ACTION RESEARCH INTO RELEASE CYCLE MANAGEMENT

BY

NEDA A. BARQAWI

May 8th, 2014

Committee Chair: *Dr. Lars Mathiassen*

Fierce competition in the market is driving software vendors to rely on Software-as-a-Service (SaaS) strategies and to continuously match new software versions with customers' needs and competitors' moves. Although release management as a recurrent activity related to SaaS arguably shapes how a vendor services its customers, the literature is surprisingly limited on how software releases are managed to support SaaS strategies. Against this backdrop, we present a collaborative action-research study with Software Inc., a large multi-national software provider, focused on improving the release cycle management process for a complex security software service. The study is part of a comprehensive intervention into Software Inc. that combines a perspective rooted in software process improvement and engineering practices with one rooted in service delivery and customer interactions. The part that is reported in this dissertation draws on the service-dominant logic framework to analyze how the release cycle management process was organized to improve Software Inc.'s ongoing value co-creation with its customers. As a result, the study contributed to improving release cycle management at Software Inc. and it expands industry knowledge about the challenges and opportunities for software vendors to manage releases and improve the value delivered to and co-created with their customers. This added knowledge is of interest to both practitioners and researchers as SaaS strategies increasingly shape the industry with important implications for how software is released.

1.0 INTRODUCTION

Software-as-a-Service (SaaS) is a software application delivery model that is rapidly growing in popularity. SaaS solutions are usually web-based and accessible via Internet browsers (M. Cusumano, 2010). Enhanced customer relationships are expected to result from the hybrid software and service features of the SaaS model (Berkovich, Esch, Leimeister, & Krcmar, 2010). Direct customer contact is expected to change the manner in which software vendors manage development, operations, and quality control (Stuckenberg & Heinzl, 2010). Market competition is driving corporations to pinpoint the timing of product introduction and to fulfill customer requirements in an increasingly expeditious manner (Krishnan, 1994; Pratim Ghosh & Chandy Varghese, 2004). A well-defined release-management process could raise the quality of building, testing, and deployment activities, thereby reducing problems occurring after product or service delivery (Lahtela & Jantti, 2011). Although release management as a recurrent activity related to SaaS arguably shapes how a vendor services its customers, the literature is surprisingly limited on how software releases are managed to support SaaS strategies.

Against this backdrop, we conducted a collaborative action research study with Software Inc. regarding the delivery of one of their SaaS solutions, *Secure-on-Request*. Specifically, we used collaborative practice research (CPR), an action research methodology that applies methodological pluralism as well as collaboration between researchers and practitioners (Mathiassen, 2002). The study adopted two complementary perspectives, one rooted in software process improvement and engineering practices and one rooted in service delivery and customer interactions (this overall research design is described in detailed in the shared platform document, Appendix A). Drawing on these complementary perspectives, the study focused on release cycle management to support Software Inc. in their *Secure-on-Request* repositioning effort and contributed to the body of knowledge simultaneously (Avison, Baskerville, & Myers, 2001; Baskerville & Wood-Harper, 1996). To ensure the rigor of the overall study, we followed the principles of canonical action research (CAR) (Davison, Martinsons, & Kock, 2004) as we enacted the dual cycles outlined by McKay and Marshall (2001). In the problem-solving cycle, we collaborated with Software Inc. to support their *Secure-on-Request* service-delivery processes. We proceeded in a stepwise, iterative fashion, based on the approach described in the IDEAL model (McFeeley, 1996). The model is an approach for innovating software practices and was developed in 1996 by the Carnegie Mellon University Software Engineering Institute. Our research cycle was guided by the style composition for action research developed by Mathiassen, et al. (2012).

As theoretical lens for the specific part of the study reported in this dissertation, we drew on service-dominant (S-D) logic, proposed by Stephen Vargo and Robert Lusch (Vargo & Lusch, 2004, 2008) to address the following research question: How can release cycle management be organized to improve Software Inc.'s ongoing value co-creation with its customers? This framing is based on an alternative logic for understanding markets and marketing, which views service, rather than goods, as the focus of economic and social exchange (i.e., service is exchanged for service) (Vargo & Lusch, 2004). Accordingly, this dissertation explored the software-release management and service-delivery processes at Software Inc. through the theoretical lens of S-D logic with a focus on the co-creation of value of the SaaS delivery model. We approached the issue from the point of view of the customer and determined how the release-management process can be organized to improve Software Inc.'s ongoing value co-creation with its customers.

We used our analysis to develop recommendations related to value creation through the service delivery of the SaaS application and to release-management process improvement at Software Inc. We propose that the insights gained from this study will both broaden our theoretical understanding of this issue and assist those in the SaaS service field. Overall then, the dissertation relied on the style composition for action research (Mathiassen, Chiasson, & Germonprez, 2012) summarized in Table 1.0 - 1. The different elements of this design will be motivated, described and further elaborated in the subsequent section of the dissertation.

Table 1.0 - 1 Research Design – Style Composition

Component	***Description***
P - Problem Setting	Improve Software Inc.'s ability to effectively service their customers and respond to their needs
A - Area of Concern	SaaS, release management, and Service Science
RQ - Research Question	How can release cycle management be organized to improve Software Inc.'s ongoing value co-creation with its customers?
F - Conceptual Framework	S-D Logic - proposed by Stephen Vargo and Robert Lusch in 2004
M - Research Method	Qualitative, action research study
CA - Contribution to Area of Concern	Empirical and theoretical contribution to SaaS, release management, and service science knowledge

2.0 AREA OF CONCERN

This dissertation focuses on SaaS, an important contemporary form of software delivery, in particular on the challenges related to recurrently releasing such services to existing customers and the market. In the following, we will review the literature on each of these two areas of contemporary software practice.

2.1 Software as a Service

SaaS refers to software applications delivered as a service over the Internet (Armbrust et al., 2010; M. Cusumano, 2010). It is one of the leading models in the service-oriented software business today and it is being increasingly adopted (M. A. Cusumano, 2008; Liu, Guo, Zhao, & Chou, 2010; Susarla, Barua, & Whinston, 2009). SaaS has been described as a delivery, business, pricing, revenue, or licensing model (Choudhary, 2007a; M. A. Cusumano, 2008; Lassila, 2006; Srikanth & Cohen, 2011; Sun, Zhang, Chen, Zhang, & Liang, 2007). Revenues for the SaaS delivery model are expected to grow by 19.4 percent overall from 2008 to 2013 (Mertz et al., 2009). In the SaaS model, the service provider hosts and manages the SaaS applications, while the “tenants” who want to use them rent the services instead of buying software licenses (Guo, Sun, Huang, Wang, & Gao, 2007). The term “cloud computing” refers to both the applications delivered as services over the Internet as well as the hardware and software systems that reside in the data sites hosted by the providers. The services themselves are referred to as SaaS (Armbrust et al., 2010) .

The SaaS model permits simultaneous utilization of the same application installation by a large number of independent users, and allows for a swift introduction of new and innovative software (Sääksjärvi, Lassila, & Nordström, 2005; Singh, Bhagat, & Kumar, 2012). SaaS also offers customers an attractive payment structure. The pricing model is based on the continuous service relationship between customers and vendors together with time- or usage-dependent metrics (Sääksjärvi et al., 2005; Singh et al., 2012; Srikanth & Cohen, 2011). The model provides customers with reductions in information technology (IT) infrastructure cost, operational flexibility, and immediate access to the latest features and innovations (Armbrust et al., 2010; Guo et al., 2007; Herrick, 2009; Singh et al., 2012).

SaaS benefits software providers as well as customers. Software providers benefit from the cost reductions gained from scalability and customization, all the while growing their customer base. Since SaaS solutions support many customers with a single-application code base, deployment time is reduced and updating of application features is centralized and simplified (Guo et al., 2007). Some authors have suggested that the SaaS model may improve the user's perception of quality and their user experience in general (Choudhary, 2007b). A number of studies have demonstrated benefits of the software-service delivery model such as cost savings, increased productivity, and improved operational efficiency (Herrick, 2009; Hudli, Shivaradhya, & Hudli, 2009).

Companies that provide SaaS solutions face the challenge of delivering and maintaining high-quality software applications that work in many different contexts. Customers can easily unsubscribe from services, so frequent updates to the software and increased investments in development are critical to retaining a competitive edge (Choudhary, 2007b; Singh et al., 2012; Srikanth & Cohen, 2011). Service quality is fundamental to the continued success of the SaaS model (Benlian, Koufaris, & Hess, 2011). The SaaS model is expected to change software vendors' management of development, operations, and quality control (Stuckenberg & Heinzl, 2010).

SaaS vendors are obliged to address the entire gamut of service-quality management processes (Benlian et al., 2011). Managers can best allocate resources for service improvements by having a measure of customer evaluation of SaaS services (Benlian et al., 2011). Although release management could impact how a software vendor support its customers (Lahtela & Jantti, 2011), research is limited on how software releases are managed to support SaaS practices. Hence, in our exploration of the release-management process of the SaaS application *Secure-on-Request* at Software Inc., we examined how customers contributed to value co-creation throughout the software release management and delivery process of the *Secure-on-Request* software.

2.2 Release Cycle Management

Software Release Management refers to the typical recurring identification, packaging, and distribution of the elements of a product (e.g., executable programs, documentation, release notes, and configuration data) (Ballintijn, 2005; Scott & Nisse, 2001). It is defined as "the process through which software is made available to and obtained by the user" (Van Der Hoek, Hall, Heimbigner, & Wolf, 1997). Quality control and the success of release management are dependent upon having the right processes in place. Well-organized release-management processes have been found to play a critical role in the success of large projects (Danesh, Saybani, & Danesh, 2011). Van der Hoek (1997) wrote that release management is "a poorly understood and underdeveloped part of the software process" and identified several obstacles to its execution. Although research on software release management is limited, both in general and as it relates to SaaS, the subject has generated both academic and practical interest. We have only identified a limited number of studies on the subject as documented in the comprehensive review in the Shared Platform Document, Appendix A. Literature is also limited on the release cycle concept which could describe how all the components in software development interconnect (Syed, 2014). A comprehensive approach is necessary to connect software development and delivery processes and the relevant functions involved in the process (Syed, 2014). In response, this action research dissertation investigates software release cycle management as an interesting starting point for improving the service quality of the SaaS application delivered by Software Inc. Against this backdrop, we contribute to the software organization and release management literature specifically in a SaaS environment, and we anticipate that the empirical insights from our problem diagnosis, interventions, and learning from Software Inc. will be helpful to both practitioners and academic researchers.

3.0 THEORETICAL FRAMEWORK

As a theoretical starting point, we will review service science background. The theoretical foundation for this dissertation is adopting S-D logic. Through its foundational premises and concepts, we studied the process of value co-creation between Software Inc. and its customers as the SaaS application service was delivered.

3.1 Service Science

The world economy is moving from being goods-based to one that is dependent on services (Bardhan, Demirkan, Kannan, Kauffman, & Sougstad, 2010; Maglio & Spohrer, 2008; Spohrer & Maglio, 2008). Services are taking on an increasing importance, and approximately 80% of all employees in western economies now work in the service sector (Kohlborn, Korthaus, Riedl, & Krcmar, 2009). Although, the service sector has matured over the last 50 years in most advanced economies, the scientific understanding of services is still in its infancy (Chesbrough & Spohrer, 2006).

Service can be defined as “acts performed for others, including the provision of resources that others will use” (Spohrer, Anderson, Pass, & Ager 2008, p. 4). In marketing and economics, service is understood as the non-material equivalent of a good. Service also has been defined as an economic activity that does not lead to ownership, and this is what distinguishes it from providing physical goods (Spohrer & Maglio, 2008). Service can be seen as a process that produces benefits by enabling either a change in customers’ physical possessions, or a change in their intangible assets (Spohrer, Maglio, Bailey, & Gruhl, 2007).

Service science is an emerging multidisciplinary field concerned with the study of service systems and value co-creation. It is an industry-led, university-supported discipline to study exchange among “service systems.” (Maglio & Spohrer, 2008; Vargo & Lusch, 2008). The field

“Combines organization and human understanding with business and technological understanding to categorize and explain the many types of service systems that exist as well as how service systems interact and evolve to co-create value” (Maglio & Spohrer 2008, p. 18). Service systems are defined as “value co-creation configurations of people, technology, value propositions connecting internal and external service systems, and shared information” (Maglio & Spohrer 2008, p. 18). Value co-creation can be defined as: “An interactive process, involving at least two willing resource integrating actors, which are engaged in specific form(s) of mutually beneficial collaboration, resulting in value creation for those actors” (Frow, Payne, & Storbacka, 2011). The actors (i.e., customers and SaaS providers) create value by cooperating and merging their resources, competences, and capabilities (Bovet & Martha, 2000; Kähkönen & Lintukangas, 2012).

Services differ from goods in that the former are intangible, inseparable, heterogeneous, and perishable (Regan, 1963; Tracy, 2012). Inseparability refers to the fact that service acts are simultaneously delivered and consumed by the customer, and consequently the customer has an active role in influencing the quality of the service (Tracy, 2012; Wolak, Kalafatis, & Harris, 1998). Goods, are produced and then sold, but services are sold and then produced and consumed (Tracy, 2012; Zeithaml, Parasuraman, & Berry, 1985) Service science refers to inseparable characteristics of service as the process of value co-creation (Spohrer et al., 2008). The idea of customers having an input in product delivery, and value, or ‘co-creation’ and ‘interactive marketing’ has been emphasized in the service-market literature (Grönroos, 1982; Gummesson, 1987; Peters, Johnston, & Pressey, 2012; Shostack, 1977). Central to service science is the role of the customer as a co-producer, where the service is adapted by customers based on their specific needs or environments (Vargo & Lusch, 2004, 2008). The understanding of service as applying resources for the benefit of others or oneself is applicable to business organizations, and is particularly consistent with service concepts from IT, such as service-oriented architecture, SaaS, and, more broadly, services computing (Lusch & Nambisan, 2012; Zhao, Tanniru, & Zhang, 2007). S-D logic has been proposed as a theoretical and philosophical foundation for the development of service science and the study of service systems (Maglio & Spohrer, 2008; Vargo & Akaka, 2009; Vargo & Lusch, 2008).

3.2 Service-dominant Logic

In 2004, Vargo and Lusch introduced an S-D logic framework for understanding the theory and practice of marketing. This perspective was presented as a more effective alternative to goods-dominant (G-D) logic—which is based on traditional economic theories—for the study of service systems (Barile & Polese, 2010; Vargo & Lusch, 2004). Vargo and Lusch have attempted to produce a general marketing theory by synthesizing the different schools of thought in the marketing literature (Vargo & Lusch, 2004, 2008; Winklhofer, Palmer, & Brodie, 2007). Service in S-D logic is defined as applying specialized competences, including knowledge and skills, through deeds, processes, and performances for the benefit of another actor or the actor itself (Lusch & Nambisan, 2012; Vargo & Lusch, 2004, 2008).

S-D logic is still evolving. Eight foundational premises were initially set out (Vargo & Lusch, 2004) and a more comprehensive conceptualization of ten foundational premises (FPs) were later introduced (Vargo & Lusch, 2008). S-D logic premises are not a set of guidelines or rules; rather, they represent a developing effort to construct a better "marketing-grounded" understanding of value and exchange (Lusch & Vargo, 2006a; A. F. Payne, Storbacka, & Frow, 2008; Vargo & Lusch, 2004). The key concepts and constructs comprising S-D logic and the transition of these constructs from G-D to S-D logic as demonstrated by the authors are listed in

Table 3.2 - (Lusch & Vargo, 2006b; Winklhofer et al., 2007). In the latest development of S-D logic, the authors identified four FPs as the fundamental axioms of S-D logic. These are illustrated in Table 3.2 - (Vargo, 2013).

Table 3.2 - 1 S-D Logic Concepts and Their Transition

Goods-dominant logic concepts	***Transitional concepts***	***Service-dominant logic concepts***
Goods	Services	Service
Products	Offerings	Experiences
Feature-attribute	Benefit	Solution
Value-added	Co-production	Co-creation of value
Profit maximization	Financial engineering	Financial feedback/learning
Price	Value delivery	Value proposition
Equilibrium systems	Dynamic systems	Complex adaptive systems
Supply chain	Value-chain	Value-creation network/constellation
Promotion	Integrated marketing communications	Dialogue
To market	Market to	Market with
Product orientation	Market orientation	Service orientation

Vargo and Lusch suggest that firms should focus on processes that are co-created with customers (Lusch & Vargo, 2006a). These co-creation processes should reflect the four fundamental building blocks forming the firm's marketing strategy: (1) service offerings; (2) value propositions; (3) conversation and dialogue; and (4) value processes and networks (Lusch & Vargo, 2006a). According to the authors, the role of the "producer" has been to create and deliver goods and services, and the role of "customer" has been to consume those goods and use those services. G-D logic understands these two roles are independent of one another, with goods being the unit of exchange. S-D logic assigns service as the foundation for exchange, value

Table 3.2 - 2 Core Foundational Premises of S-D Logic

	Premise	***Explanation***	***Application to SaaS environment***
FP1	Service is the fundamental basis of exchange.	The application of operant resources (knowledge and skills) "service," is the basis for all exchange. Service is exchanged for service.	Customers and software providers exchange skills and knowledge in creating and using the SaaS applications or solutions.
FP6	The customer is always a *co-creator of value.*	Implies value creation is interactional.	It is important for SaaS providers to understand their customers' processes and their specific requirements while developing and delivering their SaaS applications.
FP9	All economic and social actors are *resource integrators.*	Implies the context of value creation is networks of networks (resource-integrators).	Social and economic actors integrate various types of resources to create value. Software Inc.'s customers (actors) obtain *Secure-on-Request* service because they consider it part of a larger solution they need in order to integrate with other resources.
FP10	*Value* is always uniquely and phenomenologically *determined by the beneficiary*.	Value is idiosyncratic, experimental, contextual, and meaning-laden.	In the context of SaaS applications, the same service delivered to certain customers will provide different value to other customers, dependent upon their industry and their need for using that application.

creation, and marketing (Vargo & Lusch, 2004, 2008). From the perspective of S-D logic, customers and providers co-create value, whereas according to G-D logic customers only consume and buy products and services. Value, in the S-D logic approach, is co-created when customers and producers engage in a collaboration during the creation and the consuming of products and services (Vargo & Lusch, 2008).

There exist very few studies on how software releases are managed in order to support SaaS strategies and service delivery. In this study, we explored how the release-management process at Software Inc. facilitated the value co-creation between consumers and service providers in a SaaS environment. In doing so, we used S-D logic as a theoretical framework and applied the S-D logic four core foundational premises (FPs) and key constructs to the SaaS environment. Table 3.2 - shows our application of these core foundational premises to the SaaS model, and the service delivery of *Secure-on-Request*.

The first foundational premise (FP1) of S-D logic holds that service is the fundamental basis of exchange, and application of skills and knowledge is a service (Vargo, 2013; Vargo & Lusch, 2004, 2008). IT service firms traditionally provide hardware and software for organizations (Brocke et al., 2009). In S-D logic, skills and knowledge that help customers with their objectives are the units of exchange, not the hardware and software provided. In this approach, IT service providers would focus on skills and knowledge, and would use hardware and software as a means of delivering these services (Brocke et al., 2009; Vargo & Lusch, 2008). This is particularly applicable in the SaaS context, as customers and software providers exchange skills and knowledge while creating and using the SaaS applications or solutions to achieve their customers' goals.

The sixth foundational premise (FP6) states that a basic principle for successful co-creation of value for a company is to actively involve customers the process of value creation and the service delivery process (Vargo, 2013; Vargo & Lusch, 2004, 2008). The customer becomes a co-producer of value when shifting from the perspective of creating value through exchange of goods to the perspective of creating value by applying certain skills and knowledge through a service provided (Brocke et al., 2009; Vargo & Lusch, 2004). In order to provide services that can be applied within the customer's environment, SaaS providers have to understand their customers' processes and their specific requirements in developing and delivering their SaaS applications. Therefore, SaaS customers are contributing to the creation of value of the SaaS applications they require (Vargo & Lusch, 2008).

The ninth foundational premise (FP9), refers to the S-D notion that all social and economic actors integrate various types of resources to create value (Lusch & Nambisan, 2012; Vargo & Lusch, 2008). For example Software Inc.'s customers (i.e., the actors) obtain *Secure-on-Request* service because they consider it part of a larger solution they need to integrate with other resources (Lusch & Nambisan, 2012). Also, all firms are simultaneously "service offerers" (i.e., offer resources or services to other actors) and "service beneficiaries" (i.e., they themselves are beneficiaries of other firms that supply them with service or resources) (Lusch & Nambisan, 2012). This implies that SaaS solution providers have to consider the different roles of actors (e.g., customers and suppliers) in resource integration and service innovation. It also implies that SaaS providers need to understand the process of value co-creation and adapt their internal business processes to support it (Lusch & Nambisan, 2012).

Lastly, (FP10) proposes that value co-creation is contingent upon the customer's experience. Perceived value is highly context-specific. A service delivered to one customer will provide different value when delivered to another customer (Lusch & Vargo, 2006a; Vargo & Lusch, 2004, 2008). For example, in the context of the *Secure-on-Request* application, a large firm in the financial sector might utilize the service delivered by *Secure-on-Request* differently than would a firm in the pharmaceutical or retail sector.

S-D logic provides the basis on which to create a service-oriented enterprise that leverages IT for service by applying the skills of the enterprise to the requirements of customer (i.e., being service-centric and rather than company-centric) (Khoshafian, 2006; Lusch & Nambisan, 2012; Vargo & Lusch, 2008). The shift to S-D logic is particularly important for SaaS solutions providers. When firms focus on service and how it is delivered to the customer, the attention shifts from the hardware and software as products to the service-delivery responsibility expected from the firm (Brocke et al., 2009; Lusch & Nambisan, 2012). Hence, S-D logic is a highly suitable framework within which to study service delivery of SaaS application and release-management processes at Software Inc.

4.0 RESEARCH METHODOLOGY

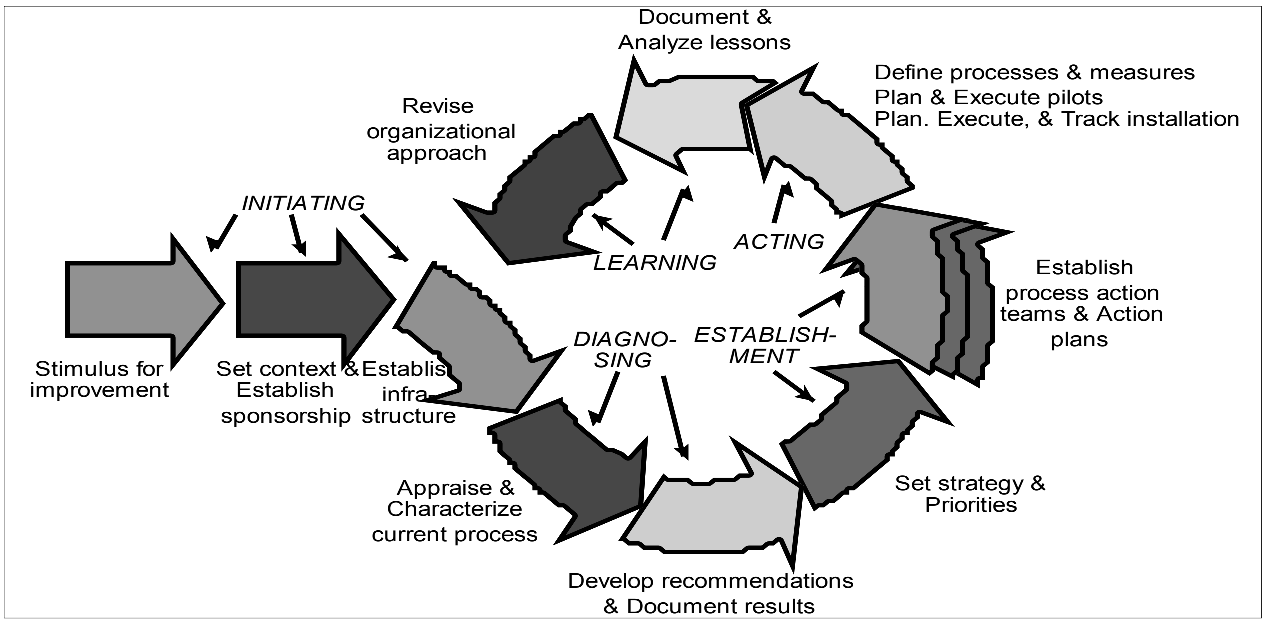
Our research was carried out as an action research study to support the SaaS solution *Secure-on-Request* repositioning effort at Software Inc. (Avison et al., 2001; Baskerville & Wood-Harper, 1996). Our general research approach was collaborative practice research (CPR), a type of action research in which methodological pluralism and collaboration between researchers and practitioners is emphasized (Mathiassen, 2002). CPR methodology works toward understanding practice through interpretation, and improving practice by making interventions (Mathiassen, 2002).

Action research was introduced by Kurt Lewin in 1951, and it uses intervention to challenging social situations as a means to develop scientific knowledge (Lewin, 1951; Rapoport, 1970). Rapoport writes that “Action research aims to contribute both to the practical concerns of people in an immediate problematic situation and to the goals of social science by joint collaboration within a mutually acceptable ethical framework” (1970, p. 499). Several action research approaches have been developed by other scholars (Davison et al., 2004; Susman & Evered, 1978). Susman and Evered described the development of a client-system infrastructure and a multi-phased cyclical process for action research consisting of diagnosing, action-planning, action-taking, evaluating, and specifying learning (Susman & Evered, 1978). To ensure the rigor of this action research, we followed the five principles and associated criteria for Canonical Action Research (CAR) suggested by Davison et al. (2004) as we enacted the dual cycles outlined by McKay and Marshall (2001). The Shared Platform Document (Appendix A) provides details on the overall research approach for this study.

5.0 PROBLEM-SOLVING CYCLE

As we engaged in the problem-solving cycle at Software Inc., we adopted the IDEAL model (McFeeley, 1996) to guide our activities. The model is an approach for innovating software practices and was developed in 1996 by the Carnegie Mellon University Software Engineering Institute (McFeeley, 1996). It is illustrated in Figure 5.0 - 1.

Figure 5.0 - 1 IDEAL Model


Document & Analyze lessons
Define processes & measures
Plan & Execute pilots
Plan. Execute, & Track installation
Revise organizational approach
INITIATING
LEARNING
ACTING
Establish process action teams & Action plans
DIAGNO-SING
ESTABLISH-MENT
Stimulus for improvement
Set context & Establish sponsorship
Establish infra structure
Appraise & Characterize current process
Set strategy & Priorities
Develop recommendations & Document results

The IDEAL model (Initiating, Diagnosing, Establishing, Acting, and Learning), is very similar to the five-phase cyclical approach (diagnosing, action planning, action taking, evaluating, and specifying learning) developed by Susman and Evered (1978). Following the phases of the IDEAL process directed our actions in the problem-solving cycle as well as provided opportunities to make research contributions as we studied the change processes over time Table 5.0 - 1 Problem Solving Time Line at Software, Inc. The Shared Platform Document (Appendix A) contains an overview and more details on the IDEAL model and the problem-solving cycle of this research.

Table 5.0 - 1 Problem Solving Time Line at Software, Inc.

Phase	*Description*
Initiation phase (January 5, 2013 - April 9, 2013)	Obtained commitment, set goals and established an improvement infrastructure.
Diagnostic phase (April 9, 2013 - June 28, 2013)	Assessed current practices; developed and prioritized recommendations for improvements.
Establishment phase (June 28, 2013 - July 2, 2013)	Created specific, focused improvement initiatives. Teams were established to deal with each of the recommended improvement areas from the diagnostic phases.
Acting phase (July 2, 2013 - October 26, 2013)	Developed and implemented solutions for each improvement area.
Learning phase (October 26, 2013 - February 28, 2014)	Evaluated results of the initiatives.

5.1 Initiation phase

In the initiation phase, we obtained commitment and set goals with Software Inc. Consequently, we established an improvement infrastructure and obtained approval for a commitment for resources to accomplish planned tasks. Key dates, and more details on the initiation phase are included in the Shared Platform Document (Appendix A).

5.2 Diagnostic phase

In the diagnostic phase, we established the groundwork for the later phases in the process. Our diagnostic work included perception-based as well as practice-based methods (Napier, Mathiassen, & Johnson, 2009). We also analyzed performance data from Software Inc.'s main tracking systems. Key dates for the diagnostic phase and more details are included in the Shared Platform Document (Appendix A).

One of the goals of the diagnostic phase was to understand the current practices and challenges related to service delivery of *Secure-on-Request* within Software Inc. We assessed existing service-delivery practices related to *Secure-on-Request* from the viewpoint of key stakeholders (Napier et al., 2009). For our practice-based assessment (Napier et al., 2009), we selected service-delivery principles identified in the service-science literature (Karpen, Bove, & Lukas, 2012; Schneider & Bowen, 2010; Vargo & Lusch, 2004). We compared these principles to current service-delivery practices at Software Inc., and provided our assessment. Based on data collected and observations, the research team assigned scores to Software Inc.'s service delivery practices as they compare to the identified principles. Service practice assessment and scores assigned are illustrated in Table 5.2 - 1.

In the perception-based part of the assessment we identified individuals from Software Inc. who are involved in the release process and service delivery of *Secure-on-Request* as well as internal and external customers (Napier et al., 2009). Participants' viewpoints were analyzed with a focus on strengths and weaknesses of service delivery practices of *Secure-on-Request*. An overview of the identified areas for improvement is included in the Shared Platform Document (Appendix A). The five areas identified for improvement are: specifying and stabilizing requirements, prioritizing requirements across channels, managing release cycles, maintaining complete service information, and communicating releases across customers. These areas are interrelated and affect the service delivery of *Secure-on-Request* in many ways.

