Users' Loyalty to Agile Information Systems

Am Fachbereich Rechts- und Wirtschaftswissenschaften
der Technischen Universität Darmstadt

eingereichte

Dissertation

vorgelegt von

Tillmann Grupp

geboren am 02.01.1982 in Bonn

zur Erlangung des akademischen Grades
Doctor rerum politicarum (Dr. rer. pol.)

Erstgutachter:	Prof. Dr. Alexander Benlian
Zweitgutachter:	Prof. Dr. Peter Buxmann

Darmstadt 2018

Abstract

Over the past few years, across many industrial sectors, Information Systems (IS) developed with the help of agile methods have become the rule rather than the exception. Because of their high flexibility, such Agile IS development methodologies help firms to keep pace with emerging market requirements. At the same time, customers are also gaining increasing market power due to an expanding digitalization of services and products, which decreases switching barriers and increases transparency. As a result, it has become crucial for firms to develop IS that continuously provide sufficient value to customers. This is one of the main reasons why firms regularly deliver increments of Agile IS for users to update outdated software versions. By doing so, firms try to bind and engage customers lastingly to capture current and future revenue streams and stay competitive. Agile IS and software updates (that deliver increments of Agile IS to users) have been researched thoroughly, however mostly from a technical point of view. Nevertheless, because updates change a system while it is already in use, they have the potential to impact users' beliefs, attitudes, behaviors, and in particular, loyalty to a software in the post-adoption phase. However, despite the importance of better understanding user responses to Agile IS to provide an adequate theoretical framework, research from a user's perspective on Agile IS, and especially software updates, is still scarce.

Against this backdrop, this thesis presents four empirical studies that were conducted to investigate whether and how Agile IS affect users' loyalty to IS, to identify potential moderators, and to understand how Agile IS should be designed to facilitate potential positive effects. In these studies, increments of Agile IS are operationalized as software updates and customer loyalty as a user's continuance intention with a system. By drawing on the IS Continuance Model in a scenario-based online experiment, the first two studies reveal empirically how Agile IS have the potential to increase user continuance intentions. Users of Agile IS show greater IS continuance intentions, despite that some functionality is provided only later on, as compared to a consistently feature-complete traditional IS. This effect is diminished somewhat when the software is introduced with an extensive feature set right from the beginning. Nevertheless, the size of an update does not seem to play a significant role. The second study reveals that this positive effect of updates only emerges if the user is not very knowledgeable regarding the software, because experts in contrast to novices seem to devalue Agile IS (their continuance intentions decrease with Agile IS in comparison to traditional IS). Additionally, the second study shows that the removal of features through updates reduces continuance intentions even more than the equivalent addition of features when considering the absolute magnitude of change. With empirical data from a laboratory experiment, the third study identifies update frequency and update type as further moderators of the effect, and confirms the hypothesized mediation mechanism presumed by the IS Continuance Model. The fourth study examines the role of update delivery strategies, i.e., the timing and presence of a notification and an installation choice. In this study, feature and security updates are distinguished, as both seem to have different characteristics with respect to the delivery strategy (i.e., users ‘need’ security but ‘want’ to add functionality). The findings show that both update types should be announced to users, in the case of a security update, only after successful installation, while presenting an installation choice to users prevents any positive effect for all types of updates.

Overall, this thesis highlights the importance of understanding Agile IS and software updates from the user’s perspective. First, the results show that Agile IS have the potential to affect user’s continuance intentions, thereby contributing to a comprehensive theoretical foundation on Agile IS. Also the findings put the user more at the center of investigations in IS. Second, the empirical findings provide evidence in support of a necessary fine-grained understanding of IT Artifacts as malleable compositions of specific features and characteristics. This answers the call of several researchers to put the IT Artifact more at the focus of IS research (Benbasat and Zmud 2003). Third, the results reveal that changes in IS might change users’ attitudes and behaviors over time, which extends the predominant view of IS in post-adoption literature from a mostly static to a more dynamic perspective. With this finding, we answer the call of several IS scholars to consider the evolution of IS more thoroughly (e.g., Jasperson et al. 2005; Benbasat and Barki 2007). For practitioners, the findings of this thesis provide empirically backed rationales to inform management decisions concerning the deployment of Agile IS and offer guidance on strategic or design considerations. Overall, the results show how and when the value provided by IS from a user’s perspective may be increased by the deployment of Agile IS and software updates.

Zusammenfassung

In den letzten Jahren sind Informationssysteme, die mit agilen Methoden entwickelt werden, in vielen Branchen zur Regel geworden. Solche agilen Informationssysteme (Agile IS) helfen wegen ihrer Flexibilität Firmen dabei, auf dem neuesten Stand bezüglich neu aufkommender Marktanforderungen zu bleiben. Jedoch gewinnen Kunden wegen der zunehmenden Digitalisierung von Services und Produkten und den damit sinkenden Wechselbarrieren und steigender Markttransparenz gleichzeitig immer mehr Marktmacht. In Folge dessen ist es für Firmen unabdingbar geworden, Informationssysteme zu entwickeln, welche dauerhaft genügend Wert aus Sicht des Kunden bieten. Das ist einer der Hauptgründe, warum Firmen regelmäßige Inkremente von agilen Informationssystemen ausliefern, um alte Softwareversionen auf den neuesten Stand zu bringen. Auf diesem Weg versuchen Firmen Kunden langfristig zu engagieren und zu binden, um gegenwärtige und zukünftige Einnahmeströme zu sichern und damit konkurrenzfähig zu bleiben. Agile IS und Software Updates (die Inkremente von agilen Informationssystemen an Nutzer ausliefern) wurden vielfach erforscht, dennoch meistens nur aus technischer Sicht. Da Software Updates jedoch ein Informationssystem verändern, während es benutzt wird, können Software Updates möglicherweise auch die Überzeugungen, Einstellungen und Verhaltensweisen von Nutzern und insbesondere die Loyalität zu einer Software in der Post-Adoptionsphase beeinflussen. Obwohl es deshalb immens wichtig ist, besser zu verstehen, wie Nutzer auf Agile IS reagieren, um ein zulängliches theoretisches Gerüst zu schaffen, gibt es jedoch nur wenig Forschung aus der Nutzerperspektive zu agilen Informationssystemen und insbesondere Software Updates.

Vor diesem Hintergrund zeigt diese Dissertation vier empirische Studien auf, welche durchgeführt wurden, um zu ergründen, ob und wie Agile IS die Loyalität von Nutzern bezüglich Informationssystemen beeinflussen können, welche möglichen Moderatoren für einen Effekt existieren und wie Agile IS gestaltet werden sollten, um mögliche positive Effekte zu fördern. In den vorliegenden Studien werden Inkremente von agilen Informationssystemen durch Software Updates und die Loyalität von Nutzern durch die Weiternutzungsabsicht bezüglich eines Systems operationalisiert. Die ersten beiden Studien zeigen empirisch anhand des IS-Continuance-Modells in einem szenariobasierten Online-Experiment, dass Agile IS das Potenzial haben, die Weiternutzungsabsicht des Nutzers zu erhöhen. Es zeigt sich ein positiver Effekt, obwohl erst später Funktionalitäten im Vergleich zu durchgängig funktionsvollständigen traditionellen IS bereitgestellt werden. Dieser Effekt wird etwas abgeschwächt, wenn die Anfangsausstattung der Software bezüglich Features bereits sehr umfangreich ist. Der Umfang des Updates selbst scheint jedoch keinen wesentlichen Einfluss auf den Effekt zu nehmen. Die zweite Studie zeigt, dass die positive Wirkung von Updates nur dann auftritt, wenn sich der Nutzer weniger gut mit der Software auskennt, da im Gegensatz zu Neulingen in diesem Fall Experten Agile IS schlechter bewerten (ihre Weiternutzungsabsicht sinkt bei agilen IS im Vergleich zu traditionellen IS). Zusätzlich zeigt die zweite Studie, dass das Entfernen von Features durch Updates die Weiternutzungsabsicht absolut betrachtet sogar stärker reduziert, als ein entsprechender Zugewinn an Features. Mit empirischen Daten aus einem Laborexperiment identifiziert die dritte Studie die Häufigkeit von Updates und die Art des Updates als weitere Moderatoren des Effekts und bestätigt den als Hypothese vermuteten Mediationsmechanismus, der auf Basis des IS-Continuance-Modell unterstellt wird. Die vierte Studie untersucht, welche Rolle die Bereitstellungsstrategie eines Updates spielt. Es wird geprüft, ob der Effekt auch dadurch beeinflusst wird, ob und wann eine Benachrichtigung zu Updates gegeben wird und ob es eine Wahlmöglichkeit zur Installation gibt. In dieser Studie werden Feature- und Sicherheitsupdates unterschieden, da beide unterschiedliche Eigenschaften zu besitzen scheinen (Im Allgemeinen „benötigen“ Nutzer Sicherheit, aber „wollen“ Funktionalitäten erhalten). Die Ergebnisse zeigen, dass beide Arten von Updates den Nutzern kommuniziert werden sollten, jedoch im Falle eines Sicherheitsupdates erst nach erfolgreicher Installation, während eine Wahlmöglichkeit zur Installation einen positiven Effekt auf Nutzer für beide Arten von Updates verhindert.

Insgesamt zeigt die Dissertation, wie wichtig es ist, Agile IS und Software-Updates aus der Sicht des Nutzers zu verstehen. Erstens zeigen die Ergebnisse, dass Agile IS das Potenzial haben, die Weiternutzungsabsichten von Nutzern zu beeinflussen, was mit Blick auf das Ziel, möglichst vollständige theoretische Grundlagen zu Agilen IS zu schaffen, berücksichtigt werden muss. Gleichzeitig wird der Nutzer durch die Erkenntnisse wieder mehr in den Mittelpunkt der IS-Forschung gerückt. Zweitens liefern die empirischen Befunde Hinweise auf ein notwendiges feingranulares Verständnis von IT-Artefakten als formbare Kompositionen aus spezifischen Funktionen und Eigenschaften. Dieses Ergebnis folgt den Forderungen mehrerer Forscher, das IT-Artefakt stärker in den Fokus der IS-Forschung zu rücken (Benbasat und Zmud 2003). Drittens zeigen die Ergebnisse, dass Veränderungen in einem Informationssystem die Einstellungen und Verhaltensweisen der Nutzer im Laufe der Zeit verändern können, was die vorherrschende eher statische Sichtweise auf Informationssystemen in der Post-Adoptions-Literatur in Richtung einer dynamischeren

Perspektive erweitert. Mit diesem Ergebnis wird der Forderung Rechnung getragen, dynamische Veränderung von Informationssystemen in der IS-Forschung gründlicher zu betrachten (Jasperson et al. 2005; Benbasat und Barki 2007). Für die Praxis schaffen die Ergebnisse dieser Dissertation stichhaltige Argumente, um Managemententscheidungen bezüglich des Einsatzes von Agilen IS zu begünstigen und bieten gleichzeitig Leitlinien zu strategischen oder Design-Überlegungen in Bezug auf Agile IS. Die Ergebnisse zeigen insgesamt, wie und wann der Wert von Informationssystemen aus der Sicht des Nutzers durch den Einsatz von Agilen IS und Software Updates erhöht werden kann.

Table of Contents

Abstract ..... **I**

Zusammenfassung ..... **III**

Table of Contents ..... **VII**

List of Tables ..... **XI**

List of Figures ..... **XIII**

List of Abbreviations ..... **XV**

Chapter 1: Introduction **1**

1.1 Motivation and Research Questions 1

1.2 Theoretical Foundations 5

1.2.1 Agile IS 5

1.2.2 Non-rational user 8

1.2.3 Software Updates and the IS Continuance Model 10

1.3 Thesis Positioning 14

1.4 Structure of the Thesis 16

Chapter 2: Effects of Updates and the Role of Update Size and Prior Endowment **23**

2.1 Introduction 24

2.2 Theoretical Foundations 26

2.2.1 Feature Updates 26

2.2.2 Information Systems Continuance 27

2.3 Hypothesis Development 28

2.3.1 The Effect of Feature Updates on Users’ Continuance Intentions 30

2.3.2 The Role of Initial Feature Endowment 31

2.3.3 The Role of Update Size 32

2.3.4 The Mediating Effect of Disconfirmation 33

2.4 Method 33

2.4.1 Experimental Design 33

2.4.2 Manipulation of Independent Variables 36

2.4.3 Measures 38

2.4.4 Participants, Incentives and Procedures 39

2.5 Data Analysis and Results 40

2.5.1 Control Variables and Manipulation Check 40

2.5.2 Measurement Validation 41

2.5.3 Hypotheses Testing 42
2.6 Discussion 44
2.6.1 Implications for Research 46
2.6.2 Implications for Practice 47
2.6.3 Limitations and Future Research 48
2.6.4 Conclusion 49
Chapter 3: Effects of Gains or Losses through Updates on Experts or Novices 51
3.1 Introduction 52
3.2 Theoretical Foundations 54
3.2.1 Software Updates 54
3.2.2 Information Systems Continuance 56
3.2.3 Information Systems Expertise 57
3.3 Hypothesis Development 59
3.3.1 IS Novices’ Response to Gaining a Feature through a Software Update 59
3.3.2 IS Experts’ Response to Gaining a Feature through a Software Update 61
3.3.3 Novices’ and Experts’ Response to Losing a Feature through a Software Update 61
3.3.4 The Mediating Effect of Disconfirmation 62
3.4 Method 63
3.4.1 Experimental Design 63
3.4.2 Manipulation of Independent Variables 64
3.4.3 Measures 66
3.4.4 Participants, Incentives and Procedures 67
3.5 Data Analysis and Results 67
3.5.1 Control Variables 67
3.5.2 Measurement Validation 68
3.5.3 Hypotheses Testing 69
3.6 Discussion 71
3.6.1 Implications for Research 73
3.6.2 Implications for Practice 74
3.6.3 Limitations and Future Research 75
3.6.4 Conclusion 76
Chapter 4: Updates and the Role of Update Frequency and Update Type 77
4.1 Introduction 78

4.2 Theoretical Foundations ... 80
4.2.1 Software Updates ... 80
4.2.2 Information Systems Continuance ... 81
4.3 Hypotheses Development ... 83
4.3.1 The Effect of Feature Updates on Users' Continuance Intentions ... 83
4.3.2 The Role of Frequency in the Delivery of Feature Updates ... 84
4.3.3 The Effect of Non-Feature Updates on Users' Continuance Intentions ... 86
4.3.4 The Role of Frequency in the Delivery of Non-Feature Updates ... 87
4.3.5 The Mediating Roles of Disconfirmation, Perceived Usefulness and Satisfaction 87
4.4 Method ... 89
4.4.1 Experimental Design ... 89
4.4.2 Manipulation of Independent Variables ... 90
4.4.3 Measures ... 93
4.4.4 Participants, Incentives and Procedures ... 94
4.5 Data Analysis and Results ... 95
4.5.1 Control Variables and Manipulation Check ... 95
4.5.2 Measurement Validation ... 96
4.5.3 Hypotheses Testing ... 98
4.6 Discussion ... 102
4.6.1 Implications for Research ... 104
4.6.2 Implications for Practice ... 105
4.6.3 Limitations and Future Research ... 106
4.6.4 Conclusion ... 107
Chapter 5: Updates and the Role of Delivery Strategy and Update Type ... 109
5.1 Introduction ... 110
5.2 Theoretical Foundations ... 112
5.2.1 Feature Updates and Security Updates ... 112
5.2.2 Information Systems Continuance ... 113
5.3 Hypotheses Development ... 114
5.3.1 Effects of Notifications for Security Updates ... 114
5.3.2 Effects of Notifications for Feature Updates ... 115
5.3.3 Effects of Non-Mandatory Security and Feature Updates ... 116
5.4 Method ... 117

5.4.1 Experimental Design ... 117
5.4.2 Manipulation of Independent Variables ... 119
5.4.3 Dependent variables, Control Variables and Manipulation Checks ... 120
5.4.4 Participants, Incentives and Procedures ... 121
5.5 Data Analysis and Results ... 122
5.5.1 Control Variables and Manipulation Check ... 122
5.5.2 Measurement Validation ... 122
5.5.3 Hypotheses Testing ... 123
5.6 Discussion ... 125
5.6.1 Implications for Research ... 126
5.6.2 Implications for Practice ... 127
5.6.3 Conclusion, Limitations, and Future Research ... 128
Chapter 6: Thesis Conclusion and Contributions ... 131
6.1 Theoretical Contributions ... 132
6.2 Practical Contributions ... 134
6.3 Limitations and Future Research ... 136
References ... 139
Eidesstattliche Erklärung ... 157
Appendix ... 159

List of Tables

Table 1-1: Overview of Update Types 12
Table 1-2: Overview of Articles 17
Table 2-1: Results of Confirmatory Factor Analysis for Core Variables 41
Table 2-2: Means, Standard Deviations and Correlation Matrix for Core Variables 42
Table 3-1: Results of Confirmatory Factor Analysis for Core Variables 68
Table 3-2: Means, Standard Deviations, and Correlation Matrix for Core Variables 69
Table 3-3: Means, Mean Differences and Significance Levels for Continuance Intention 70
Table 3-4: Results from Serial Multiple Mediation Analysis of Novices in Groups B and C (Bootstrapping Results for Indirect Paths) 71
Table 4-1: Results of Confirmatory Factor Analysis for Core Variables 97
Table 4-2: Means, Standard Deviations, and Correlation Matrix for Core Variables 98
Table 4-3: Mean Values for Dependent Variables 98
Table 4-4: Mean Differences from Baseline (No Updates, Control Group A) and Significance Levels 99
Table 4-5: Direct Comparisons of Update Types 99
Table 4-6: Direct Comparisons of Update Frequencies 99
Table 4-7: Results from Serial Multiple Mediation Analysis, Groups A and E 101
Table 5-1: Experimental design and experimental groups (N: notification, F: feature added) 119
Table 5-2: Results of confirmatory factor analysis for core variables 122
Table 5-3: Mean values, differences and significance levels for security update groups 124
Table 5-4: Mean values, differences and significance levels for feature update groups 125

List of Figures

Figure 1-1: Thesis Positioning 14
Figure 1-2: Overarching Article Contributions 16
Figure 1-3: Research Framework and Specific Article Contributions 17
Figure 2-1: Experimental Procedure 35
Figure 2-2: Average Continuance Intentions, Mean Differences and Significance Levels 42
Figure 2-3: Feature Updates Disconfirming prior Expectations Regarding Software 44
Figure 3-1: Experimental Setup, Groups, and Treatments 64
Figure 3-2: Expert and Novice Responses to Gaining and Loosing Features from an Update 70
Figure 3-3: Mediation Mechanism Behind Novices' Positive Response to Gaining a Feature through an Update 71
Figure 4-1: Research Model 88
Figure 4-2: Experimental Setup, Groups, and Treatments 90
Figure 4-3: Sample Screenshots of Text Editor. 92
Figure 4-4: Mediation Analysis for Groups A and E 101
Figure 5-1: IS Continuance Model (Following Bhattacherjee, 2001) 113
Figure 5-2: Sample screens of app with no, post-, pre-notification, and additional choice (l.t.r.) 119
Figure 5-3: Mean values, differences and significance levels for CI between groups 123

List of Abbreviations

ANOVA	Analysis of Variance
AVE	Average Variance Extracted
CFA	Confirmatory Factor Analysis
CI	Continuance Intentions
DevOps	Development-Operations
DISC	Positive Disconfirmation
ECT	Expectation-Confirmation Theory
EVM	Experimental Vignette Methodology
IS	Information Systems
IT	Information Technology
ISD	Information Systems Development
LLCI	Lower Limit of Confidence Interval
MANOVA	Multivariate Analysis of Variance
PEoU	Perceived Ease of Use
PU	Perceived Usefulness
RQ	Research question
SAT	Satisfaction
SD	Standard Deviation
SE	Standard Error
TAM	Technology Acceptance Model
TBP	Travel Booking Platform

ULCI	Upper Limit of Confidence Interval
XP	eXtreme Programming

Chapter 1: Introduction

1.1 Motivation and Research Questions

Worldwide, firms are continuously striving for more agility. Due to the ever increasing rate of technological progress and increased market transparency, firms need to keep pace with shifting user requirements in order to maintain relevance. For example, Microsoft who formerly produced monolithic operating system versions every few years, in a huge effort over a five-year period has transformed itself into an agile company that now produces a more flexible and evolving operating system that is frequently updated (Denning 2015a; Denning 2015b). This transformation has allowed them to stay relevant in the market, especially with respect to the emergence of new digital ecosystems, such as the mobile operating systems iOS and Android. More recently, Microsoft along with many other firms is striving to design the update process as unobtrusive and convenient for users as possible (e.g., Bowden 2017).

To account for these changes, over the past few decades, the understanding of markets as balanced places where supply meets demand has changed. In a more recent understanding, the customer has noticeably gained in power, and it is assumed that the access to customers might play a much more central role than thought before. In the field of Information System Development (ISD) this is reflected in continuous efforts to become more user centered (e.g., Fowler and Highsmith 2001; Hong et al. 2011). Moreover, the ability of a firm to capture and maintain a user's attention has dramatically increased in importance (Hong et al. 2004). Many contemporary business models rely on recurring revenue streams from users or, in the beginning, even only on potential future revenues from an expanded user base and the access to the users' attention (e.g., Google, Facebook, Snapchat, or Instagram). An example in IS research in this respect is web-personalization that deals with engaging users more intensely to tie their attention to a service (e.g., Benlian 2015b). However, considering the low switching barriers of users due to a progressing digitalization of services and the success of cloud solutions in many fields (e.g., Benlian and Hess 2011a; Harnisch and Buxmann 2013), it has also become of crucial importance for firms to adapt and adjust to user requirements quickly to keep their customers satisfied. Only by doing so are firms able to engage and bind their customers successfully by providing a high value-to-customer. Quick responses to new requirements are even more necessary due to the growing importance of providing safe and stable solutions to users in times of substantial cybercrime and with increased privacy and security demands (Ackermann and Buxmann 2010). Considering all these indications together, it becomes clear that firms need to put their users in a position where users predominantly perceive themselves as ‘in good hands’ to ensure their own viability.

Out of this position, firms have developed more flexible development methods (e.g., Hirotaka and Nonaka 1986). These development methods are collectively referenced to as agile development methods (Maruping et al. 2009). Some prominent methods include eXtreme Programming (XP) (Beck 1999), Kanban (Ohno 1988), and Scrum (Rising and Janoff 2000; Schwaber and Beedle 2002), which are still the most prevalent ones today (Versionone 2017). In general, agile methods propagate an iterative and self-governing development approach that is aligned with customer and company goals. This offers flexibility in the ISD process and enables firms to cope with dynamic and changing environments. Recently, agile methods have once more received significant attention in the context of digital transformation of classic industries, in the discussion of bimodal IT functions (Haffke et al. 2017), and particularly in the field of operations. In operations, the cross-functional participation of developers and operation engineers based on agile principles over the entire product lifecycle, so called ‘DevOps’, has vastly benefited from previous insights on agile methods and shown a significant increase in service quality (Juner and Benlian 2017; Banica et al. 2017). However, although research has provided a significant theoretical foundation and profound understanding of agile methods in many fields (e.g., Fowler and Highsmith 2001; Cockburn 2001; Conboy 2009; Maruping et al. 2009), the results are largely based on a firm’s perspective. Yet, IS that are developed incrementally with the help of agile methods might be perceived differently by users and even change users’ experiences (in the following IS developed by agile methods are referenced as Agile Information Systems ‘Agile IS’). However, despite the potential effects of Agile IS on users, this subject remains understudied with little scholarly attention, only few scholars have researched this perspective (e.g., Hong et al. 2011). Therefore, this thesis aims to extend the knowledge on Agile IS from the user’s perspective to increase the explanatory power of IS theory on user responses to Agile IS and potential moderators for an effect.

Nonetheless, a user’s evaluation of an Agile IS might be prone to potential biases. Research in psychology has repeatedly shown that users, due to the limited cognitive resources, may utilize rules of thumbs, thereby not assessing situations in an entirely objective manner (e.g., Simon 1959; Tversky and Kahneman 1973, 1974; Kahneman and Tversky 1979; Thaler 1979). Such simplifications – heuristics – and resulting cognitive biases have been studied and incorporated in theorizing in IS research widely (e.g., Benlian 2013a; Benlian 2013b;

Fleischmann et al. 2014; Benlian and Haffke 2016). In most of the cases, it could be shown that due to the use of heuristics, users' reactions and decisions are biased with systematic errors (e.g., Rafaeli and Raban 2003; Vetter et al. 2011). Emanating from these findings, recent research has shown that such biases can be used to carefully lead users to beneficial or desired decisions by presenting little cues, so called 'nudges', that compensate for, or explore biases (Weinmann et al. 2016; Thaler 2016). However, even if nudges are not considered directly to utilize or compensate cognitive biases, research on Agile IS must consider that users may not exhibit fully rational responses to Agile IS. When assessing Agile IS, and in particular additionally delivered increments that change the previously installed software, users may fall prey to systematic errors originating from such simplifications (Tversky and Kahneman 1973, 1974; Kahneman and Tversky 1979). Therefore, this somewhat more subjective and heuristic understanding of a user's cognitive processes is incorporated into the theoretical framework adopted by this thesis.

Out of the user's point of view, increments of Agile IS are delivered through software updates. An example of this is the 'Creators Update' of Windows 10 that delivered several security features, a faster browser, and 3d painting capabilities to its users in 2017 (Ruiz-Hopper 2017). Because software updates are delivered to users when the system is already in use, they have the potential to change users' experiences. However, despite the ubiquitous use of software updates in practice to implement Agile IS, research on the impact of updates on users' beliefs, attitudes, and users' loyalty to the updated software in particular is scarce (Hong et al. 2011; Claussen et al. 2013). Also, there is no comprehensive understanding of the mechanisms by which users perceive updates, which factors may strengthen or mitigate a potential effect, and how Agile IS should be designed accordingly. This not only leaves practitioners without guidance, but also without a solid theoretical framework on how update processes should be designed specifically. Current research explores software updates mostly from an engineering perspective and thus from the supply side. This includes research on software engineering (Sommerville 2010), software product lines (Clements and Northrop 2002), software release planning (Svahnberg et al. 2010), and software evolution and maintenance (Mens and Demeyer 2008). The user's side, and in particular, users' perceptions of Agile IS which characteristics change over time, remains largely unexplored. Following Karahanna et al. (1999) and Bhattacherjee (2001), such changes in IS during use, however, may have the potential to alter users' beliefs, attitudes, and behaviors in the post-adoption stage. In particular, Bhattacherjee (2001) proposes the IS Continuance Model, which suggests that users would compare pre-usage expectations of a system with post-adoption experiences to form their beliefs, attitudes, and behaviors regarding the system. In consequence, through the lens of the IS Continuance Model, Agile IS might cause similar changes in users' beliefs, attitudes, and behaviors after initial adoption due to their changing nature. Therefore, a better understanding of software updates as the medium to deliver Agile IS increments to users has the potential to complement existing post-adoption research in significant ways.

Moreover, research on post-adoption phenomena still has the tendency to conceptualize IS as static and monolithic systems, rather than as a dynamic assembly of specific features that can be altered over time (Jasperson et al. 2005). Although some studies have explored IS usage at a feature level (e.g., Benlian 2015), they usually do not consider changes in the available feature set over time. Understanding the details and specifics of Agile IS and dynamic changes in such IS through software updates may help to gain insights on to how user's beliefs, attitudes, and behaviors fluctuate over time, due to the flexible nature of Agile IS. Moreover, there are several calls for research from IS scholars who criticize the negligence of the IT Artifact in IS research (Orlikowski and Iacono 2001; Benbasat and Zmud 2003; Hevner et al. 2004). They suggest that more research on the IT Artifact itself and on changes in beliefs, attitudes, and behaviors emanating from the IT Artifact itself rather than from other IT-unrelated stimuli is required. Summing up, due to the underexplored user perspective on Agile IS, poor knowledge on potential user responses to software updates and potential moderators, and a mostly monolithic view on IS that neglects the central role of the IT Artifact and its specifics, this thesis addresses the following research questions:

RQ1: Do incrementally developed Information Systems (Agile IS) have the potential to increase users' loyalty and if so, how does an effect emerge?

RQ2: What are important moderators for the effect of Agile IS on users and how should Agile IS be designed in consequence?

To answer the overarching research questions, existing research on Agile IS, non-rational user behavior in IS, software updates, and the IS Continuance Model is drawn. Based on these theoretical grounds, a research framework is posited to systematically investigate several factors that may constitute and affect potential user responses to software updates. Moreover, based on the IS Continuance Model (Bhattacherjee 2001) that originates from the Expectation-Confirmation Theory (Oliver 1980), a potential mechanism on how users might perceive software updates is set forth. With the help of this theoretical lens, users' loyalty to

Agile IS is operationalized as users' intentions to continue using a system in the empirical studies (i.e. continuance intentions).

To obtain valid and reliable answers to our questions, four empirical studies across different contexts were conducted, which were operationalized with slightly different experimental methods. This not only allows uncovering multiple facets of potential effects, but also confirming and retesting core effects. The peer-reviewed articles that present the results of the studies are included in this thesis. The articles have been published in established IS outlets and provide causal evidence based on previous theoretical reasoning with respect to the research questions. In the following, the theoretical foundation on which this reasoning is based is presented. Subsequently, the thesis is positioned in the context of previous research and the underlying framework and structure are outlined.

1.2 Theoretical Foundations

This section begins with a review of literature on Agile IS and agile methods to better understand the nature and characteristics of Agile IS. Subsequently, to base the understanding of potential user-reactions on a solid theoretical reasoning, literature on user behavior and potential biases is summarized. Finally, current research on software updates and theory on the IS Continuance Model is laid out to substantiate our reasoning about potential effects of software updates on users based on previous theoretical knowledge.

1.2.1 Agile IS

IS have been evolving increasingly fast. And because traditional development methods show persistent limitations in addressing rapidly emerging new requirements, engineers have sought of new approaches to cope with constraints and changing environments effectively, while producing tangible results quickly. These methods have the potential to transform the nature of an IT Artifact, as they change the way how an IT Artifact is composed over time while it is already in use. Most often, such methods are referred to as agile development methods (Maruping et al. 2009). Frequently named examples for agile methods include eXtreme Programming (XP) (Beck 1999), Crystal (Cockburn 2001), Lean Programming (Poppendeick 2001), Kanban (Ohno 1988) and Scrum (Rising and Janoff 2000; Schwaber and Beedle 2002). Today, Scrum, XP, and Kanban are the most prominent agile methods and are employed across different company functions and industry sectors, sometimes in hybrid or blended variations (Versionone 2017).