To help with identifying the gaps and areas for improvement for the service-delivery process of *Secure-on-Request*, we used a practical technique called Service *Blueprinting* (Bitner, Ostrom, & Morgan, 2008). Given the intangible and complex nature of services, blueprinting helps create a visual depiction of the service process, the points of customer contact, and the

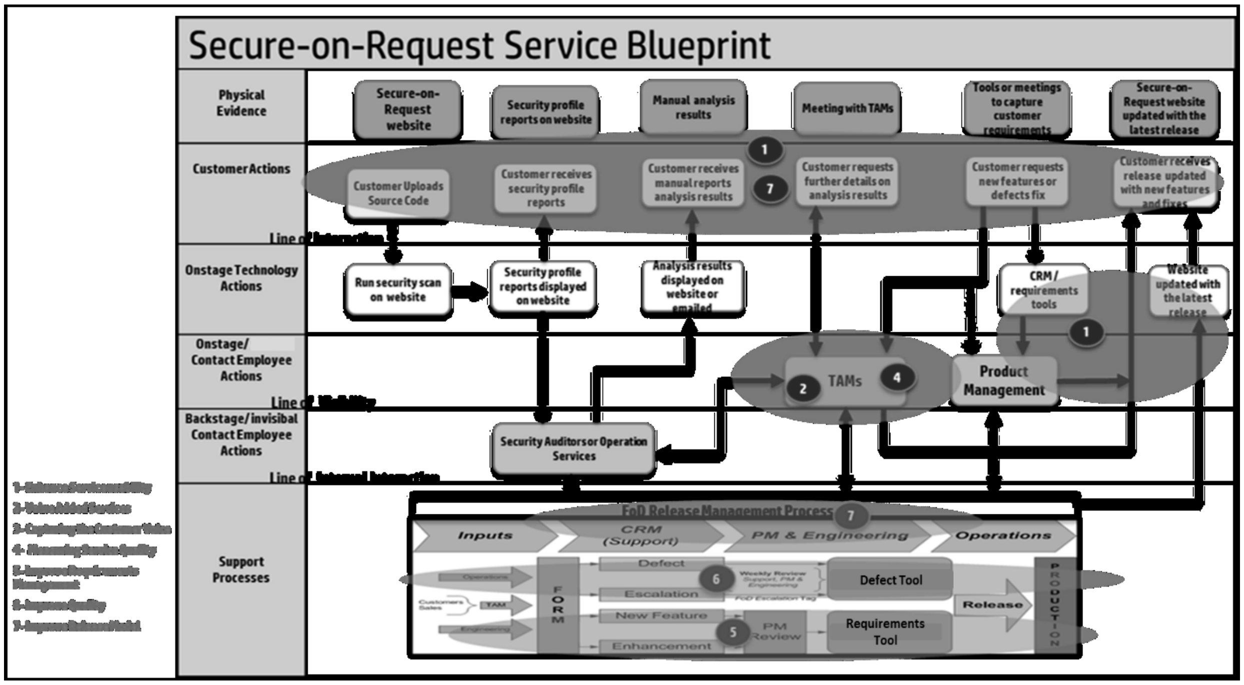
Table 5.2 - 1 Service Delivery Practice-Based Assessment - Diagnostic Phase

	Principle	*Score*
1	Support fair and non-opportunistic customer-service provisioning.	High
2	Ensure connections and relationships with customers during service provisioning.	High
3	Ensure alignment between *Secure-on-Request* directions and the strategic focus of Software Inc.	Medium
4	Establish process to capture customer needs and have them influence the service.	Medium
5	Understand customers' service contexts, processes, and expected outcomes.	Medium
6	Share information on customer perceptions of service value across *Secure-on-Request* teams.	Medium
7	Coordinate and integrate the service to allow customization to individual customers.	Low
8	Ensure clear communications of release features to provide new value to all customers.	Low
9	Maintain complete service information to assist customers' knowledge and competence.	Low
10	Measure the gap between customer expectations and perceptions of the service.	Low

evidence of service from the customer's point of view (Bitner et al., 2008). Using service blueprinting (Bitner et al., 2008) for *Secure-on-Request*, we displayed possible areas for improvement and assigned the recommended project for improvement as it is illustrated in Figure 5.2 - 1.

The steering committee was kept informed of the activities through weekly status reports and status meetings. The assessment findings and improvement options and recommendations were shared with the steering committee meeting on June 20, 2013, as is described in greater detail in the Shared Platform Document (Appendix A).

Figure 5.2 - 1 *Secure-on-Request* Service Blueprint at Software, Inc.


Secure-on-Request Service Blueprint
Physical Evidence
Customer Actions
Onstage Technology Actions
Onstage/ Contact Employee Actions
Backstage/invisibal Contact Employee Actions
Support Processes
Secure-on-Request website
Security profile reports on website
Manual analysis results
Meeting with TAMs
Tools or meetings to capture customer requirements
Secure-on-Request website updated with the latest release
Customer Uploads Source Code
Customer receives security profile reports
Customer receives manual reports analysis results
Customer requests further details on analysis results
Customer requests new features or defects fix
Customer receives release updated with new features and fixes
Line of Interaction
Run security scan on website
Security profile reports displayed on website
Analysis results displayed on website or emailed
CRM / requirements tools
Website updated with the latest release
TAMs
Product Management
Line of Visibility
Security Auditors or Operation Services
Line of Internal Interaction
FoD Release Management Process
Inputs
CRM (Support)
PM & Engineering
Operations
Defect
Escalation
New Feature
Enhancement
FORM
Defect Tool
Requirements Tool
PM Review
Release
PRODUCTION

5.3 Establishment phase

In the establishment phase, the issues identified during the diagnostic phase were prioritized and strategies were developed for improvements, as explained in greater detail in the Shared Platform Document (Appendix A). The steering committee approved three projects: improvement of customer relations, improvement of requirements and quality, and improvement of release cycle. Three teams were formed and specific roles were assigned for each project. Projects schedules and milestones were determined as illustrated in Table 5.3 - 1.

Table 5.3 - 1 Project Schedule

Projects Milestones	***Target Dates***
Project Start Date	7/2/2013
Implementation Decision	By 8/15/2013
Implementation Complete	By 9/30/2013
Lessons Learned	By 10/15/2013

Improvement projects that are related to the enhancement of service delivery of *Secure-on-Request* were developed through working with key stakeholders at Software Inc. Our recommendations were also informed by current literature. For example, research suggests that collaboration between the service provider and the customer must involve the whole value chain in order to co-create value (Schmidt, Dengler, & Kieninger, 2010). There are a number of challenges for the co-creation of value in service processes, and changing the focus of cooperation with customers is required (Schmidt et al., 2010). The deliverables and assigned roles of the first project (Improvement of Customer Relationship) are illustrated in 5.3 - 2.

5.3 - 2 Improvements in Customer Relationship Project

Project Roles	***Project Deliverables***	
• Project Manager: Release Manager • Project Contributors: Business Owner, Product Manager, Technical Account Managers, Selected External Customers • Project Consultants: Research team • Project Sponsor: *Secure-on-Request* business owner	Enhanced Service Usability	• Identify ways to enhance the usability of *Secure-on-Request* website, from the end-user's perspective • Effective and smooth communication of new features and releases to customers
	Value-Added Services	• Enhance TAMs team weekly status report • Identify measurements that are related to SaaS service quality and establish a process for reporting them
	Capturing The "Voice" of The Customer	• Early Adopters Program • Customer Advisory Board • Web-based collaborative customer service software

As part of the project of improving the customer relationship, the research team working with Software Inc. key stakeholders recommended enhancing service usability for *Secure-on-*

Request customers. The team suggested that focusing on the usability features of the *Secure-on-Request* tool would enhance the service quality from the end-user perspective. Also, improving the release documentation process would result in smooth communication of new features and releases to customers, and consequently enhance service usability.

For value-added services, the team recommended bolstering the TAMs team weekly status report, which summarizes customer contact and concerns, along with Software, Inc. responsiveness. The report is used by management as a measure of transparency and readiness to deal with customers' issues. Many organizations have established measurement and management approaches to improve their service delivery (Zeithaml, Berry, & Parasuraman, 1996). Our recommendation was to identify measurements as shown in Table 5.3 - 3, for *Secure-on-Request* service delivery processes that could be mapped to SaaS-Qual service quality factors defined in the literature (Benlian et al., 2011). The research team recommended this set of measurements and establishing a process for communicating it to management and other relevant stakeholders through the weekly report.

Capturing the "voice" of the customer is essential to improving the customer relationship with the *Secure-on-Request* product. An active dialog between companies and customers is needed to enhance value (Grissemann & Stokburger-Sauer, 2012; Lusch & Vargo, 2006b). S-D logic holds that value is not only created by the delivery of the service, but also during the service development process (Grissemann & Stokburger-Sauer, 2012; Lusch & Vargo, 2006b). The Early Adopters Program is a forum wherein Software Inc. elicits from select customers feedback on new product features prior to the official release to a wider customer base. Research investigating the notion of perceived empowerment to engage in new product development has

Table 5.3 - 3 Conceptual Definition of Six SaaS-Qual Factors

Factor	***Conceptual Definition***
Rapport	Includes all aspects of a SaaS provider's ability to provide knowledgeable, caring, and courteous support (e.g., joint problem solving or aligned working styles) as well as individualized attention (e.g., support tailored to individual needs).
Responsiveness	Consists of all aspects of a SaaS provider's ability to ensure that the availability and performance of the SaaS-delivered application (e.g., through professional disaster-recovery planning or load balancing) as well as the responsiveness of support staff (e.g., 24-7 hotline support availability) is guaranteed.
Reliability	Comprises all features of a SaaS vendor's ability to perform the promised services in a timely, dependable, and accurate fashion (e.g., providing services at the promised time, provision of error-free services).
Flexibility	Covers the degrees of freedom customers have to change contractual (e.g., cancellation period, payment model) or functional/technical (e.g., scalability, interoperability, or modularity of the application) aspects in the relationship with a SaaS vendor.
Features	Refers to the degree the key functionalities (e.g., data extraction, reporting, or configuration features) and design features (e.g., user interface) of a SaaS application meet the business requirements of a customer.
Security	Includes all aspects to ensure that regular (preventive) measures (e.g., regular security audits, usage of encryption, or antivirus technology) are taken to avoid unintentional data breaches or corruptions (e.g., through loss, theft, or intrusions).

shown that changes that support co-creation encourage customer creativity and appreciation (Franke, Schreier, & Kaiser, 2010; FüLler, MüHlbacher, Matzler, & Jawecki, 2009; Grissemann & Stokburger-Sauer, 2012). The Customer Advisory Board is another forum wherein Software Inc. engages with customers and gathers their feedback on service delivery and future requirements, thereby improving the value co-creation process. The web-based customer service collaborative tool is yet another powerful way to work with customers. Studies indicate that the use of communication tools that improve information and knowledge exchange result in a reduction of organizational and technical barriers for customers to contribute ideas for improving service (Schmidt et al., 2010). The deliverables and assigned roles of the second project (Improvement in Requirements and Quality) are illustrated in Table 5.3 - 4. An accurate understanding of customers' requirements is crucial for proper service delivery. The team recommended using specialized software tools for developing visual templates of requirements to help *Secure-on-Request* development team in the implementation of customers' requirements. The team also recommended that meetings be held to validate and align requirements coming from different stakeholder.

Table 5.3 - 4 Improvements in Requirements and Quality Project

Project Role	***Project Deliverables***	
• Project Manager: Release Manager • Project Contributors: Development Manager, Product Managers, QA Managers • Project Consultants: Research team • Project Sponsor: *Secure-on-Request* business owner	Requirement Management Process	• Visualization of requirements (wireframes) using software tools. • Validation of requirements through meetings and sessions and unifying statements of all stakeholders.
	Quality Improvement Process	• QA to develop end-to-end scenario-based testing for each user

To improve the quality of the service delivered through *Secure-on-Request*, it is important to ensure the quality of the SaaS product. We recommended that the QA team develop and run more end-to-end scenario-based testing, which depicts actual procedures of most *Secure on-Request* customers. The assigned roles and deliverables of the third project (Improvements in Release Cycle) are illustrated in Table 5.3 - 5.

Table 5.3 - 5 Improvements in Release Cycle Project

Role	***Deliverables***	
• Project Manager: Release Manager • Project Contributors: Development Manager, Product Manager, QA Manager • Project Consultants: Research team • Project Sponsor: *Secure-on-Request* business owner	Revised Release Model	Change the release frequency from 30 days to 60 days. Longer release cycles will allow for process improvement and thereby improve quality and service delivery
	Customer Communication Strategy	Revised release frequency to be communicated to customers, and benefits of these changes to be explained

Improving the release cycle of *Secure-on-Request* will contribute to improving the service delivered to their customers. The team recommended changing the release frequency from 30 days to 60 days. This change will impact other areas in the release-cycle process and contribute to enhancement of service delivery quality. For example, adequate time will be allotted for enacting the requirement and quality process improvements suggested above. The longer release cycle will also allow for the recommended documentation process improvement that in turn will enchain customer communication and ultimately upgrade service quality. The team also recommended a strategy for communicating this change to customers via product management and technical account management teams.

All stakeholders agreed on the suggested improvement strategy and implementation plan of the three projects. Leadership team support and operational preparedness were also part of the three projects deliverables committed by Software Inc. In the next phase we enact the approved plans. More details on the release cycle model are included in the other dissertation developed as part of our study at Software Inc. (Syed, 2014)

5.4 Acting phase

In the acting phase, we positioned the improvement strategy approved by Software Inc. The Shared Platform Document (Appendix A) has details and key dates of the acting phase activities at Software Inc. The steering committee held a kick-off meeting for each improvement project, and objectives were set. Meetings to work on the projects and evaluate progress took place between research team members and Software Inc.'s key stakeholders.

The final deliverables from each project were submitted on October 19, 2013. The acting phase was completed on October 26, 2013. Table 5.4 - 1, Table 5.4 - 2, and Table 5.4 - 3 give an overview of our activities during the acting phase. These activities will be discussed in more detail in the data analysis and findings sections.

Table 5.4 - 1 Improvements in Customer Relationship - Acting Phase

Project Deliverables		**Acting Phase Activities**
Enhanced Service Usability	Identify ways to enhance the usability of *Secure-on-Request* website, from the end user's perspective	Research team worked with TAM team to provide a list of requirements that could enhance portal usability. The list was prioritized and communicated to PM and Engineering. Most of the items from the list are on the product management roadmap
	Effective and smooth communication of new features and releases to customers	PM took ownership of coordinating documentation process. Documentation team and PM worked early in the release cycle to review and identify relevant activities
Value-Added Services	Enhance TAM team weekly status report	Research team discussed the summary report with management and TAM. A summary section was suggested as an addition to the report which include main items for quick review
	Measuring Service Quality	Research team discussed SaaS service quality measures with TAM and PM teams. A list of measurements are being considered: renewal rates, expansion (new customers), open and closed tickets
Capturing The Voice of The Customer	Early Adopters Program	Introductory meetings between PMs and identified early adopters' customers were completed. Customers reported positive feedback and more meetings for discussing requirements and evaluating features are scheduled
	Customer Advisory Board	TAM management and research team worked on this initiative. Information and sample agenda were discussed and a list of customers was identified. A CAB meeting was held at a Software Inc. conference for customers
	Web-based collaborative customer service software ("Help Desk")	Demos of the proposed software solution were done by potential vendors. The solutions included live chat, ticketing, and knowledge-management systems. A solution was chosen and development is reviewing the implementation steps to integrate the tool within *Secure-on-Request* website.

Table 5.4 - 2 Improvements in Requirements and Quality - Acting Phase

Project Deliverables		***Acting Phase Activities***
Requirement Management Process	Visualization of requirements using specialized software tools.	A software tool is being used by PM to develop visualization templates of requirements to be used by development and documentation teams.
	Validation of requirements through meetings and sessions and unifying statements of all stakeholders.	Validation of requirements meetings. Unifying statements of all stakeholders including PMs, TAMs, QA, and development during the requirement gathering process. An acceptance criteria for requirements implementation was put in place.
Quality Improvement Process	QA to develop end to end scenario base testing for each user.	TAMs and business owner of *Secure-on-Request* shared end to end testing scenarios with QA and development. These scenarios are documented and being used by QA for testing.

Table 5.4 - 3 Improvements in Release Cycle - Acting Phase

Project Deliverables		***Acting Phase Activities***
Revised Release Model	Change the release frequency from 30 days to 60 days. Longer release cycles will allow for processes improvement and consequently improve quality and service delivery	A release model was developed by the release manager and was agreed upon by all stakeholders. The *Secure-on-Request* release following this model was released on October, 2013.
Customer Communication Strategy	Revised release frequency to be communicated with customers, and benefits of these changes to be explained	A strategy for communicating these changes to customers was followed by PMs and TAMs and in other appropriate forums.

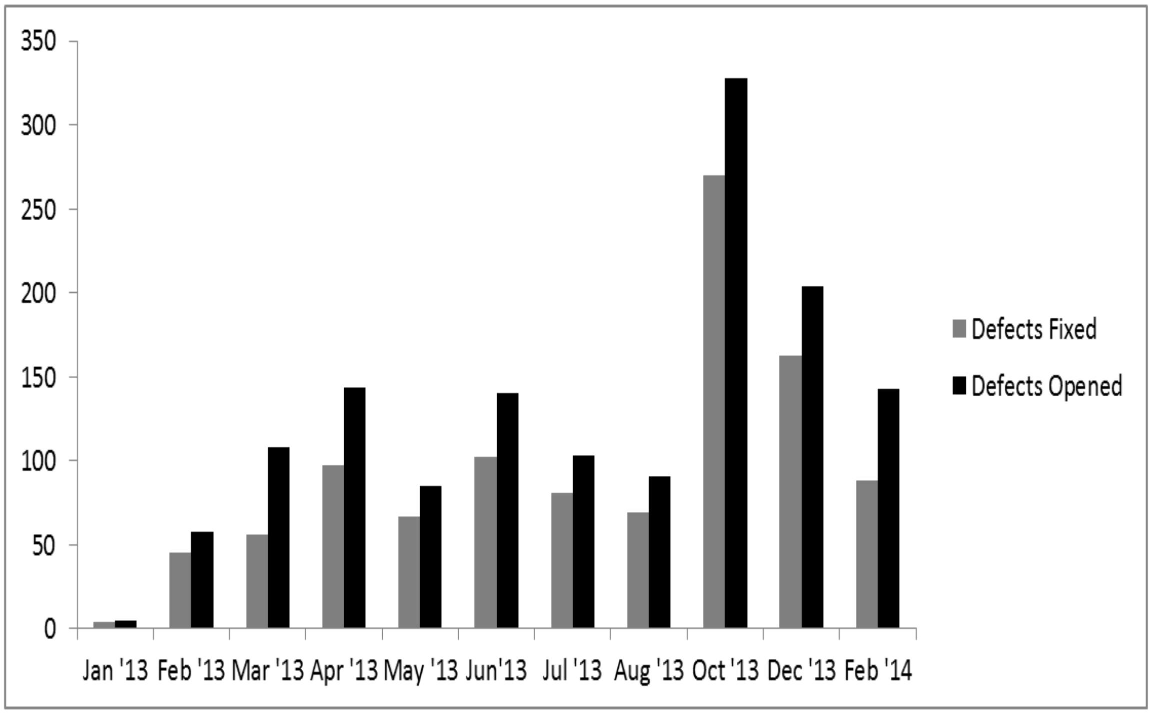
5.5 Learning Phase

In the learning phase, we reviewed the implemented solutions as well as evaluated the outcome of the three improvement projects. The details and key dates of the learning phase activities at Software Inc. are specified in the Shared Platform Document (Appendix A). Our learning-phase assessments incorporated both perception-based and practice-based methods (Napier et al., 2009). The assessments were geared toward evaluating the impact on the service-delivery process of *Secure-on-Request*. Our goals were to identify changes in each of the three project-improvement areas, determine their effect on the processes with an eye toward noting challenges that arose while implementing the changes, and make suggestions for further improvement. For the practice-based part of the assessment, we applied the norms and practices from release-management and service-delivery literature identified in the diagnostic phase (Schneider & Bowen, 2010; Vargo & Lusch, 2004) and compared them after the implementation of the improvement projects to software release management service-delivery practices at Software Inc. The research team assigned scores based on collected data and observations, and the assessment results were compared against those from the diagnostic phase. The resulting assessments are summarized in Table 5.5 - 1. Additionally, the data we collected from Software Inc.'s systems showed that the new release model allowed time for addressing service quality and for more service issues to be reported. However, the subsequent release cycles showed better stability of *Secure-on-Request* software and better service quality as illustrated in Figure 5.5 - 1. An overall assessment of the improvement projects will be discussed in Section 7.0 and Section 8.0.

Table 5.5 - 1 Service Delivery Practice-Based Assessments – Learning Phase

	Principle	***Diagnostic Phase Score*** *(June, 2013)*	***Learning Phase Score*** *(February, 2014)*
1	Support fair and non-opportunistic customer-service provisioning.	High	High
2	Ensure connections and relationships with customers during service provisioning.	High	High
3	Ensure alignment between *Secure-on-Request* directions and the strategic focus of Software Inc.	Medium	High
4	Establish process to capture customer needs and have them influence the service.	Medium	Medium
5	Understand customers' service contexts, processes, and expected outcomes.	Medium	High
6	Share information on customer perceptions of service value across *Secure-on-Request* teams.	Medium	Medium
7	Coordinate and integrate the service to allow customization to individual customers.	Low	Medium
8	Ensure clear communications of release features to provide new value to all customers.	Low	Medium
9	Maintain complete service information to assist customers' knowledge and competence.	Low	Medium
10	Measure the gap between customer expectations and perceptions of the service.	Low	Medium

Figure 5.5 - 1 *Secure-on-Request* Reported Issues - Learning Phase



6.0 RESEARCH CYCLE

Our research cycle was guided by the style composition for action research developed by Mathiassen, et al. (2012) (Table 1.0-1). We reviewed SaaS, Service Science, and software release-management streams of literature. This dissertation adopted the S-D logic framework (Vargo & Lusch, 2004). Our research process was a collaborative and iterative process focused on problem-diagnosis, change, and reflection (Avison et al., 2001). Furthermore, our study satisfied the three methodological characteristics that were described across action-research cycles (Baskerville & Wood-Harper, 1996). Rapoport (1970) identified three characteristic dilemmas of action research, which relate to ethics, goals and initiative. Details on how our study satisfied the three methodological characteristics (Baskerville & Wood-Harper, 1996) and dealt with the three dilemmas (Rapoport, 1970) are covered in the Action Research Design section in the Shared Platform Document (Appendix A).

We followed the CAR principles of action research to ensure rigor in our study (Davison et al., 2004). CAR suggests that action research is directed by five principles:

1) Researcher-client agreement;
2) Cyclical process model;
3) Theory
4) Change through action; and
5) Learning through reflection (Davison et al., 2004).

CAR provides specific questions and criteria for each principle. The Shared Platform Document (Appendix A) covers in detail how these principles were followed during our action study at Software Inc. As we followed the principles of canonical action research, evaluated the data through our analytical framework, and triangulated, we managed the action research dilemmas that occurred (Rapoport, 1970). This also helped us deal with the issue of insider bias (Coghian, 2001).

6.1 Data Collection

Our research objective was to analyze how the release-management process impacted the value co-creation in a SaaS environment at Software Inc. We collected data from multiple primary and secondary sources (Myers, 2008) throughout our collaborative study period.

Following the guidelines found in Yin (2008) and Miles and Huberman (1994), the principle data sources included semi-structured interviews and problem-solving cycle documentation. We identified key individuals from Software Inc. to be interviewed for our study. For our diagnostic-phase assessments, sixteen interviews were conducted. For our learning-phase assessments, fourteen interviews were conducted. These were face-to-face interviews of approximately one hours' duration. All interviews were recorded, and detailed notes were taken. During the course of our data collection, we used triangulation (Miles & Huberman, 1994) to counterbalance insider bias (Coghian, 2001). Table 6.1 - 1 outlines the specific primary and secondary data sources used in our research.

Table 6.1 - 1 Primary and Secondary Data Sources

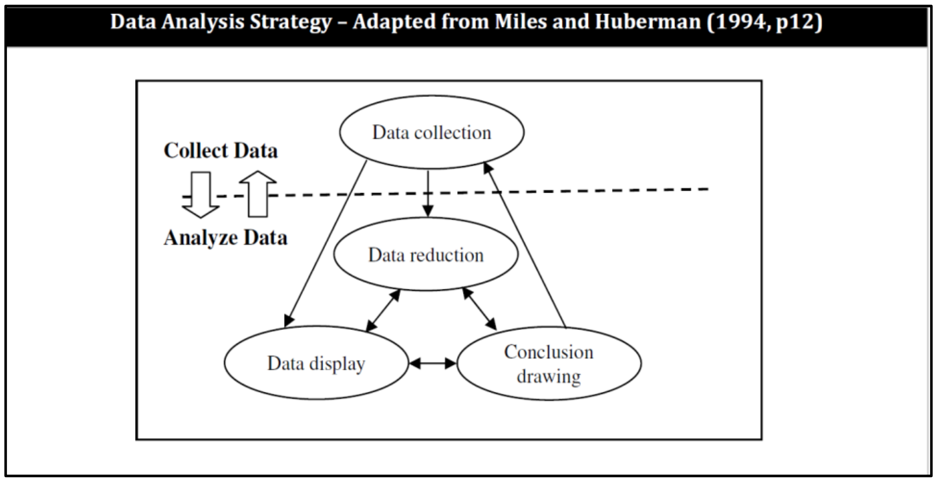
Primary Data Sources	***Secondary Data Sources***
Meetings: • Release Management Meetings (Weekly) • Bi-Weekly Scrums • Monthly Release Planning and Demos • Daily Customer Escalation Calls	Release management documentation tools: • Requirement Management Tool • Defect Management Tool • Customer Relationship Management Tool
Semi-structured interviews: • Professional Services • Sales • Quality Assurance • Product Management • Operational Services • Development • Business Unit Owner • Technical Account Management • Project Managers • External Customer	

6.2 Data Analysis

We produced our data analysis using a variety of qualitative data-analysis techniques as we enacted the cyclical process of diagnosing, action-planning, action-taking, evaluating and specifying learning during our problem-solving phase (Susman & Evered, 1978). We adopted the concepts and constructs of S-D logic (Lusch & Vargo, 2006a; Vargo & Lusch, 2004) in developing our coding scheme and analyzing our data. We used triangulation throughout our data analysis to offset potential insider bias related to the role played in Software Inc. by one of our research team members (Coghian, 2001). Our team of researchers independently analyzed meetings and interview transcripts, and used qualitative data analysis software (NVIVO) to classify, tabulate, and visualize the data.

We followed the qualitative data analysis strategy offered by Miles and Huberman (1994). These researchers suggest three concurrent flows of activities: data reduction, data display, and conclusion-drawing and verification. These activities, done continuously throughout the data-collection process, helped us determine the subsequent data-collection actions needed for evaluating the outcome of the problem-solving phase and applying the S-D logic theoretical framework. Figure 6.2 - 1 represents the data-analysis process we performed during the research cycle.

Figure 6.2 - 1 Data Analysis Activities



Miles and Huberman define data reduction as “The process of selecting, focusing, simplifying, abstracting, and transforming the data that appear in written-up field notes or transcriptions” (1994, p10). They stress that data reduction is meant to be done continuously throughout the duration of the qualitative study. As we proceeded with our data collection, we employed data-reduction techniques through identifying emerging themes, coding, and writing summaries.

Our process of data reduction started immediately upon engagement with Software Inc. During this engagement, the data-reduction process consisted of a weekly status report that was sent to all stakeholders involved in the study, Appendix B, and a bi-monthly meeting update that was sent to the business owner, who is the sponsor of our action study at Software Inc. Additional data-reduction was accomplished by detecting major practical themes relating to service delivery quality improvement, as well as identifying problem areas for refinement and conveying this information to the steering committee, Appendix C.

Data display has been described as "an organized, compressed assembly of information that permits conclusion drawing and action" (Miles and Huberman 1994, p11). Data displays may take the form of tables, graphs, and charts that organize information and make it available for quick demonstration. We developed data displays in an iterative manner during our data-collection process and after its completion. Our data displays included tables, graphs, and flowcharts (Table 5.4 - 1, Table 5.4 - 2, and Table 5.4 - 3 above). The service blueprint (Figure 5.2 - 1) also served as a data-display tool that helped to identify the complexity of the service-delivery process of *Secure-on-Request*, and refined our understanding of the overall workflow and team activities related to the service-delivery process and the release cycle at Software Inc.

Drawing conclusions involves "identifying regularities, patterns, explanations, possible configurations, causal flows, and propositions from available data" (Miles and Huberman 1994, p11). Miles and Huberman (1994) underscore the importance for research validity of iterating between drawing conclusions and verifying those conclusions in a continuous manner and reaching conclusions that may not appear until data collection is completed. Our data-analysis conclusion-drawing and verification activities took place during both the problem-solving cycle and the research cycle. During the problem-solving cycle, our twin diagnostic methods (Napier et al., 2009) provided a framework to identify primary areas for improvement relating to the service delivery of *Secure-on-Request*. Using these assessment methods, we determined the major challenges at Software Inc., and were able to provide to the steering committee both an initial diagnosis and several ideas for upgrading the system (Appendix C). These recommendations reflected the conclusions drawn from our diagnostic-phase interviews and meetings with key stakeholders during our action study at Software Inc.

Alongside each intervention, we collected additional data and conducted data analyses. Our analysis material included transcribed interviews and meetings, researchers' notes, email communications, and system-performance data (Denzin & Lincoln, 2005). These data analyses further clarified our understanding of the issues at Software Inc., and helped us to adjust our interventions based on feedback and review of the initial results. The research team also conducted ongoing discussions and debriefing sessions about the observations to advance our understanding of the problem-context at Software Inc. Additionally, we regularly referred back to the meetings and interview transcripts, researchers' notes, meeting summaries, status updates, and other material to pinpoint substantive themes related to the challenges at Software Inc. (Boyatzis, 1998; Eisenhardt, 1989; Yin, 2003). In Section 7.0, we verify the applicability of the S-D logic theoretical framework concepts in the context of the SaaS delivery environment at Software Inc. and present our study findings.

7.0 IMPROVING SAAS RELEASES AT SOFTWARE INC.

In the following section, we present the empirical results of our study and provide contextual accounts of the interventions at Software Inc. As we interpret the findings through the prism of SD-Logic concepts, we examine how the release cycle management process at Software Inc. was informed by and impacted the value co-creation process, particularly in the service delivery of the SaaS solution *Secure-on-Request*.

7.1 Value Proposition

Software Inc. proposes value in the market based on certain competences and capabilities through its SaaS solution *Secure-on-Request*. Potential customers assess this value proposition in light of their needs and compare it to competing value propositions in the market. Customers perceived the value of *Secure-on-Request* as a security solution backed by specialized services which enabled them to proactively and effectively protect their applications and processes. From the vantage point of customers, the ability of Software Inc. to respond quickly to a wide range of needs was a pivotal part of the value proposition. Software Inc. provided customers with features and services that were customized to their organizational processes. As one customer noted, "I've been asking for these thing from your competitor for a year and you guys did it in…two months."

Software Inc. offered a value proposition that was consistent with customer perception. Teams involved in the service delivery of *Secure-on-Request*, reported that Software Inc. gained value in terms of profitability, revenue, and market share. They also reported that the knowledge and expertise gained through the service delivery process resulted in a competitive edge and a strong market position for Software Inc. In the words of one *Secure-on-Request* product manager, “We won some very big deals, and in part because of our ability to turn pretty quickly on features and functions and requirements, many of our customers feel we’re pretty nimble.”

Internal teams at Software Inc. shared this perception of the *Secure-on-Request* solution value proposition. They understood the nature of the proposed value of the service and expressed the importance of delivering it well to customers. “Secure-on-Request is a software as a service and that means instead of just selling a box and a machine, customers use our software as a rental and they can use it and gain the advantages from the services side,” explained a software engineer. In fact, we found a homogeneous perception of the value proposition at Software Inc. across teams and individuals who worked with customers directly and those that indirectly supported the service delivery process. Moreover, this homogenous value proposition perception was sustained throughout the study.

At the same time, we identified several ways in which the value proposition of Software Inc. could be enhanced, including tracking customer information and measuring service quality. Our initial assessment (Appendix A, Table 4.2-3) revealed that management and decision makers of the *Secure-on-Request* team did not have easy access to the bulk of customer information and the service quality measurements pertaining to these customers. This was important, since our diagnosis of the service delivery process showed that *Secure-on-Request* serviced a large number of customers from a variety of industries. Customer segmentation was, in fact, crucial to its success. For instance, certain customers were willing to pay a premium price for specialized service, whereas smaller customers purchased a type of service that was expected to have a completely different value. Since our diagnosis showed that important customer information and service quality measurements were not readily available to management, we worked with management to refine an existing weekly report that included customer information and recommended service quality measurements (Table 5.3 - 3 above and Table 5.4 - 1 above). This report was given to the business owner and contained customer information such as number of customers, contract renewals, new accounts, and details pertaining to lost accounts. Also, we worked with the *Secure-on-Request* business owner and TAM manager to introduce a summary review of the most critical information and include service quality measurements. This new weekly report made it simple for decision makers to navigate massive amounts of customer information and helped management to identify new value propositions. *Secure-on-Request* management judged the summary review favorably. While not all suggestions were initially implemented due to time constraints of the TAM manager, she committed to implementing the remainder of the proposed changes including the suggested service quality measurements when it became clear that the management and team members were fully supportive of the initiative.

In summary, we found that Software Inc. and its customers applied resources and worked together in mutually beneficial ways. Software Inc. provided service by applying skill and knowledge combined with processes and technologies through *Secure-on-Request*. This service was deployed in combination with customers' knowledge and alongside their existing applications that needed security protection. Focusing on the value proposition related to *Secure-on-Request* created a platform upon which we worked with Software Inc. to upgrade service delivery quality and advance the value co-creation process. This will be demonstrated in the following sections.

7.2 Service Dominance

Software Inc. prioritizes responding to its customers' needs and maintaining close relationships with them. The quality of the service delivery and the relationships were sustained through the work of dedicated teams such as Sales, TAMs, and Product Management. While

TAMs were tasked with resolving customers' problems, Product Management made certain that service requirements were implemented to customers' satisfaction and beyond. As one of the product managers shared, "my work is focused around which functionality we need to provide for the product as a service solution to even exceed the customers' expectations and help cover customers' needs from that perspective." Customer-oriented and dominated by a service mindset, these teams interacted directly with their customers.

Teams and individuals, which supported service production, also reported awareness of the service-driven nature of *Secure-on-Request* and of the importance of delivering quality to customers. Naturally, the teams that did not directly interact with customers such as Product Development and Quality Assurance had less of a service mindset and were less customer-oriented. These teams were tasked with improving the basic infrastructure and software that made delivering a quality SaaS to customers possible. While these internal teams were more product-oriented, they accessed customer information needed for service production through communication with the teams that had direct customer contact. As one of the Quality Assurance members commented, "I usually reach out to one of the TAMs for a better analysis for our scanning process with our solution."

Our study at Software Inc. revealed that the overall approach to the service delivery of *Secure-on-Request* was both service- and customer-oriented. However, as we evaluated the service delivery and the release cycle process of the SaaS solution, we identified several gaps related to service dominance and customer orientation, including capturing clear service requirements, production and completion of service information, and service usability of the *Secure-on-Request* portal. We then worked collaboratively with Software Inc. to develop a number of pertinent interventions.

One gap that was affecting the service delivery of Secure-on- Request pertained to capturing clear service requirements from customers. Unclear service requirements and the lack of a verification process for their implementation caused confusion during service production and a reduction in service delivery quality (Appendix A, Table 4.2-3). Hence, there was a need for ensuring that service requirements were clearly and effectively communicated and managed across all stakeholders. Accordingly, we worked with Software Inc. on improving the service requirement process through release cycle management (Table 5.4 - 2 above). First, product management identified a third-party tool that provided a way to depict service requirements through a visual representation specifically designed for user interface. This tool enabled the product manager to ensure that customers' and other stakeholders' service requirements were accurately captured and clearly communicated across all teams involved. Second, the release manger introduced a multi-step process to ensure accurate verification and validation of requirement implementation in the early stages of service production. Meetings were held between stakeholders such as TAMs, product management, quality assurance, and development teams to align their understanding of the prioritized list of service requirements. Third, a list of requirement acceptance criteria was compiled. And, finally, a sign-off by product management development and quality assurance teams was established. Release management and product management feedback confirmed that these changes, incorporating as they did feedback from stakeholders who represented the customers’ point of view early in the release cycle (Appendix D), resulted in better customer service and improved release cycle management process. Moreover, these changes established a new, continuously evolving service requirement process that reinforces service dominance and efficiency.

A second area for service delivery improvement pertained to the production and completion of service information and communicating *Secure-on-Request* releases to customers. The information is related to communication of newly developed features required for service delivery. We determined that service information construction processes were not well established for *Secure-on-Request* releases (Appendix A, Table 4.2-3). Information from the engineering and development teams were not readily available because these teams did not view this as a high priority. The release-cycle processes and related communications were unclear for the teams that worked directly with customers. The description offered by one of the TAMs encapsulates the problem neatly, “Sometimes I feel the need to hedge our release communication to avoid failing to meet customers’ expectations.” At times, inadequate information had a negative impact on customers’ procedures, which in turn reflected badly on their perception of service quality. Communication pertaining to the newly released functionalities was not always released in a timely fashion. Thus, customers had difficulty preparing for integrating the service with their process. In the words of one customer, "You guys just released all that stuff and we were not expecting it, we are glad you are doing all that kind of stuff, but we want more notice.”

Consequently, we worked with product management to develop and implement a process that would produce and maintain complete customer service information and communication (Table 5.4 - 1 above). Product management agreed to take responsibility for the service information production process, as suggested by the GSU research team. The product manager worked with the documentation team point-of-contact and managed the related communications with the engineering and development teams. The new process also included walkthrough meetings with product management and the documentation team to ensure the accuracy and completeness of this service information (Appendix D). In the first release after we implemented the new release cycle, TAMs reported a slight improvement in the service information quality. The Release Manager also noted that this process established a platform for improvement in forthcoming release cycles. However, the progress of the production of service information was somewhat disrupted after the first release due to departure of the product managers assigned to the task. Other product managers worked with the development manager to compensate for the missing resource, and the product management started to find a replacement person to take on the responsibility. “

Lastly, TAMs identified an important problem area in the service usability of the *Secure-on-Request* portal. TAMs wanted to boost the usability of the portal from the end user’s perspective. They also pointed out the absence of several major usability features which the business owner believed already existed in the portal (Appendix A, Table 4.2-3). One of TAMs explained, “Lack of certain usability features is seen as defects by customers.” Thus, the GSU research team requested from the TAM manager a list of features that would strengthen portal usability. We asked the TAMs to prioritize this list on the basis of ease of implementation and predicted improvement on service quality. The features that scored highest based on these two criteria were considered of highest priority. A list of 30 requirements was compiled and shared with key stakeholders (Table 5.4 - 1 above). Product Management and Development committed to the implementation of requested usability features based on their priority; some features were included in the first release after our interventions; and, most features were incorporated into the product management map. As a result, TAMs and other teams involved in the service delivery reported improvement in portal usability. Most importantly, this intervention established a process wherein Product Management and Development communicated regularly with TAMs regarding service usability requirements. Development and Product Management valued the input from TAMs and its effect on service quality. TAMs expressed that they now felt their voice was being heard in the service production process, a situation that contributed to service delivery quality and a better release cycle management process.

In summary, the above-mentioned changes leveraged collaboration between different teams involved in the service delivery process. This allowed for better knowledge-sharing of customer information and experiences, which heightened the service quality and moved the organizational thinking further towards service dominance. In turn, this resulted in a better understanding of customers' issues during service production and allowed Software Inc. to service its customers better. Service dominance and understanding the important role of customers in improving *Secure-on-Request* continued to be a high priority for key stakeholders. As we shall see in the next sections, this provided a platform for the process of value co-creation and quality service delivery.

7.3 Value Co-Creation

Our analysis showed that Software Inc. as a service provider engaged in an interactive value co-creation process with its customers. The value co-creation process of *Secure-on-Request* centered on integrating customer-specific solutions and the core value was created as the service was used by the customer.

We discovered specific evidence of value co-creation with customers as we analyzed the service delivery of *Secure-on-Request*. These came to light especially in the context of teams and individuals that worked with customers directly. These teams had a good grasp of their customers' processes and supported their requirements accordingly. As the *Secure-on-Request* product manager commented, "I think because the software is as a service, it is an evolving software, we always have the ability to go back and retool certain aspects of the solution itself, adapting it to customers perspective." This approach was crucial for the value co-creation process that occurred between Software Inc. and its customers and created benefits for both. Customers benefited from the service offered to them, and Software Inc. enhanced its value offering and benefited from the expansion of its existing customer base.

Teams that supported the service delivery process played an indirect but important role in the value co-creation process. These supporting functions collected valuable information from the teams that interacted directly with *Secure-on-Request* customers. They subsequently incorporated this information during the service production process and provided new features and service functionalities in the solution. This served to further hone the value co-creation process at Software Inc. and the quality of the service delivery.

Broadly speaking, Software Inc. was well connected to the market and its customers, both of which provided important input for the value co-creation process. However, our diagnosis identified certain areas in which the value co-creation process at the company could be reinforced (Appendix A, Table 4.2-3). Hence, we worked with *Secure-on-Request* service delivery stakeholders on introducing several changes that enriched relationships with customers by eliciting customer feedback regarding the service processes.

Our assessment of the service delivery and value co-creation process at Software Inc. revealed a need for fine-tuning communications and relationships with customers. *Secure-on-Request* services a large and diverse customer base, which necessitated the development of heterogeneous service features. Interviewees shared with us that they felt the need to better understand and address their customer expectations and needs. Teams that work with customers directly such as TAMs and Product Management reported that customers needed better access to comprehensive service information and the SaaS solution. At the same time, TAMs and Product

Management expressed a desire for technical solutions and processes that would improve their insight into customer needs and expectations. Overall, we noted a widespread desire for regular dialog between customers and Software Inc. toward the goal of creating better customer relationships, which was understood as crucial for the value co-creation process. The interventions detailed below, included adding “Help Desk” to the *Secure-on-Request* portal, introducing “Customer Advisory Board” (CAB) and “Early Adopters Program” to the release cycle management represent our work in these areas (Table 5.4 - 1 above).

First, Product Management and the business owner led an effort to add a technical solution to the *Secure-on-Request* portal that supported customers’ activities and facilitated communication directly with them (Table 5.4 - 1 above). This effort was based on the feedback from the TAM manager during our assessment, which highlighted the importance of such a capability in order to serve customers better by establishing a convenient channel for communicating with them. After considering a list of third-party tools, Software Inc. decided to integrate a solution called “Help Desk” into the portal. This integration effort was high on the priority list of the Development Manager due to the support of the Steering Committee for this project. The integration of the third-party solution into the *Secure-on-Request* portal was completed and delivered as part of the first release after the new release cycle was implemented. This newly integrated capability served as a medium for knowledge management. Customers could use the solution to report service problems and propose ideas for enhancing service value. Additionally, the solution provided a way for customers to conveniently access support personal or TAMs through a “Live Chat” feature. Immediately upon release of these features, TAMs noted they had improved their communication with customers. However, it took more time for customers to become familiar with the new capability. Most customers were introduced to this function, with a consequent enhancement of customer communications and greater customer involvement in the co-creating process.

Second, in collaboration with the TAM manager and the business owner, we developed a customer-focused interaction process called the "Customer Advisory Board" (CAB) in which customers' concerns featured prominently. CAB was a way for Software Inc. to keep its finger on the pulse of the market in general and of its customers in particular, and to keep the *Secure-on-Request* service abreast of both. The GSU research team worked with the TAM manager to develop a sample agenda, and formal invitations were sent to a select list of customers identified for the meeting, see Appendix E. The first *Secure-on-Request* CAB meeting was held at a conference that occurred during the acting phase of our study. During the CAB meeting, the TAM Manager, Product Management, and the business owner collected customer feedback that was valuable for the value co-creation process. According to the TAM manager and the business owner, customers appreciated the CAB meeting as a joint learning experience and information exchange. They took the opportunity to comment on the *Secure-on-Request* strategy roadmap and reported enjoying co-creating strategies for improved services. According to the *Secure-on-Request* business owner, the exchange provided valuable knowledge which the company incorporated in its service delivery and production planning. The company intends to hold quarterly CAB meetings with select *Secure-on-Request* customers on both a domestic and global basis.

A third initiative was the Early Adopters Program. Early in the production process and as part of the release cycle management process, Product Management introduced select customers to the newly developed features and service function and solicited their participation in pre-release trials (Table 5.4 - 1 above). Meetings were held with the six customers who were selected to participate in the process. In this program customers helped to test and evaluate the latest service function added to the solution. Customers offered suggestions and feedback to Product Management. Software Inc. received helpful feedback from customers that helped enhance the release cycle process, and further meetings were scheduled. This process transformed Software Inc. customers into partners, thus reinforcing the process of value co-creation.

In conclusion, we found that Software Inc. considers its customers an important source of information. The customers' stamp was clearly visible in the service delivery, release cycle, and creation process. Thus, we built on this foundation with Software Inc. to introduce processes and tools that deepened customer involvement and incorporated the customers' points of view. This enabled Software Inc. to advance into a co-creating process environment where internal and external customers collaborated with, and contributed to, the process. In the next section we will analyze the activities of the service delivery process of *Secure-on-Request* as it engaged in value co-creation and provided service and value to its customers.

7.4 Service Delivery Process

Software Inc. serviced customers in diverse industries and the service delivery process of *Secure-on-Request* involved a number of service systems. Understanding these systems and the activities involved in the service delivery process was crucial to our analysis and the application of S-D logic concepts. These service systems were made up of resources such as people, organizations, technology and shared information. Value co-creation between consumers and Software Inc. resulted from the interaction of these service systems.

Our analysis of the service delivery process of *Secure-on-Request* centered on the activities of the relevant individuals and teams in the Software Inc. service systems. We acquired detailed information on how the service delivery process of *Secure-on-Request* was conducted.

Mapping and analyzing the service delivery process of *Secure-on-Request* using the service blue print technique (Figure 5.2 above) resulted in a shared understanding of activities of supporting functions as well as internal and external customers. This analysis also enabled us to identify opportunities to improve the service delivery quality as well as the release cycle and value co-creation processes. Furthermore, identifying and determining service delivery activities triggered important discussions between the managers and the other stakeholders at Software Inc.

The delivery process of *Secure-on-Request* starts with "customer actions." Customer actions include customers accessing the *Secure-on-Request* portal to upload and check the security of their applications and request the specialized service from Software Inc. (Figure 5.2 above). Customers' actions were also the first step in the value co-creation process in which customers evaluated the service delivery quality and the perceived value of *Secure-on-Request*.

Customer actions ran in parallel to Software Inc. "onstage contact employee actions." These took the form of direct interactions with customers and were provided by different contact persons for different matters (Figure 5.2 above). In the case of *Secure-on-Request*, the teams that interacted directly with customers during the service delivery process were mainly TAMs and Product Management. The TAMs' primary role was to resolve customers' issues. Product management defined functions in the software to meet and exceed customers' service requirements. As one of the product mangers shared with us, "My work is focused around which functionality we need to provide for the product as a service solution to even exceed the customers' expectations and help cover customers' needs from that perspective." Onstage contact employees actions were typically service transactions in which customers contacted Software Inc. for support. The nature of these interactions depended on the customers and their industries. *Secure-on-Request* service delivery was usually performed in close contact with customers. Further, the service delivery was highly knowledge-intensive. This resulted in the service quality being quote dependent on customer inputs. A good grasp of customer needs was critical for high-level service delivery and creation of value.

Supporting the direct contact employees were the “backstage contact employees.” *Secure-on-Request* backstage contact employees assisted in the service delivery process and solved customer problems without directly interacting with the customers themselves (Figure 5.2 above). In the case of *Secure-on-Request*, the Service Operations team provided support to TAMs as they assisted customers. These service activities were performed on behalf of *Secure-on-Request* customers without direct interaction. In some cases Service Operations interacted with customers directly, but with the involvement of the TAMs who worked on resolving the issue and delivering the service.

Although Software Inc. had in place certain mechanisms for supporting service delivery and garnering customer involvement, there were a number of gaps in customer service and communication. During our assessment and diagnosis, we identified areas for improvement that affected customer actions as well as onstage and backstage employee actions. These areas included capturing clear service requirements, production of service information and release communication, and lack of certain service-usability features. The gaps in these areas were addressed through introducing changes in a number of processes, as explained in Section 7.2. Further, as noted in Section7.3, we worked with Software Inc. on introducing efficient and effective means to achieve customer communication.

The service delivery of *Secure-on-Request* included the essential “Support Processes” (Figure 5.2 above). The activities of the teams and individuals involved in the support processes focused on the service production and helped with the quality of the service delivery. While these individuals and teams did not interact directly with customers, they collaborated with the "back stage" and "on stage" teams. Our analysis looked at the processes and sub-processes associated with supporting the service delivery of *Secure-on-Request*. The release cycle and the service production of *Secure-on-Request* were at the core of the service delivery, and thus contributed greatly to the value creation and quality of service. The teams and individuals involved in these processes were technically skilled and created value for customers through their technical capabilities and service production processes.

The SaaS delivery model of *Secure-on-Request* meant that Software Inc. was challenged with simultaneously designing software and delivering services. An important part of the value co-creation process was incorporating the information gained through the service delivery interactions of *Secure-on-Request* teams and the release cycle process. Our analysis revealed that although the release cycle process enabled the supporting functions to collaborate with customer facing teams, there were gaps in certain areas related to communication across the teams involved in the service delivery and release cycle process of *Secure-on-Request*.

The release frequency represented one major problem area that affected service delivery quality of the *Secure-on-Request* solution. Our assessment revealed that the monthly release of the SaaS solution had a broad negative influence on service delivery quality (Appendix A, Table 4.2-3). Interviewees across all functions expressed that monthly releases did not allow enough time for requirement analysis, quality testing, completing service information or adequate customer communication related to service delivery of the solution. As one of the TAMs shared with us, “Frankly, the customers can’t absorb these frequent updates and changes, and in the process we haven’t been giving the customers enough time to know it is changing”.

As a result, we worked with the steering committee members to change the release frequency from 30 days to 60 days. This change resulted in a reduction of these service delivery issues. Furthermore, the release manager developed a new release model (Appendix D) that systematically incorporated most of the changes that we introduced to improve the service delivery and release cycle process of *Secure-on-Request*, including the changes related to service dominance and value co-creation discussed in Sections 7.2 and 7.3. After implementing the new release cycle model and extending the release duration (Table 5.4 - 3 above), relevant teams reported an across-the-board improvement in the processes and service delivery of *Secure-on-Request*. In the extended release cycle and new release model, sufficient time was allotted for the service requirement process, service quality testing, service information completion, and advanced service delivery communication to customers. The new release model also allowed for better communication through weekly demonstration meetings and for better knowledge sharing of customer information across different teams. The TAMs and the other teams with direct customer contact had access early in the production process to the latest release information through these weekly demonstrations (Appendix D). Hence, they were ready to provide customers with the right communications.

Another issue that affected these internal support processes was related to requirement prioritization across channels (Appendix A, Table 4.2-3). In this situation, expectations were high, resources were limited, and the release timeline was short. The major challenges included prioritization for new features development, escalations from customers on defects, and technical debt. As one of the engineers stated, “Our maturity and our ability to move forward with requirements prioritization process isn’t still 100% there, and we all agree that is not what we want to be in the long term.”

Thus, we worked with Software Inc. to revamp the service requirement prioritization process (Table 5.4 - 2 above). The release manager introduced a process to ensure clear prioritization of requirements from the different stakeholders (Appendix D). The goal of this process was to avoid confusion and ensure efficiency in implementing these requirements. For instance, it was decided that a meeting of key stakeholders would be held two weeks prior to each release cycle. The key stakeholders include the business owner, product manager, TAMs manager, and development manager. In that meeting each manger presented a list of requirements and at the end of the meeting a finalized prioritized list was drawn up, to be shared among all stakeholders. According to the release manager, this turned out to be a major step forward because it ensured that key stakeholders agreed on the requirements and how they were to be prioritized. Further, this meant that requirements were shared across developers, QA and service information production.

Another challenge pertains to quality testing of the solution. The QA team was new and the processes of quality assurance for *Secure-on-Request* were immature (Appendix A, Table 4.2-3). Unclear and changing requirements as well as lack of visibility of planned features for releases added to the confusion. The short release duration also adversely affected the quality assurance process. As one of the QA engineers shared with us, "We don't have enough time between the end of the release and the time we put it out to get full quality regression tests done."

We worked with Software Inc. to resolve this issue through altering the release frequency and the development of a new release model (Appendix D). The new release model allowed more time for testing and for the quality assurance team to do regression testing. It also involved QA early in the process through the weekly development team demonstration of the new features and strengthened collaboration between the two teams. Moreover, post-release meetings in which key stakeholders analyzed strengths and defects were built into the model. This created a feedback mechanism for applying learning gained in the previous release cycle to the next release cycle.

In summary, we found multiple ways in which Software Inc. interacts with its customers during the release cycle and service delivery of *Secure-on-Request*. A thorough understanding of these was important for the enrichment of both the service delivery and the value co-creation process. Identifying the *Secure-on-Request* service-delivery activities resulted in a comprehensive view of the process and an upgrading of SaaS quality and service delivery. Release Management commented that the evaluation of customer activities related to the *Secure-on-Request* service provided them with valuable insight pertaining to the SaaS solution delivery. Improving the relationship with, and information delivery to, customers in the service delivery and value co-creation process through the release cycle management benefited both the customers and the company. This reciprocal enhancement is in line with value co-creation and the main concepts of S-D logic.

8.0 DISCUSSION

In this dissertation, we have presented our collaborative action research study with Software Inc. We aimed to help the company upgrade their release-cycle management process and service-delivery practices. Specifically, the goal was to overcome the challenges of repositioning their SaaS application *Secure-on-Request*. Although current literature reflects both the challenges in release-cycle management and the importance of the SaaS model to the software industry, research about release-cycle management in SaaS environments is limited.

This dissertation adopted the S-D logic framework (Vargo & Lusch, 2004, 2008) to explore how the release-cycle management process could be organized to improve the process of value co-creation in a SaaS environment. S-D logic's prioritization of service makes it a particularly appropriate lens through which to analyze the SaaS environment (Vargo & Lusch, 2004, 2008). In the following, we present the empirical and theoretical contributions that emerged from our action research study. Additionally, we present a grounded-process model that illustrates the roles and activities of service delivery and value co-creation processes in the SaaS environment.

8.1 Software Service Innovation at Software Inc.

Adopting S-D logic as a framework can help SaaS providers enhance the service quality that they deliver to their customers (Vargo & Lusch, 2004, 2008). S-D logic's four foundational premises (Table 3.2 - 1 above), provide a general framework for service innovation and value co-creation processes to service providers. By applying a combination of insights from our action research with S-D logic principles, SaaS providers will be particularly fortified to raise their service quality and advance their value co-creation process. In this section, we provide an account of how the managers at Software Inc. adapted S-D logic premises and organized release-cycle management and heightened the company's ongoing value co-creation with its customers.

1) *Platforms for engaging customers were established:* Software Inc. adopted several approaches to better understand their customers' organizational processes and their precise utilizations of *Secure-on-Request*. The company's managers recognized the importance of this understanding for boosting the value proposition and the value co-creation process. During our action study, we collaborated with Software Inc.'s managers and established engagement platforms such as the Early Adopters Program and the Customer Advisory Board meetings (Table 5.4 - 1 above). During these interactions, customers and users provided valuable feedback for the recurrent release-cycle and service-innovation processes at Software Inc. Moreover, these interactions created partnerships with customers to further strengthen the value co-creation process and the firm's value proposition. Recent studies on improving value co-creation and furthering S-D logic have noted the need for establishing specific mechanisms to engage with customers to co-create value (Maglio & Spohrer, 2008; Ramaswamy & Gouillart, 2010). Our analysis adds to these studies by highlighting the value of proactivity on the part of SaaS providers to gather information from their customers to maintain and enrich their service innovation. Although value co-creation involves actions on the parts of both providers and customers, we found that SaaS providers' initiation of customer engagement was pivotal to the promotion of the value-creation process.
2) *Technology was leveraged for continuous customer interaction:* Software Inc. invested resources in integrating a technological capability to efficiently and

effectively capture "the voice" of their customers. The company integrated "Help Desk" within the *Secure-on-Request* portal (Table 5.4 - 1, above). The technology enabled direct interactions with customers through a feature called "live chat", and included knowledgebase and ticket-tracking systems. This technology-based interaction provided the company with a steady flow of up-to-date information on customer service utilization, and empowered the company to quickly pinpoint customer challenges in this area. Using this feedback, the service teams were able to achieve a higher level of response to customers. Previous literature has demonstrated how the leveraging of information technology contributes to the value co-creation process (Burgoon et al., 2002; Rust & Kannan, 2003; Vargo, Maglio, & Akaka, 2008). Our analysis broadened the scope of this knowledge, and showed that introducing and adopting such technological capability required a commitment from teams interacting directly with customers (Walker, Craig-Lees, Hecker, & Francis, 2002). As the technology permitted information to be shared in new ways, it also bettered company-customer relationships as customers became participants in service innovation and value co-creation. This progression is consistent with S-D logic (Vargo & Lusch, 2008).

3) *New release-cycle management process provided effective service-systems coordination:* We found that coordination and communication between the teams responsible for supporting and developing the service offered by SaaS providers was vital. During our action research study, Software Inc. adopted a release-cycle management model that permitted such interaction to occur by granting it adequate time and by establishing specific meetings throughout the release- cycle process

among the relevant stakeholders. (Table 5.3 - 5, above). As a result, systematic communication across the teams involved in supporting and developing the service process became the norm. Moreover, the release-cycle model allowed for effective information sharing and knowledge incorporation in relation to the value co-creation process. There has been little discussion in the literature on the role of release-cycle management in service delivery and the value co-creation process. However, scholars agree on the pivotal part played by communication between the service systems for advancement of the value co-creation process (Larsson & Bowen, 1989; Maglio & Spohrer, 2008). Our study confirms the centrality of communication between the service system participants. At the same time, our research explores the role of the release-cycle management process in facilitating the coordination and information-sharing activities that lie at the heart of service innovation and value co-creation.

4) *Issues with capturing service requirements were addressed:* It is imperative that SaaS providers respond swiftly and accurately to their customers' service requirements (Berkovich et al., 2010). Software Inc. utilized the release-cycle management model to upgrade the service-requirement process. During our action research, Software Inc. introduced a tool that depicted service requirements visually, and a multi-step process to ensure accurate verification and validation of service-requirement implementation. These changes refined customer service, and established a new service-requirement process that reinforces service quality and efficiency. The current literature stresses the importance of understanding customers within the service delivery and value co-creation process (Vargo & Lusch, 2004). Our study corroborates these findings, and expands on them by suggesting the introduction to the release-cycle management

process of verification and validation processes and specific technological capabilities. We found that a well-established service-requirement process positively impacted on service innovation and value co-creation.

5) *A process for maintaining complete customer-service information was introduced:* Customers need complete service information in order to get the best service value and to be able to contribute in a meaningful way to the value co-creation process (Lusch & Nambisan, 2012; A. Payne, Storbacka, Frow, & Knox, 2009). During our action research, Software Inc. implemented a process that is designed to maintain up-to-date customer service information and communication. In this process, the product management and documentation teams collected and verified service information as they communicated with the service development teams (Table 5.4 - 1, above). Thus, we recommend assigning appropriate ownerships and establishing walkthrough meetings among the relevant stakeholders to ensure accurate service-information production. In this manner, a platform was established for improvement in release-cycle management and service-delivery quality, and TAMs reported improvement in the quality of the service information received. We found that keeping customers continuously informed about services enhanced their contribution to the value co-creation process and therefore the quality of the service they receive (Lusch & Nambisan, 2012).

6) *Software Inc. Stakeholders reported satisfaction with the new release-cycle model*: We used perception-based as well as practice-based methods (Napier et al., 2009) to evaluate the impact of our interventions and the new release-cycle model on the service-delivery process of *Secure-on-Request*. Our learning-phase interviews

revealed that Software Inc. stakeholders were satisfied with the new release-cycle model, and that they perceived improvement in the areas of service requirements, service quality, and company-customer communication. Consistent with this, our learning-phase practice-based assessment showed improvement compared to the assessment conducted in the diagnostic phase, as illustrated in Table 5.5 - 1, above. Additionally, the data we collected from Software Inc.'s systems showed that the new release model allowed time for addressing service quality. There was an increase in the reported issued initially, however, the subsequent release cycles showed a decline in the number of the issues reported, indicating heightened stability of *Secure-on-Request* software and better service quality, as illustrated in Figure 5.5 - 1 above. In summary, the extended-release cycle and new release model allowed for adequate time to fulfill service requirements, attend to the process of service quality, and provide customers with on-target communications. Hence, the changes that we made to the service delivery- and release cycle- processes also improved service dominance and value co-creation, as discussed in Sections 7.2 and 7.3.

8.2 S-D Logic Perspective on SaaS

Our study contributes to the existing body of knowledge by providing insight into the area of SaaS thorough an action research study on the release-cycle management of a large SaaS provider. Specifically, this study adopted S-D logic (Vargo & Lusch, 2004, 2008) as an analytical lens to explore how release-cycle management can be organized to positively impact on the value co-creation processes and the quality of service delivery in SaaS environments.

Extant SaaS literature has investigated the benefits to customers provided by the SaaS model. These benefits include immediate access to the latest innovations (Sääksjärvi et al., 2005; Singh et al., 2012), attractive payment structure (Sääksjärvi et al., 2005; Singh et al., 2012; Srikanth & Cohen, 2011), and reductions in IT infrastructure cost (Armbrust et al., 2010; Guo et al., 2007; Herrick, 2009; Singh et al., 2012). At the same time, research has looked at the model's benefits to SaaS providers in terms of cost reductions gained from scalability and customization, and deployment efficiency (Guo et al., 2007). The literature has also reflected the challenge of delivering and maintaining high-quality SaaS applications and retaining a competitive edge (Choudhary, 2007b; Singh et al., 2012; Srikanth & Cohen, 2011). There has been limited discussion of release-cycle management in SaaS environments, particularly in the context of service delivery and the value co-creation process.

Based on the analyses of our collaboration with Software Inc., our study adds to existing knowledge by extending our current understanding of service-innovation dynamics in SaaS environments. As explained below, our study furthers the discussion on the role of release-cycle management in realizing service dominance, clarifies the impact of adapting S-D logic principles (Vargo & Lusch, 2004, 2008) in SaaS environments, and explicates the roles of individuals and teams as they interact in the value co-creation and service-delivery processes.