Collectively, agile methods are chosen over traditional approaches because they offer flexibility in the Information Systems Development (ISD) process and enable firms to cope with a volatile and changing environment, i.e., they can increase a firm's agility (Beck 1999). In the development context agility is the defined as *"the continual readiness of an Information System development method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment."* (Conboy 2009, p. 338). Over the years, a precise definition and formal taxonomy has been developed to better understand agility, but also has the introduction, promotion and development of agile methods been researched in diverse contexts such as management, project management, and ISD (Conboy 2009; Ågerfalk et al. 2009; Lee and Xia 2010). Most of research related to this topic provides a better understanding on how to successfully implement agile methods and what factors are important for the subsequent success of agile ISD efforts. For example, Lee and Xia 2010 empirically explore the impact of team autonomy and diversity in the context of software development agility on software development performance (i.e., on-time completion, on-budget completion, and software functionality). In another example, Mishra et al. 2012 explore the role of communication in agile systems development and find that co-location and open offices improve communication quality, and thereby results.[1] Other scholars advance a more strategic understanding of a firm's agility as a factor related to IT capabilities, IT expenditures, and IT alignment (e.g., Lu and Ramamurthy 2011; Talon and Pinsonneault 2011). Surprisingly, they find that only expenditures directed to IT capabilities increases agility while other expenditures do not, and that IT infrastructure flexibility increase firms' performance in volatile markets in particular (Lu and Ramamurthy 2011; Talon and Pinsonneault 2011). More recent approaches consider using a bimodal IT that uses both agile and traditional modes contingent on the functions purpose and requirements (Haffke et al. 2017). Summing up, altogether, most research considers personal skills, culture, infrastructure, resources, organizational design, and methods as fundamental factors that moderate a firm's ability to increase its agility, in particular with agile development approaches (Salmela et al. 2015). Finally, given that such agile development approaches not only support a firm to gain agility, but also, change the nature of IT Artifacts with potential further consequences and the characteristics of such IS developed by agile methods need a

[1] A structured literature review of research on agile development can be found in Hummel et al. 2013.

better understanding. Therefore, theoretical perspectives on Agile IS are presented in the following.

The recent body of research on the characteristics of Agile IS explains many fundamental properties of Agile IS and benefits for the supply side very well; however it is still scarce with respect to many aspects out of the user's perspective. In this context, Hong et al. (2011) define Agile IS as IS that are developed using agile methods. Such systems are implemented incrementally, in each release only the smallest set of most valuable functionality is delivered to the user (Hong et al. 2011). The particular requirements for each increment are prioritized based on users' input – and should represent the users' most urging or valued requirements (Lee and Xia 2010). Clearly, these requirements may change and fluctuate over time (Maruping et al. 2009). However, fortunately, agile methods embrace such dynamic contexts by design because of the constant and periodical re-evaluation of priorities and needs (Fowler and Highsmith 2001; Cockburn 2001; Conboy 2009). Nevertheless, as a result, Agile IS provide only a limited feature-set in the first release, and this feature-set will only subsequently be extended in each release with additional, limited set of functionality. Thereby, the system evolves periodically, most often with a preset release frequency (i.e., every few weeks or months depending on the development cycle length) (Hong et al. 2011). Therefore, because the composition of IS changes with later increments of the software, Agile IS have the potential to alter users' perceptions of the IT Artifact, because the system's nature changes over time. This users' perspective needs a better understanding, not only in the context of Agile IS, but also in the overall field of IS, which is reflected in scholars emphasizing the need to put the user and their perceptions more at the center of all investigations (e.g., Brenner et al. 2014).

However, the existing body of knowledge on agile methods, implementation of agile methods, and Agile IS has mainly focused on cultural, structural, processual, and technical facets that all lie within the domain of a firm (e.g., Fruhling and Vreede 2006; Ågerfalk et al. 2009; Harris and Collins 2009; Maruping et al. 2009; Vidgen and Wand 2009). Although this developer's perspective advances the understanding on how to implement and benefit from agile methods, as well as how Agile IS should be composed, nevertheless, it does not consider the user's perspective. Most often, the user is only incorporated into the theorizing as a source of input and feedback (Chan and Thong 2009). In contrast, Hong et al. 2011 are among of the first to empirically study the impact of Agile IS from the user's perspective. However, they do not isolate the base effect as compared to Non-agile IS, nor do they consider the specifics of changing compositions of IS and related consequences. Overall, a recent review of literature on agile ISD identifies only 6% of all studies in the field as somewhat considering the users' perspective (Hummel 2014). Therefore, given the vast spread of Agile IS and the shift of focus to the user, further research to close this existing gap is inevitable.

In the following, to better understand and predict user responses to Agile IS, research on potential biases that might influence users' perceptions is reviewed. Subsequently, software updates and the IS Continuance Model are introduced. The IS Continuance Model will serve as a theoretical framework to understand the implementation of Agile IS and potential user responses in terms of continuance intentions (CI) better.

1.2.2 Non-rational user

Modern theory on behavioral economics suggests that subjects instead of being fully rational economic actors are to some extend 'more human' making subjective choices. Simon (1959) was one of the first to develop this idea by proposing that understanding subjects to make somewhat irrational decisions would provide a more accurate tool for modeling human behavior than classic economic theory does (Simon 1959). Even more, due to cognitive load and the required mental efforts of decisions, humans overcome cognitive limitations by using rules of thumbs – heuristics – that help them to reduce the amount of information that is needed to be processed (Tversky and Kahneman 1973, 1974). Such simpler heuristics may result in subjective biases and systematic errors that lead to decisions that are not congruent with classic utilitarian theory (Tversky and Kahneman 1973, 1974; Kahneman and Tversky 1979; Evans 2006; Evans 2008). Given this stream of research, it has become clear that although economic theory helps to understand what consumers should do; only theory that takes both psychological and economic factors into account can help to understand what consumers actually do (Thaler 1979).

Further research on heuristics and cognitive biases over the last decades has revealed several situations in which outcomes of decisions are not in line with optimal predictions from economic theory. Some of the very fundamental principles that may bias such decisions are set forth in the Prospect Theory developed by Kahneman and Tversky (1979). First, Kahneman and Tversky (1979) describe the subjective value function to reflect *"changes in wealth or welfare, rather than final states"* (Kahneman and Tversky 1979, p. 277). They assume most user responses to emerge relative to a prior adapted level, a subjective reference point. As a result, this reference point may be biased and experiences may therefore not solely relate to an absolute magnitude of an event, but more to the perceived change with respect to the status quo. Next, based on these findings, they introduce the principle of diminishing utility. Because of this relative perception, if a value is already high, a fixed size change seems relatively small compared to a small value that is changed by the same amount. Therefore, *"the marginal value of both gains and losses generally decreases with their magnitude"* (Kahneman and Tversky 1979, p. 278). Lastly, they expect losses to be weighted more than gains: *"The aggravation that one experiences in losing a sum of money appears to be greater than the pleasure associated with gaining the same amount"* (Kahneman and Tversky 1979, p. 279). This suggests, for example, that users would pay more to not lose a certain item they possess than what they would pay for the item to gain it in the first place (Kahneman et al. 1991; Benartzi and Thaler 1995). Considered together, these and several more related crucial findings have demonstrated that cognitive heuristics and biases need to be accounted for to correctly understand user behavior. Therefore, this understanding is incorporated into the theoretical framework related to the studies' hypotheses.

Heuristics have been studied in IS research in various ways, most often offering interesting new insights and providing explanations for somewhat irrational behaviors. For example, Vetter et al. (2011) show that IT decision makers draw near targets of past decisions as reference points for building future risk preferences. In another example, Rafaeli and Raban (2003) show an endowment effect in an information trading situation. People would value information they have more than information they do not have. Also, Gupta and Kim (2007) show how in a transaction process perceived value and perceived convenience are evaluated relative to previous anchors that are formed from previous expectations. They show that this subjective perception affects changes in beliefs and attitudes. Many examples of IS research exist, in which cognitive biases contribute significantly in explaining the observed results (e.g., Benlian 2013a; Benlian 2013b; Benlian and Haffke 2016).[2] These examples underline empirically that cognitive biases may clearly enrich existing theories and models in IS, and that potential cognitive biases therefore need to be considered thoroughly when theorizing about users' behaviors (Fleischmann et al. 2014).

Building on these rich findings related to cognitive biases in human behavior, present research efforts are directed towards understanding how such biases may be reduced or

[2] A scientometric analysis by Fleischmann et al. (2014) provides a systematic overview on the role of cognitive biases in IS research and suggests a categorization into specific classes of biases.

exploited by using interface or product design elements to guide consumer behavior in favorable ways (e.g., Weinmann et al. 2016). Such adjustments and informational cues provided to users to remedy or facilitate biases are termed ‘nudges’ (Thaler 2008). The beneficial use of nudges is currently receiving increasing research interest in diverse fields (Thaler 2016). In line with this development, the amount of research on nudges in digital choice environments is growing steadily (e.g., Schneider et al. 2017). Following this notion, fostering a positive and considered behavior of users that are confronted with Agile IS can play a major role. For example, if a system is vulnerable in its current state and should be updated to close a security vulnerability quickly, guiding users to understand the benefits of an update will become outstandingly important. Nonetheless, also without making explicit use of them, understanding cognitive biases in the context of IS research has strongly contributed to studies of IS usage and must therefore be accounted for.

Finally, to understand how Agile IS are delivered to users, the medium that delivers increments of Agile IS to users – software updates – are thoroughly explored in the following section. Subsequently, potential user responses to Agile IS in terms of their continuance intentions are examined through the lens of the Expectation-Confirmation Theory and the IS Continuance Model that is also introduced in the following section.

1.2.3 Software Updates and the IS Continuance Model

Software updates are self-contained modules of IS that are provided to users with the purpose to extend or modify the system after its initial deployment when the software is already in use (e.g., Dunn 2004). Software updates are most commonly used by firms to repeatedly roll out new versions and recent increments of Agile IS to users who currently use a first or earlier version of the system. Thereby, they constitute a fundamental component of Agile IS. Software updates are usually provided for free, this distinguishes them from software upgrades which are typically paid for. Moreover, a software update is not a stand-alone program – it rather depends on the base software and modifies or extends the IS once it has been applied.

Software updates are addressed in software engineering literature with a range of ambiguous terms (e.g., update, upgrade, patch, bug fix, or hotfix) (Sommerville 2010). This technical literature addresses software updates with respect to software release planning, software maintenance, and evolution, as well as software product lines (Weyns et al. 2011). Following Svahnberg et al. (2010), software updates fall within the strategic considerations as part of the software release planning process. Firms need to decide when to deliver what type of functionality to the user. With the help of release plans they can either fix in advance what will be developed next over longer timespans, or they can determine the scope for the next release more dynamically for each individual release just in time. Such flexible practices are inherent to most agile methods. Finally, research on software evolution and maintenance addresses later stages in the software development lifecycle, where updates are used to adjust systems to changing requirements or repair emerging flaws while the software is already in use (Shirabad et al. 2001).

Similar to literature on Agile IS, technical literature has comprehensively dealt with software updates from a developers’ perspective. However, in contrast, literature on users’ beliefs and attitudes regarding updates is scarce. Only few IS studies have dealt with updates so far (e.g., Hong et al. 2011; Amirpur et al. 2015). While Benlian (2015) explores IT feature repertoires and their impact on the task performance of users, however, the study does not consider changes in functionality over time through updates. Hong et al. (2011) are among the first to explorer user’s acceptance of IS that received additional new functionality after the initial release. However, they do not compare the observed effects to comparable baseline software (i.e., non-agile software with afull feature set) to understand the fundamental differences, nor do they consider feature-level compositions of the software. Other studies explore different feature sets, their use, valuation, consequences, and usage over time (e.g., Turoff 1981; DeSanctis and Poole 1994; Hiltz and Kay and Thomas 1995; Benlian and Hess 2011b; Sun 2012; Leonardi 2013; Benlian 2015a), nevertheless, they do not consider changes in the feature set after the initial release. Lastly, almost all other studies in the field have pushed updates to the sidelines, only considering them as instrumental to study other phenomena instead of investigating them to better understand the user’s perceptions of changes in an IT Artifact (e.g., Claussen et al. 2013).

In the present thesis, three types of updates are distinguished: feature updates, non-feature updates, and security updates. Table 1-1 provides an overview on how these types of updates are defined. However, because all three update types occur during the use of a system and manifest themselves in various ways, they have the potential to affect users’ post-adoption beliefs, attitudes, and behaviors regarding IS, including continuance intentions. To better understand this potential, a review of theory on IS continuance follows.

Table 1-1: Overview of Update Types

Update Type	Definition	Examples
Feature update	Feature updates are updates that change the core functionality of software by adding or removing discernible functionality. Functionality thereby refers to features in the software which are deliberately employed by users in accomplishing the core task for which the software is used.	E.g., the update of the popular Instagram-App in 2016 that enabled users to add personal momentary stories to the feed in addition to the possibility to add permanent posts (Constine 2016).
Non-feature update	Non-feature updates are updates that do no change the core functionality but correct flaws (e.g., bug fixes) or change software properties that are not directly related to its core functionality (e.g., improvements in stability or performance) (Popović et al. 2001).	E.g., the regular 'hot fixes' that Microsoft rolls out to the windows platform via its automatic update service to fix minor bugs.
Security update[3]	Security updates are updates that close vulnerabilities of a software or enhance its protective powers against security breaches (Dinev and Hu, 2007; Ng et al., 2009).	E.g., the firmware update rolled out to Intel Processors to close the security vulnerability in the preload functionality of processors that allows malicious software to read and intercept all processed data (Shenoy 2018).

One of the most mature fields in IS research is the field of 'IS usage' (Jasperson et al. 2005). It is constituted by research on the pre-adoption phase, the actual adoption decision, and the post-adoption phase. With respect to research on the post-adoption phase (Karahanna et al. 1999; Bhattacherjee 2001; Benlian et al. 2011; Goldbach et al. 2017), the term information systems continuance is defined as the "*sustained use of an IT by individual users over the long-term after their initial acceptance*" (Bhattacherjee and Barfar 2011, p. 2). In the last decade, the interest in IS continuance has nurtured a better theoretical and empirical understanding of the post-adoption phase. Bhattacherjee (2001) for example, with the aim to study users' intentions to continue or discontinue the use of IS, has suggested the IS Continuance Model.

[3] Clearly, security updates can be classified as a sub group of non-features updates, however, because of their distinct nature (i.e., they are not equally perceptible by users compared to performance or stability improvements, but there is a strong need to apply them consistently), they are considered as different types of updates for the sake of explicit research.

The IS Continuance Model is based on Expectation-Confirmation Theory (ECT) (Locke 1976; Oliver 1980, 1993; Anderson and Sullivan 1993). Expectation-Confirmation Theory posits that repurchase intention and post-purchase satisfaction are formed by a positive or negative disconfirmation of beliefs that results from a comparison of prior expectations with post-hoc performance. In its reasoning, ECT understands expectations as a relative and subjective baseline that is *not* an objective value. Following Helson's (1964) adaption level theory and reasoning in prospect theory (Kahneman and Tversky 1979), this baseline to which stimuli are compared to, is a subjective reference point. ECT however, though it has been successfully applied in many IS research contexts, puts customers' repurchase intentions at the center of investigation. Therefore, to study the sustained use of IS, Bhattacherjee (2001) replaces this repurchase intention in ECT by users' intention to continue using an IS, the core variable in the IS Continuance Model. In line with ECT the IS Continuance Model suggests that users compare their pre-usage expectations of an IS with the perceived performance regarding this IS during actual usage (Bhattacherjee 2001). The discrepancy manifests in users experiencing positive or negative disconfirmation (DISC). Subsequently, this positive or negative disconfirmation increases or decreases users' satisfaction (SAT) regarding the IS (Bhattacherjee and Barfar 2011). As a result, satisfied users intend to continue using the IS, while dissatisfied users discontinue its subsequent use (Oliver 1980; Bhattacherjee 2001). This relative mechanism in the IS Continuance Model allows the potential presence of a non-rational response to feature updates that may challenge the idea of a 'rational user' in the IS continuance literature (Ortiz de Guinea and Markus 2009; Bhattacherjee and Barfar 2011; Ortiz de Guinea and Webster 2013).