First, we address the impact of release-cycle management on service dominance and service quality in SaaS environments. Our findings revealed important insight into how release-cycle management can be organized to incorporate practices that boost service dominance. Existing literature underscores the importance of service dominance for software organizations that are adopting SaaS delivery models (Khoshafian, 2006; Lusch & Nambisan, 2012; Vargo et al., 2008), shifting the thinking from the hardware and software as products to the service-delivery responsibility expected from these providers (Brocke et al., 2009; Lusch & Nambisan, 2012). This was the case at Software Inc., as our study revealed that the overall approach to service delivery was both service- and customer-oriented. However, as our evaluation identified several gaps related to service dominance, we introduced a number of practices through release-cycle management such as honing the service usability of the Secure-on-Request portal, improving the service requirement, and maintaining customer service information processes (Table 5.4-1, above). In turn, these changes allowed for better knowledge-sharing of customer information and experiences, which moved the organizational thinking further towards service dominance and upgraded service quality. So, while the literature is centered on the importance of service dominance (Brocke et al., 2009; Lusch & Nambisan, 2012) and to a lesser extent on how this is accomplished through release-cycle management, our study adds to existing research by explicating how organizing the release-cycle management could be the means by which service dominance could be systematically heightened in SaaS environments. As a result, by extrapolation to broader SaaS environments, our findings indicates that organizing release-cycle management can be instrumental and an integral part of the service innovation and enhancing service dominance in such environments.

Second, since this dissertation investigates release-cycle management in a large SaaS software provider firm, we further our understanding of how adopting the S-D logic framework in a SaaS environment will enhance the value co-creation process with SaaS customers. As evidenced in the literature, S-D logic helps SaaS providers to apprehend the process of value co-creation and adapt their internal business processes to support it (Khoshafian, 2006; Lusch & Nambisan, 2012; Vargo & Lusch, 2008). S-D logic is highly relevant to SaaS solutions providers, as they co-create value with customers and concentrate on service-delivery (Brocke et al., 2009; Vargo & Lusch, 2008). Our analysis showed that Software Inc. is a SaaS provider engaged in an interactive value co-creation process with its customers. However, we identified certain areas in which the value co-creation process could be reinforced, and we thus introduced several S-D logic-informed changes. These changes improved relationships with customers and served to elicit customer feedback regarding the service processes. Although the value co-creation process has been recognized as significant in SaaS environments (Kähkönen & Lintukangas, 2012; Lusch & Nambisan, 2012; Vargo et al., 2008), little has been written about how to reinforce its processes specifically through release-cycle management. As demonstrated in Table 5.4-1, above, the value co-creation process at Software Inc. was intensified through changes such as adding a technical solution that facilitated direct communication with customers, and the development of customer-focused interaction processes such as the Customer Advisory Board and the Early Adopters Program. These release-cycle management actions ultimately enabled Software Inc. to advance into a co-creating process environment where internal and external customers cooperated and contributed to the process (Maglio & Spohrer, 2008; Spohrer et al., 2007). Hence, when applied to the broader SaaS environments, we add to the current body of knowledge by presenting a process of adopting S-D logic principles through changing the release-cycle management, and exploring its implications for improving both the value co-creation process and the quality of service delivery.

Finally, our study furthers the understanding of the roles and activities of individuals and teams involved in the service-delivery process in a SaaS environment. The S-D logic framework made it possible to understand how various stakeholders communicated information as the release-cycle management process unfolded, and value was co-created. The importance of identifying the role of service system participants as they engage in knowledge-based interactions to co-create value is discussed in the literature (Maglio & Spohrer, 2008; Spohrer et al., 2007). In particular, studies indicate that developments in service innovation are only possible when a service system has information about their customers, and each other (Lusch & Nambisan, 2012; Spohrer & Maglio, 2008; Vargo & Lusch, 2004). Research also points to the notion that service-system resources have different arrangements of competencies that are distributed among them and connected by the value co-creation (Maglio & Spohrer, 2008). Expanding on this research, our study clarifies the roles and interaction of teams and individuals in these service systems within a SaaS environment. We combined S-D logic (Vargo & Lusch, 2004, 2008) and a service blueprint (Bitner et al., 2008) to closely analyze how the service systems interacted internally with Software Inc. and with customers externally to co-create value. These service systems included resources such as people, organizations, technology and shared information (Maglio & Spohrer, 2008). A thorough understanding of roles and interactions was crucial for a polishing of both the service delivery and the value co-creation process. This enhancement, in turn, benefited both the customers and the company.

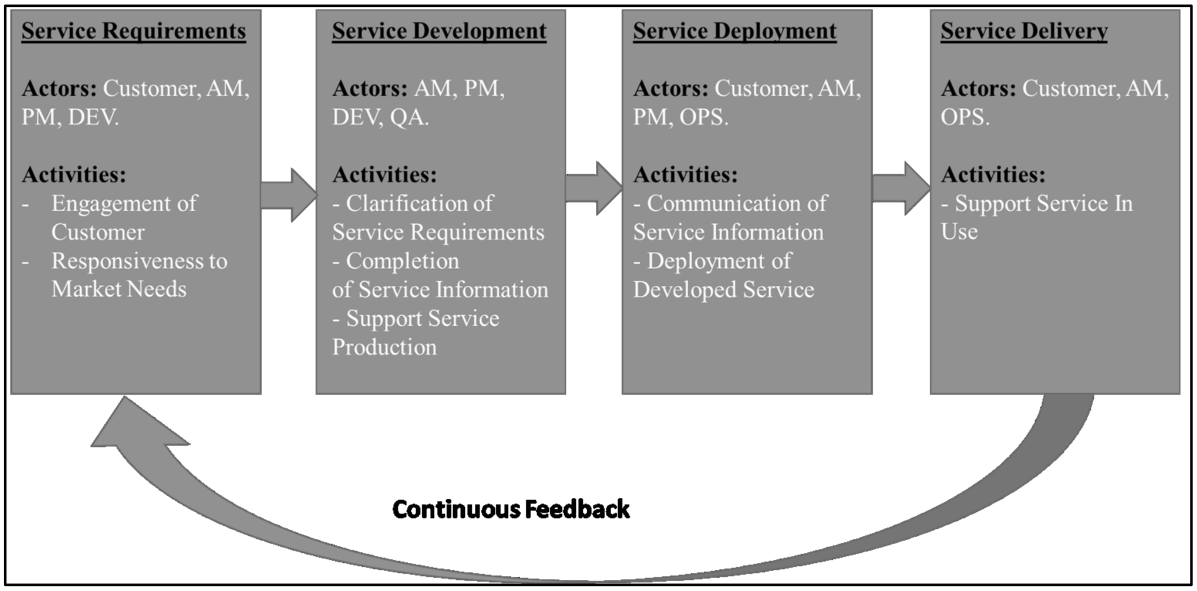
In conclusion, our analyses suggest that the S-D framework offered a powerful approach to understand and improve the service-delivery process in a SaaS environment and expand knowledge as it relates to release-cycle management and value co-creation.

8.3 Grounded SaaS Delivery Model

The service literature discusses several instruments designed to enhance the depiction of the service delivery process. One of these tools is the “Service Blueprint” technique (Bitner et al., 2008): “Services are dynamic, unfolding over a period of time through a sequence or constellation of events and steps” (Bitner et al., 2008, p. 68), and service systems can be defined as “value co-creation configurations of people, technology, value propositions connecting internal and external service systems, and shared information" (Maglio & Spohrer 2008, p. 18). These actors create value by cooperating and merging their resources, competencies, and capabilities (Bovet & Martha, 2000; Kähkönen & Lintukangas, 2012). We coded and analyzed our data (Sections 6.1, 6.2) using S-D logic (Vargo & Lusch, 2004, 2008), and adopted the framework as an analytical lens to make sense of the rich data we had gleaned from our collaborative action study with Software Inc. As a result, we developed a detailed account of how teams and individuals collaborated during the service-delivery process, and by extension the value co-creation process, at Software Inc. The framework of S-D logic enabled us to learn how the service teams engaged in the value co-creation process over the period of our action study. Specifically, S-D logic (Vargo & Lusch, 2004, 2008) and the service blueprint technique (Figure 5.2 - 1 above) (Bitner et al., 2008) made it possible for us to tease out the ways in which teams and individuals adopted various changes in order to refine the value co-creation process (Maglio & Spohrer, 2008; Vargo et al., 2008).

Based on the empirical accounts of our analysis and previous literature, we offer a grounded- process model of how individuals and teams interacted as they engaged in the service-delivery process, Figure 8.3 - 1. This model illustrates the activities of each team in relation to the value co-creation process at Software Inc., and pinpoints the service components involved as per the service blueprint technique (Bitner et al., 2008). Moreover, the model identifies the role of each team as it adopted changes in the release-cycle management and consequently the value co-creation process. The ability to describe service process to SaaS managers and customers will help them recognize what the service process encompasses and understand their corresponding roles in the value co-creation process.

Figure 8.3 - 1 Grounded Process Model for Value Co-Creation in SaaS


Service Requirements
Actors: Customer, AM, PM, DEV.
Activities:
- Engagement of Customer
- Responsiveness to Market Needs
Service Development
Actors: AM, PM, DEV, QA.
Activities:
- Clarification of Service Requirements
- Completion of Service Information
- Support Service Production
Service Deployment
Actors: Customer, AM, PM, OPS.
Activities:
- Communication of Service Information
- Deployment of Developed Service
Service Delivery
Actors: Customer, AM, OPS.
Activities:
- Support Service In Use
Continuous Feedback

Additionally, we draw upon our empirical results and propose theoretical statements or principles (Lee & Baskerville, 2003) related to service innovation in SaaS environments, as demonstrated in Table 8.3 - 1. The first principle states that value co-creation requires that SaaS providers and customers engage in continuous quality interactions. The proposed grounded-process model illustrates that activities that are related to engagement and continuous interaction with the customer occur mainly during the service requirement and service delivery stages. This principle is consistent with one of the main foundational premises of S-D logic (FP 6), which states that the customer is always a co-creator of value (Vargo & Lusch, 2008). The value co-creation process, in which the customer plays a central role, demands continuous interaction between SaaS provider and the customer.

Table 8.3 - 1 Grounded Process Model for Value Co-Creation in SaaS

	Stages	***Actors***	***Activities***	***Service Components***
Continuous Feedback	1. Service Requirements	*Customer* *AM* *PM* *DEV*	• Engagement of Customer through Various Platforms such as CAB and Early Adopters Program • Responsiveness to Market Needs	• Customer Actions • Onstage employee Actions • Backstage employee Actions
	2. Service Development	AM PM DEV QA	• Clarification and Prioritization of Service Requirements • Completion of Service Information • Support Service Production	• Backstage employee Actions • Support Processes
	3. Service Deployment	Customer AM PM OPS	• Communication of Service Information • Deployment of Developed Service	• Customer Actions • Onstage employee Actions • Backstage employee Actions • Support Processes
	4. Service Delivery	Customer AM OPS	• Support Service In Use • Leveraging technology similar to "Help Desk" for knowledge sharing with customers • Utilization of service usability in the SaaS portal	• Customer Actions • Onstage employee Actions • Backstage employee Actions

The second principle states that SaaS providers must understand their customers' requirements and processes while developing and delivering the service: this is related to the service requirement and service delivery stages as demonstrated in the grounded- process model. This principle is in accordance with S-D logic (FP 10), which states that value is always uniquely determined by the beneficiary (Vargo & Lusch, 2008). In the SaaS context, a particular service delivered to a particular customer is understood to provide a specific value; the same service delivered to another customer might provide a very different value. A customer’s industry and his or her need for that service constitute the determining factors. Hence, it is crucial for SaaS providers to have a good grasp of their customers’ processes and specific requirements while developing and delivering their software-as-a-service.

The third principle states that SaaS providers adopting service logic are required to implement processes that facilitate close interactions between teams developing and supporting the service. Service development-stage activities are related to this principle. The fourth principle proposes that customers require complete information as they obtain and integrate the service with other resources to create value. This principle is mainly associated with service-deployment activities. These two principles are related to S-D logic (FP 9), which maintains that all economic and social actors are "resource integrators." This term implies that the context of value creation is a network of networks (resource integrators) and that social and economic actors integrate various types of resources to create value (Vargo & Lusch, 2008). We take this general notion and zero in on SaaS providers’ service teams and customers as the main actors in this large network. The service developed and deployed is considered part of a larger solution required by customers, and certain processes are required to facilitate close interactions for efficient resource integration between all actors.

The fifth and final principle states that customers and SaaS providers exchange skills and knowledge through developing and using the service. This principle is related to the service-delivery stage illustrated in the grounded-process model. In addition, this principle is associated with the S-D logic premise that service is the fundamental basis of exchange (FP1), and that the application of operant resources (knowledge and skills) “service,” is the basis for all exchange (Vargo & Lusch, 2008). Customers and SaaS providers exchange skills and knowledge in creating and using the software as a service, and co-create value with their customers in the process.

Finally, although such analytical generalizations are not validated beyond the observed case, as noted by Yin (2009), they combine empirical and theoretical insights in a way that informs further research in this important area. As our analysis incorporates empirical observations and contributions from earlier studies, the proposed model might be applicable, with minor variations and modifications, to other SaaS environments.

Table 8.3 - 2 Service Innovation Principles in SaaS Environments

	Service Innovation Principles in SaaS Environments	***Process Model Stage***	***Related S-D Logic FP***
1.	Value co-creation requires that SaaS providers and customers engage in continuous quality interactions	Service Requirements Service Delivery	**FP 6 -** The customer is always a co-creator of value
2.	SaaS providers are required to understand their customers' requirements and processes while delivering the service	Service Requirements Service Delivery	**FP 10 -** Value is always uniquely and phenomenologically determined by the beneficiary
3.	SaaS providers adopting a service logic are required to implement processes that facilitate close interactions between teams developing and supporting the service	Service Development	**FP 9 -** All economic and social actors are resource integrators
4.	Customers require complete information as they obtain and integrate the service with other resources to create value	Service Deployment	
5.	Customers and SaaS providers exchange skills and knowledge through developing and using the service	Service Delivery	**FP 1 -** Service is the fundamental basis of exchange

9.0 CONCLUSION

During our action research engagement at Software Inc., we collaborated with key stakeholders and conducted research with the dual objectives of advancing academic knowledge and enlightening professional practices (Van de Ven, 2007) . Thus, our research demonstrated value in both theoretical and practical areas (Baskerville & Myers, 2009; Baskerville & Wood-Harper, 1996). Accordingly, this research contributed to theory and sharpened the value co-creation process and service quality of a SaaS provider through intervening in its release-cycle management practices. However, as always the study has important limitations; these relate to generalizability, research bias, and theoretical framing approach. It is to be noted that we developed a research methodology which minimized these concerns and increased the reliability and validity of our study.

First, the single-environment study sample used in this study may limit generalizability (Miles & Huberman, 1994; Myers, 2008). However, this limitation should be considered against the benefits of drawing attention to the details of processes and multiple stakeholder perspectives (Miles & Huberman, 1994). Additionally, it is important to examine opportunities for engaging in analytical generalizations that connect empirical insights to existing theory and into suggestions for future research (Lee & Baskerville, 2003; Yin, 2009). Accordingly, the study provides theoretical contributions and a grounded-process model of the value co-creation process at Software Inc. so that other researchers may evaluate the results and their applicability to other SaaS environments (Lee & Baskerville, 2003; Lincoln & Guba, 1985).

Second, research bias was a concern as one of our researchers is an “insider” (Coghian, 2001) and played multiple roles as both researcher and release manager at Software Inc. To minimize this limitation, we gathered rich data through interviews, meetings, researchers' notes, and documentation from different primary and secondary sources (Miles & Huberman, 1994; Myers, 2008; Yin, 2009). We triangulated the data with the involvement of the other two research members and between the different data sources (Miles & Huberman, 1994). Additionally, we followed the principles of canonical action research (Davison et al., 2004) as set out in (Appendix A) to minimize insider bias and ensure research rigor.

Finally, the data analysis might have been susceptible to interpretive biases due to the adoption of the S-D logic framework (Vargo & Lusch, 2004, 2008). Different theoretical frameworks could have been applied to explore service delivery and value co-creation process at Software Inc. However, as we evaluated the problem situation through the dual-cycle process (McKay & Marshall, 2001), S-D logic (Vargo & Lusch, 2004, 2008) offered an appropriate theoretical frame as we positioned the study in relation to extant SaaS, release-cycle management, and service innovation literature.

Stakeholders at Software Inc. reported that our interventions improved the company's release-cycle process and service quality. This helped Software Inc. to reposition their SaaS application *Secure-on-Request*. Additionally, the interventions strengthened relations among the service teams at Software Inc. and between the company and its customers. Thus, our interventions at Software Inc. produced notable outcomes relating to release-cycle management and service quality. The lessons learned by Software Inc. could well be relevant to other SaaS providers in similar settings. Our findings have implications for SaaS managers seeking to strengthen their service quality and enhance their value proposition in the market. Based on our study at Software Inc., we recommend that SaaS managers:

1) *Concentrate on knowledge-sharing with customers:* SaaS providers would do well to use to the fullest their direct interactions with customers, and actively seek to create additional opportunities for knowledge-sharing. SaaS managers might implement practical interaction forums such as CAB and the Early Adopters Program to solicit customer feedback and grasp customer needs. Direct interactions with customers during the service-delivery process and customer-engagement platforms should be harnessed for knowledge-sharing and value co-creation.
2) *Ensure communication among teams supporting the service:* A critical lesson derived from our collaboration with Software Inc. is that co-creating value and delivering quality service depends upon a thorough understanding of customer needs. We further learned that this understanding can only be gained when those responsible for service delivery are functioning as a smooth-running unit. That is, fine-tuned communication among the different stakeholders in turn allows the customer "voice" to ring out loud and clear. Service quality and value co-creation were found to closely follow suit.
3) *Re-organize release cycle to enhance the value co-creation process:* We addressed practical issues and enhanced the value co-creation process at Software Inc. by re-organizing the release-cycle process. In like manner, SaaS managers might re-organize their release-cycle process to systematically incorporate changes related to service dominance and value co-creation, thus improving the SaaS quality and service delivery process. Critically, the release-cycle process could be re-organized to allow for better communication and knowledge-sharing of customer information across different teams. Furthermore, it should allow adequate time for service-requirement

processes, service-quality testing, service-information completion, and advancing service-delivery communication to customers.

4) *Shift emphasis to service dominance to enhance SaaS quality:* The SaaS managerial approach should be dual-pronged: it should take into account both service- and customer-orientation. Teams that support service internally and that do not directly interact with customers should be helped to understand the importance of service dominance. Gaps related to service dominance and customer orientation in SaaS environments may be addressed through introducing into release-cycle management certain goals such as enhancing service usability, capturing clear service requirements, and completing service information.
5) *Utilize technology to improve customer service experience:* SaaS managers might consider introducing technological capability to upgrade company-customer interactions. These technology-assisted interactions with customers could give SaaS providers up-to-the-minute information that may be germane to service innovation, and permit a timely identification of customer problems. In this way, company-customer relationships will be bolstered and the customer's role in value co-creation and service innovation will be reinforced.
6) *Utilize service mapping to improve the release cycle and service quality:* We utilized service-blueprinting to map out the service-delivery process at Software Inc. SaaS managers might employ service-mapping techniques and similar tools to identify opportunities for improvement, and to clarify the respective roles of the teams and individuals who are participating in the process. The service mapping may uncover opportunities for re-organizing the release-cycle model, identifying failure points, and

improving customer experience, which would in turn enrich SaaS delivery and service quality.

This research contributed to the body of knowledge by supplementing the literature through exploring service innovation in SaaS environments and providing insights into the role of release-cycle management, service systems and the adoption of S-D logic principles for the value co-creation process and service quality. Accordingly, our research began with an effort to grasp how release-cycle management impacted on value co-creation in SaaS environments. As we continued our work, we advanced the understanding of service innovation and proposed a grounded-process model to describe the activities and roles of teams involved in the value co-creation process in SaaS environments. Future studies might explore further the role of release-cycle management in SaaS value co-creation and service dominance, adopt alternative theoretical frameworks, and expand the proposed grounded model to the broader SaaS field.

APPENDIX A: SHARED PLATFORM DOCUMENT

Improving Processes and Services in a Software Unit: An Action Research Study into Release Cycle Management

Neda Barqawi and Kamran Syed

J. Mack Robinson College of Business

Georgia State University

A1.0 PROBLEM SETTING

As part of its corporate business strategy, Software Inc. has decided to develop and reposition its on-line security testing solution, *Secure-on-Request*. This Software-as-a-Service (SaaS) application enables an organization to test the security of its software quickly, accurately, affordably, and without installing additional software. This action research investigated the challenges around the recurrent release management and the continuous service delivery functions of *Secure-on-Request* at Software Inc. The release management team of the application faces four significant problems: (1) the recent acquisition of the software; (2) the complexity of service delivery; (3) a new engineering and product management team; and (4) software engineering process immaturity.

A1.1 Recently Acquired Software

Software Inc. inherited *Secure-on-Request* through a recent acquisition. The company plans to develop and reposition this SaaS to realize its full potential. There were issues with *Secure-on-Request* stemming from before the acquisition: the original design needed rethinking, parts of the system were difficult to use, and the system's use of resources was less than optimal. Overall, the software is complicated, and its components need better alignment and consistency. As a result, the SaaS is somewhat fragile and until recently, the engineering team would not modify its core. Instead, they built everything around it for new functionality, and consequently the advancement of *Secure-on-Request* has been severely limited.

This innovation challenge is a predicament for the production group. The group is facing difficult to manage technology at a time when Software Inc. faces serious challenges from startup companies that threaten its market position with new, innovative technology. In this situation, Software Inc. needs to find ways to respond to customer needs and market demands as quickly as its smaller competitors. The company's best option is to adopt more agile approaches and business technology systems that respond nimbly to both changing market conditions and competitive challenges.

"Security testing as a service is a way for enterprises to reduce upfront costs and to augment limited internal resources when undertaking a software security program. This technology area is growing and will have a significant impact on the application security market over the next 12-18 months." — Joseph Feiman, Ph.D., Research Vice President and Gartner Fellow

A1.2 Complexity of Service Delivery

Secure-on-Demand is a complex, SaaS-based security-testing solution. Each customer application submitted for security analysis is unique. A team of experts conducts a thorough audit of each application for security vulnerabilities and provides a comprehensive and accurate analysis. This service tests a variety of technologies (21 different development languages) for back-end, web, mobile or cloud-based applications. It encompasses the testing of thousands of applications, security expert teams located on four continents, services provided to sixteen diverse industries including civilian and defense agencies, and companies of various sizes.

A1.3 New Engineering and Product Management Team

Due to the repositioning of *Secure-on-Request*, Software Inc. has formed several new teams to support the recurrent release of the software. These teams, each with a specific function, include engineering development, quality assurance, product management, program management, and infrastructure operations. These functional teams are heterogeneous with unique skills and knowledge. Across these teams, there are disparities in commitment due to competing priorities. In this complex organizational set-up, the newly formed teams face two critical issues: establishing appropriate collaboration patterns and effective processes, and developing the capability to recurrently release new versions of the SaaS to market.

A1.4 Low Software Engineering Process Maturity

Processes for recurrent release-management and related activities are mostly ad hoc. On the whole, software development is performed informally without proper documentation. As a result, the release-management function does not operate in a repeatable fashion. Due to this less than optimal software-development lifecycle maturity, the release-management team must work overtime to meet set deadlines and customer expectations. There are some mature tracking mechanisms and defined standards in place. However, quality issues are mainly addressed by individual team members that are technically strong and experienced. As a result, the degree of predictability in schedule, budget, scope and quality is not high and the success of a release depends upon the heroism of a few key team members. Moreover, because there are no effective mechanisms for organizational learning, the know-how of the software can easily be lost if an engineer leaves the company.

A1.5 Actors

The key functional leaders associated with this challenging situation include the head of the program management office, the development manager, the product manager and the business owner of the services provided by the application. Each of these people faces different but overlapping problems.

The head of the program management office is frustrated by the low visibility, weak predictability, and inefficient processes in delivering quality software to the market. He believes that these problems make it difficult to quickly and flexibly respond to problems and address the needs of end-users. Fluctuating and conflicting requirements is a problem for the development manager. The business owner of the service delivery of the software application is unhappy with the quality and the speed at which solutions are being delivered. The product manager feels he is sucked into day-to-day issues due to weak engineering processes which do not allow him sufficient time to focus on customer needs. Together, these players seek intervention to improve this problematic situation. Toward this end, we agreed to conduct an action research study with the above-mentioned individuals as collaborators.

We consider release management a good starting point for intervention to improve Software Inc.'s capabilities related to *Secure-on-Request*. Release management is the nub at which all of the above-described functions meet. The release-management area oversees end-to-end software engineering functions including requirement gathering, planning, designing, developing, testing, and coordinating deployment activities in the Software Development Lifecycle (SDLC). Looking at release management from the perspective of the product management and engineering teams provided a rich, internal picture emphasizing software engineering and management. At the same time, looking at the release-management function from a customer-perspective provided an external, service-oriented view. Hence, release management served as a platform for addressing the observed portfolio of problems, and drove improvements both in software process improvement and service innovation.

A2.0 RELEASE CYCLE MANAGEMENT

Software release management is defined as “the process through which software is made available to and obtained by the user” (A. Van Der Hoek, Hall, Heimbigner, & Wolf, 1997). It includes the typically recurrent identification, packaging, and distribution of the elements of a product such as an executable program, documentation, release notes, and configuration data (Ballintijn, 2005; Scott & Nisse, 2001). The term “release” refers to the distribution of software outside of the development activity, and this includes internal releases as well as outside customers (Scott & Nisse, 2001). A well-defined release-management process can be the crux of increased quality of release- planning, building, testing, and deployment activities. This will likely reduce the number of problems occurring after delivering the release to customers (Lahtela & Jantti, 2011).

The fact that *Secure-on-Request* was inherited through acquisition might be part of the problem in the release-management process. High-tech companies acquire commercial off-the-shelf software components as a strategy to achieve efficient new product development (Brown & Eisenhardt, 1995; Kakola, Koivulahti-Ojala, & Liimatainen, 2009; Meyer & Seliger, 1998). Companies try to shorten the cycle of new product development while reducing cost and improving product quality and service delivery of their products in order to succeed in the global markets of software-intensive products and services (Kakola et al., 2009; Krishnan, 1994; Prasad, 1994). In general, software release management is further complicated by the increasing tendency for software to be assembled as a “system of systems," constructed from pre-existing, independently created systems. Both developers and users of such software are affected by these trends (André Van der Hoek & Wolf, 2002)

Releasing a large software application is a complex procedure. In the case of *Secure-on-Request*, this complexity is heightened by the number of customers that use the service. A diverse and large customer base indicates a need for a substantial number of features to be included in the service. Furthermore, as the service evolves over time to incorporate the changing needs of customers, the release takes a great deal of effort and tends to be error-prone (Ballintijn, 2005). Delivering features that reliably meet customer requirements is an essential part of the release-management process; low-quality releases affect customer operations and the long-term relationship with their software providers (M. Kajko-Mattsson & Yulong, 2005). On-time delivery is equally critical to customer satisfaction (Prasad, 1994). Creating a robust software-release model and an effective release-management process will benefit business by reducing general cost and enhancing customer satisfaction (Rana & Arfi, 2005) .

Release management involves technical and management activities that take a release from a set of requirements to the final-delivery stage of the software (Danesh, Saybani, & Danesh, 2011). New management of the *Secure-on-Request* team adds challenges to the release process, since software typically result from the efforts of multiple individuals and teams (Otte, Moreton, & Knoell, 2008). Managing the work of multiple teams requires careful planning to ensure the quality of every part of the application. Meeting deadlines and documenting milestones is equally important. A release manager can be appointed to coordinate the teams and to identify problems that might affect the software-release process (C. Jensen & Scacchi, 2005).

Release managers play the diverse role of interacting, planning and coordinating with different stakeholders, as well as understanding technical issues (C. Jensen & Scacchi, 2005; Michlmayr, Hunt, & Probert, 2007) .

Software quality and the success of release management hinge on having the right processes in place. Managers and developers must be provided with accurate information and guidelines to improve decision-making processes, plan and schedule activities, predict bottlenecks, allocate resources, and optimize implementation of change requests (Basili et al., 1996). Van der Hoek et al. (1997) noted that release management is “a poorly understood and underdeveloped part of the software process,” and they pointed out several pertinent issues. Because efficient management of new-release production can improve software quality and customer satisfaction, the release-management process is crucial to the success of large software projects (Danesh et al., 2011) .

Software release management has garnered substantial academic and practical interest. We categorized the reviewed articles into four areas: standardization and development of models, process improvement, software quality, and customer and business perspectives. Standardization was the focus of several studies on software release management (Ballintijn, 2005; Biswas, 2007; M. Kajko-Mattsson & Yulong, 2005; Ramakrishnan, 2004; A. Van Der Hoek et al., 1997; André Van der Hoek & Wolf, 2002). Two studies identified specific issues in software-release management, offered a list of requirements and proposed a prototype for a software release management tool called “SRM.” The tool was designed to aid both customers and developers in the software-release management process (A. Van Der Hoek et al., 1997; André Van der Hoek & Wolf, 2002). Several studies examined the overall release process. These studies identified problems and practices for release-management processes and offered practical suggestions (Bjarnason, Wnuk, & Regnell, 2010; Danesh et al., 2011; Erenkrantz, 2003; Kakola et al., 2009; Lahtela & Jantti, 2011). Release management has also been looked at in terms of release-quality (Boote et al., 2007; Michlmayr, 2005; Prasad, 1994; Rana & Arfi, 2005). For instance, Michlmayr (2005) found that improvement of release management impacted on quality issues facing open-source development. This research identified problems in release practices, and developed ways to improve release management in free-software projects. Finally, release management has been investigated from business and customer perspectives (B. B. Jensen, Lyngshede, & Søndergaard; M Kajko-Mattsson & Meyer, 2005; Krishnan, 1994). Krishnan (1994) presented an economic model to evaluate the tradeoffs involved in software-release decisions, and discussed techniques to achieve optimal software-release time (Krishnan, 1994) .

Research on software release management is limited. Consequently, no major improvements have been seen in tools and processes used in this area. Furthermore, it has been suggested that software-release processes have been “ad hoc and homegrown” in nature (Wright, 2009). Fierce market competition is now demanding a transformation of development strategies that provides timely product introduction and responsiveness to customer need (Krishnan, 1994; Pratim Ghosh & Chandy Varghese, 2004). Therefore, we are proposing an action research study at Software Inc. on software rerelease management. Improvements in both software processes and service-delivery quality are targeted results. The theory and practice of release management is likely mainly instrumental in nature when focusing on the activity itself, that is, the perspective is of a first-order nature. We also zoomed in on and explored release management on a second-order level, that is, as an approach to organizational learning and innovation. In addition, we looked at release management from both an internal (engineering orientation) and external (customer orientation) perspective. Accordingly, our study contributed to the software organization and release-management literature regarding development of high-reliability capability, and to the SaaS and service-innovation literature regarding enhancing service-delivery quality by improving the release-management process. This knowledge will be of both practical and academic interest, as currently, significant resources are being expended on the software-release management process.