Undoubtedly, the IS Continuance Model has contributed significantly to post-adoption research (Bhattacherjee 2001). Yet, in its original form, the IS Continuance Model established a static view on IS continuance decisions, thereby failing to consider changes in user's beliefs and attitudes over time, after initial adoption. Therefore, to compensate for this limitation, Bhattacherjee and Premkumar (2004) propose a more dynamic view on the IS Continuance Model that also considers changes in beliefs and attitudes during the ongoing usage of an IS and not only from the pre-usage to the actual usage stage. In line with this proposal, several scholars have found evidence that an IT Artifact itself can affect users' beliefs and attitudes during use and in later usage stages (e.g., Kim and Malhotra 2005; Kim and Son 2009; Ortiz de Guinea and Markus 2009; Ortiz de Guinea and Webster 2013). Following Bhattacherjee and Premkumar (2004), it is therefore reasonable to assume that a change in IS, due to an update for instance, can also induce changes in users' beliefs and attitudes towards it. Hence, to investigate the changing nature of Agile IS and its impact on users' beliefs, attitudes and behaviors during post-adoption use, software updates are studied through the lens of the IS Continuance Model.

1.3 Thesis Positioning

While IS scholars have explored many aspects of IS in depth, nevertheless, additional research is required in some areas, particularly related to core aspects of IS. This assessment stems from a vital commentary by Benbasat and Zmud (2003) in which they analyze the central identity of IS research. Comparing it to extant research in the field, they argue that phenomena directly related to IT-based systems (i.e., IT managerial, methodological, and technological capabilities, practices, the IT Artifact itself, its usage, and its impact) are under-investigated and distantly associated phenomena that do not directly include the IT Artifact (i.e., consumer behavior, trust-building, decision-making, and so forth) are over-investigated (Benbasat and Zmud 2003). Moreover, Benbasat and Barki (2007) set forth that some of the most accepted models in IS research while providing solid ground for many studies, have a very narrow focus on user behaviors and do not accurately capture evolving IT contexts in which a dynamic interplay in user behaviors occurs between various software states (Benbasat and Barki 2007). Therefore, to account for this somewhat ambiguous shift of the discipline's central focus and the still mostly narrow and static view on IT Artifacts, this thesis is clearly positioned to gain knowledge *directly related* to the *core aspects* of IS research, on the *IT Artifact* itself (i.e., Agile IS), through a contemporary perspective, the *IS Continuance Model,* and with a *dynamic understanding* of IS (see Figure 1-1).

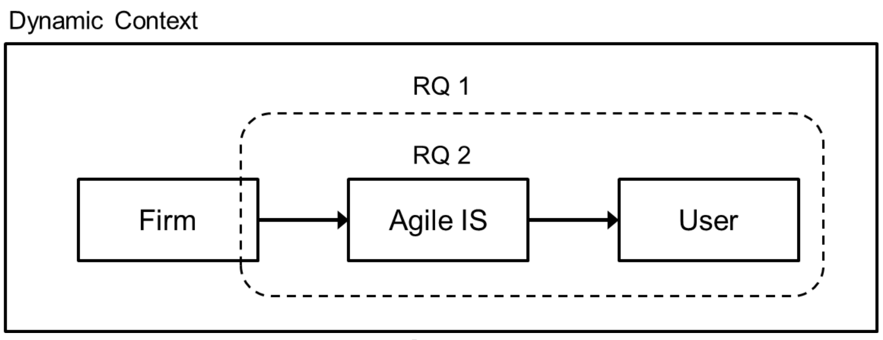


Figure 1-1: Thesis Positioning

As discussed in the theoretical section, extant research on Agile IS, agile methods, and agile practices has a strong focus on the supply side whereas research on the user's perspective on Agile IS is still scarce. Considering the established research in this field, most of the studies only investigate aspects related to agile methods themselves (e.g., Fowler and Highsmith

2001), the implementation of such practices (e.g., Chan and Thong 2009), technical aspects (e.g., Fruhling and Vreede 2006), and the firm's agility (e.g., Lu and Ramamurthy 2011; Talon and Pinsonneault 2011). Considering this vast amount of knowledge, it seems surprising that only little is known about the perceptions of Agile IS from the user's perspective. Given the clear call for more research from scholars that puts the user and his needs at the center of all investigations (e.g., Brenner et al. 2014), this thesis aims to contribute to this insufficient body of knowledge and focuses on understanding Agile IS from the user's perspective.

Bearing in mind that software updates are the vehicle through which Agile IS are changed over time, it is surprising that only few IS studies have dealt with them (e.g., Hong et al. 2011; Amirpur et al. 2015). And because software updates are applied during use, in the post-adoption stage, they may have the potential to alter users' beliefs, attitudes, and behaviors regarding the software after initial adoption (Karahanna et al. 1999; Bhattacherjee 2001). Nonetheless, still, researchers studying post-adoption phenomena often tend to conceptualize IS as monolithic and static black boxes, rather than as collections of finer-grained features that are alterable over time (Jasperson et al. 2005). Only few studies have explored IS usage at a feature level (e.g., Benlian 2015). However, these studies do not consider changes in the feature set over time, during usage, that may have the potential to impact user's beliefs, attitudes and behaviors after initial acceptance. Hence, the results of this thesis may help to increase the explanatory and predictive power of existing post-adoption theory, by better understanding dynamic changes in software through software updates and potential impacts from a user's perspective.

Summing up, this thesis aims to extending the limited understanding of user's beliefs, attitudes, and intentions with respect to an Agile IS. It is clearly positioned with a focus on the IT Artifact itself, and in the field of IS post-adoption research within a dynamic context (see Figure 1-1). Moreover, out of the user's point of view, this thesis also provides new and important insights that can be inferred for the supply side of Agile IS (i.e. firms).

1.4 Structure of the Thesis

To answer the overarching research questions, four empirical studies were conducted. This allows the identification of causal relationships with respect to our central questions on Agile IS and their impact on users. All studies were published in peer-reviewed scientific IS outlets. Overall, the thesis is organized into six chapters: an introductory chapter, four chapters that present the published articles as shown in Table 1-2, and a final chapter that concludes with key findings, contributions, and limitations. The articles included into the thesis were slightly revised to achieve a consistent layout throughout the thesis.

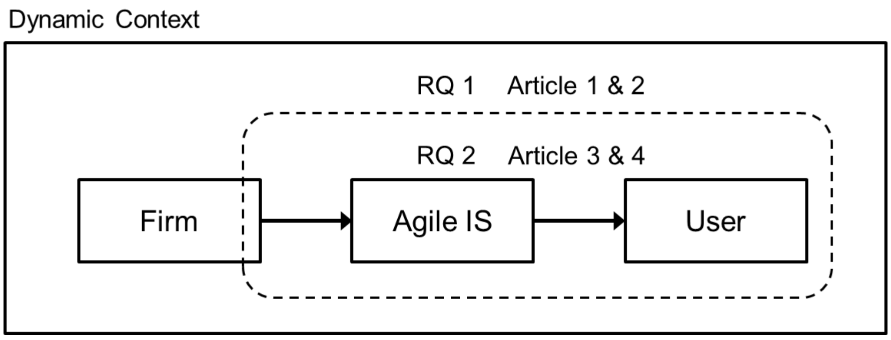


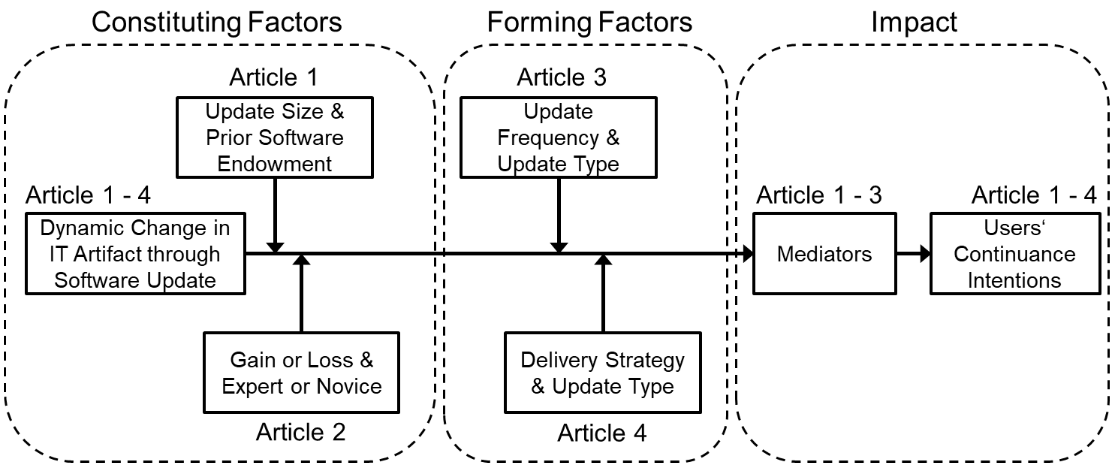
Figure 1-2: Overarching Article Contributions

Primarily, the first two articles aim to identify the underlying effect of Agile IS on users in comparison to traditional IS, and to understand basic principles. The following two articles focus more on the tangible design of Agile IS and consequences for users. This embodies the meta-structure of the thesis that is shown in Figure 1-2.

Table 1-2: Overview of Articles

Study 1	**Chapter 2**	**Effects of Updates and the Role of Update Size and Prior Endowment**
	Article 1	Fleischmann, M., Grupp, T., Amirpur, M., Benlian, A., and Hess, T. 2015. "When Updates Make a User Stick: Software Feature Updates and their Differential Effects on Users' Continuance Intentions," *Thirty Sixth International Conference on Information Systems (ICIS),* Fort Worth, USA. **VHB: A**
Study 2	**Chapter 3**	**Effects of Gains or Losses through Updates on Experts or Novices**
	Article 2	Fleischmann, M., Grupp, T., Amirpur, M., Benlian, A., and Hess, T. 2015. "Gains and Losses in Functionality – An Experimental Investigation of the Effect of Software Updates on Users' Continuance Intentions," *Thirty Sixth International Conference on Information Systems (ICIS),* Fort Worth, USA. **VHB: A**
Study 3	**Chapter 4**	**Updates and the Role of Update Frequency and Update Type**
	Article 3	Fleischmann, M., Amirpur, M., Grupp, T., Benlian, A., and Hess, T. 2016. "The Role of Software Updates in Information Systems Continuance – An Experimental Study from a User Perspective," *Decision Support Systems* (83), pp. 83-96. **VHB: B**
Study 4	**Chapter 5**	**Updates and the Role of Delivery Strategy and Update Type**
	Article 4	Grupp, T., and Schneider, D. 2017. "Seamless Updates – How Security And Feature Update Delivery Strategies Affect Continuance Intentions With Digital Applications," *Proceedings of the 25th European Conference on Information Systems (ECIS),* Guimarães, Portugal. **VHB: B**

Specifically, the introductory chapter motivates this research, presents the research questions, lays the theoretical foundations for the subsequent studies, and provides an overview on the thesis positioning and structure. Subsequently, the four studies are presented to answer our research questions as outlined in the research framework's detailed structure in Figure 1-3.


Constituting Factors
Forming Factors
Impact
Article 1
Update Size & Prior Software Endowment
Article 3
Update Frequency & Update Type
Article 1 - 4
Dynamic Change in IT Artifact through Software Update
Article 1 - 3
Mediators
Article 1 - 4
Users' Continuance Intentions
Gain or Loss & Expert or Novice
Article 2
Delivery Strategy & Update Type
Article 4
Figure 1-3: Research Framework and Specific Article Contributions

First, article 1 (chapter 2) introduces software updates – the manifestation of Agile IS increments from the user's perspective – and investigates their impact on users' continuance intentions as compared to a feature-complete software. Also, the study evaluates two crucial moderators for the identified effect: update size and prior software endowment. Second, article 2 (chapter 3) extends this first understanding of a somewhat irrational effect of software updates on users' continuance intentions by distinguishing the reactions of experts and novices. Furthermore, the second study evaluates the case where an update removes a feature from the software. Both studies isolate and demonstrate the effect of how Agile IS constitute a change in users' perceptions of the IT Artifact in comparison to a traditional IS. Third, article 3 (chapter 4) further evaluates software updates by considering update frequency as an additional moderator for the identified update-effect, a factor, that can be deliberately designed by the firm. In this study, to understand whether the effect applies in this case or not, feature and non-feature updates are distinguished. All three studies examine and confirm the hypothesized mediating mechanism that underlies the update-effect and this third study in particular confirms the mediation mechanisms by thought-listing. Lastly, article 4 (chapter 5) evaluates the role of the delivery strategy of an update, i.e., the timing and presence of a notification and an installation choice. In this study, feature and security updates are distinguished, as both possess different characteristics with this respect. Finally, chapter 6 concludes the thesis by providing a summary of key findings, underlining theoretical and practical contributions, and pointing out limitations and opportunities for further research.

In the following, each of the four articles is briefly summarized. The motivation and the main contributions of each article are positioned within the context of the overall research questions, and the relations of the articles are outlined. The summaries and the articles will use the first-person plural (i.e., 'we'), as the studies were conducted in cooperation with co-authors.

Article 1 (Chapter 2):
When Updates Make a User Stick: Software Feature Updates and their Differential Effects on Users' Continuance Intentions.

Increments of Agile IS are usually delivered through software updates, which have been researched in software engineering and development literature, but have been neglected in research from the users' perspective despite their prevalence in practice. This study soughs to reduce this gap, thereby mainly contributing to the first overarching research questions of the thesis. By drawing on theory of IS Continuance and the ECT, we hypothesize about the effect of updates on users' continuance intentions as compared to a traditional IS and the role of update size and initial feature endowment. We conducted a vignette-based online experiment with 261 participants to identify causal relationships and thereby answer these questions. The results of the study show that continuance intentions are increased in all update-conditions compared to the non-update conditions. This increase in CI can be interpreted a as a somewhat counter-intuitive finding because users had access to less functionality over the total timespan compared to user who had all features right from the beginning. Despite this objective disadvantage, participants in all update groups indicated significantly higher scores in CI. Moreover, the experiment revealed that update size does not seem to constitute a significant boundary condition for the positive effect of feature updates on users' CI. However, contrary to our hypothesis, feature endowment appears to moderate the effect of feature updates on CI by decreasing its magnitude. This finding may be explained by the concept of diminishing sensitivity (Tversky and Kahneman 1992; Nowlis and Simonson 1996). Additionally, we could demonstrate that the positive effect of feature updates on CI was mediated by a serial chain of relations that originates from a positive disconfirmation of previous expectations. Our results contribute to our overarching research questions in two ways: first, the surprising results of this study underline the understanding that Agile IS are able to increase users' continuance intentions. Thereby the study reveals non-rational user responses to updates and identifies a beneficial strategy for the firm. Second, it shows how software updates are perceived by users; hence it strengthens the understanding of the user's perspective on Agile IS.

Article 2 (Chapter 3):
Gains and Losses in Functionality – An Experimental Investigation of the Effect of Software Updates on Users' Continuance Intentions.

Firms usually update Agile IS in order to correct flaws or extend functionality. However, sometimes updates also remove functionality, for example when license deals run. Moreover, as more and more people gain access to information technology, the amount of non-expert users increases (Rogers 1995). To theoretically account for these developments, it becomes important for IS research on Agile IS to explore the heterogeneity in users' beliefs, attitudes and behaviors contingent on the user's expertise. Drawing on theory of IS Continuance and IS Expertise, this study therefore extends the findings of the first study by considering both the addition and removal of functionality through updates while simultaneously distinguishing between experts and novices to better understand potential differences. In a scenario-based online experiment with 178 participants we found that expert and novice users showed different responses to updates. In the case of experts, any type of update (e.g., loss or gain in functionality) led to a decrease in CI compared to a feature-complete software. Probably, experts are more likely to identify functionality as common features that should have been available right from the beginning which explains this result. However, novices showed different reactions. While they also had a lower CI when losing a feature, their CI was significantly higher in the positive update condition, confirming the somewhat counter-intuitive finding of the first study. When comparing the absolute values of the novices' responses to gaining or losing a feature, their evaluations seem even less rational. The perceived loss from removing a feature from the software through an update was higher in magnitude than the perceived gain from receiving the exact same feature through an update. This suggests the presence of loss aversion (Kahneman and Tversky 1979). Collectively, the results from this study reveal two additional aspects that help answering our first overarching research question. On the one hand, we find that only if the hold back functionality is not expected to be available at first, updates have the potential to increases users' continuance intentions (which often is the case for users that are not very knowledgeable regarding the software). On the other hand, we find that users devalue losses in functionality through software update more than gains, which again demonstrates somewhat irrational user reactions to software updates.

Article 3 (Chapter 4):
The Role of Software Updates in Information Systems Continuance – An Experimental Study from a User Perspective.

With software updates firms roll out increments of Agile IS that either deliver specific features that can be deliberately employed by the user, or they roll out technical improvements that correct flaws, increase performance, or security. Moreover, several such feature or non-feature functionalities can be gathered into one chunk to be released as one capital update, or they can be distributed across multiple updates to be released separately. The latter would lead to a higher update frequency, given a continuous development output. Assuming a subjective and relative evaluation of the updates' contents by users, the effect of updates may change contingent on the update's frequency. We therefore assume two fundamental aspects to have a notable impact on our results of the previous studies: 1) update type (feature or non-feature update), and 2) update frequency. Again drawing on the IS Continuance Model, we investigated our hypotheses based on a controlled lab experiment with 135 participants. For the purpose of the study we developed a fully functional text editor application, to perfectly mimic a real-world scenario. The results of the experiment reveal that only in the feature-update conditions CI was significantly higher than in the non-update condition. Non-feature updates could not increase users' CI compared to the condition without updates. This identifies update type as a distinct and crucial moderator for the effect of software updates on CI. Moreover, we find that users prefer features to be delivered in individual updates over a delivery of features in larger but less frequent update packages comprising several features. Update frequency thus seem to clearly moderate the effect of software updates on users' continuance intentions. Lastly, by mediation-analysis and thought-listing, we substantiate the understanding of users' reactions and the hypothesized mediation mechanism in detail. This study thereby contributes somewhat to our first overarching research questions by deepening the understanding of the mediating mechanism of user beliefs. However, it mainly contributes to our second overarching research question by identifying a new crucial moderator that can be deliberately designed by firms – update frequency – that impacts the effect of updates on users' continuance intentions. Moreover, the study extends our previous findings by demonstrating that users' reactions are contingent on the update type (i.e., feature vs. non-feature update).

Article 4 (Chapter 5):
Seamless Updates – How Security and Feature Update Delivery Strategies Affect Continuance Intentions with Digital Applications.

If updates are rolled out to users, developers of applications or platforms have various options to make them available to users. Updates may be applied consistently or only optional and they may be announced before or after successful implementation. Regarding the choice of delivery strategy, one must also think about the contents of an update. We distinguish two major update types, delivering either additional functionality or security enhancements. Feature updates are clearly noticeable by users because they provide specific functionality that can be deliberately employed. Security updates however, remove potential vulnerabilities or enhance the software's security and only indirectly contribute to the value of the software and thus cannot be observed directly. Drawing again on the Expectation-Confirmation Theory (Oliver 1980), embedded in the IS Continuance Model (Bhattacherjee 2001), we conducted an online experiment with 282 participants to understand the role of update delivery strategies. The empirical results reveal surprising insights. In the case of a security update with post-update notification, users showed a significant higher CI. However, in the case of a security update with ex-ante notification, no significant change in CI could be observed. Regarding feature updates, users receiving additional new functionality without further notification, did not show a significant increase in CI, despite this increased value provided by the software.

This somewhat unexpected result may be explained by the users' attention bound to the task users had to accomplish (Tversky and Kahnemann 1973), leaving the additional functionality unnoticed. Only in both cases when the feature update was announced before or after successful implementation, we found a significant increase in users' CI. In addition, we could evidence that updates that are delivered with a voluntary strategy do not increase users' CI. The diverse findings of this study help to answer the second overarching research questions in considerable ways. We could differentiate, in which cases a positive update-effect can be obtained, by identifying update notification, timing, and installation choice as crucial moderators for the effect. Furthermore, we could show that user responses to updates fundamentally depend on the update type and the user's perceptions of the update's delivery process, which can be shaped by explicitly designing this experience.

Additional Articles (not included):

Moreover, in addition to the above articles, I co-authored the following study which is not part of this thesis:

Schneider, D., Lins, S., Grupp, T., Benlian, A., and Sunyaev, A. 2017. "Nudging Users Into Online Verification: The Case of Carsharing Platforms," *Proceedings of the 38th International Conference on Information Systems (ICIS)*, Seoul, South Korea. **VHB: A**

Chapter 2: Effects of Updates and the Role of Update Size and Prior Endowment

Title: When Updates Make a User Stick: Software Feature Updates and their Differential Effects on Users' Continuance Intentions

Authors: Marvin Fleischmann, Ludwig-Maximilians-Universität München, Germany
Tillmann Grupp, Technische Universität Darmstadt, Germany
Miglena Amirpur, Technische Universität Darmstadt, Germany
Alexander Benlian, Technische Universität Darmstadt, Germany
Thomas Hess, Ludwig-Maximilians-Universität München, Germany

Published in: International Conference on Information Systems (ICIS 2015), Fort Worth, USA.

Abstract

Although software updates are extensively used to enhance software while already being used, their impact on users' post-adoption beliefs and attitudes has received little attention. Drawing on expectation-confirmation-theory and the IS continuance model, we investigate if and how feature updates affect users' continuance intentions (CI) and what role initial feature endowment and update size play. In an online experiment, we find a positive effect of feature updates on users' CI. According to this effect, software vendors can increase users' CI by delivering features later, through updates instead of providing them right with the first release. While this positive effect persists despite a small update size and high initial feature endowment, the latter diminishes the effect. We also unveil positive disconfirmation of previous expectations regarding the updated software as crucial mediating mechanism between feature updates and CI. Implications for research and practice as well as directions for future research are discussed.

Keywords: Feature updates, continuance intentions, feature endowment, update size, expectation-confirmation-theory, IS post-adoption theory

2.1 Introduction

In many cases, software vendors nowadays no longer sell their applications as monolithic packages but instead constantly enhance and extend their products after their first release and while the software is already in use by their customers. This is a phenomenon that is particularly prevalent in the field of consumer software such as apps for smartphones and tablet computers but also applies to desktop computer software and web services. For example, Microsoft's Office 365-Suite received 127 updates since its release in June 2011 (Microsoft 2015b). Another example is Facebook. The popular social network received over ten major software feature enhancements (e.g., keyword search in all posts and read-it-later-feature) only in 2014 (Facebook 2015). In the case of such 'agile software' (Hong et al. 2011), vendors have to make two strategic decisions regarding the implementation of software features. First, they have to decide what and how many functionalities the first release of the software should comprise, i.e., at what stage of development the software should be released. Second, after this first release, vendors have to decide how to deliver the new features that result from the ongoing development. Functionality that is delivered after the first release is usually delivered to users through free 'feature updates'. These feature updates are no discrete and standalone programs themselves but are rather integrated into the base software once they are applied to it (e.g., Dunn 2004). In practice, features are sometimes delivered in individual updates or in larger update-packages, containing several new features at once.

For vendors of agile software, it is important to understand how their customers perceive these feature updates. Because they alter software during use, feature updates may impact users' post-adoption beliefs and attitudes regarding the software and thus even affect their intentions to continue using the software (Hong et al. 2011). More specifically, the continued use of software by users (i.e., customer loyalty) has become an important goal for vendors because business models in the software industry increasingly rely on recurring revenues from subscriptions or the sale of ads and therefore a large and active user base. However, despite this common practice of extending and enhancing software during use and its potential impact on vendors' revenues, the effect of feature updates on users' continuance intentions (CI) remains largely unexplored. While there is an extensive body of research on software engineering (Sommerville 2010), software product lines (Clements and Northrop 2002), software release planning (Svahnberg et al. 2010) and software evolution and maintenance (Mens and Demeyer 2008), this primarily addresses technical considerations. Post-adoption research which explores the user's perspective, on the other hand, often tends to conceptualize information systems as a monolithic and coarse-grained black box, rather than as modular composition of specific and finer-grained features which may be altered after the software's first release (e.g., Bhattacherjee 2001). Even the few post-adoption studies that do consider changes in the software's functionality after its first release do not account for different ways of delivering features (e.g., the number of features delivered in one update) and a potential interaction with the initial feature endowment (i.e., a software's level of functionality at its first release) (Hong et al. 2011). To address these shortcomings, we will study the impact of different feature delivery strategies on users' continuance intentions regarding software in non-mandatory usage settings (e.g., software use by consumers). Specifically, we address the following research questions:

RQ1: How and why do feature updates impact users' continuance intentions regarding software?

RQ2: How do initial feature endowment and update size affect the potential impact of feature updates on users' continuance intentions?

To answer these questions, we conducted a vignette-based online experiment with 261 participants, allowing us to identify and isolate causal evidence of the effect of feature updates on users' CI. In doing so, our study contributes to post-adoption research in three considerable ways. First, we identify a somewhat counter-intuitive, positive effect on user's CI from a deferred delivery of software features. Users who receive features through updates during software use have higher CI than users who have access to all features right from the beginning of their software usage. This phenomenon seems to be robust to manipulations of update size (i.e., the number of features delivered in one update) and occurs even under a high initial feature endowment. However, the positive effect seems to diminish with increasing initial feature endowment. Second, by disclosing the relative nature of positive disconfirmation of (subjective) previous expectations regarding the software as the mediating mechanism behind this positive effect of updates, we find a possible empirical evidence for reference point dependency in users' perception of software (Kahneman and Tversky 1979). Third, we advance the understanding of IS post-adoption behavior by conceptualizing and exploring information systems as a fine-grained, malleable and dynamic collection of modular features rather than a monolithic block which is static over time. From a practitioner's perspective, our study offers important implications for vendors of consumer software. First, we describe how vendors might increase their customers' loyalty by strategically deferring the delivery of features through the use of software updates. Moreover, if holding back features is strategically not feasible (e.g., due to competition) our findings also suggest that it still beneficial to release a software early on and only subsequently roll out additional features, once they are developed. Our study also suggests that this measure should work with software that has a low initial feature endowment as well as for more mature and feature-rich software. Ruling out update size as potential boundary condition to this effect moreover implies that vendors should deliver feature innovations individually, instead of bundling them in large update packages.

2.2 Theoretical Foundations

2.2.1 Feature Updates

Consistent with previous research (e.g., Dunn 2004), software updates can be defined as self-contained modules of software that are provided to the user for free in order to modify or extend software after it has been rolled out and is already in use. Software updates are no discrete and stand-alone programs themselves, but rather integrate into the software to which they are applied. With varying terminology (e.g., update, upgrade, patch, bug fix, or hotfix), the concept of software updates is repeatedly addressed throughout the software engineering literature (Sommerville 2010). This includes software release planning, software maintenance and evolution and software product lines (Shirabad et al. 2001; Svahnberg et al. 2010; Weyns et al. 2011). In this context, software release planning or strategic release planning refers to the "*idea of selecting the optimum set of features or requirements to deliver in a release within given constraints*" (Svahnberg et al. 2010, p. 1). Following this definition, an update is the delivery of features after the first release of a software and falls within a vendor's strategic considerations regarding when to deliver what type of functionality to the user. Literature on software evolution and maintenance addresses the later stages in the software lifecycle, where updates are utilized to adjust software to changing requirements or repair emerging flaws in the software while it is already in use (Shirabad et al. 2001). In contrast to this rich stream of technical literature dealing with software updates, research on user's beliefs and attitudes regarding updates has so far been very limited (Amirpur et al. 2015). Hong et al. (2011), for example, explore users' acceptance of information systems that frequently change through the addition of new functionality. And while Benlian (2015) examines IT feature repertoires and their impact on individual task performance, he does not consider changes in these repertoires through updates. Other studies that investigate updates have often pushed them to the sidelines, treating them as control variables for studying other phenomena (e.g., Claussen et al. 2013). Existing IS research has, however, not explored the specific impact of updates on users' perceptions of an IS. Specifically, essential system characteristics such as the pre-update feature endowment of a software or the number of features in one update have so far not been explored in this context.

For the purpose of this study, we distinguish two basic types of software updates: feature updates and non-feature updates. Feature updates change the core functionality of the software to which they are applied. Functionality thereby refers to distinct, discernible features which are deliberately employed by the user in accomplishing the task or goal for which he or she uses the software. In contrast to feature updates, technical non-feature updates do not change the core functionality of software but only correct flaws or change software properties. Non-feature updates usually do not directly affect the user's interaction with the software and are typically not even visible to the user (e.g., improvements in stability, security or performance) (Popović et al. 2001). Because the core functionality is frequently utilized for accomplishing the task for which the software is used, a change in the software induced by feature updates is most often notable for users. If the software's core functionality is changed, the user's interaction with the software may also change. As we will outline later on, we argue that feature updates thus have the potential to influence users' beliefs, attitudes, and behaviors regarding the updated software in the postadoption stage of IS usage. This may even affect their decisions on continued use or discontinuation. Before further substantiating this claim, we proceed by reviewing research on IS continuance.

2.2.2 Information Systems Continuance

Post-adoption research studies users' beliefs, attitudes and behaviors after the initial adoption of an IS (Benlian et al. 2011; Bhattacherjee 2001; Karahanna et al. 1999). One of the main goals of post-adoption research is the exploration of users' information system continuance, which is defined as the "*sustained use of an IT by individual users over the long-term after their initial acceptance*" (Bhattacherjee and Barfar 2011, p. 2). To explore users' continuance behaviors, Bhattacherjee (2001) adopts expectation confirmation theory (ECT) (Anderson and Sullivan 1993; Locke 1976; Oliver 1980; Oliver 1993) and proposes a model to explain users' intentions to continue using an information system as a result of satisfaction (SAT), perceived usefulness (PU) which are in turn determined by a confirmation or disconfirmation of previous expectations regarding the software (DISC). Following ECT, the IS continuance model suggests that users compare their pre-usage expectations of an IS with their perception of the performance of this IS during actual usage (Bhattacherjee 2001). This comparison of expectations with usage experiences has also been shown to occur in later stages of use, where expectations are sequentially updated through ongoing usage experiences Kim and

Malhotra (2005). If perceived performance exceeds expectations, users experience positive disconfirmation (DISC) which has a positive impact on their satisfaction with the IS. If, on the other hand, perceived performance falls short of the expectations, negative disconfirmation occurs and users are dissatisfied with the IS (Bhattacherjee and Barfar 2011). Positive (negative) disconfirmation thus comprises two essential elements: unexpectedness and a positive (negative) experience. ECT moreover posits expectations as a relative reference point or baseline (i.e., not an absolute, objective one) upon which the user makes a comparative judgment (Oliver 1980). This idea of a subjective, relative reference point is based on Helson's (1964) adaptation level theory, which proposes that human beings perceive stimuli relative to or as a deviation from an 'adapted level' or baseline stimulus level. "*This adapted level is determined by the nature of the stimulus, the psychological characteristics of the individual experiencing that stimulus, and situational context*" (Bhattacherjee 2001, p. 354).