A3.0 RESEARCH METHODOLOGY

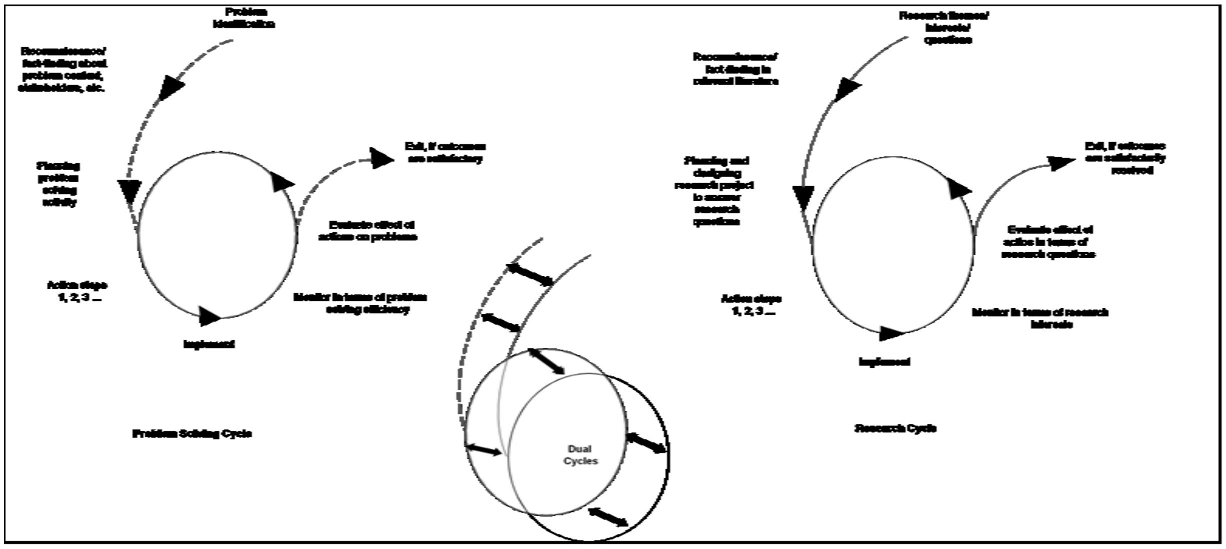
A3.1 Engaged Scholarship

To achieve deep insight into the process, we applied the principles of engaged scholarship, implying “negotiation and collaboration between researchers and practitioners in a learning community; such a community jointly produces knowledge that can both advance the scientific enterprise and enlighten a community of practitioners” (Van de Ven (2007), p.7).

Van de Ven describes engaged scholarship as a participative form of research for obtaining the views of key stakeholders to understand a complex problem. By exploiting differences between these viewpoints, he argues that engaged scholarship produces knowledge that is more penetrating and insightful than when researchers work alone. Four alternative forms of engaged scholarship are defined by Van de Ven: (1) informed basic research with stakeholder advice that is undertaken to describe, explain or predict a social phenomenon; (2) co-produced knowledge with collaborators entailing a greater sharing of power and participation between researchers and stakeholders; (3) policy, design and evaluation research undertaken to develop knowledge related to design and evaluation of policies, programs and models for addressing practical and professional problems; and (4) action and intervention research for solving a client’s problem while at the same time, contributing to the academic body of knowledge (Van de Ven, 2007). Of the four forms of engaged scholarship, we adopted action research for a number of reasons: we had unlimited access to Software Inc., we had close relationships to the leadership of *Secure-on-Request*, we wanted to actively contribute to addressing the problems faced by the *Secure-on-Request* teams, and, we assumed such interventions would provide new valuable insights into release management and service provisioning in recurrent software practices. As a result, we adopted a clinical intervention approach to diagnose and resolve a portfolio of problems in a specific client context.

Action research was introduced by Kurt Lewin, and it makes use of intervention within challenging social situations as a means of developing scientific knowledge (Lewin, 1951; Rapoport, 1970). Rapport described action research as aiming “to contribute both to the practical concerns of people in an immediate problematic situation and to the goals of social science by joint collaboration within a mutually acceptable ethical framework” (1970, p. 499). Several action research approaches have been developed by subsequent scholars. Susman and Evered developed what has become known as Canonical Action Research (CAR) by expanding the work of Lewin and Rapoport to develop a client-system infrastructure and a multi- phased cyclical process for action research consisting of diagnosing, action planning, action taking, evaluating, and specifying learning (Davison, Martinsons, & Kock, 2004; Susman & Evered, 1978). McKay & Marshall, 2001 further developed the cyclical process of action research and introduced the two simultaneous cycles of research and problem-solving. McKay and Marshall’s dual cycle framework enables researchers to diagnose problems and develop solutions in the problem-solving cycle while working closely with key stake holders. The research cycle allows researchers to focus on developing and evaluating theory, while they start with an initial area of research interest and adopt the appropriate theoretical framework (McKay & Marshall, 2001). Figure 3.0 illustrates the two cycles and the exchange of information between them.

Figure 3.0: Dual Cycle Model of Action Research at Software Inc. (McKay and Marshall 2001)


Problem identification
Reconnaissance/ fact-finding about problem context, stakeholders, etc.
Planning problem solving activity
Action steps 1, 2, 3 ...
Implement
Monitor in terms of problem solving efficiency
Evaluate effect of actions on problems
Exit, if outcomes are satisfactory
Problem Solving Cycle
Dual Cycles
Research themes/ interests/ questions
Reconnaissance/ fact finding in relevant literature
Planning and designing research project to answer research questions
Action steps 1, 2, 3 ...
Implement
Monitor in terms of research interests
Evaluate effect of action in terms of research questions
Exit, if outcomes are satisfactorily resolved
Research Cycle

A3.2 Action Research Design

Our action research study aimed to simultaneously support the *Secure-on-Request* repositioning effort at Software Inc. and contribute to the body of scientific knowledge (Avison, Baskerville, & Myers, 2001; Baskerville & Wood-Harper, 1996). The general research approach is collaborative practice research (CPR). It is an action research methodology that advocates methodological pluralism and collaboration between researchers and practitioners (Mathiassen, 2002). CPR methodology goal is to understand practice through interpretation, and to improve practice through interventions (Mathiassen, 2002). CPR suggests ways to achieve the right balance between relevance and rigor, requiring a dedicated effort involving both research and organizational work. Throughout our study we facilitated collaboration and managed the different agendas involved (Mathiassen, 2002). CPR disciplines complemented our action research approach, and allowed for collecting data systematically in addition to applying methods of interventions appropriately (Mathiassen, 2002).

We followed McKay and Marshall (2001) and organized our research into two parallel cycles: the problem-solving cycle and the research cycle. We adopted the IDEAL model (McFeeley, 1996) to guide our activities in the problem-solving cycle. Moreover, to ensure applicability and accuracy, we followed the five principles and associated criteria for Canonical Action Research (CAR) suggested by Davison et al. (2004). In Section 5, we provide a detailed account of how these principles were applied to our research at Software Inc.

Our action research was collaborative and iterative and focused on problem diagnosis, change, and reflection (Avison et al., 2001). Three methodological characteristics apply across the action research cycles (Baskerville & Wood-Harper, 1996). First, the researcher is actively involved with expected benefits for both the researcher and the organization. In our case, one of the researchers is the release manager of the project we are studying at Software Inc. His organization benefited from the ideas developed during the problem-solving cycle through the enhancement of the knowledge base of their release management process. Second, immediate application of the knowledge obtained, and cyclical process linking theory and practice. As we moved forward with our activities, we applied the knowledge gained. Finally, the cyclical process should link theory and practice. Most participants were, to some extent, involved in all aspects of the action research cycles.

Rapoport (1970) identified three characteristic dilemmas of action research: ethics, goals and initiative. He suggested that a resolution in the science direction could lead away from action and vice versa. He also argued that “good” action research selectively combines elements of both directions. We were on the look-out for these dilemmas in our research with Software Inc. Examples of ethical dilemmas include researcher reactions to the client, managing confidentiality of participants, being approached by a competitor of a client, and personal involvement in the client’s organization (Rapoport, 1970). Since one of the researchers is a manager at Software Inc., we were conscious of his dual role as researcher and employee of the client for whom we conducted the study. We consider that working with two other researchers and other stakeholders, and triangulating the data, will reduce the risks associated with dual allegiance. The discrepancy between practice and academic goals is the second dilemma identified by Rapport. We managed this dilemma by applying the recommended style composition practices (Mathiassen, Chiasson, & Germonprez, 2012), identifying the dual cycles of action research (McKay & Marshall, 2001), and recognizing the role duality as an insider action research project raised by (Coghian, 2001). Initiative, which in this context concerns the solving of a client’s problem as opposed to the pursuit of knowledge for knowledge’s sake, is the third dilemma identified by Rapoport (Rapoport, 1970). The combined effort of multiple stakeholders when conducting engaged scholarship and action research provided the proper platform for us to deal with this dilemma.

A4.0 PROBLEM-SOLVING CYCLE

We worked in a collaborative, stepwise, iterative fashion as we engaged in the problem-solving cycle to support the release-management and service-delivery processes at Software Inc. To guide our activities in the problem-solving cycle, we adopted the IDEAL model (McFeeley, 1996). This model is an approach for innovating software practices and was developed in 1996 by the Carnegie Mellon University Software Engineering Institute (McFeeley, 1996). The IDEAL model (Initiating, Diagnosing, Establishing, Acting, and Learning), illustrated in Figure 4.0, is very similar to the CAR five-phase cyclical approach (diagnosing, action planning, action taking, evaluating, and specifying learning) developed by Susman and Evered (1978). Enacting the phases of the IDEAL process guided our activities in the problem-solving cycle as well as provided opportunities to make research contributions as we studied the change processes over time.

Figure 4.0: IDEAL Model (McFeeley, 1996)

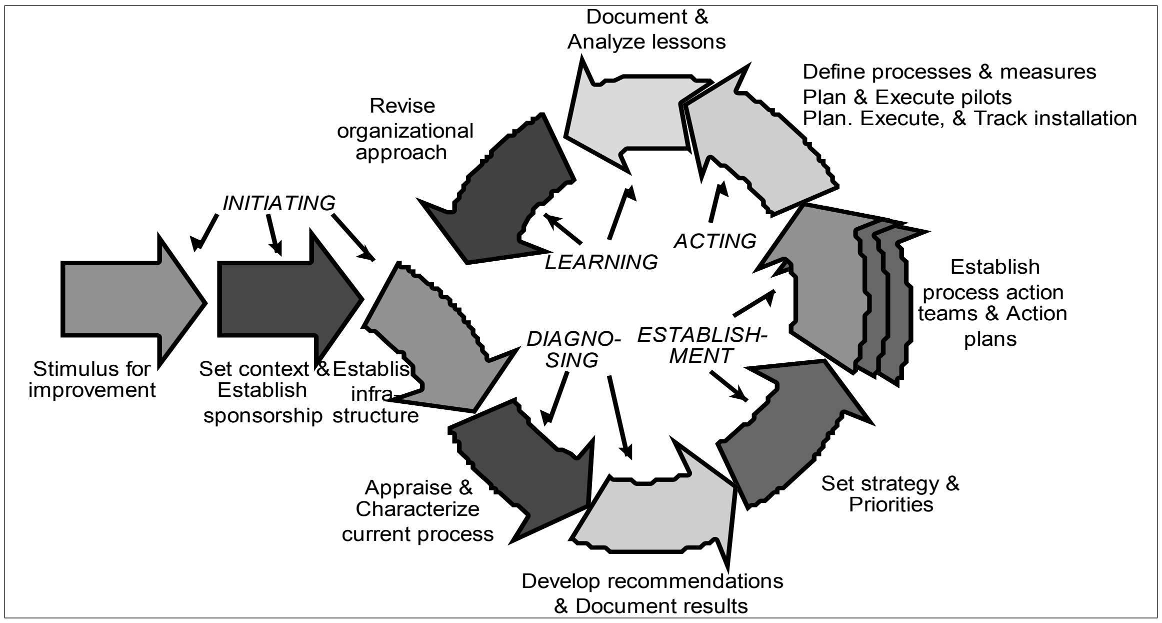

Document & Analyze lessons
Define processes & measures
Plan & Execute pilots
Plan. Execute, & Track installation
Revise organizational approach
INITIATING
LEARNING
ACTING
Establish process action teams & Action plans
DIAGNO-SING
ESTABLISH-MENT
Stimulus for improvement
Set context & Establish sponsorship
Establish infra-structure
Appraise & Characterize current process
Set strategy & Priorities
Develop recommendations & Document results

Table 4.0: IDEAL Model Phases (McFeeley, 1996)

Initiation phase	Obtaining commitment, setting goals and establishing an improvement infrastructure
Diagnostic phase	Assess current practices; develop and prioritize recommendations for improvements
Establishment phase	Create specific, focused improvement initiatives. Teams are established to deal with each of the recommended improvement areas from the diagnostic phases
Acting phase	Develop and implement solutions for each improvement area.
Learning phase	Develop plan based on the results of the initiatives. Improvements data are collected and new evaluation is prepared

A4.1 Initiation Phase

In the initiation phase, we created an initial improvement infrastructure and established the "mutually acceptable ethical framework" (Rapoport, 1970) that served as the foundation for our study. We also secured a commitment from Software Inc. to work on the possible improvement areas (McFeeley, 1996). Table 4.1: Initiation Phase Key Dates provides a summary of key dates during the initiation phase at Software Inc. The research team received Institutional Review Board approval (IRB) on March 8 2013. The research team created a memorandum of understanding (MOU) which functioned as the researcher-client agreement (RCA) (Davison et al., 2004) for the study. The MOU defined the initial roles and responsibilities of both Software Inc. and the research team. It also clarified the dual objectives of contributing to research and practice, and provided an overview of project outcomes. Subsequently, we obtained approval for the improvement plans as well as a commitment for resources to accomplish future tasks.

Table 4.1: Initiation Phase Key Dates

Date	***Activity***
January 5, 2013	Email sent to Software Inc. senior manager regarding possible collaboration
January 12, 2013	Invitation to collaboration meeting with Software Inc. senior management
March 08 , 2013	IRB Approval for Protocol Application Number: H13290
March 11, 2013	The *Memorandum of Understanding* was shared and agreed to by Software Inc.
March 15, 2013	First meeting for the project steering committee
April 09, 2013	Starting Diagnostic Phase : First diagnostic interview was conducted

A4.2 Diagnostic Phase

In the diagnostic phase, we established the foundation for the later phases in the process. The goal of the diagnostic phase was to understand the current practices and challenges related to software release management and service delivery within Software Inc.

We assessed existing software-release and service-delivery practices related to *Secure-on-Request* at Software Inc. and established our baseline. We collected data between March 2013 and June 2013 to assess current practices from the viewpoint of key stakeholders at Software Inc. (Table 4.2-1: Diagnostic Phase Key Dates). Our diagnostic work included 16 semi-structured interviews, several meeting with Software Inc. stakeholders, and a review of performance data extracted from Software Inc. internal tracking tools and systems. Our assessment included perception-based methods constructed from our interviews and meetings with Software Inc. stakeholders (Napier, Mathiassen, & Johnson, 2009). It also included practice-based methods, derived from a review of release-management and service- delivery practices in the literature.

Finally, we analyzed the performance data and reported results extracted from the main tracking systems of Software Inc.

Table 4.2-1: Diagnostic Phase Key Dates

Date	*Activity*
April 09, 2013	Starting Diagnostic Phase : First diagnostic interview was conducted
April 10, 2013	Meetings with product management team of *Secure-on-Request* started
April 11, 2013	Meetings with software development team of *Secure-on-Request* started
May 22, 2013	Last interview for initial diagnosis was completed
June 05, 2013	Release-management standards assessment completed
June 10, 2013	Service-quality standards assessment completed
June 14, 2013	First draft of diagnostic report completed
June 20, 2013	Steering committee meeting to share and discuss diagnostic findings
June 28, 2013	Establishment phase begins: First meeting to plan improvement projects

For the practice-based part of the assessment, the research team selected norms and practices that were identified in the release-management literature (Elephant, 2006; Team, 2006), and compared them to current release practices at Software Inc. We also selected service-delivery principles identified in the service-science literature (Karpen, Bove, & Lukas, 2012; Schneider & Bowen, 2010; Vargo & Lusch, 2004), and compared them to current service-delivery practices at Software Inc. The research team assigned scores based on data collected and observations, as it will be illustrated in the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014)

In the perception-based part of the assessment we identified individuals from Software Inc. who were involved in the release process of *Secure-on-Request* as well as internal and external customers (Napier et al., 2009). The research team created an interview guide that discussed objective and subjective information about the release cycle and service-delivery processes related to *Secure-on-Request*. The research team conducted semi-structured interviews with the individuals listed in Table 4.2-2: Diagnosing Interview Sources.

Table 4.2-2: Diagnosing Interview Sources

Group	***Role***	***Count***
Software Development	*Manager* *Engineer*	*2*
Quality Assurance	*Manager* *Engineer*	*2*
Product Management	Manager PM	2
Project Management	Manager Release Manager	2
Internal Customers	*Business Owner* *Professional Services* *Sales* *Technical Account Managers*	*6*
External Customers	*Manager*s	2
	Total	16

The research team met and analyzed the interviews to reflect upon emerging themes on release-management and service-delivery practices related to Secure-on-Demand. Participants' viewpoints were analyzed with a focus on strengths and weaknesses of current release-management and service-delivery practices. The identified areas for improvement are illustrated in Table 4.2-3. We will expand on these identified areas in the research team members' individual dissertation documents (Barqawi, 2014; Syed, 2014), as it relates to their research focus.

Table 4.2-3 Identified Possible Areas for Improvement at Software Inc.

Area	***Identified Issues***
Specifying and Stabilizing Requirements	• *Unclear requirements cause confusion, rework, delayed releases and adverse effects on our ability to ensure software quality.* • *Inadequate verification of requirements quality* *"In detailing our requirements there should always be a picture or a screenshot (wireframe) of what it should look like if it is a customer facing thing, so there will be no confusion"*
Prioritizing Requirements Across Channels	• *Expectations are high, release timeline is short, and resources are limited* • *Too many inputs for requirements for detailed analysis due to time constraint* • *Prioritization within and between new features development, escalations, fixing defects and technical debt are major challenges* *"Our maturity and our ability to move forward with the prioritization process isn't still 100% there, and we all agree that is not what we want to be in the long term"*
Managing Technical Debt	• *Inherent product maturity issues* • *Deadline pressure due to short release cycle* • *Lack of unit test, peer code review, definition of "done"* • *Technical debt often results in escalation of customer problems* *"We definitely have some technical debt, and I would say moderate quality, it is not high quality, I think it is important to say that our technical debt in January was much higher than it is now"*

Area	***Identified Issues***
Testing Releases	• *New quality assurance team and new management. Continue to mature quality assurance processes* • *Unclear and changing requirements adversely affect ability to ensure software quality* • *Lack of visibility of planned features for releases: adding features late in the sprint creates challenges for QA* • *Frequency of releases is affecting the time allowed for better testing for and stabilization of the software* *"We don't have enough time between the end of the release and the time we put it out to get full quality regression tests done"*
Managing Release Cycles	• *Monthly releases help catch up with competition in market* • *Monthly releases does not allow enough time for requirements analysis, testing, documentation and customer communication* *"Frankly the customers can't absorb this frequent updates and changes, and in the process we haven't been given the customers enough time to know it is changing"* *"We could do a 90 day cycle that could give us more time to provide more components and focus on the core capability of the application"*
Maintaining Complete Service Information	• *Information about features in new releases is not effectively communicated to TAM's and customers* • *Release frequency is not allowing enough time for generating complete service information* *"Release notes and user guide documentations, have been a real challenge because we have a monthly release cycles and how can you write documentation if you are actually writing codes the night before it goes out, it is pretty hard"*
Communicating Releases Across Customers	• *Release process is unclear for internal customers* • *Technical account managers feel the need to "hedge" their communication to avoid failure to meet customers' expectations* • *Customers require early notice of new features released* • *Engineering work closely with Technical account managers, Beta is an initiative in this direction, Recent UI changes made to help* *"Customers commented on one of latest releases as the following: you guys just released all that stuff and we were not expecting it, we are glad you are doing all that kind of stuff, but we want more notice"*

Area	***Identified Issues***
Giving Customers a Voice	• *Servicing large and diverse customer base allows for developing heterogeneous functions and features* • *A need for better way to understand and address customer expectations and needs* • *Fixing problems without changing the user interface making it difficult for customers to appreciate the enhancement* *"Lack of certain usability features is seen as defects by customers, but this not how we see it"*

During the course of the study, the steering committee was kept informed of the activities through weekly status reports and periodic status meetings. The research team documented the assessment findings in a complete diagnostic report, and a steering committee meeting was held on June 20, 2013 to describe the findings and overall recommendations. Table 4.2-4 illustrates the list of improvement options and recommendations shared with the steering committee during that meeting.

Table 4.2-4 Suggested Improvement Options at Software Inc.

Area	***Improvement Options***
Release Frequency	*Move from 30 day to 90 day release model*
Service Requirements	• *Allow more time for requirements analysis* • *Ensure key stakeholders agree on requirements and how they are prioritized* • *Ensure requirements are explicated and effectively shared across developers, QA and documentation* • *Ensure requirements changes are managed explicitly and shared effectively* • *Use Wireframes to ensure effective communication between technical and business people* • *Early demo of feature for key stakeholders*
Software Quality	• *Allow time for testing by reducing release frequency* • *Involve QA early in the process to support development of test cases based on requirements* • *Strengthen collaboration between development and QA about requirements, test cases, test results, and defect fixing* • *Introduce automatic testing to free resources from mundane testing, provide quick feedback to developers, and focus on high-priority issues*
Customer Relationships	• *Help customers build knowledge and competence by maintaining complete service information and scheduling monthly customer webinars* • *Gain better insight into customer needs and expectations by integrating support capability directly in the portal and scheduling quarterly on site reviews with customers* • *Improve communication of releases across TAMs and customers by providing updates and notifications in the system on new features upon application access* • *Continue assessments with key people, TAM's and customers to create stronger basis for improving customer relationships*

A4.3 Establishment Phase

In the establishment phase, we prioritized the issues that Software Inc. would address and we developed strategies for reaching solutions (Table 4.3-1: Establishment Phase Key Dates).

Table 4.3-1: Establishment Phase Key Dates

Date	***Activity***
June 28, 2013	Establishment phase begins: First meeting to plan improvement projects
July 1 , 2013	Meetings with steering committee members to agree on strategy and deliverables of improvement projects
July 2, 2013	Acting phase begins: Kick-off meetings for improvement projects started

We completed the detailed process-improvement plan based on the agreed-upon strategy, and designed plans to execute it. The suggested improvement strategy were implemented through a number of dedicated project teams with clear timelines and identified deliverables. The steering committee members agreed to form three teams to work on three improvement projects: customer relations, software quality, and release cycle. The details of these improvement projects will be discussed in the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014). Table 4.3-2 shows an overview of the three improvement projects approved by the steering committee members. The steering committee was responsible for approving the overall plans for the improvements identified in the diagnostic phase.

Table 4.3-2 *Secure-on-Request* Release Management and Service Delivery

Project Name	***Project Roles***	***Project Deliverables***
Improve Customer Relationship	• Project Manager: Release Manager • Project Contributors: Business Owner, Product Manager, Technical Account Managers, Selected External Customers • Project Consultants: Research team • Project Sponsor: *Secure-on-Request* business owner	• Enhanced Service Usability • Value Added Services • Capturing The Voice of The Customer • Operational Preparedness • Implementation Plan • Leadership Team Commitment
Improve Requirements And Quality	• Project Manager: Release Manager • Project Contributors: Development Manager, Product Managers, QA Managers • Project Consultants: Research team • Project Sponsor: *Secure-on-Request* business owner	• Requirement Management Process • Requirement Specification Formats • Development–Test Exchange Process • Development–Test–Documentation Management • Operational Preparedness • Implementation Plan • Leadership Team Commitment
Improve Release Cycle	• Project Manager: Release Manager • Project Contributors: Development Manager, Product Manager, QA Manager • Project Consultants: Research team • Project Sponsor: *Secure-on-Request* business owner	• Revised Release Model • Customer Communication Strategy • Operational Preparedness • Implementation Plan • Leadership Team Commitment

A4.4 Acting Phase

In the acting phase, we positioned the improvement projects agreed on at Software Inc., to address the areas for improvement identified during the diagnosing phase (Table 4.4: Acting Phase Key Dates). The strategy and prioritization as well as deliverables were agreed upon in the establishment phase. The research team and steering committee members held a kick-off meeting for each improvement project. At the kick-off meetings, the teams were given a set of objectives and deliverables. The teams were provided with draft project plans along with expected delivery dates. Numerous meetings were held between research team members and improvement teams to work on the deliverables and assess progress. An interim status meeting for the steering committee was held on August 19, 2013, where a status update on the three projects was presented and progress was discussed.

Table 4.4: Acting Phase Key Dates

Date	*Activity*
July 2, 2013	Acting phase begins: Kick-off meetings for improvement projects started
July 2 , 2013	Kick-off meeting for improved customer relationship project
July 3, 2013	Kick-off meeting for improved requirements and quality project
July 5, 2013	Kick-off meeting for improved release cycle project
August 19, 2013	Interim status meeting for steering committee members
September 30, 2013	Deliverables from project teams due
October 26, 2013	Learning Phase begins: acting phase completion meeting

The project team members provided projects deliverables for review on September 30, 2013. The completion meeting to close this phase was conducted on October 19, 2013. The details and key outcomes for each project are included in the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014).

A4.5 Learning Phase

In the learning phase, we reviewed the implemented solutions as well as evaluated the outcome of the three improvement projects (Table 4.5: Learning Phase Key Dates). Our learning phase assessments included perception-based as well as practice-based methods (Napier et al., 2009) with a focus on evaluating the impact on the release cycle and service-delivery process of *Secure-on-Request*. our goal was to identify changes in each of the three project improvement areas, the effect on the processes as well as the challenges that occurred during implementing the changes, and suggestions for improvement. For the perception-based assessment, we conducted fourteen semi- structured interviews with the key stakeholders. Each interview was around 45 minutes, and was recorded, and later transcribed. Our goal was to determine how different stakeholders perceived the overall value of the improvement projects implemented, their satisfaction with their own level of involvement, as well as suggestions for future improvement. For the practice-based part of the assessment, we used the norms and practices from release management and service-delivery literature identified in the diagnostic phase (Elephant, 2006; Team, 2006; Karpen, Bove, & Lukas, 2012; Schneider & Bowen, 2010; Vargo & Lusch, 2004) and compared them to software release management service-delivery practices at Software Inc. after implement the improvement projects. The research team assigned scores based on data collected and observations, and the assessment results were compared against those from the diagnosing phase as it will be illustrated in the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014). The resulting assessments and findings were summarized.   An overall assessment of the value of the improvement projects will be discussed in details the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014).

Table 4.5: Learning Phase Key Dates

Date	***Activity***
October 26, 2013	Learning Phase started
November 14, 2013	First learning phase interview was conducted
December 5, 2013	Last learning phase interview was completed
February 28, 2014	Release-management standards assessment completed
February 28 , 2014	Service-quality standards assessment completed

A5.0 RESEARCH CYCLE

The research cycle for this study was guided by the style composition for action research developed by Mathiassen, et al. (2012). Our research explored software release management, software improvement, and software-as-a-service and service-science streams of literature. The study employed Pettigrew's contextualist inquiry theory (Pettigrew, 1985) to analyze how release cycle management can be improved in the context of recurrent development of software. Additionally, the study adopted Service-dominant logic as a theoretical framework (Vargo & Lusch, 2004) to analyze how the release management process can be organized to improve Software Inc.'s ongoing value co-creation with its customers. Our research process was a collaborative and iterative process highlighting problem diagnosis, change, and reflection (Avison et al., 2001). Furthermore, our study satisfied the three methodology characteristics that were described across action research cycles (Baskerville & Wood-Harper, 1996). First, the researcher is actively involved with expected benefits for both the researcher and the organization. In our case, one of the researchers was the release manager of the project we are studying at Software Inc. We expect that as a manager, his organization will benefit from the suggestions developed during the problem-solving cycle and add to the understanding of their release-management process. Secondly, we linked theory and practice through immediate application of the knowledge obtained, and by following the cyclical process. Using our research at Software Inc., we applied knowledge gained as we moved forward to the next set of activities.

We followed CAR principles of action research to guarantee rigor as we conducted our study and depicted the research cycles (Davison et al., 2004). As explained in Section 3 on the adopted action research design, the authors provided specific questions and criteria for each principle (Davison et al., 2004) to guide the study.

A5.1 Data Collection

Action research and qualitative research require rigorous documentation, data collection, and documentation methods (Avison et al., 2001; Miles & Huberman, 1994). Our study employed several sources for data collection, which include interviews, meetings, field observations, researchers' notes, and unlimited access to Software Inc. internal systems reports and process documentation. For our diagnostic phase, we identified key individuals from Software Inc. to be interviewed for our study. We conducted sixteen one-hour face-to-face as well as phone interviews. All interviews were conducted in English, and detailed notes were taken. All interviews were recorded. During the course of our data collection, we used triangulation (Miles & Huberman, 1994) to counterbalance any insider bias (Coghian, 2001). Table 5.1 outlines the specific primary and secondary data sources for our data collection phase. Data collection methods for the study are discussed in more detail in the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014).

Table 5.1: Primary and Secondary Data Sources

Primary Data Sources	***Secondary Data Sources***
Meetings: • Release Management Meetings (Weekly) • Bi-Weekly Scrums • Monthly Release Planning and Demos • Daily Customer Escalation Calls	Release management documentation tools: • Requirements Management tool • Defect Management tool • Customer Relationship Management tool
Semi-structured interviews: • Professional Services • Sales • Quality Assurance • Product Management • Operational Services • Development • Business Unit Owner • Technical Account Management • Project Managers • External Customer	

A5.2 Data Analysis

Analysis was performed using a variety of qualitative data analysis techniques and followed the guidelines suggested by Miles and Huberman (1994). We used Pettigrew's contextualist inquiry theory and its adopted constructs (Pettigrew, 1985) in analyzing the data related to the study of release management focused on the internal software process improvement at Software Inc. We also used Service-dominant logic as framework (Vargo & Lusch, 2004, 2008) in analyzing the data related to the service delivery practices of *Secure-on-Request*. Additionally, our study followed the qualitative data analysis strategy offered by Miles and Huberman (1994). They propose three concurrent flows of activities: data reduction, data display, and conclusion drawing and verification. These activities were enacted continuously throughout the data collection process as it is explained in more detail in the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014).

Our team of researchers independently analyzed the interviews and meetings transcripts and used triangulation throughout the data analysis to offset potential for insider-bias related to the role held by one of our research team members in Software Inc. (Coghian, 2001). Qualitative data analysis software (NVIVO) was used to classify, tabulate, and visualize the data. We used the constructs and concepts from the adapted theoretical framework to analyze and code our data. Data analysis strategy and outcome of the study will be discussed in more detail in the individual dissertation documents for the research team members (Barqawi, 2014; Syed, 2014).