While applications of the continuance model have made valuable contributions in exploring postadoption phenomena, IS researchers often tend to conceptualize the studied information systems as a monolithic and coarse-grained black box, rather than as collection of specific and finer-grained features that are alterable after the first release. However, accounting for the granularity of software would help to explain how users respond to different compositions of software features and how changes in this composition through e.g., software updates after the first release might affect users' beliefs, attitudes, and behaviors regarding an information system (Benlian 2015). In addition, there are several calls for research from IS scholars who criticize the negligence of the IT artifact's role in IS research (Benbasat and Zmud 2003; Hevner et al. 2004; Orlikowski and Iacono 2001). They advocate for focusing more on changes in users' beliefs, attitudes and behaviors, emanating from the IT artifact itself rather than from other IT unrelated environmental stimuli. By studying the impact of software updates on users' continuance intentions, we account for this malleable nature of the IT artifact and address these calls for research.

2.3 Hypothesis Development

Our study is motivated by the overarching idea that the interplay between initial feature endowment and post-release update size may impact users' continuance intentions regarding an IS. The expectation confirmation mechanism (Oliver 1980) incorporated in the IS continuance model (Bhattacherjee 2001) serves as theoretical lens through which we will investigate the roles of feature endowment and update size and develop our hypotheses. When receiving updates during use of an IS, ECT implies that users' continuance intentions crucially depend on a comparison between pre-update expectations and post-update experiences with the IS. Specifically, we theorize (1) how a deferred delivery of features through updates can increase users' continuance intentions, how this might be affected by (2) initial (pre-update) feature endowment and (3) the number of features delivered in a post-release update (i.e., update size). We also hypothesize how this proposed effect is mediated through a chain of relations, initiated by a positive disconfirmation of previous expectations (4).

As highlighted in the introduction, when considering software as a flexible combination of modular functionalities, a vendor of an agile information system has to balance two crucial design parameters and their reciprocal interaction: (1) the initial set of features (hereafter called feature endowment) of a software at the time of its first release and (2) the set of features which is added to the software through updates after its first release (hereafter called update size). We investigate how a given set of features can be balanced between the initial release of a software and the later delivery through updates and how this design choice affects users' continuance intentions regarding the software. The settings which we investigate are widely used in practice (and promise interesting theoretical insights). They comprise (1) a low initial feature endowment and large update size, (2) a high initial feature endowment and a small update size and (3) a low initial feature endowment and a small update size.

Our theorizing about the differential allocation of features between the initial release and the later delivery requires the assumption that each feature provides an equal value to the user. This is necessary, because features with different levels of importance could interfere with our attempts to conceptually isolate the effects of different feature allocations between initial release and later update on users' CI. While being restrictive, this is a necessary assumption for identifying the proposed causal effects. In addition, we assume that the number of planned features for a certain time period is predetermined. This time period may vary depending on the planning period of a vendor in which release decisions are taken. Moreover, because we investigate users' perceptions during usage, we focus on feature updates with explicit user notification and neglect silent, background updates which are unnoticed by the user. Lastly, we will focus on updates for consumer software applications with an unobtrusive update process (in contrast to e.g., some desktop operating system updates processes where system usage may be severely disturbed by updates).

2.3.1 The Effect of Feature Updates on Users’ Continuance Intentions

We argue that feature updates, which provide additional functionality that directly serves users in accomplishing their IS-based tasks, will be perceived as a *positive* experience with the software. Furthermore, it is reasonable to assume that feature updates are usually not anticipated by users and are thus *unexpected* experiences with the software. According to ECT, if feature updates are perceived as *unexpected* and *positive* experiences during usage, they should consequently induce perceived positive disconfirmation (Anderson and Sullivan 1993). Following the IS continuance model (Bhattacherjee 2001), it is plausible to assume that this perceived positive disconfirmation during software use will increase users’ CI regarding the updated software. We will further elaborate on these arguments below.

In the context of software features, ECT also implies that positive disconfirmation from a feature update results from a *relative* change in functionality compared to the user’s subjective reference point (i.e., the pre-update configuration of the software) rather than an absolute change (Helson 1964; Oliver 1980). A software vendor should thus be able to induce positive disconfirmation and therefore increase the user’s CI by applying the strategy of simply holding back features (functionality) in the first release of the software and delivering this functionality only later on, through feature updates (Sen and Morwitz 1996). Under this deferred feature delivery strategy, a feature-complete software package can be designed and developed by the software vendor, but certain features might be removed from the initially shipped version of the software. The user is usually unaware of the existence of these remaining features. Once they are delivered through an update, they likely elicit positive disconfirmation because the user may perceive them as a ‘gift’ from the vendor. Consistent with the IS continuance model, this could then lead to an increase in CI (Bhattacherjee and Barfar 2011). This deferred feature delivery strategy is thus to be distinguished from an all-at-once feature delivery strategy under which all developed features are delivered in the first release. Nonetheless, both feature delivery strategies are assumed to comprise the same type and number of features overall.

Regarding our assumptions about feature updates, we acknowledge that in practice, there might be cases, where feature updates are perceived *negatively* by users. For example, if features are intentionally removed (e.g., because of expired licensing deals), software functionality is unintentionally impaired or if updates bring major changes to the software which necessitates users to learn and adjust (Polites and Karahanna 2012; Mukherjee and Hoyer 2001). Nevertheless, we argue that in most cases, feature updates are intended to enhance the software with regard to its core purpose and are thus perceived positively (Larsen et al. 2009). Furthermore, it is reasonable to assume that users perceive updates as *unexpected* events during usage. In many cases, updates are not announced beforehand and even if software vendors announce release plans about future updates, most end-users (and consumers in particular) likely do not follow them in detail. Moreover, if a feature is delivered through an update, it may 'stick out' more than if it is included in the first release where it is one among many other features. The positive perception of this feature received through an update and its effect on CI may thus be increased even further. To summarize, because of the nature of the disconfirmation mechanism in ECT, which operates through an evaluation of relative instead of absolute change (Oliver 1980), users who receive functionality through feature updates will likely have a higher intention to continue using a software than users who received all these features right with the first release. Accordingly, we propose our first hypothesis:

H1: Software that receives functionality through feature updates induces a higher continuance intention compared to software that includes all this functionality right with the first release.

2.3.2 The Role of Initial Feature Endowment

As discussed before, vendors of agile software can decide what features to include in the first release and what features to deliver only later on, after the first release. Thus, in the first release, different levels of feature endowment are possible. A vendor might decide for a small initial feature set or for a more comprehensive set of features. In both cases, additional features may be added to the software through updates after its first release. Then, the possibility arises that the previously proposed positive effect of feature updates on users' continuance intentions might only occur under a low initial feature endowment and that it might disappear under high initial feature endowment. One reason for this that the addition of one feature to an already well-endowed software might be perceived as negligible by users.

As implied by ECT, the positive effect of feature updates on users' continuance intentions requires—in addition to unexpectedness—a positive experience (Oliver 1980). In the context of software features, this experience is positive when it exceeds previous expectations regarding the functional capabilities of the information system (i.e., the subjective reference point) (Helson 1964). Users form this *subjective* reference point based on *their* overall pre-update experience with the software. We acknowledge that competing software could serve as *objective* reference for user's evaluations of feature endowment. However, in practice, competing software is usually sufficiently differentiated with regard to functionality (i.e., does not have *identical* functionality) to prohibit a direct and objective comparison of feature endowment. Moreover, users (and consumers in particular) likely do not know about each individual feature that is available in every other software product on the market. Furthermore, feature updates usually provide functionalities which are unique and serve a distinct purpose, making them different from any other feature that is already included in the initial release of the software. Because disconfirmation is based on a relative comparison of the pre-update and post-update state of the software, *any* feature update that improves the functional capabilities of the software will likely induce positive disconfirmation—independent from the factual feature endowment of the software in its pre-update state (Anderson and Sullivan 1993). We thus suggest that even if software has a high initial feature endowment and receives a feature update, users' CI will nonetheless increase compared to software that provides all these features right with the first release. To summarize, from the perspective of an objective evaluation one might expect a different outcome, but the outlined theory suggests that the positive effect from a deferred delivery of features through updates persists despite a high initial feature endowment. We argue that this is the result of the interplay between users' subjective evaluations of feature endowment, the relative nature of the ECT mechanism and the functional uniqueness of features. We thus hypothesize:

H2: The higher continuance intention induced by feature updates is independent of the initial feature endowment of the software.

2.3.3 The Role of Update Size

From an objective point of view, one might also expect users to value larger update packages that comprise many features (i.e., large update size) more than smaller update packages which contain only a few features (i.e., small update size). However, as stated above, ECT implies that positive disconfirmation depends on a *relative* change in functionality compared to the user's reference point rather than on an absolute change (Helson 1964; Oliver 1980). In the ECT mechanism, this subjective evaluation does not only comprise the pre-update state of the software which forms the baseline for the comparison (i.e., the reference point) but also the *post-update* state of the software and the evaluation of the feature update itself (Oliver 1980). We thus argue that while users will likely perceive a feature update as a 'free gift' from the vendor, they are not likely to evaluate the magnitude of its functional value objectively. Because users do usually not expect an update and have no objective comparison regarding the comprised features, any update—irrespective of the number of contained features—will likely induce the positive effect as suggested in hypothesis 1. Moreover, we argue that an objective evaluation of an update through the comparison with similar, competing software as discussed in hypothesis 2 is even less likely to occur in practice. This is, because competing software differs even more with regard to the timeframes and extent of feature delivery through updates than it does with regard to initial feature endowment. Therefore, we argue that it is likely that users will not be able to acknowledge the distinction between large and small update sizes. We suggest that the magnitude of the effect of disconfirmation should thus not depend on update size. To sum up, it is reasonable to assume that the number of features in an update is likely to be neglected when users' continuance intentions are formed after an update. We thus hypothesize:

H3: The higher continuance intention induced by feature updates is independent of the update size.

2.3.4 The Mediating Effect of Disconfirmation

As outlined before, we suggest that feature updates work through a disconfirmation of previous expectations regarding the software (Oliver 1980; Anderson and Sullivan 1993). In terms of the continuance model, disconfirmation should thus mediate the effect of feature updates on continuance intentions (Bhattacherjee 2001). The IS continuance model further posits that this disconfirmation in turn impacts perceived usefulness and satisfaction of an information system (Bhattacherjee and Barfar 2011). Both, satisfaction and perceived usefulness will furthermore drive users' intentions to continue using an updated IS. We therefore argue that the higher levels of CI regarding software that receives functionality through feature updates compared to software that includes all features right with the first release, will likely be the result of the mediating effect of DISC and in turn affect the downstream variables of the IS continuance model (i.e., perceived usefulness and satisfaction). We thus hypothesize:

H4: The positive effect of feature updates on continuance intentions is mediated by disconfirmation of initial expectations regarding the software.

2.4 Method

2.4.1 Experimental Design

To examine the effect of feature updates on users' CI and the roles of initial feature endowment and update size as suggested by our hypotheses, we conducted a vignette-based online experiment with manipulations of update size (small vs. large) and initial feature endowment (low vs. high). Additionally, we controlled for the user's construal level. Construal level is an individual's mental mode of decision making and has been shown to crucially influence decisions depending on the subject matter's perceived levels of abstraction and detail (Trope and Liberman 2003). According to construal level theory, individuals with high construal levels think more abstractly, focus on bigger picture concerns and show less myopia by pursuing long term goals compared to individuals with low construal levels. These individuals are more focused on short term interests and are subject to myopia (Fujita et al. 2006; Mehta et al. 2014; Wan and Agrawal 2011). We deliberately controlled for these different ways in thinking because we sought to show that our hypothesized negligence of initial feature endowment (H2) and update size (H3) even holds true for these two modes of thinking which seem likely to affect the perception of different allocations of features (i.e., software capabilities) between initial release and later updates. Specifically, individuals with high construal levels and long term orientations might favor software that receives features through updates over its usage cycle. A short term orientation from low construal levels, on the other hand, might lead to a preference of high initial feature endowments because this might satisfy short term goals.

To account for the different, feasible delivery strategies for a given set of features as proposed in our hypotheses, we defined one control group and three incomplete factorials. The groups had (A) all features right with the first release and no updates (control), (B) low initial feature endowment and large update size, (C) high initial feature endowment and small update size, (D) low initial feature endowment and small update size. For each of these groups, construal level (low vs. high) was additionally manipulated though textual vignette treatments. The used software and the task for which the software had to be used were deliberately held constant across all conditions. Overall, this lead to the 4x2 between-subjects design depicted in Figure 2-1. Subjects were randomly assigned to one of the eight groups. We realized the manipulations of feature endowment, update size and construal level by presenting participants with carefully constructed textual scenarios (vignettes) that precisely described a person (user), task and software characteristics (vignette setting), software usage and a conditional update during usage (vignette usage) (see Figure 2-1). The experimental vignette methodology (EVM) (e.g., Aguinis and Bradley 2014) was used because it provided us with the possibility to control for the user's construal level, avoid social desirability bias and eliminate undesired learning effects of participants. Even though this method comes with some downsides such as simplifications and constructed hypothetical usage scenarios, it also enabled us to isolate and precisely manipulate the dependent variables while nevertheless accurately identifying the hypothesized causal relationships. Compared to a laboratory experiment that e.g., uses simplified prototype software as a treatment, we favored vignettes to achieve a high external validity by being able to design a realistic scenario. Researchers in IS and other disciplines have repeatedly shown that individuals respond quite similarly whether they are presented with a hypothetical situation using vignettes or a hypothetical situation in a traditional laboratory experiment. Therefore a scenario based manipulation can be assumed to work appropriately if constructed carefully (De Cremer et al. 2007; Dennis et al. 2012; Rahman 1996; Shaw et al. 2003). This makes the method suitable for our needs.

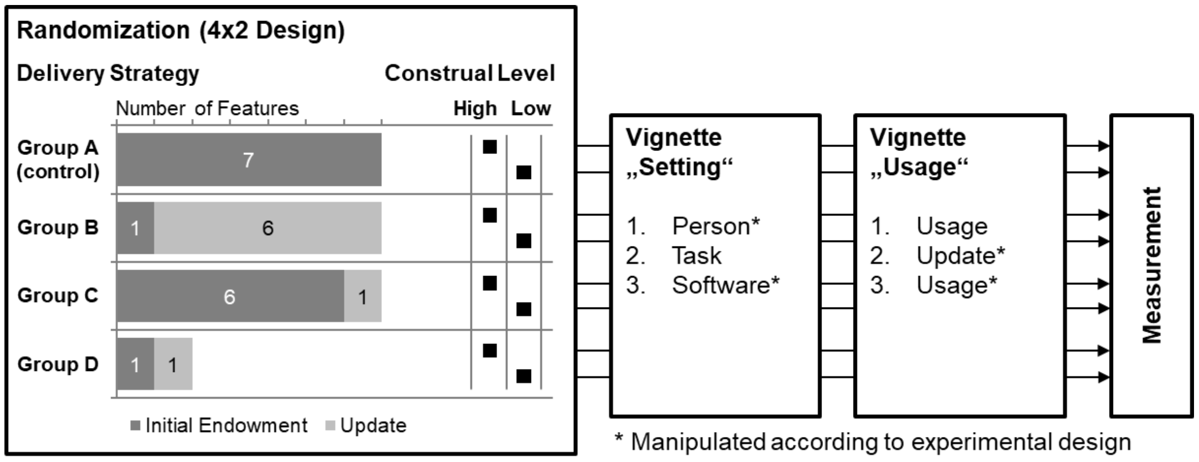


Figure 2-1: Experimental Procedure

The experiment proceeded in four major steps: First, upon arrival at the website, subjects were told to read instructions carefully and to answer questions to the best of their knowledge, followed by questions about subjects' motivation to process information. Second, subjects were randomly assigned to one of the eight experimental groups and then presented with the corresponding vignettes (see Figure 2-1). We instructed subjects to carefully read the vignettes and put themselves in the hypothetical setting described in the scenario, before answering the subsequent questions. The vignettes described a user in a high or low construal state of mind (person), a travel booking task (task) and a travel booking platform (TBP) including its initial feature endowment depending on the experimental condition (software). Third, on the next page, subjects were presented with part two of the vignette. This part described how the person in the scenario uses the TBP to accomplish his task. Halfway through the usage time, the availability of new functionality through an update was described according to the respective experimental condition (not applicable for the control group). After the update, this part of the vignette ended with further description of TBP usage up until the task was completed. Finally, a post-experimental survey on the following pages asked subjects to respond to questions measuring their evaluation of the CI of the person described in the scenario with regard to the TBP and all further variables (see Measures). On the last page of the survey, subjects were debriefed and thanked for their participation.

2.4.2 Manipulation of Independent Variables

Regarding the described software, for our experiment we opted for an online travel booking platform because we wanted to ensure that subjects had previous usage experience and would thus understand the outlined scenario quickly and well. Moreover, we had the goal that the findings regarding updates which we obtained from this exemplarily setting could be generalized across a wide range of software types. The choice of an online service (i.e., the travel booking platform) allowed us to isolate the effect of receiving features through updates from other intervening factors such as waiting times or technical difficulties during the installation of updates which might occur on e.g., desktop computers (Sykes 2011; Tyre and Orlikowski 1994). These may have traditionally been issues associated with updates. However, we argue that such technical downsides of software updates have been reduced over time and have mostly disappeared in online services (e.g., Facebook), platforms for mobile devices (e.g., Apple iOS) and modern desktop operating systems (e.g., Microsoft Windows 10) where updates are now mostly unobtrusive and frictionless. We are thus confident that with the described travel booking platform, we can derive viable implications that are generalizable across a large part of the modern software market.

The specific task was booking a two-week vacation including flight and hotel with a limited monetary budget and further constraints which fostered the use of the individual features on the TBP. The person described in the scenario was a student called ‘Tom’ and the travel booking task was chosen to be typical for a student. To construct the different stimuli with respect to update size and initial feature endowment, we identified features of a TBP that satisfied three criteria: 1) they needed to be self-explanatory, 2) they had to be perceived as useful for the task by participants, 3) when absent, the TBP still needed to be functional and the general task—while being more difficult—could still be accomplished. Through interviews and research, we compiled 22 features that meet these criteria. The importance of each feature was evaluated in a pre-study[4]. Eventually, seven features were identified as

[4] 20 subjects participated in the pre-study. They resembled the demographics of the main study and rated 22 identified TBP features with regard to perceived importance on seven-point-Likert scales. Seven features with similar but high levels of importance were selected for the main study. Holding the importance of features constant within and across treatments allowed us to isolate the effects of initial feature endowment and

appropriate and relevant to establish the different update and endowment stimuli. The features were: *rating an accommodation with stars (5 levels); viewing the average rating*; *filtering for price, rating etc.; sorting for price, rating etc.; calendar functionality to plan arrival, departure and duration of stay; viewing professional holiday reviews; a budget calculator to find and plan fixed budget vacations*. To implement our required assumption regarding an equal value of the employed features (which we raised in our hypotheses), we deliberately defined the task in a way so that it could—in principle—be accomplished without using any of the manipulated features. Nonetheless, each feature made the accomplishment of one part of the task or a specific constraint easier for 'Tom', thus providing approximately equal benefits. The vignettes specifically described the usage of each available feature in each condition in order to highlight the benefit of each feature. Regarding the assignment to initial endowment and later update, the order of the seven features was random (as listed above) but held constant across the experimental groups. Group A had all seven features right from the beginning; group B had one feature right from the beginning (*rating an accommodation with stars*) and the remaining six features were added through an update; group C had the first six features right from the beginning and one feature was added through an update (*budget calculator to find and plan fixed budget vacations*) and group D had one feature right from the beginning (*rating an accommodation with stars*) and one feature was added through an update (*viewing the average rating*). Figure 2-1 depicts this assignment of features.

To operationalize the manipulation, we constructed textual scenarios and presented them to participants in an online based questionnaire that comprised several consecutive pages. On a first page, we described Tom and his personality. For a high construal level mindset, we described Tom as a person *'who is considering the big picture for making decisions', 'who establishes an overview on superior goals', 'who makes gut decisions and focuses on essentials',* for a low construal level mindset we described Tom as a person *'who wants to consider all details before deciding', 'who establishes a broad information basis to consider all facets of a problem', 'who wants to decide rational and therefore focuses on details'*. Subsequently, we introduced a task of finding a cheap 2-week vacation to Madrid, Spain, with a price limit of 800 Euros, full board, clean restrooms and, among other things, a modern ambience. Third, the software and its initial feature endowment were described as follows: *'To find a suitable flight and hotel, he [Tom] uses the TBP Journey4You. In addition to the*

update size on the dependent variables and avoid potential confounding effects that might result from variations in the importance of features.

simple search for flights and hotels, the platform offers the following functionalities:' followed by the abovementioned features. On the second page—part two of the vignette—we first described Tom's repeated visits of the TBP over several days to find a suitable flight and hotel including the usage of the currently available features. For the three groups which received an update, we subsequently included a section that introduced the update after Tom revisits the TBP one day (halfway through the described usage time) *'... the website Journey4You shows a message that Journey4You offers new functionality to its users:'* followed by a list of one or six of the features. After this conditional section describing the update, a description of further usage including the use of the new features (only in the update conditions) followed (see Figure 2-1): *'Tom uses the new function in addition to available functionality [...]. Finally, Tom locates the most attractive offer and books his journey'*. Except for the manipulated text passages, all other parts of the scenario were kept constant across the groups. After this second page, participants started to answer the questionnaire. Participants could only proceed to the next page when all questions were answered and returning to previous pages, including the vignettes, was not possible. Following common vignette procedures (Aguinis and Bradley 2014), we ensured that our vignettes illustrated realistic situations and that participants identified well with the character described by conducting several revision cycles based on qualitative interviews. Furthermore, the so designed vignettes were tested in a pilot study with ten subjects to ensure that our treatments were manipulated according to the experimental design (Perdue and Summers 1986). Specifically, subjects of the pilot study were asked about the comprehensiveness of the instructions, the vignettes' realism and their ability to put themselves in the hypothetical scenario as well as the clarity of questions in the subsequent questionnaire. Feedback and suggestions were obtained from participants and the vignettes and the questionnaire were accordingly revised for the main experiment.

2.4.3 Measures

Dependent Variables

For the questionnaire that succeeded our vignettes, we used validated scales to measure dependent variables with slight changes in wording to adapt the items to our experimental setting. Unless stated otherwise all items were measured on seven-point-Likert scales anchored at 1 = strongly disagree and 7 = strongly agree. CI was measured with items adapted from (Hong et al. 2011): CI1. *Tom intends to continue using the TBP rather than discontinue its use;* CI2. *Tom's intentions are to continue using the TBP than use any alternative means;* DISC was based on items adapted from Bhattacherjee (2001): DISC1. *Tom's experience with* *using the TBP were better than what he expected;* DISC2. *The service level provided by the TBP was better than what Tom expected;* DISC3. *Overall, most of Tom's expectations from using the TBP were confirmed;* While we did not explicitly hypothesize about SAT and PU, we also included these variables in our measurement, to capture the entire continuance model. PU and SAT were measured with items from Kim and Son (2009): PU1. *Using the TBP enhances Tom's effectiveness;* PU2. *Using the TBP increases Tom's productivity;* PU3. *Using the TBP improves Tom's performance;* SAT1. *Tom is content with the features provided by the TBP;* SAT2. *Tom is satisfied with the features provided by the TBP;* SAT3. *What Tom gets from using the TBP meets what he expects for this type of software.*

Control Variables and Manipulation Check

In our study, we controlled for subjects' expertise with regard to TBPs with an established four item, seven-point semantic differential scale with the items *know very little about/know very much about*, *inexperienced/experienced*, *uniformed/informed*, *novice buyer/expert buyer* (Mishra et al. 1993). Furthermore, we also captured participants' *motivation to process information* with one item to control for motivational influences on response behavior (Suri and Monroe 2003). Additionally, in the post experimental survey, we measured (1) subjects' level of understanding of the scenario, (2) the scenario's realism, (3) subjects' ability to put themselves in the hypothetical setting described in the scenario, (4) subjects' level of understanding of the instructions and questionnaire items. We further collected the participants' age, gender, education and profession to control for a homogeneous distribution of participants across groups with regard to these core demographics.

2.4.4 Participants, Incentives and Procedures

The outlined online experiment was conducted between November 2014 and January 2015. In line with best-practices of augmented number and diversity of participants for vignette studies (Aguinis and Bradley 2014), we invited subjects to participate in an online survey about software usage by consumers through several postings in social networks, via word-of-mouth and emails. Overall, 386 subjects started the experiment. The rate of completion was 74%, i.e., a total number of 285 subjects completed the questionnaire. 24 participants were excluded from our final analysis because they either did not answer control questions correctly or completed the experiment in less than five minutes (the average time to completion was about ten minutes). Of the 261 remaining participants that were used in the following analysis, 107 were females and 153 were males (one not specified). Subjects' average age was 28.47 (SD=9.08) years. On average, 78% of the subjects used TBPs less than one hour per month, 20% one to five hours and 2% more than five hours per month. The average reported expertise with a TBP was 4.26 (SD=1.63) on a seven-point semantic differential scale. 48% of subjects were students, 29% employees, 8% self-employed and the remainder had various occupations.

2.5 Data Analysis and Results

2.5.1 Control Variables and Manipulation Check

To confirm a successful randomization of participants to the different experimental conditions, we searched for differences between groups with regard to the collected control variables by performing a one-way MANOVA. The results showed no significant differences between groups $\lambda = 0.90$, F[33,717]= 0.78, p>0.1. Neither control variable showed significant differences: age (F=0.72, df=3, p>0.1), gender (F=1.40, df=3, p>0.1), occupation (F=1.38, df=3, p>0.1), usage intensity (F=0.54, df=3, p>0.1), product expertise (F=0.91, df=3, p>0.1), motivation to process information (F=1.21, df=3, p>0.1), understanding of story (F=0.24, df=3, p>0.1), story's realism (F=1.31, df=3, p>0.1), ability to put oneself in the scenario (F=0.14, df=3, p>0.1), understanding questions (F=0.23, df=3, p>0.1) and instructions (F=0.17, df=3, p>0.1). Hence, we conclude that subjects were distributed homogenously across our experimental groups. As indicators for the external validity of our findings, we reviewed the participants' answers regarding their understanding, realism and adaption of the vignettes. For all three measures, the participants reported high levels on the seven-point-Likert-scales: understanding (M=6.32; SD=1.00), realism (M=5.84; SD=1.31) and adaption (M=5.77; SD=1.36). Moreover, in qualitative open text questions we observed that subjects described Tom's instead of their own feelings and thoughts. It is therefore reasonable to assume that our experimental manipulations using textual vignettes worked as intended and that participants thought and acted like the fictitious character. To control for potential differences in the effect of updates on CI from different construal levels, we ran two one-way ANOVA tests. We assessed whether there were any differences in CI between high and low construal conditions across the control group and three update groups. The results indicated no significant differences for CI in the control group (F=1.16, df=1, p>0.1) and all three update groups (F=0.06, df=1, p>0.1). We may therefore conclude that construal level did not interact with the effect of feature updates on CI. In our subsequent analysis we thus combined participants who received high and low construal level treatments.

2.5.2 Measurement Validation

Because we adopted established constructs for our measurement, confirmatory factor analysis (CFA) was conducted to test the instruments' convergent and discriminant validity (Levine 2005). Table 2-1: Results of Confirmatory Factor Analysis for Core Variables reports the CFA results regarding convergent validity using SmartPLS 3.0 (Chin et al. 2003; Ringle et al. 2014).

Table 2-1: Results of Confirmatory Factor Analysis for Core Variables

Latent Construct	Items	Range of std. Factor Loadings*	Cronbach's alpha	Composite Reliability (ρ_c)	Average Variance Extracted
Continuance intention (CI)	2	0.925 - 0.929	0.836	0.924	0.859
Disconfirmation (DISC)	3	0.779 - 0.879	0.793	0.879	0.708
Perceived Usefulness (PU)	3	0.783 - 0.870	0.796	0.879	0.709
Satisfaction (SAT)	3	0.772 - 0.920	0.840	0.905	0.761
* All factor loadings are significant at the p<0.01 level					

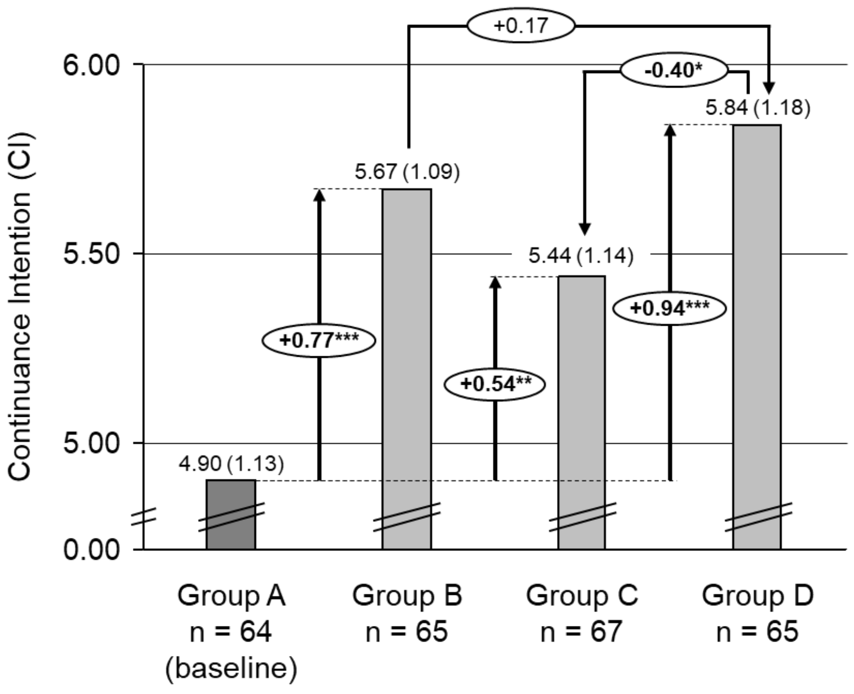
Constructs were assessed for reliability using Cronbach's alpha (Cronbach 1951). Values above 0.70 are considered to provide adequate reliability (Nunnally 1994). The alphas for all constructs were well above 0.7. Moreover, the composite reliability of all constructs exceeded 0.70, which is considered the minimum threshold (Hair et al. 2011). Values for AVEs for each construct ranged from 0.708 to 0.859, exceeding the variance due to measurement error for that construct (that is, AVE exceeded 0.50). Moreover, we examined cross correlations (see Table 2-2: Means, Standard Deviations and Correlation Matrix for Core Variables). All square roots of AVE exceeded inter-construct correlations, providing strong evidence of discriminant validity. Hence, the constructs in our study represent concepts that are both theoretically and empirically distinguishable. After ensuring the validity of our measured constructs, summated scales based on the average scores of the multi-items were used to calculate the constructs for our later analysis (Zhu et al. 2012).