A6.0 PRINCIPLES OF CANONICAL ACTION RESEARCH

We followed the principles of CAR to ensure rigor as we conducted our study at Software Inc. Davison, Martinsons and Kock write that CAR is directed by five principles: 1) researcher-client agreement; 2) cyclical process model; 3) theory; 4) change through action; and 5) learning through reflection (2004). The authors provide criteria for each principle that we followed to ensure the rigor and relevance of our study (Davison et al., 2004).

Following the principle of Researcher-Client Agreement (Davison et al., 2004), we provided a framework for our research by communicating the overall objectives of the study and by explaining the roles of research team members. The Memorandum of Understanding on Research Collaboration (MoU) that we initially shared with Software Inc. clearly stated the objective of the research project. Software Inc. committed the time and resources needed to complete the study. The business owner of the product *Secure-on-Request* at Software Inc. became the sponsor of the project and helped identify the roles of the steering committee as well as those of the problem-solving project's team members. Key deliverables and evaluation criteria were communicated to all stakeholders. Software Inc. also agreed to our data collection methods including interviews, meeting attendance, and data and reports from internal systems and internal communications. Table 6.1 lists the evaluation of the principle of Researcher-Client Agreement criteria of our study.

Table 6.1: Criteria for the Researcher-Client Agreement

Principle 1 – Criteria for the Researcher - Client Agreement	***Applied to Software Inc.***	
1a – Did both the researcher and the client agree that CAR was the appropriate approach for the organizational situation?	No	No explicit agreement with Software Inc., but we followed the CAR principles to guide our research effort.
1b – Was the focus of the research project specified clearly and explicitly?	Yes	Our MoU with Software Inc. clearly stated the objective of the study: Improving processes and services in a software unit: An action research study into release management.
1c – Did the client make an explicit commitment to the project?	Yes	Software Inc. committed to the project the time and resources needed to complete the study.
1d – Were the roles and responsibilities of the researcher and client organization members specified explicitly?	Yes	Steering committee as well as the problem solving team were specified.
1e – Were project objectives and evaluation measures specified explicitly?	Yes	Key deliverables and evaluation criteria were communicated to all stakeholders.
1f – Were the data collection and analysis methods specified explicitly?	Yes	Software Inc. approved our data collection methods, including interviews, meeting attendance, data and reports from internal systems, and internal communications.

The principle of the Cyclical Process Model evaluates the relationship between diagnosing and acting (Davison et al., 2004). It emphasizes the need for modifying processes based on continuing evaluations. We followed McKay and Marshall's (2001) dual-cycle model; therefore, the information gleaned from the problem-solving cycle was incorporated into the research cycle, and the knowledge from the research cycle was integrated in the problem-solving cycle. We modified our project plans throughout the course of our study in response to challenges encountered and new knowledge gained. Continuous evaluation of our strategy and results were discussed in meetings held between steering committee members. Table 6.2 summarizes the evaluation of the principle of Cyclical Process Model criteria of our study.

Table 6.2: Criteria for the Cyclical Process Model

Principle 2– Criteria for the Cyclical Process Model (CPM)	***Applied to Software Inc.***	
2a – Did the project follow the CPM or justify any deviation from it?	Yes	We followed McKay and Marshall's (2001) dual-cycle model, therefore the information from the problem-solving cycle added to the research cycle while the knowledge from the research cycle was employed in the problem-solving cycle.
2b – Did the researcher conduct an independent diagnosis of the organizational situation?	Yes	
2c – Were the planned actions based explicitly on the results of the diagnosis?	Yes	
2d – Were the planned actions implemented and evaluated?	Yes	
2e – Did the researcher reflect on the outcomes of the intervention?	Yes	
2f – Was this reflection followed by an explicit decision on whether or not to proceed through an additional process cycle?	Yes	Throughout the course of our study we modified our project plans based on challenges encountered and new knowledge gained. Continuous evaluation of our strategy and results were discussed in meetings held between steering committee members.

The Principle of Theory focuses the research cycle and the project by ensuring that the research is guided by a theoretical framework (Davison et al., 2004). We adopted Pettigrew's contextualist inquiry theory as a framework to analyze how release cycle management can be improved in the context of recurrent development of software (Pettigrew, 1985). Based on insights from our analysis, the study developed recommendations for software providers to manage their software releases and software processes. Our study also adopted the service-dominant logic framework (Vargo & Lusch, 2004) to analyze how the release-management process can be organized to improve Software Inc.'s ongoing value co-creation with its customers. As a result, the study contributed to improving release management at Software Inc. and added to knowledge about the challenges and opportunities for software vendors to manage releases and improve the value delivered to and co-created with their customers. The theoretical frameworks chosen for our study guided our interventions and research activities as well as helped in evaluating the outcomes. Table 6.3 summarizes the evaluation of the Principle of Theory criteria of our study.

Table 6.3: Criteria for the Principle of Theory

Principle 3 – Criteria for the Principle of Theory	***Applied to Software Inc.***	
3a – Were the project activities guided by a theory or set of theories?	Yes	We adopted Pettigrew's *contextualist inquiry* theory as a framework to analyze how release cycle management can be improved in the context of recurrent development of software. *Service-dominant logic* framework was adopted to analyze how the release management process can be organized to improve Software Inc.*'s* ongoing value co-creation with its customers.
3b – Was the domain of investigation and the specific problem setting relevant to, and significant for, the interest of the researcher's community of peers as well as the client?	Yes	
3c – Was a theoretically based model used to derive the causes of the observed problem?	Yes	
3d – Did the planned intervention follow from this theoretically based model?	Yes	The theoretical frameworks chosen for our study guided our intervention and research activities at Software Inc. as well as helped in evaluating the outcomes.

The principle of Change through Action helps researchers and clients isolate and resolve problems (Davison et al., 2004). Research team members and the steering committee agreed to improve both the release process of *Secure-on-Request* and the service quality delivered to their customers. The researchers and steering committee members identified specific areas for improvement after a comprehensive assessment was conducted. The research team ensured that decisions were made with the involvement of all relevant stakeholders at Software Inc. The process and plans for the project were documented and progress was communicated to all stakeholders. Consequently, Software Inc. was supportive of our efforts throughout the project and was appreciative of the work done to improve their release-management process and service quality. Table 6.4 summarizes the evaluation of the principle of Change through Action criteria.

Table 6.4: Criteria for the Principle of Change through Action

Principle 4 – Criteria for the Principle of Change through Action	***Applied to Software Inc.***	
4a – Were both the researcher and client motivated to improve the situation?	Yes	Software Inc. and the research team members agreed on improving the release process of *Secure-on-Request* and improving the service quality delivered to customers.
4b – Were the problem and its hypothesized cause(s) specified as a result of the diagnosis?	Yes	Specific areas for improvement were identified after a comprehensive assessment was conducted at Software Inc.
4c – Were the planned actions designed to address the hypothesized cause(s)	Yes	
4d – Did the client approve the planned actions before they were implemented?	Yes	Decisions were made with the involvement of all relevant stakeholders. Project plans were documented and progress was communicated to all stakeholders.
4e – Was the organization situation assessed comprehensively both before and after the intervention?	Yes	
4f – Were the timing and nature of the actions taken clearly and completely documented?	Yes	

The principle of Learning through Reflection concerns learning through reflection from practical work as well as research (Davison et al., 2004). The research team discussed in a meeting with the steering committee members the areas targeted for improvement in the software-release and the service-delivery process. Shortly thereafter, initial recommendations for improvement in these areas were communicated to Software Inc. The research team provided an update on the status of each improvement project in a weekly communication that was sent out to key stakeholders. Several meetings were held with key stakeholders from Software Inc. to assess progress and discuss ways to ensure continuous improvement and rigorous data collection. Table 6.5 summarizes the evaluation of the principle of the Learning through Reflection criteria.

Table 6.5 Criteria for the Principle of Learning through Reflection

Principle 5 – Criteria for the Principle of Learning through Reflection	***Applied to Software Inc.***	
5a – Did the researcher provide progress reports to the client and organizational members?	Yes	The research team provided an update on the status of each improvement project, in a weekly communication material that was sent out to Software Inc. key stakeholders.
5b – Did both the researcher and the client reflect upon the outcomes of the project?	Yes	The research team discussed the areas needed for improvement Software Inc. Initial recommendations for improvement were communicated to key stakeholders shortly thereafter.
5c – Were the research activities and outcomes reported clearly and completely?	Yes	
5d – Were the results considered in terms of implications for further action in this situation?	Yes	Several meetings were held with key stakeholders from Software Inc. to assess progress and discuss ways to ensure continuous improvement and rigorous data collection
5e – Were the results considered in terms of implications for actions to be taken in related research domains?	Yes	
5f – Were the results considered in terms of implications for the research community (general knowledge, informing/re-informing theory)?	Yes	
5g – Were the results considered in terms of the general applicability of CAR?	Yes	

In sum, we applied literature-derived knowledge on, Pettigrew's contextualist inquiry theory and service-dominant logic as theoretical frameworks (Pettigrew, 1985; Vargo & Lusch, 2004, 2008), and action research as a methodology (Davison et al., 2004; Lewin, 1951; Mathiassen, 2002; McKay & Marshall, 2001; Rapoport, 1970), and engaged in collaborative research and problem-solving at Software Inc. Our research aimed to provide rich data for software-process and service-delivery improvements at Software Inc.

APPENDIX A REFERENCES

Avison, D., Baskerville, R., & Myers, M. (2001). Controlling action research projects. Information technology & people, 14(1), 28-45.

Ballintijn, G. (2005). A case study of the release management of a health-care information system. Paper presented at the proceedings of the IEEE International Conference on Software Maintenance, ICSM2005, Industrial Applications track.

Barqawi, N. (2014). Software Service Innovation: An Action Research into Release Cycle Management.

Basili, V., Briand, L., Condon, S., Kim, Y.-M., Melo, W. L., & Valett, J. D. (1996). Understanding and predicting the process of software maintenance release. Paper presented at the Proceedings of the 18th international conference on Software engineering.

Baskerville, R. L., & Wood-Harper, A. T. (1996). A critical perspective on action research as a method for information systems research. Journal of Information Technology, 11(3), 235-246.

Biswas, P. K. (2007). Autonomic Software Release Management for Communications Networks. Paper presented at the Integrated Network Management, 2007. IM'07. 10th IFIP/IEEE International Symposium on.

Bjarnason, E., Wnuk, K., & Regnell, B. (2010). Overscoping: Reasons and consequences—A case study on decision making in software product management. Paper presented at the Software Product Management (IWSPM), 2010 Fourth International Workshop on.

Boote, J. W., Hanemann, A., Kudarimoti, L., Louridas, P., Marta, L., Michael, M., . . . Tsompanidis, I. (2007). Quality assurance in perfSONAR release management. Paper presented at the Quality of Information and Communications Technology, 2007. QUATIC 2007. 6th International Conference on the.

Brown, S. L., & Eisenhardt, K. M. (1995). Product development: Past research, present findings, and future directions. Academy of management review, 343-378.

Coghian, D. (2001). Insider Action Research Projects Implications for Practising Managers. Management Learning, 32(1), 49-60.

Danesh, A. S., Saybani, M. R., & Danesh, S. Y. S. (2011). Software release management challenges in industry: An exploratory study. African Journal of Business Management, 5(20), 8050-8056.

Davison, R., Martinsons, M. G., & Kock, N. (2004). Principles of canonical action research. Information Systems Journal, 14(1), 65-86.

Elephant, P. (2006). ITIL IT Service Management Essentials. Course Workbook. Burlington, Ontario: Pink Elephant Inc.

Erenkrantz, J. R. (2003). Release management within open source projects. Proceedings of the 3rd Open Source Software DevelopmentWorkshop, 51-55.

Jensen, B. B., Lyngshede, S., & Søndergaard, D. Quality Assurance Recommendations for Open Source Developers.

Jensen, C., & Scacchi, W. (2005). Collaboration, leadership, control, and conflict negotiation and the netbeans. org open source software development community. System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on, 196b-196b.

Kajko-Mattsson, M., & Meyer, P. (2005). Evaluating the acceptor side of EM< sup> 3</sup>: release management at SAS. Paper presented at the Empirical Software Engineering, 2005. 2005 International Symposium on.

Kajko-Mattsson, M., & Yulong, F. (2005). Outlining a model of a release management process. Journal of Integrated Design and Process Science, 9(4), 13-25.

Kakola, T., Koivulahti-Ojala, M., & Liimatainen, J. (2009). An Information Systems Design Theory for Integrated Requirements and Release Management Systems. Paper presented at the System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on.

Karpen, I. O., Bove, L. L., & Lukas, B. A. (2012). Linking Service-Dominant Logic and Strategic Business Practice A Conceptual Model of a Service-Dominant Orientation. Journal of Service Research, 15(1), 21-38.

Krishnan, M. S. (1994). Software release management: a business perspective. Paper presented at the Proceedings of the 1994 conference of the Centre for Advanced Studies on Collaborative research.

Lahtela, A., & Jantti, M. (2011). Challenges and problems in release management process: A case study. Paper presented at the Software Engineering and Service Science (ICSESS), 2011 IEEE 2nd International Conference on.

Lewin, K. (1951). Field theory in social science: selected theoretical papers (Edited by Dorwin Cartwright.).

Mathiassen, L. (2002). Collaborative practice research. Information Technology & People, 15(4), 321-345.

Mathiassen, L., Chiasson, M., & Germonprez, M. (2012). Style composition in action research publication. MIS Quarterly, 36(2), 347-363.

McFeeley, B. (1996). IDEAL: A User's Guide for Software Process Improvement: DTIC Document.

McKay, J., & Marshall, P. (2001). The dual imperatives of action research. Information Technology & People, 14(1), 46-59.

Meyer, M. H., & Seliger, R. (1998). Product platforms in software development. Sloan Management Review, 40(1), 61-74.

Michlmayr, M. (2005). Quality improvement in volunteer free software projects: Exploring the impact of release management. Paper presented at the Proceedings of the First International Conference on Open Source Systems.

Michlmayr, M., Hunt, F., & Probert, D. (2007). Release management in free software projects: Practices and problems. Open Source Development, Adoption and Innovation, 295-300.

Miles, M. B., & Huberman, A. M. (1994). Qualitative data analysis: An expanded sourcebook: Sage Publications, Incorporated.

Napier, N. P., Mathiassen, L., & Johnson, R. D. (2009). Combining perceptions and prescriptions in requirements engineering process assessment: an industrial case study. Software Engineering, IEEE Transactions on, 35(5), 593-606.

Otte, T., Moreton, R., & Knoell, H. D. (2008). Applied quality assurance methods under the open source development model. Paper presented at the Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International.

Pettigrew, A. M. (1985). Contextualist research and the study of organizational change processes. Research methods in information systems, 53-78.

Prasad, R. K. (1994). Towards a zero-defect product-the End-To-End test process. Paper presented at the Software Testing, Reliability and Quality Assurance, 1994. Conference Proceedings., First International Conference on.

Pratim Ghosh, P., & Chandy Varghese, J. (2004). Globally distributed product development using a new project management framework. International Journal of Project Management, 22(8), 669-678.

Ramakrishnan, M. (2004). Software release management. Bell Labs Technical Journal, 9(1), 205-210.

Rana, A. I., & Arfi, M. W. (2005). Software Release Methodology: A Case Study. Paper presented at the Engineering Sciences and Technology, 2005. SCONEST 2005. Student Conference on.

Rapoport, R. N. (1970). Three dilemmas in action research with special reference to the Tavistock experience. Human relations, 23(6), 499-513.

Schneider, B., & Bowen, D. E. (2010). Winning the service game: Springer.

Scott, J. A., & Nisse, D. (2001). Software configuration management. SWEBOK, 103.

Susman, G. I., & Evered, R. D. (1978). An assessment of the scientific merits of action research. Administrative Science Quarterly, 582-603.

Syed, K. (2014). Improving Recurrent Software Development: A Contextualist Inquiry into Release Cycle Management.

Team, C. P. (2006). CMMI for Development, version 1.2.

Van de Ven, A. H. (2007). Engaged Scholarship: A Guide for Organizational and Social Research: A Guide for Organizational and Social Research: Oxford University Press.

Van Der Hoek, A., Hall, R., Heimbigner, D., & Wolf, A. (1997). Software release management. Software Engineering—ESEC/FSE'97, 159-175.

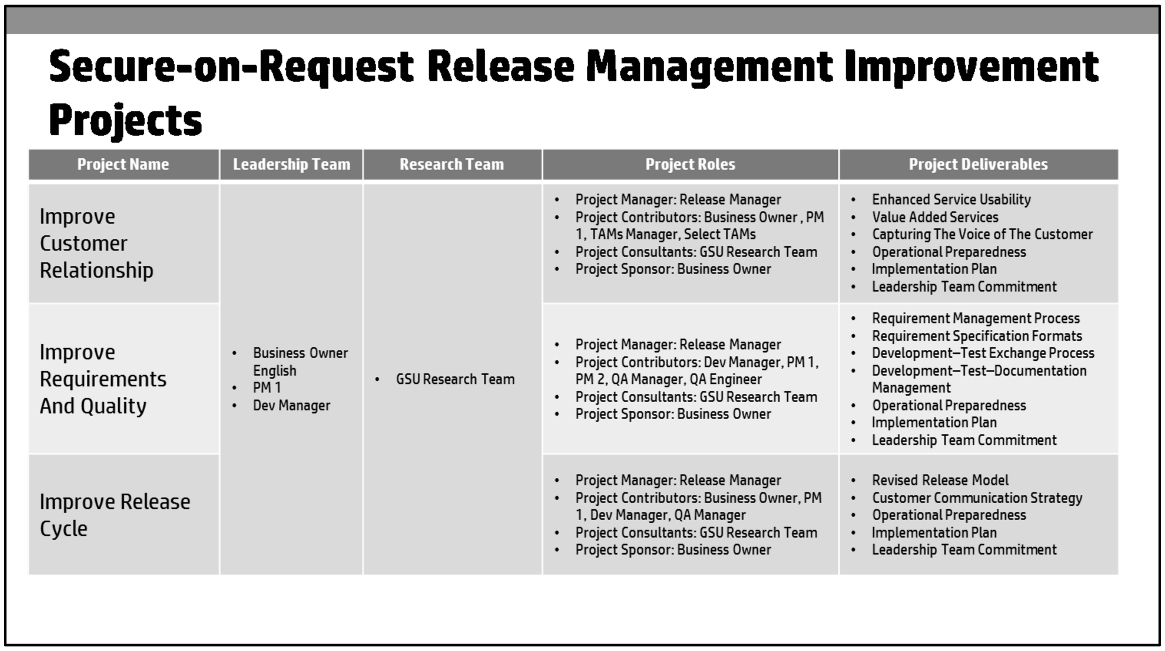
Van der Hoek, A., & Wolf, A. L. (2002). Software release management for component-based software. Software: Practice and Experience, 33(1), 77-98.

Vargo, S. L., & Lusch, R. F. (2004). Evolving to a new dominant logic for marketing. Journal of marketing, 1-17.

Vargo, S. L., & Lusch, R. F. (2008). Service-dominant logic: continuing the evolution. Journal of the Academy of Marketing Science, 36(1), 1-10.

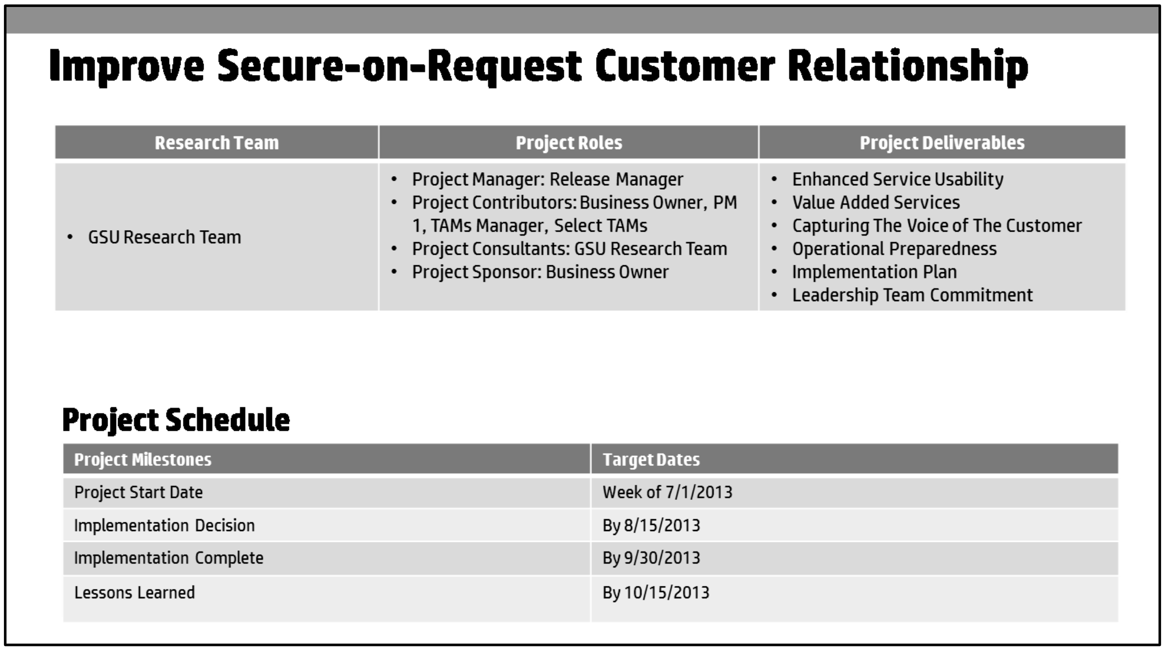
Wright, H. K. (2009). Release engineering processes, models, and metrics. Paper presented at the Proceedings of the doctoral symposium for ESEC/FSE on Doctoral symposium.

APPENDIX B: SECURE-ON-REQUEST IMPROVEMENT PROJECTS – STATUS REPORT



Secure-on-Request Release Management Improvement Projects

Project Name	Leadership Team	Research Team	Project Roles	Project Deliverables
Improve Customer Relationship	• Business Owner English • PM 1 • Dev Manager	• GSU Research Team	• Project Manager: Release Manager • Project Contributors: Business Owner , PM 1, TAMs Manager, Select TAMs • Project Consultants: GSU Research Team • Project Sponsor: Business Owner	• Enhanced Service Usability • Value Added Services • Capturing The Voice of The Customer • Operational Preparedness • Implementation Plan • Leadership Team Commitment
Improve Requirements And Quality			• Project Manager: Release Manager • Project Contributors: Dev Manager, PM 1, PM 2, QA Manager, QA Engineer • Project Consultants: GSU Research Team • Project Sponsor: Business Owner	• Requirement Management Process • Requirement Specification Formats • Development–Test Exchange Process • Development–Test–Documentation Management • Operational Preparedness • Implementation Plan • Leadership Team Commitment
Improve Release Cycle			• Project Manager: Release Manager • Project Contributors: Business Owner, PM 1, Dev Manager, QA Manager • Project Consultants: GSU Research Team • Project Sponsor: Business Owner	• Revised Release Model • Customer Communication Strategy • Operational Preparedness • Implementation Plan • Leadership Team Commitment

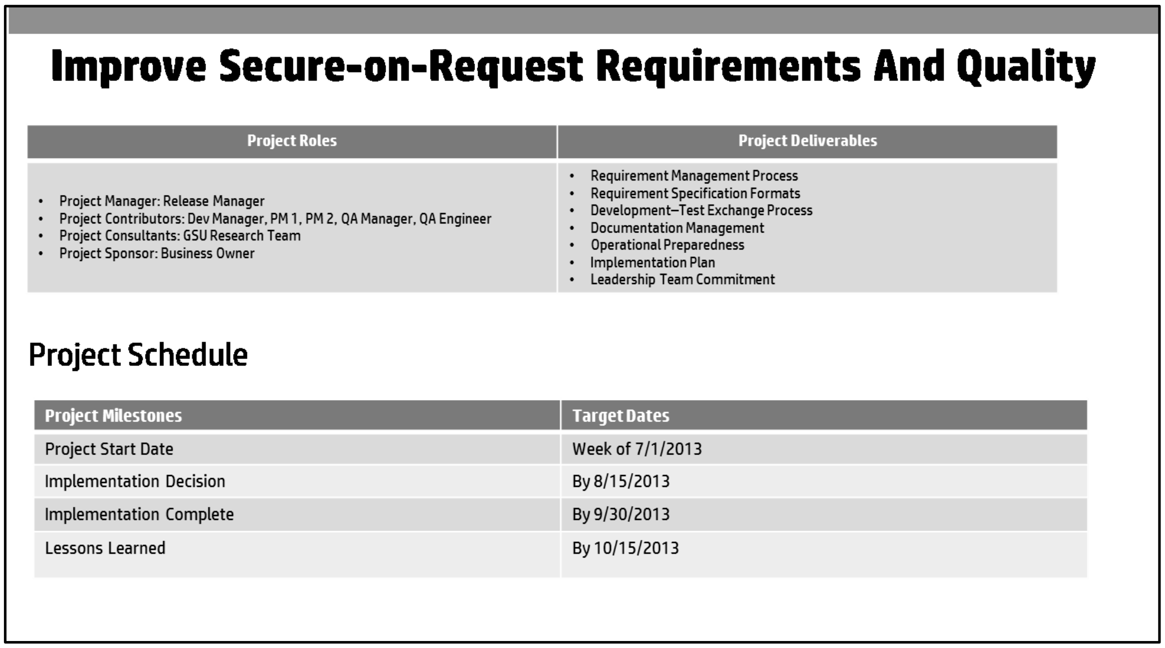


Improve Secure-on-Request Customer Relationship

Research Team	Project Roles	Project Deliverables
• GSU Research Team	• Project Manager: Release Manager • Project Contributors: Business Owner, PM 1, TAMs Manager, Select TAMs • Project Consultants: GSU Research Team • Project Sponsor: Business Owner	• Enhanced Service Usability • Value Added Services • Capturing The Voice of The Customer • Operational Preparedness • Implementation Plan • Leadership Team Commitment

Project Schedule

Project Milestones	Target Dates
Project Start Date	Week of 7/1/2013
Implementation Decision	By 8/15/2013
Implementation Complete	By 9/30/2013
Lessons Learned	By 10/15/2013

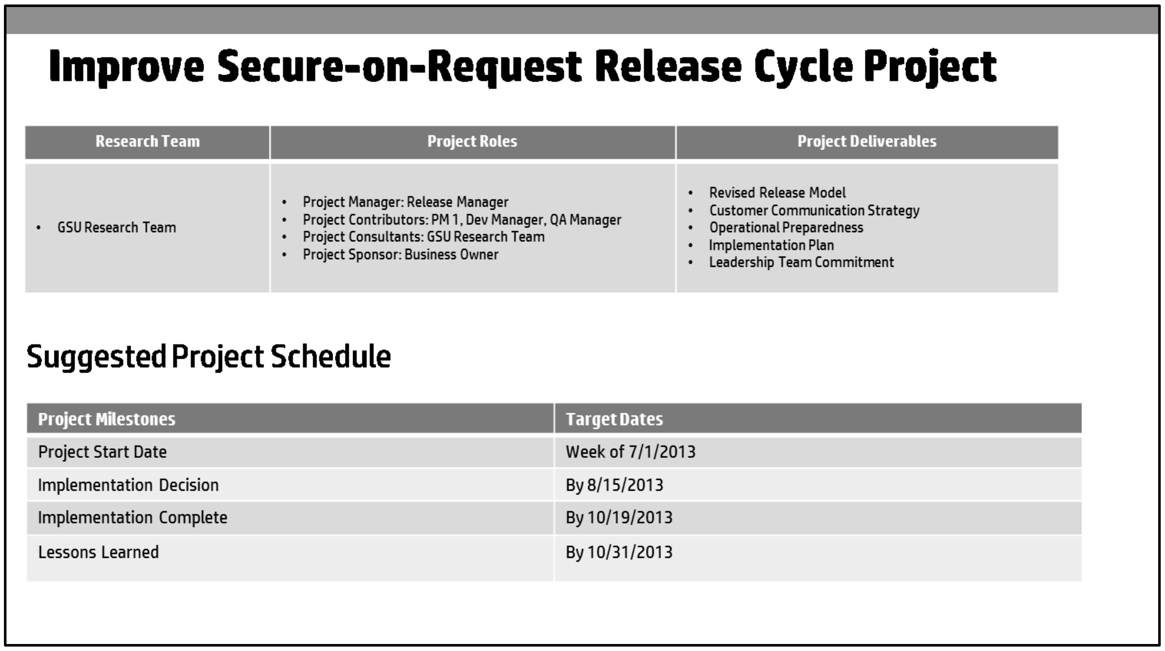


Improve Secure-on-Request Requirements And Quality

Project Roles	Project Deliverables
• Project Manager: Release Manager • Project Contributors: Dev Manager, PM 1, PM 2, QA Manager, QA Engineer • Project Consultants: GSU Research Team • Project Sponsor: Business Owner	• Requirement Management Process • Requirement Specification Formats • Development–Test Exchange Process • Documentation Management • Operational Preparedness • Implementation Plan • Leadership Team Commitment

Project Schedule

Project Milestones	Target Dates
Project Start Date	Week of 7/1/2013
Implementation Decision	By 8/15/2013
Implementation Complete	By 9/30/2013
Lessons Learned	By 10/15/2013



Improve Secure-on-Request Release Cycle Project

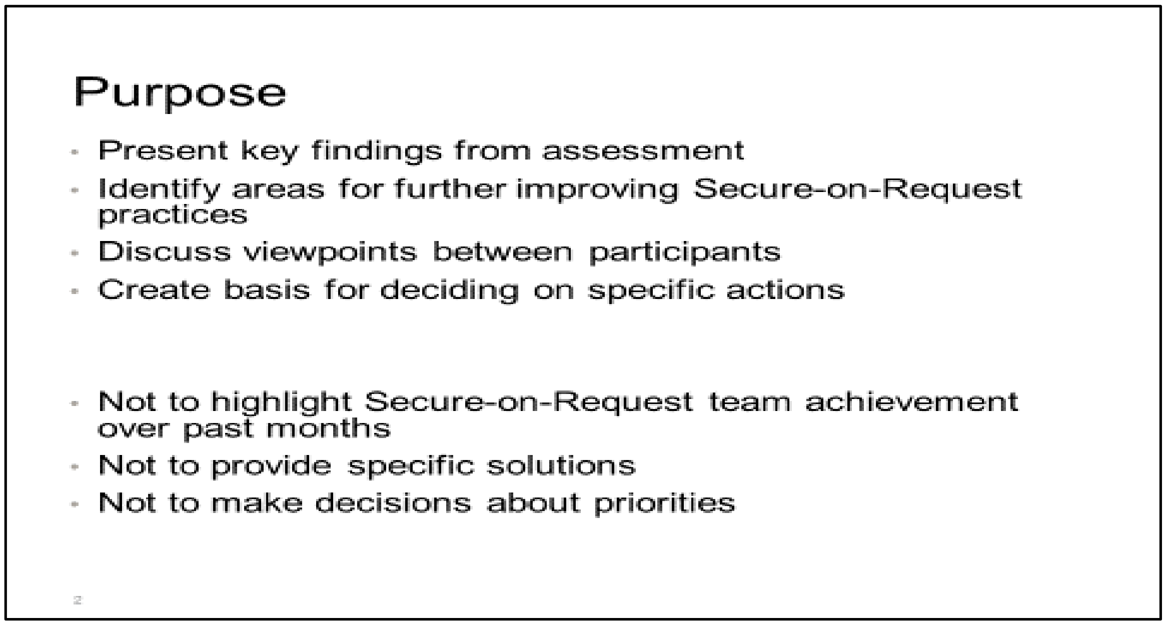
Research Team	Project Roles	Project Deliverables
• GSU Research Team	• Project Manager: Release Manager • Project Contributors: PM 1, Dev Manager, QA Manager • Project Consultants: GSU Research Team • Project Sponsor: Business Owner	• Revised Release Model • Customer Communication Strategy • Operational Preparedness • Implementation Plan • Leadership Team Commitment

Suggested Project Schedule

Project Milestones	Target Dates
Project Start Date	Week of 7/1/2013
Implementation Decision	By 8/15/2013
Implementation Complete	By 10/19/2013
Lessons Learned	By 10/31/2013

APPENDIX C: SECURE-ON-REQUEST PROCESSES ASSESSMENT AND IMPROVEMENT OPTIONS

Meeting with Steering Committee - June 20th 2013

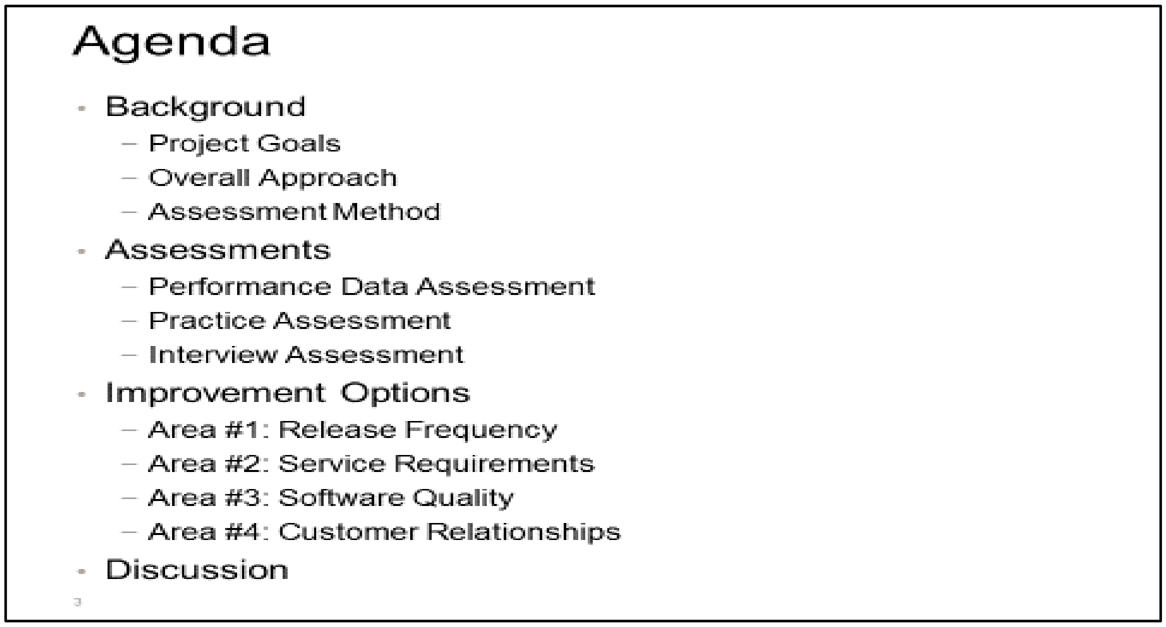


Purpose

- Present key findings from assessment
- Identify areas for further improving Secure-on-Request practices
- Discuss viewpoints between participants
- Create basis for deciding on specific actions

- Not to highlight Secure-on-Request team achievement over past months
- Not to provide specific solutions
- Not to make decisions about priorities

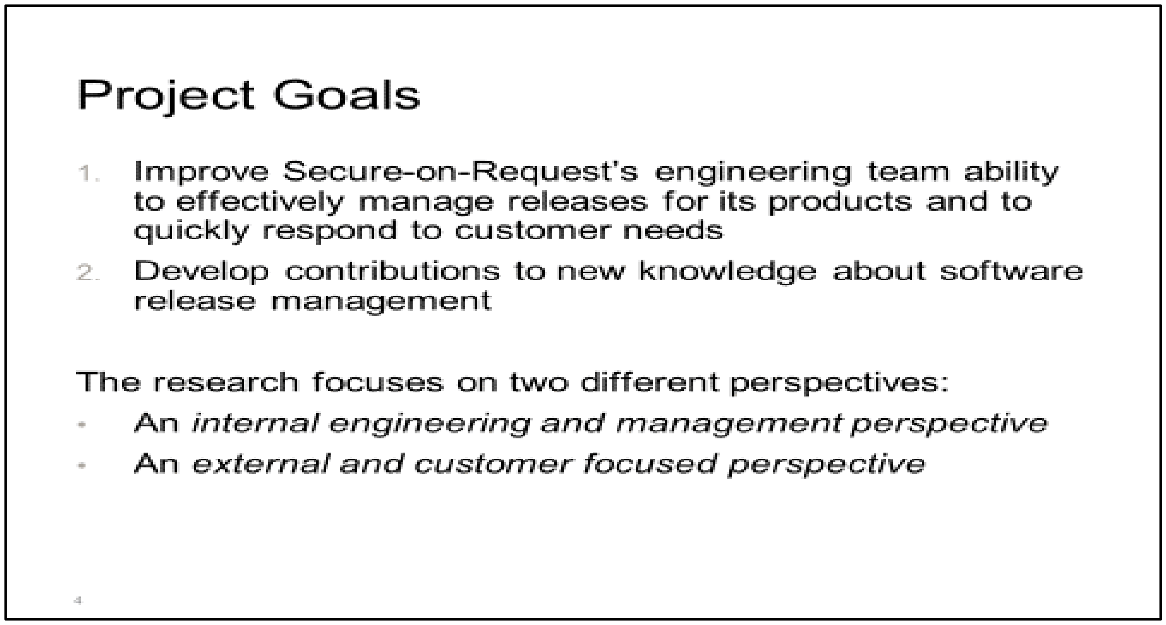
2



Agenda

- Background
  - Project Goals
  - Overall Approach
  - Assessment Method
- Assessments
  - Performance Data Assessment
  - Practice Assessment
  - Interview Assessment
- Improvement Options
  - Area #1: Release Frequency
  - Area #2: Service Requirements
  - Area #3: Software Quality
  - Area #4: Customer Relationships
- Discussion

3



Project Goals

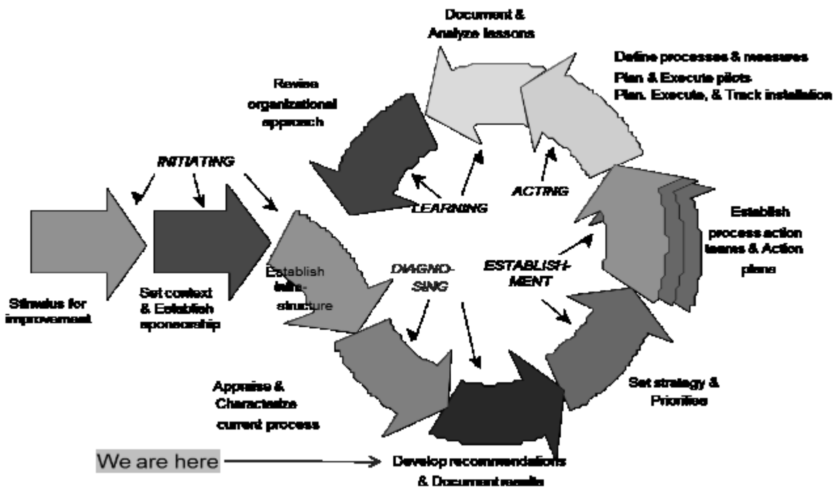
1. Improve Secure-on-Request's engineering team ability to effectively manage releases for its products and to quickly respond to customer needs
2. Develop contributions to new knowledge about software release management

The research focuses on two different perspectives:

- An *internal engineering and management perspective*
- An *external and customer focused perspective*

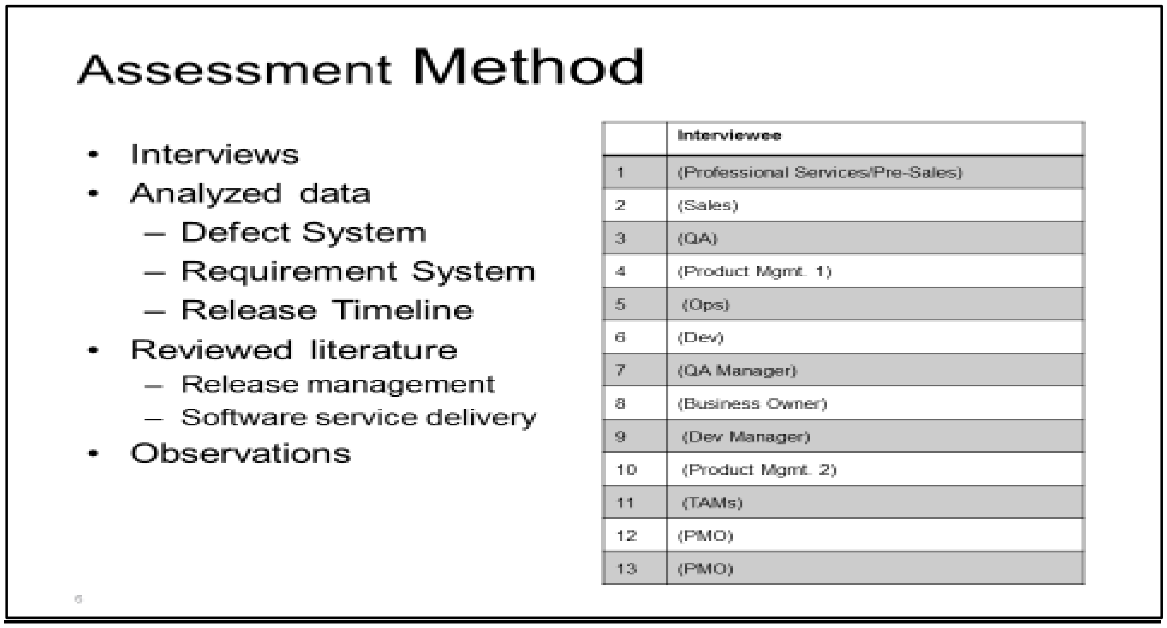
4

Assessment Context


Document & Analyze lessons
Define processes & measures
Plan & Execute pilots
Plan. Execute, & Track installation
Revise organizational approach
INITIATING
LEARNING
ACTING
Establish process action teams & Action plans
Establish infra-structure
DIAGNO-SING
ESTABLISH-MENT
Stimulus for improvement
Set context & Establish sponsorship
Set strategy & Prioritize
Appraise & Characterize current process
We are here
Develop recommendations & Document results

This research is based on the IDEAL model, a software process improvement framework by the Software Engineering Institute (SEI)

5



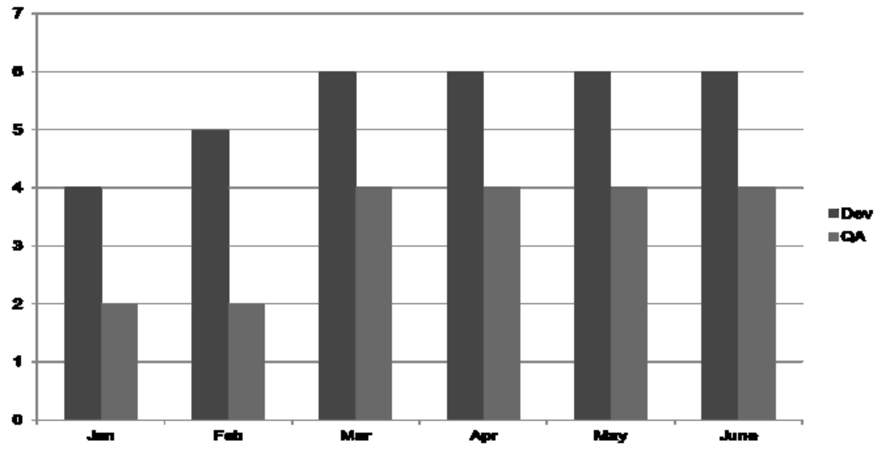
Assessment Method

- Interviews
- Analyzed data
  - Defect System
  - Requirement System
  - Release Timeline
- Reviewed literature
  - Release management
  - Software service delivery
- Observations

	Interviewee
1	(Professional Services/Pre-Sales)
2	(Sales)
3	(QA)
4	(Product Mgmt. 1)
5	(Ops)
6	(Dev)
7	(QA Manager)
8	(Business Owner)
9	(Dev Manager)
10	(Product Mgmt. 2)
11	(TAMs)
12	(PMO)
13	(PMO)

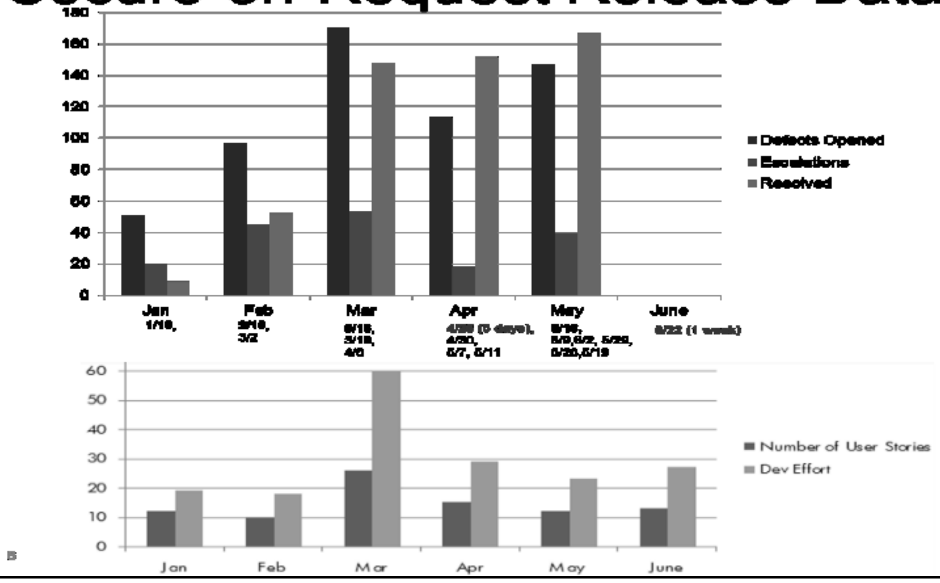
6

Secure-on-Request Engineering Team Resources

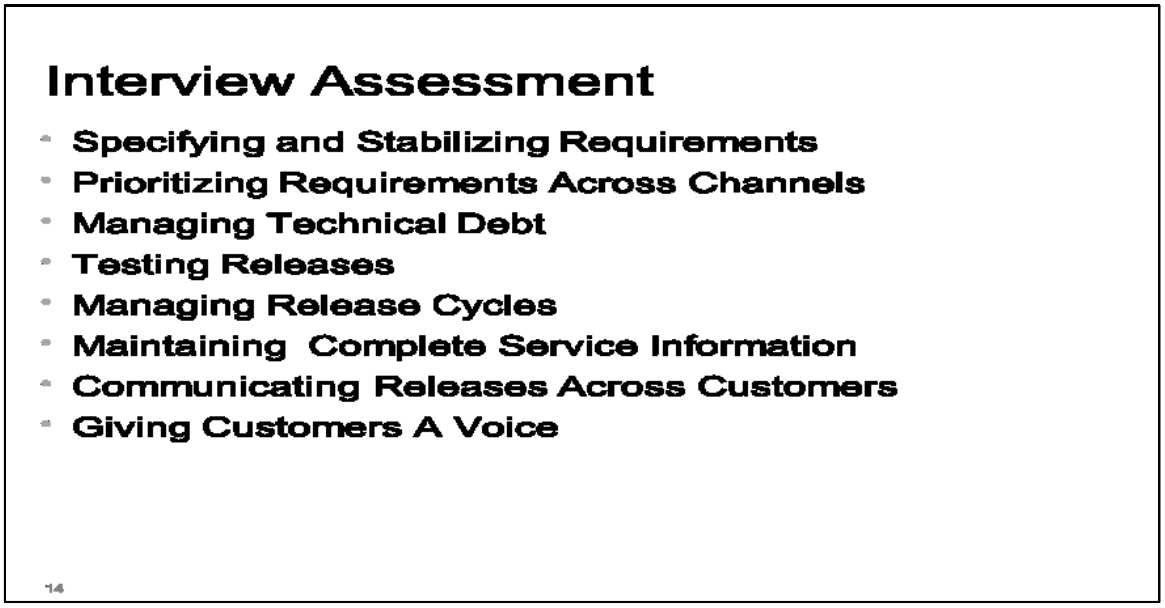


7

Secure-on-Request Release Data



13



Interview Assessment

- Specifying and Stabilizing Requirements
- Prioritizing Requirements Across Channels
- Managing Technical Debt
- Testing Releases
- Managing Release Cycles
- Maintaining Complete Service Information
- Communicating Releases Across Customers
- Giving Customers A Voice

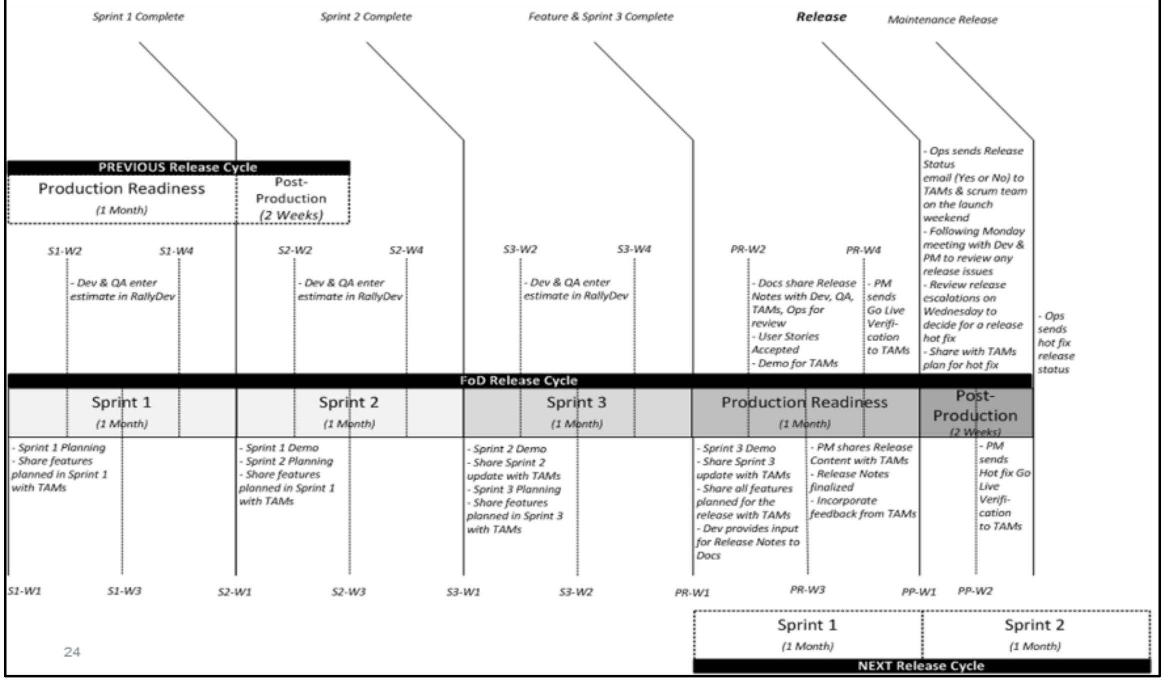
14

Improvement Options

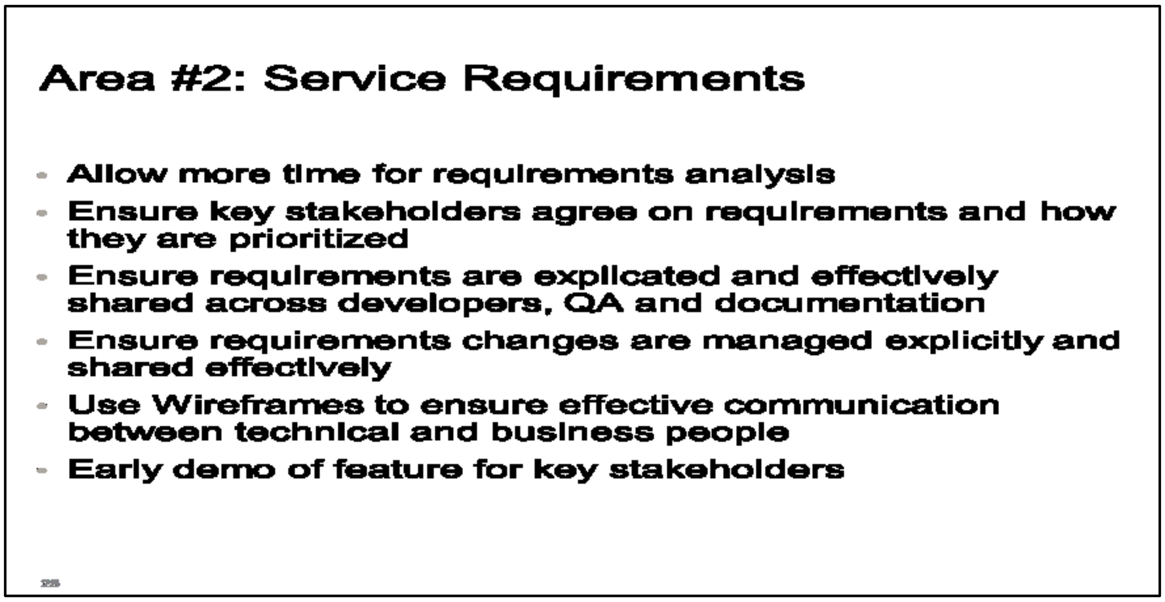
- **Area #1: Release Frequency**
- **Area #2: Service Requirements**
- **Area #3: Software Quality**
- **Area #4: Customer Relationships**

23

Area #1: Release Frequency (90 day model)


Sprint 1 Complete
Sprint 2 Complete
Feature & Sprint 3 Complete
Release
Maintenance Release
PREVIOUS Release Cycle
Production Readiness (1 Month)
Post-Production (2 Weeks)
S1-W2
S1-W4
S2-W2
S2-W4
S3-W2
S3-W4
PR-W2
PR-W4
- Dev & QA enter estimate in RallyDev
- Dev & QA enter estimate in RallyDev
- Dev & QA enter estimate in RallyDev
- Docs share Release Notes with Dev, QA, TAMs, Ops for review - User Stories Accepted - Demo for TAMs
- PM sends Go Live Verifi-cation to TAMs
- Ops sends Release Status email (Yes or No) to TAMs & scrum team on the launch weekend - Following Monday meeting with Dev & PM to review any release issues - Review release escalations on Wednesday to decide for a release hot fix - Share with TAMs plan for hot fix
- Ops sends hot fix release status
FoD Release Cycle
Sprint 1 (1 Month)
Sprint 2 (1 Month)
Sprint 3 (1 Month)
Production Readiness (1 Month)
Post-Production (2 Weeks)
- Sprint 1 Planning - Share features planned in Sprint 1 with TAMs
- Sprint 1 Demo - Sprint 2 Planning - Share features planned in Sprint 1 with TAMs
- Sprint 2 Demo - Share Sprint 2 update with TAMs - Sprint 3 Planning - Share features planned in Sprint 3 with TAMs
- Sprint 3 Demo - Share Sprint 3 update with TAMs - Share all features planned for the release with TAMs - Dev provides input for Release Notes to Docs
- PM shares Release Content with TAMs - Release Notes finalized - Incorporate feedback from TAMs
- PM sends Hot fix Go Live Verifi-cation to TAMs
S1-W1
S1-W3
S2-W1
S2-W3
S3-W1
S3-W2
PR-W1
PR-W3
PP-W1
PP-W2
Sprint 1 (1 Month)
Sprint 2 (1 Month)
NEXT Release Cycle

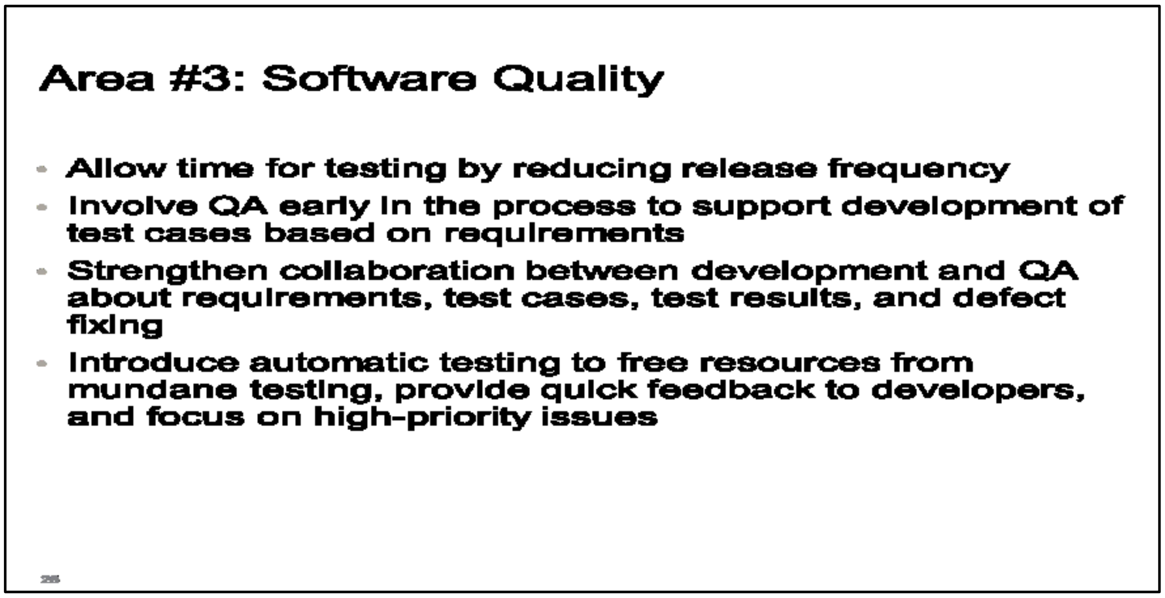
24



Area #2: Service Requirements

- Allow more time for requirements analysis
- Ensure key stakeholders agree on requirements and how they are prioritized
- Ensure requirements are explicated and effectively shared across developers, QA and documentation
- Ensure requirements changes are managed explicitly and shared effectively
- Use Wireframes to ensure effective communication between technical and business people
- Early demo of feature for key stakeholders

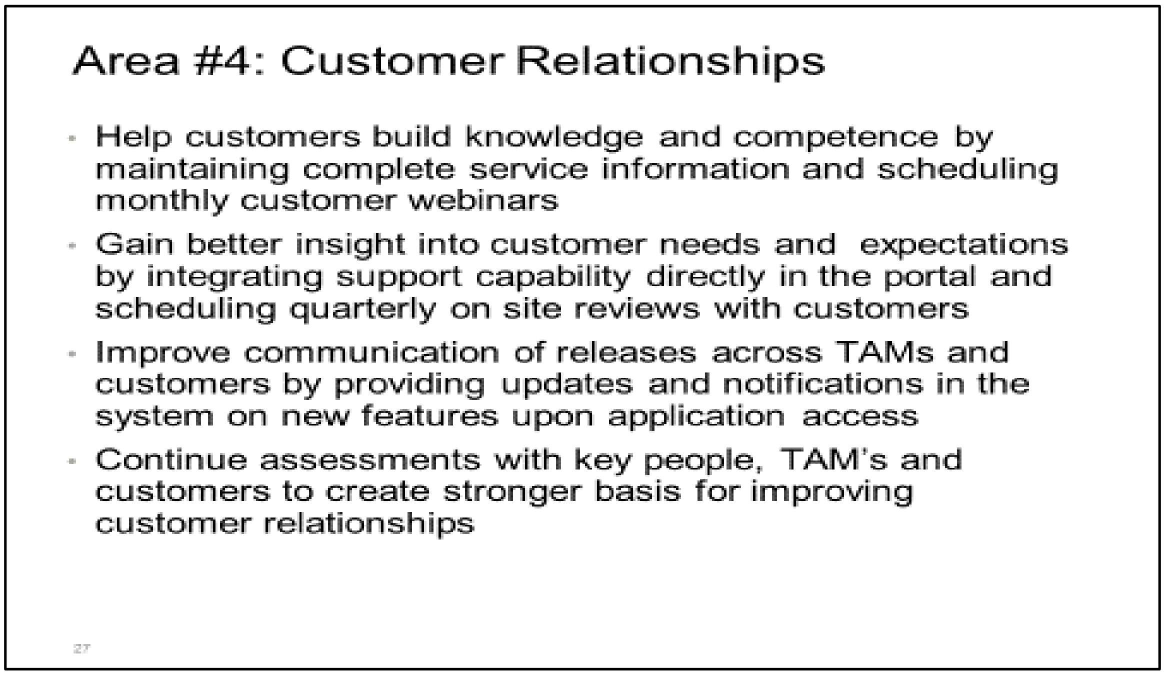
25



Area #3: Software Quality

- Allow time for testing by reducing release frequency
- Involve QA early in the process to support development of test cases based on requirements
- Strengthen collaboration between development and QA about requirements, test cases, test results, and defect fixing
- Introduce automatic testing to free resources from mundane testing, provide quick feedback to developers, and focus on high-priority issues

26

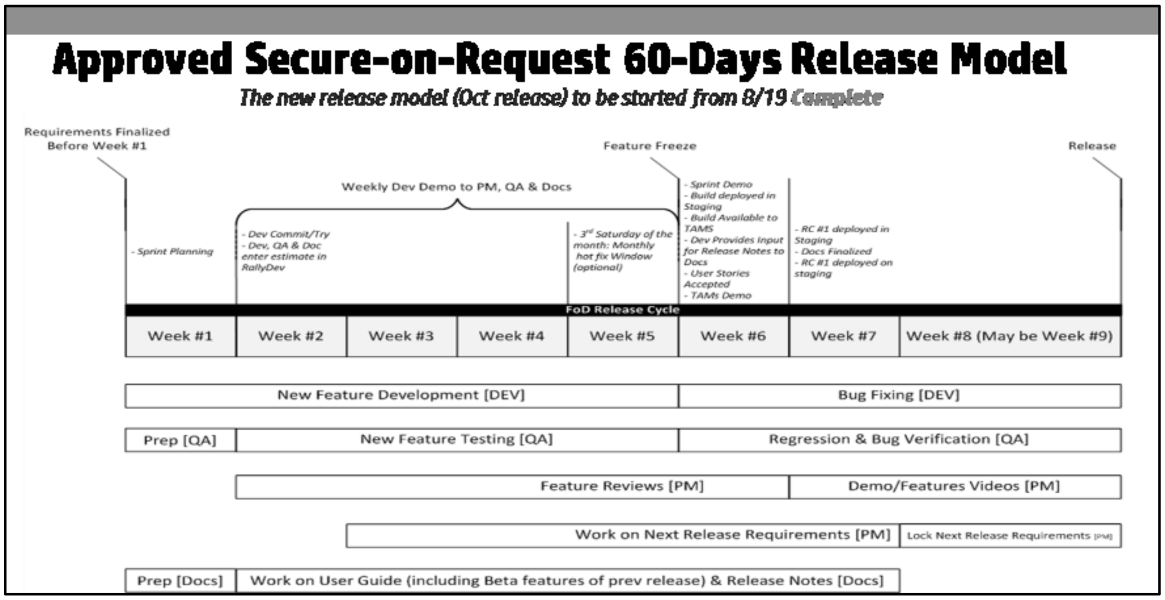


Area #4: Customer Relationships

- Help customers build knowledge and competence by maintaining complete service information and scheduling monthly customer webinars
- Gain better insight into customer needs and expectations by integrating support capability directly in the portal and scheduling quarterly on site reviews with customers
- Improve communication of releases across TAMs and customers by providing updates and notifications in the system on new features upon application access
- Continue assessments with key people, TAM's and customers to create stronger basis for improving customer relationships

27

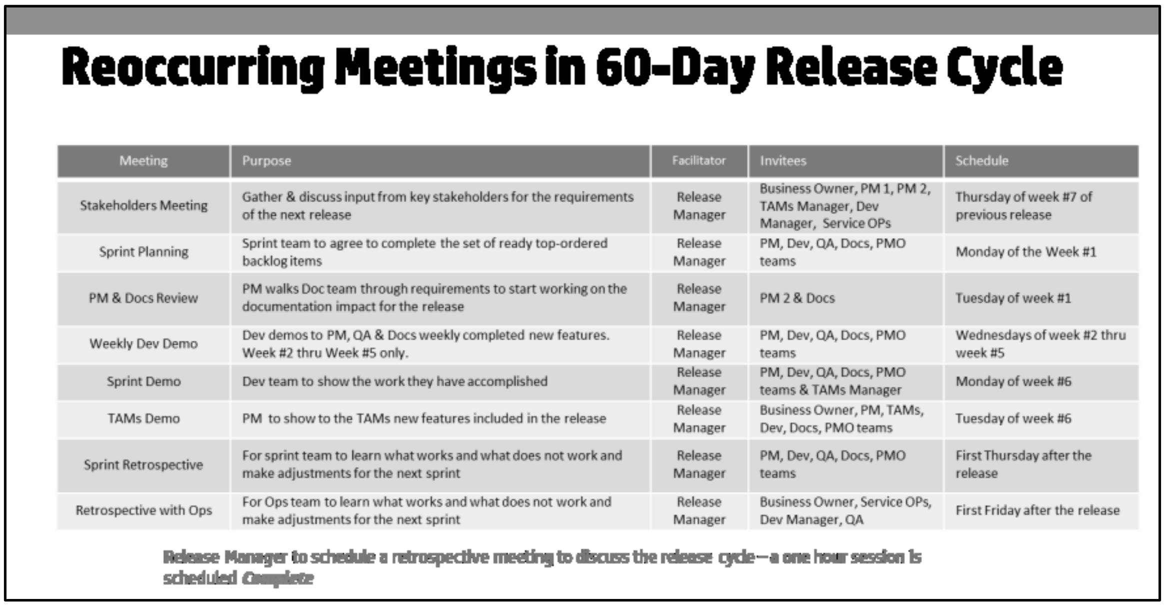
APPENDIX D: SECURE-ON-REQUEST NEW RELEASE CYCLE MODEL



Approved Secure-on-Request 60-Days Release Model

The new release model (Oct release) to be started from 8/19 **Complete**

Requirements Finalized Before Week #1
Feature Freeze
Release
Weekly Dev Demo to PM, QA & Docs
- Sprint Planning
- Dev Commit/Try
- Dev, QA & Doc enter estimate in RallyDev
- 3rd Saturday of the month: Monthly hot fix Window (optional)
- Sprint Demo
- Build deployed in Staging
- Build Available to TAMS
- Dev Provides Input for Release Notes to Docs
- User Stories Accepted
- TAMs Demo
- RC #1 deployed in Staging
- Docs Finalized
- RC #1 deployed on staging
FoD Release Cycle
Week #1
Week #2
Week #3
Week #4
Week #5
Week #6
Week #7
Week #8 (May be Week #9)
New Feature Development [DEV]
Bug Fixing [DEV]
Prep [QA]
New Feature Testing [QA]
Regression & Bug Verification [QA]
Feature Reviews [PM]
Demo/Features Videos [PM]
Work on Next Release Requirements [PM]
Lock Next Release Requirements [PM]
Prep [Docs]
Work on User Guide (including Beta features of prev release) & Release Notes [Docs]

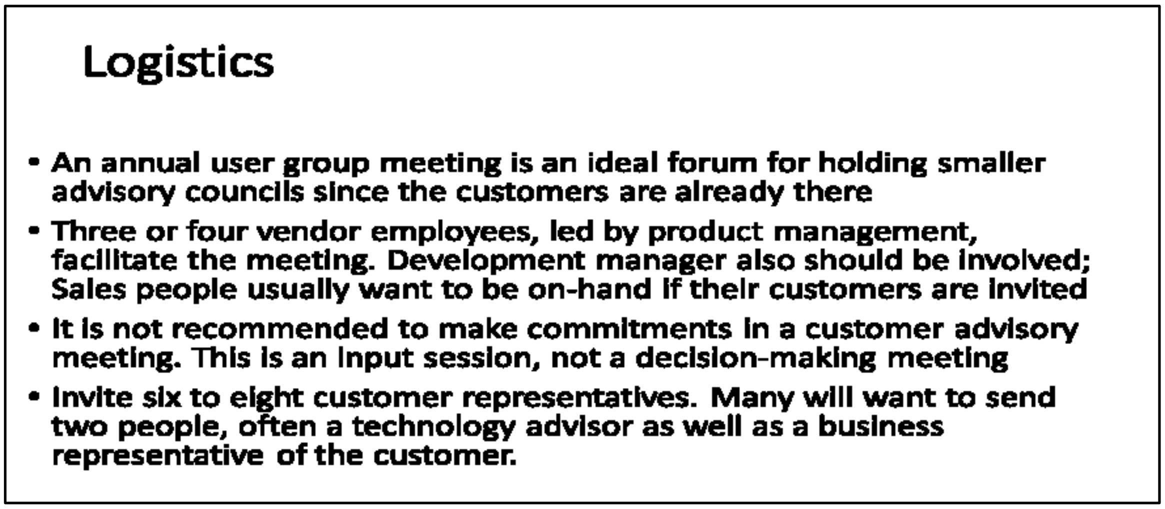


Reoccurring Meetings in 60-Day Release Cycle

Meeting	Purpose	Facilitator	Invitees	Schedule
Stakeholders Meeting	Gather & discuss input from key stakeholders for the requirements of the next release	Release Manager	Business Owner, PM 1, PM 2, TAMs Manager, Dev Manager, Service OPs	Thursday of week #7 of previous release
Sprint Planning	Sprint team to agree to complete the set of ready top-ordered backlog items	Release Manager	PM, Dev, QA, Docs, PMO teams	Monday of the Week #1
PM & Docs Review	PM walks Doc team through requirements to start working on the documentation impact for the release	Release Manager	PM 2 & Docs	Tuesday of week #1
Weekly Dev Demo	Dev demos to PM, QA & Docs weekly completed new features. Week #2 thru Week #5 only.	Release Manager	PM, Dev, QA, Docs, PMO teams	Wednesdays of week #2 thru week #5
Sprint Demo	Dev team to show the work they have accomplished	Release Manager	PM, Dev, QA, Docs, PMO teams & TAMs Manager	Monday of week #6
TAMs Demo	PM to show to the TAMs new features included in the release	Release Manager	Business Owner, PM, TAMs, Dev, Docs, PMO teams	Tuesday of week #6
Sprint Retrospective	For sprint team to learn what works and what does not work and make adjustments for the next sprint	Release Manager	PM, Dev, QA, Docs, PMO teams	First Thursday after the release
Retrospective with Ops	For Ops team to learn what works and what does not work and make adjustments for the next sprint	Release Manager	Business Owner, Service OPs, Dev Manager, QA	First Friday after the release

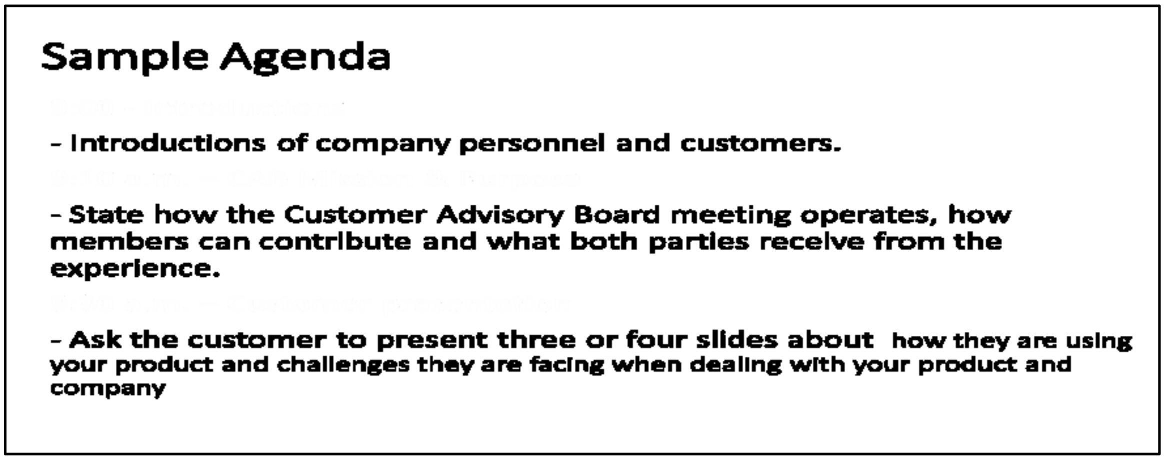
Release Manager to schedule a retrospective meeting to discuss the release cycle—a one hour session is scheduled *Complete*

APPENDIX E: CUSTOMER ADVISORY BOARD - MEETING ITEMS



Logistics

- An annual user group meeting is an ideal forum for holding smaller advisory councils since the customers are already there
- Three or four vendor employees, led by product management, facilitate the meeting. Development manager also should be involved; Sales people usually want to be on-hand if their customers are invited
- It is not recommended to make commitments in a customer advisory meeting. This is an input session, not a decision-making meeting
- Invite six to eight customer representatives. Many will want to send two people, often a technology advisor as well as a business representative of the customer.



Sample Agenda

8:00 – Introductions

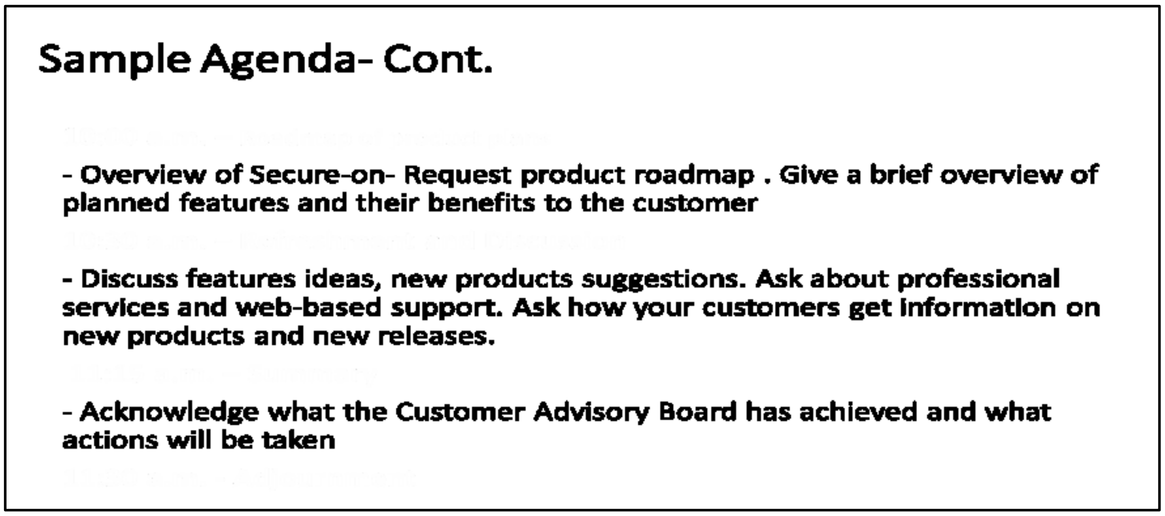
- Introductions of company personnel and customers.

8:10 a.m. – CAB Mission & Purpose

- State how the Customer Advisory Board meeting operates, how members can contribute and what both parties receive from the experience.

8:30 a.m. – Customer presentation

- Ask the customer to present three or four slides about how they are using your product and challenges they are facing when dealing with your product and company



Sample Agenda- Cont.

10:00 a.m. – Roadmap of product plans

- Overview of Secure-on- Request product roadmap . Give a brief overview of planned features and their benefits to the customer

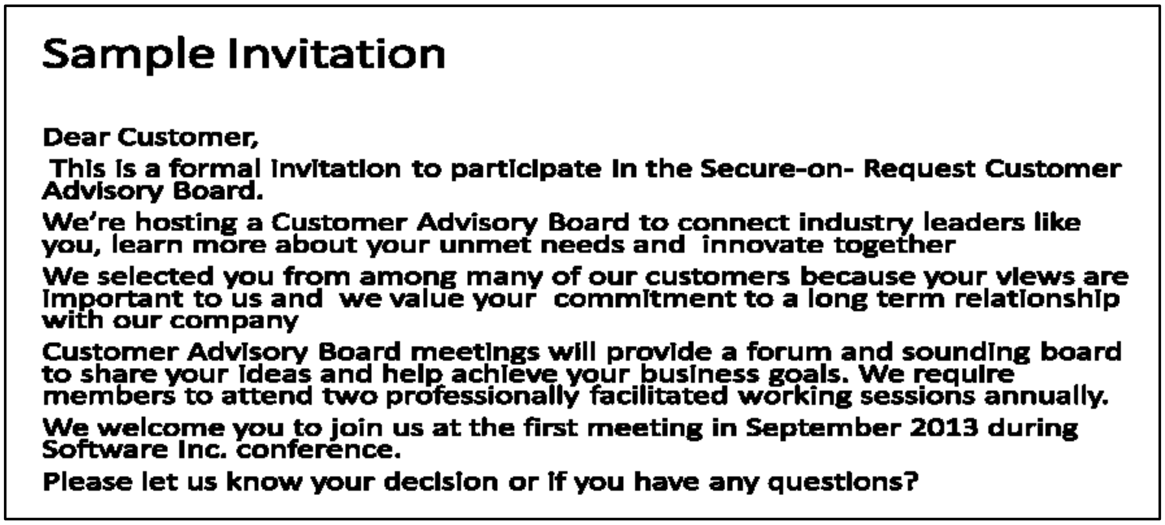
10:30 a.m. – Refreshment and Discussion

- Discuss features ideas, new products suggestions. Ask about professional services and web-based support. Ask how your customers get information on new products and new releases.

11:15 a.m. – Summary

- Acknowledge what the Customer Advisory Board has achieved and what actions will be taken

11:30 a.m. – Adjournment



Sample Invitation

Dear Customer,

This is a formal invitation to participate in the Secure-on- Request Customer Advisory Board.

We're hosting a Customer Advisory Board to connect industry leaders like you, learn more about your unmet needs and innovate together

We selected you from among many of our customers because your views are important to us and we value your commitment to a long term relationship with our company

Customer Advisory Board meetings will provide a forum and sounding board to share your ideas and help achieve your business goals. We require members to attend two professionally facilitated working sessions annually.

We welcome you to join us at the first meeting in September 2013 during Software Inc. conference.

Please let us know your decision or if you have any questions?

REFERENCES

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., . . . Stoica, I. (2010). A view of cloud computing. *Communications of the ACM, 53*(4), 50-58.

Avison, D., Baskerville, R., & Myers, M. (2001). Controlling action research projects. *Information technology & people, 14*(1), 28-45.

Ballintijn, G. (2005). *A case study of the release management of a health-care information system.* Paper presented at the proceedings of the IEEE International Conference on Software Maintenance, ICSM2005, Industrial Applications track.

Bardhan, I. R., Demirkan, H., Kannan, P., Kauffman, R. J., & Sougstad, R. (2010). An interdisciplinary perspective on IT services management and service science. *Journal of Management Information Systems, 26*(4), 13-64.

Barile, S., & Polese, F. (2010). Smart service systems and viable service systems: Applying systems theory to service science. *Service Science, 2*(1-2), 21-40.

Baskerville, R. L., & Myers, M. D. (2009). Fashion waves in information systems research and practice. *Mis Quarterly, 33*(4).

Baskerville, R. L., & Wood-Harper, A. T. (1996). A critical perspective on action research as a method for information systems research. *Journal of Information Technology, 11*(3), 235-246.

Benlian, A., Koufaris, M., & Hess, T. (2011). Service quality in software-as-a-service: developing the SaaS-Qual measure and examining its role in usage continuance. *Journal of Management Information Systems, 28*(3), 85-126.

Berkovich, M., Esch, S., Leimeister, J. M., & Krcmar, H. (2010). Towards Requirements Engineering for "Software as a Service". *Multikonferenz Wirtschaftsinformatik 2010*, 107.

Bitner, M. J., Ostrom, A. L., & Morgan, F. N. (2008). Service blueprinting: a practical technique for service innovation. *California Management Review, 50*(3), 66.

Bovet, D., & Martha, J. (2000). *Value nets: breaking the supply chain to unlock hidden profits*: John Wiley & Sons.

Boyatzis, R. E. (1998). *Transforming qualitative information: Thematic analysis and code development*: Sage.

Brocke, H., Hau, T., Vogedes, A., Schindlholzer, B., Uebernickel, F., & Brenner, W. (2009). *Design Rules for User-Oriented IT Service Descriptions.* Paper presented at the System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on.

Burgoon, J. K., Bonito, J. A., Ramirez, A., Dunbar, N. E., Kam, K., & Fischer, J. (2002). Testing the interactivity principle: Effects of mediation, propinquity, and verbal and nonverbal modalities in interpersonal interaction. *Journal of Communication, 52*(3), 657-677.

Chesbrough, H., & Spohrer, J. (2006). A research manifesto for services science. *Communications of the ACM, 49*(7), 35-40.

Choudhary, V. (2007a). Comparison of software quality under perpetual licensing and software as a service. *Journal of Management Information Systems, 24*(2), 141-165.

Choudhary, V. (2007b). *Software as a service: Implications for investment in software development.* Paper presented at the System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on.

Coghian, D. (2001). Insider Action Research Projects Implications for Practising Managers. *Management Learning, 32*(1), 49-60.

Cusumano, M. (2010). Cloud computing and SaaS as new computing platforms. *Communications of the ACM, 53*(4), 27-29.

Cusumano, M. A. (2008). The changing software business: Moving from products to services. *Computer, 41*(1), 20-27.

Danesh, A. S., Saybani, M. R., & Danesh, S. Y. S. (2011). Software release management challenges in industry: An exploratory study. *African Journal of Business Management, 5*(20), 8050-8056.

Davison, R., Martinsons, M. G., & Kock, N. (2004). Principles of canonical action research. *Information Systems Journal, 14*(1), 65-86.

Denzin, N. K., & Lincoln, Y. S. (2005). *The Sage handbook of qualitative research*: Sage.

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

Franke, N., Schreier, M., & Kaiser, U. (2010). The "I designed it myself" effect in mass customization. *Management science, 56*(1), 125-140.

Frow, P., Payne, A., & Storbacka, K. (2011). *Co-creation: a typology and conceptual framework.* Paper presented at the Proceedings of the 2011 Anzmac conference.

FüLler, J., MüHlbacher, H., Matzler, K., & Jawecki, G. (2009). Consumer empowerment through internet-based co-creation. *Journal of Management Information Systems, 26*(3), 71-102.

Grissemann, U. S., & Stokburger-Sauer, N. E. (2012). Customer co-creation of travel services: the role of company support and customer satisfaction with the co-creation performance. *Tourism Management, 33*(6), 1483-1492.

Grönroos, C. (1982). An applied service marketing theory. *European Journal of Marketing, 16*(7), 30-41.

Gummesson, E. (1987). The new marketing—Developing long-term interactive relationships. *Long Range Planning, 20*(4), 10-20.

Guo, C. J., Sun, W., Huang, Y., Wang, Z. H., & Gao, B. (2007). *A framework for native multi-tenancy application development and management.* Paper presented at the E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, 2007. CEC/EEE 2007. The 9th IEEE International Conference on.

Herrick, D. R. (2009). *Google this!: using Google apps for collaboration and productivity.* Paper presented at the Proceedings of the 37th annual ACM SIGUCCS fall conference.

Hudli, A. V., Shivaradhya, B., & Hudli, R. V. (2009). *Level-4 SaaS applications for healthcare industry.* Paper presented at the Proceedings of the 2nd Bangalore Annual Compute Conference.

Kähkönen, A.-K., & Lintukangas, K. (2012). The underlying potential of supply management in value creation. *Journal of Purchasing and Supply Management, 18*(2), 68-75.

Karpen, I. O., Bove, L. L., & Lukas, B. A. (2012). Linking Service-Dominant Logic and Strategic Business Practice A Conceptual Model of a Service-Dominant Orientation. *Journal of Service Research, 15*(1), 21-38.

Khoshafian, S. (2006). *Service oriented enterprises*: CRC Press.

Kohlborn, T., Korthaus, A., Riedl, C., & Krcmar, H. (2009). *Service aggregators in business networks.* Paper presented at the Enterprise Distributed Object Computing Conference Workshops, 2009. EDOCW 2009. 13th.

Krishnan, M. S. (1994). *Software release management: a business perspective.* Paper presented at the Proceedings of the 1994 conference of the Centre for Advanced Studies on Collaborative research.

Lahtela, A., & Jantti, M. (2011). *Challenges and problems in release management process: A case study.* Paper presented at the Software Engineering and Service Science (ICSESS), 2011 IEEE 2nd International Conference on.

Larsson, R., & Bowen, D. E. (1989). Organization and customer: managing design and coordination of services. *Academy of Management Review, 14*(2), 213-233.

Lassila, A. (2006). Taking a service-oriented perspective on software business: How to move from product business to online service business. *IADIS International Journal on WWW/Internet, 4*(1), 70-82.

Lee, A. S., & Baskerville, R. L. (2003). Generalizing generalizability in information systems research. *Information systems research, 14*(3), 221-243.

Lewin, K. (1951). Field theory in social science: selected theoretical papers (Edited by Dorwin Cartwright.).

Lincoln, Y. S., & Guba, E. G. (1985). Establishing trustworthiness. *Naturalistic inquiry*, 289-331.

Liu, F., Guo, W., Zhao, Z. Q., & Chou, W. (2010). *SaaS integration for software cloud.* Paper presented at the Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on.

Lusch, R. F., & Nambisan, S. (2012). Service Innovation: A Service-Dominant (SD) Logic Perspective. *Retrieved August, 12*, 2012.

Lusch, R. F., & Vargo, S. L. (2006a). Service-dominant logic as a foundation for a general theory. *The Service-Dominant Logic of Marketing: Dialog, Debate, and Directions, ME Sharpe, Armonk, NY*, 406-420.

Lusch, R. F., & Vargo, S. L. (2006b). Service-dominant logic: reactions, reflections and refinements. *Marketing theory, 6*(3), 281-288.

Maglio, P. P., & Spohrer, J. (2008). Fundamentals of service science. *Journal of the Academy of Marketing Science, 36*(1), 18-20.

Mathiassen, L. (2002). Collaborative practice research. *Information Technology & People, 15*(4), 321-345.

Mathiassen, L., Chiasson, M., & Germonprez, M. (2012). Style Composition in Action Research Publication. *Mis Quarterly, 36*(2).

McFeeley, B. (1996). IDEAL: A User's Guide for Software Process Improvement: DTIC Document.

McKay, J., & Marshall, P. (2001). The dual imperatives of action research. *Information Technology & People, 14*(1), 46-59.

Mertz, S., Eschinger, C., Eid, T., Huang, H., Pang, C., & Pring, B. (2009). Market trends: Software as a service, worldwide, 2008-2013. *Gartner, Stamford, CT.*

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*: Sage Publications, Incorporated.

Myers, M. D. (2008). *Qualitative research in business & management*: SAGE Publications Limited.

Napier, N. P., Mathiassen, L., & Johnson, R. D. (2009). Combining perceptions and prescriptions in requirements engineering process assessment: an industrial case study. *Software Engineering, IEEE Transactions on, 35*(5), 593-606.

Payne, A., Storbacka, K., Frow, P., & Knox, S. (2009). Co-creating brands: diagnosing and designing the relationship experience. *Journal of Business Research, 62*(3), 379-389.

Payne, A. F., Storbacka, K., & Frow, P. (2008). Managing the co-creation of value. *Journal of the Academy of Marketing Science, 36*(1), 83-96.

Peters, L., Johnston, W., & Pressey, A. (2012). Involving clients in innovation: Exploring expectation, knowledge, and competency gaps.

Pratim Ghosh, P., & Chandy Varghese, J. (2004). Globally distributed product development using a new project management framework. *International Journal of Project Management, 22*(8), 669-678.

Ramaswamy, V., & Gouillart, F. (2010). Building the co-creative enterprise. *Harvard Business Review, 88*(10), 100-109.

Rapoport, R. N. (1970). Three dilemmas in action research with special reference to the Tavistock experience. *Human relations, 23*(6), 499-513.

Regan, W. J. (1963). The service revolution. *The Journal of Marketing*, 57-62.

Rust, R. T., & Kannan, P. (2003). E-service: a new paradigm for business in the electronic environment. *Communications of the ACM, 46*(6), 36-42.

Sääksjärvi, M., Lassila, A., & Nordström, H. (2005). *Evaluating the software as a service business model: From CPU time-sharing to online innovation sharing.* Paper presented at the IADIS International Conference e-Society.

Schmidt, R., Dengler, F., & Kieninger, A. (2010). *Co-creation of value in IT service processes using semantic mediawiki.* Paper presented at the Business Process Management Workshops.

Schneider, B., & Bowen, D. E. (2010). *Winning the service game*: Springer.

Scott, J. A., & Nisse, D. (2001). Software configuration management. *SWEBOK*, 103.

Shostack, G. L. (1977). Breaking free from product marketing. *The Journal of Marketing*, 73-80.

Singh, R., Bhagat, A., & Kumar, N. (2012). *Generalization of Software Metrics on Software as a Service (SaaS).* Paper presented at the Computing Sciences (ICCS), 2012 International Conference on.

Spohrer, J., Anderson, L., Pass, N., & Ager, T. (2008). *Service science and service-dominant logic.* Paper presented at the Otago Forum.

Spohrer, J., & Maglio, P. P. (2008). The Emergence of Service Science: Toward Systematic Service Innovations to Accelerate Co-Creation of Value. *Production and Operations Management, 17*(3), 238-246.

Spohrer, J., Maglio, P. P., Bailey, J., & Gruhl, D. (2007). Steps toward a science of service systems. *Computer, 40*(1), 71-77.

Srikanth, H., & Cohen, M. B. (2011). *Regression testing in Software as a Service: An industrial case study.* Paper presented at the Software Maintenance (ICSM), 2011 27th IEEE International Conference on.

Stuckenberg, S., & Heinzl, A. (2010). The Impact of the Software-as-a-Service Concept on the Underlying Software and Service Development Processes. *PACIS 2010 Proceedings*.

Sun, W., Zhang, K., Chen, S.-K., Zhang, X., & Liang, H. (2007). Software as a service: An integration perspective *Service-oriented computing–ICSOC 2007* (pp. 558-569): Springer.

Susarla, A., Barua, A., & Whinston, A. B. (2009). A transaction cost perspective of the" software as a service" business model. *Journal of Management Information Systems, 26*(2), 205-240.

Susman, G. I., & Evered, R. D. (1978). An assessment of the scientific merits of action research. *Administrative Science Quarterly*, 582-603.

Syed, K. (2014). Improving Recurrent Software Development: A Contextualist Inquiry into Release Cycle Management.

Tracy, S. (2012). *Service Systems and Social Enterprise: Beyond the Economics of Business.*

Van de Ven, A. H. (2007). *Engaged scholarship: A guide for organizational and social research*: OUP Oxford.

Van Der Hoek, A., Hall, R., Heimbigner, D., & Wolf, A. (1997). Software release management. *Software Engineering—ESEC/FSE'97*, 159-175.

Vargo, S. L. (2013). Service-dominant logic reframes (service) innovation. *VTT Technical Research Centre of Finland*, 7.

Vargo, S. L., & Akaka, M. A. (2009). Service-dominant logic as a foundation for service science: clarifications. *Service Science, 1*(1), 32-41.

Vargo, S. L., & Lusch, R. F. (2004). Evolving to a new dominant logic for marketing. *Journal of marketing*, 1-17.

Vargo, S. L., & Lusch, R. F. (2008). Service-dominant logic: continuing the evolution. *Journal of the Academy of Marketing Science, 36*(1), 1-10.

Vargo, S. L., Maglio, P. P., & Akaka, M. A. (2008). On value and value co-creation: A service systems and service logic perspective. *European management journal, 26*(3), 145-152.

Walker, R. H., Craig-Lees, M., Hecker, R., & Francis, H. (2002). Technology-enabled service delivery: an investigation of reasons affecting customer adoption and rejection. *International Journal of Service Industry Management, 13*(1), 91-106.

Winklhofer, H., Palmer, R. A., & Brodie, R. J. (2007). Researching the Service Dominant Logic– Normative Perspective Versus Practice. *Australasian Marketing Journal (AMJ), 15*(1), 76-83.

Wolak, R., Kalafatis, S., & Harris, P. (1998). An investigation into four characteristics of services. *Journal of Empirical Generalisations in Marketing Science, 3*(2), 22-43.

Yin, R. K. (2003). *Case study research: Design and methods* (Vol. 5): sage.

Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5): Sage.

Zeithaml, V. A., Berry, L. L., & Parasuraman, A. (1996). The behavioral consequences of service quality. *The Journal of Marketing*, 31-46.

Zeithaml, V. A., Parasuraman, A., & Berry, L. L. (1985). Problems and strategies in services marketing. *The Journal of Marketing*, 33-46.

Zhao, J. L., Tanniru, M., & Zhang, L.-J. (2007). Services computing as the foundation of enterprise agility: Overview of recent advances and introduction to the special issue. *Information Systems Frontiers, 9*(1), 1-8.