Table 2-2: Means, Standard Deviations and Correlation Matrix for Core Variables

Latent Construct	M	SD	1	2	3	4
(1) Continuance intention (CI)	5.644	1.183	**0.927**			
(2) Disconfirmation (DISC)	5.450	1.028	0.458***	**0.841**		
(3) Perceived Usefulness (PU)	5.393	1.159	0.484***	0.549***	**0.842**	
(4) Satisfaction (SAT)	5.865	0.973	0.478***	0.654***	0.608***	**0.872**
Note: Bolded diagonal elements are the square root of AVE. These values should exceed inter-construct correlations (off-diagonal elements) for adequate discriminant validity; ***p<0.01, **p<0.05, *p<0.1.						

2.5.3 Hypotheses Testing

In order to test our hypotheses H1 - H3, we conducted a one-way ANOVA with planned contrast tests using IBM SPSS Statistics 22. A significant effect of feature updates on CI was found (F=8.362, p<0.01). Post hoc contrast analysis revealed that participants in all three update groups (B, C and D) showed significant higher levels of CI compared to the control group (see Figure 2-2).



Note: ***p<0.01, **p<0.05, *p<0.1 (one-sided); ANOVA-test with planned contrast analysis

Figure 2-2: Average Continuance Intentions, Mean Differences and Significance Levels

In hypothesis H1, we argued that software that receives additional functionality via feature updates will induce higher CI compared to software that includes all these features right with the first release. The results from our experiment indicate that participants' CI in group B (one initial feature, update adds six features) was on average significantly higher (+0.77) than in group A (seven initial features, no updates). Hence, H1 is supported. Hypothesis H2 posits that this effect would persist, regardless of the initial feature endowment. As hypothesized, our results showed that participants' CI in group C (six initial features, update adds one feature) was on average significantly higher than in group A (+0.54). Furthermore, H2 implies that the increase in user's CI should not be lower when an update is applied to software with high initial feature endowment (compared to software with low initial feature endowment). However, the results from our experiment indicated that this idea is not supported, since subject's CI in group C was on average significantly lower (-0.40) than in group D (one initial feature, update adds one feature). H2 is thus only supported partly. A discussion of this finding follows in the next section. In hypothesis H3 we proposed that the positive effect of feature updates on user's CI (compared to software that includes all these features right from the first release) persists regardless of the update size. Supporting this hypothesis, our results showed that subject's CI in group D was on average significantly higher (+0.94) than in the control group A. Moreover, hypothesis H3 also implies that the increase in user's CI should not be higher for a feature update with large size compared to a small update size. Since subject's CI in group D (one initial feature, update adds on feature) was not significantly different than in group B (one initial feature, update adds six features), we conclude that H3 is fully supported.

Finally, to evaluate the explanatory mechanism behind the impact of feature updates on CI, we conducted a mediation analysis using partial least squares structural equation modeling with SmartPLS 3.0 with the bootstrapping resampling procedure (Ringle et al. 2014). In line with previous post-adoption continuance studies (Bhattacherjee 2001; Bhattacherjee and Premkumar 2004; Kim and Son 2009; Ortiz de Guinea and Webster 2013), a component based structural equation modeling approach using SmartPLS was preferred over a covariance-based one because it does not impose sample size restrictions or require multivariate normality distributions of the underlying data. A complete bootstrapping with 10,000 samples was conducted, bias-correction was enabled and test type was set to two tailed. The validation of the employed reflective measurement model is reported in Tables 2-1 and 2-2 (Chin et al. 2003). Following hypothesis H4, we included our experimental treatment (no update vs. feature update) as dichotomous independent variable and driver of DISC in the continuance model (Bhattacherjee 2001) (see Figure 2-3).

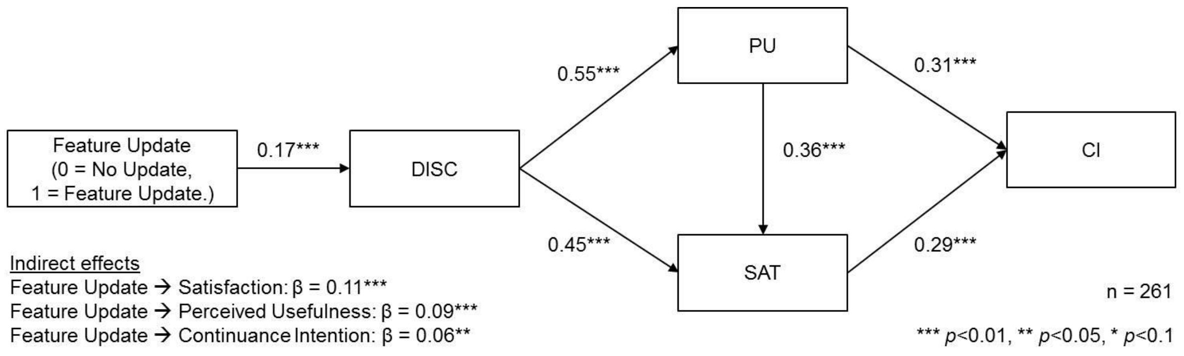

Feature Update
(0 = No Update,
1 = Feature Update.)
0.17***
DISC
0.55***
0.45***
PU
SAT
0.36***
0.31***
0.29***
CI
Indirect effects
Feature Update → Satisfaction: β = 0.11***
Feature Update → Perceived Usefulness: β = 0.09***
Feature Update → Continuance Intention: β = 0.06**
n = 261
*** p<0.01, ** p<0.05, * p<0.1

Figure 2-3: Feature Updates Disconfirming prior Expectations Regarding Software

Following Hair et al. (2014), our analysis revealed significant paths between all core variables of the continuance model (Bhattacherjee 2001). Overall, the continuance model was strongly supported and explained about 28% of the variance in continuance intention (R^2=0.282). Moreover, the positive and highly significant path from feature update to DISC (β = 0.17, p<0.01), supports our hypothesis H4 which suggested that the positive effect from feature updates on CI is partially carried over by DISC to the downstream factors of the IS continuance model to affect CI. In order to further examine the mediation by disconfirmation, additional models were tested by including direct links from feature update to continuance intention (Venkatesh 2000). The effect of feature update on continuance intention was partially mediated by disconfirmation (and the subsequent variables perceived ease of use and satisfaction) (Hair 2014).

2.6 Discussion

This study sought to achieve three main objectives: (1) examine the effects of feature updates on users' intentions to continue using an information system (i.e., whether there is a discernible effect from updates), (2) to investigate two possible boundary conditions (i.e., when there is an effect from updates and when not), namely, update size and initial feature endowment and (3) to unravel the explanatory mechanism through which such an effect occurs (i.e., how and why such an effect from updates occurs). To achieve these objectives, we drew on the IS continuance model that is embedded in the Expectation-Confirmation theory and investigated our hypotheses based on a vignette-based online experiment.

Drawing on the advantages of the experimental method, which allows to isolate the effects of manipulated stimuli on user responses from other confounding variables to unveil causal relationships, we found that CI was rated significantly higher in all update-conditions (groups B, C and D) than in the non-update condition (A). This increase in CI can be interpreted as a somewhat counter-intuitive finding because in the groups in which features were delivered via updates only halfway through the described time span, users had access to less functionality to accomplish their task compared to the user who had all features right at the beginning: They had to use the software prior to the update with less features. In particular in group D, even after the update, the user had in sum fewer features to accomplish his task compared to group A. Despite this objective disadvantage, participants in all update groups, including D, indicated significantly higher scores in CI. This suggests the presence of a positive, somewhat non-rational user response to feature updates and challenges the idea of a ‘rational user’ in the IS continuance literature (Bhattacherjee and Barfar 2011; Ortiz de Guinea and Markus 2009; Ortiz de Guinea and Webster 2013).

Moreover, our experiment revealed that update size does not seem to constitute a boundary condition to this positive effect of feature updates on users’ CI (groups B and D did not differ significantly). However, contrary to our hypothesis and to what the continuance model and the underlying ECT mechanism would suggest regarding the subjective and relative evaluation of feature endowment and updates, groups C and D differed significantly with regard to CI. Feature endowment therefore appears to moderate the effect of feature updates on CI. While this finding partly contradicts our second hypothesis, the observed effect may be explained by the concept of diminishing sensitivity (Tversky and Kahneman 1992). This concept suggests that the characteristics of a product to which a new feature is added determine the impact of this feature on e.g., the sales of the product. Specifically, a feature that is added to a relatively superior product increases the overall perceived value of the product less than the same feature that is added to a relatively inferior product (Nowlis and Simonson 1996). Nonetheless, we suggest further research to substantiate our interpretation of this finding. Additionally, we could demonstrate that the positive effect of feature updates on CI was mediated by a serial effect chain of relations that originates in a positive disconfirmation of previous expectations (DISC). This emphasizes the relative nature of users’ evaluations of changes in software features and validates the IS continuance model for IS that are conceived as a dynamic collection of features rather than one monolithic and static block. These changes in beliefs and attitudes over time which are induced by changes in the IT artifact may be explained by sequential belief updating (Kim 2009; Maier et al. 2015).

Lastly, our experiment revealed that these findings even persist under different user's construal levels, which are modes of thinking that seemed likely to affect the perception of the allocation of features between initial release and later updates.

2.6.1 Implications for Research

This study makes three contributions to literature. First, our main contribution lies in the detection of a positive user reaction to feature updates. Specifically, delivering software functionality through feature updates has a stronger and more positive impact on IS users' CI than providing the entire feature set at once and right with the first release. We reveal that users evaluate software functionality not objectively and that evaluations of feature updates are based on relative comparisons to a subjective baseline of functionality. While this effect persists despite a high initial feature endowment, its magnitude diminishes with increasing endowment. This diminishing sensitivity to new features is consistent with findings from psychology and marketing research (Nowlis and Simonson 1996) and should be considered when studying users' perceptions of software features. With regard to update size, we find that the positive effect of feature updates is independent from different update sizes. This implies that users do not assess changes in an information system through updates objectively. Moreover, in our study we could empirically demonstrate that the observed effects even withstand different construal levels, a crucial user characteristic with respect to preferences of initial over later benefits. The lack of a significant influence of construal level further substantiates the robustness of our findings. Our second contribution lies in shedding light on the explanatory mechanism behind the identified positive effect of feature updates on CI. In particular, we find that this positive effect is mediated by a serial chain of relations which originates in the positive disconfirmation of previous expectations. This finding highlights the subjective and relative nature of users' perceptions of IS changes which lead to somewhat non-rational responses (Fleischmann et al. 2014). These results may also be interpreted as a possible empirical evidence for reference point dependency in users' perception of software (Kahneman and Tversky 1979) and deserve further research. Our third and overarching contribution lies in the extension of the predominant view of information systems in the post-adoption literature from a mostly monolithic one to a finer-grained and dynamic perspective by showing how modular features can be strategically combined by vendors and that the specific composition of features and their changes over time can influence users' beliefs and attitudes regarding a software. In doing so, we answer calls of several IS scholars (e.g., Benbasat and Barki 2007; Jasperson et al. 2005) to consider the granularity of information systems in IS research. Our study thus offers a complement to the existing IS post-adoption literature by showing that user beliefs and attitudes change alongside modifications of the IT artifact during usage. Moreover, through this notion, our study also contributes to the stream of IS research on belief updating (Kim 2009; Maier et al. 2015).

2.6.2 Implications for Practice

Our results also have important implications for practice. First, despite the extensive use of feature updates by vendors to maintain, alter and extend their products after they have already been rolled out, it is surprising that insights on how these updates are perceived by users are still scarce. This leaves vendors without guidance *when* to provide *which* functionality to customers. From the results of our experimental study we can conclude that it might be advisable for vendors to deliver features over time, via updates, because feature updates can induce a positive experience, which, in turn, increases users' CI. For vendors, users with a high CI are a particularly desirable goal because these are the loyal, returning customers who ensure the long term profitability of their businesses in the highly competitive software industry. Moreover, a high CI is particularly important for the increasing share of subscription-based business models in the software industry (Veit et al. 2014). An incremental delivery of features may also be beneficial to vendors, providing them with a competitive advantage due to shorter times-to-market when developing new software. Instead of waiting for the completion of all planned features, they could release their software with a smaller feature endowment and deliver additional functionalities successively through incremental feature updates when their development is completed. An additional benefit of this quicker time-to-market strategy is that revenues start to flow earlier than under an all-at-once feature delivery strategy with a later release. However, the identified positive effect of feature updates needs to be well understood and correctly applied to achieve the desired outcomes. The findings of this study reveal that this effect works only if users' experiences exceed their prior expectations when receiving an update (positive disconfirmation). Vendors should thus avoid announcing feature updates in advance as this would annihilate the required element of surprise. Our results regarding initial feature endowment show that while this positive effect from updates decreases with endowment it still persists. A deferred delivery of features as suggested by our hypothesis H1 may thus be applied for lean software as well as for software with a high initial feature endowment. Vendors of advanced, mature software may therefore also take advantage of this effect. Furthermore, because the size of updates was revealed to not affect this positive effect, it is not necessary for vendors to pack several features into one update in order to obtain this positive user response. However, vendors should not overdraw holding back functionality. Starting out with a too small feature set might render the first release of a software almost useless and lead to discontinuation before the program can be updated or even prohibit the adoption in the first place. Finally, our findings highlight the benefit from using a modular software-architecture. Aside from an increased flexibility in the development and maintenance, a modular architecture also facilitates the use of feature updates. When features are encapsulated in discrete modules, they may be delivered in small packages (updates) and can be integrated easily in existing systems that are already being used.

2.6.3 Limitations and Future Research

Five limitations of this study are noteworthy and provide avenues for future research. First, this study made some crucial assumptions that can be revisited in future research. We conceptualized individual features to provide equal value to the user and thus held the relative importance of features constant. This was also reflected in the design of our experiment. To increase the external validity of our findings, future studies should investigate features with varying relative importance and account for different valuations of features across users. Second, our treatment was realized through vignettes in an online questionnaire. As such, typical limitations of this methodology apply (Aguinis and Bradley 2014): the setting was fictitious and demanded subjects to put themselves in the position of the person in the scenario, while no instructor was available if questions arose. We thus controlled for motivation to process information, understanding and realism of the scenario and have strong reason to rule out bias in our results from these limitations. Nevertheless we encourage researchers to conduct longitudinal field studies or experiments with real software usage to further validate our findings. Third, we only observed a short usage time span and one update in our experiment. Future studies could explore user responses to repeated updates to understand the interplay between update size and update frequency as other possible boundary conditions and thus also deepen the understanding of sequential belief updating triggered by feature updates. Experiments conducted on longer time spans with users' evaluations measured at several points in time could also provide additional evidence for the robustness of the positive effect of feature updates on users' CI. Fourth, to control for the potential impact of different construal levels on the perception of updates, we split our experimental groups, resulting in reduced cell sizes for analysis. Although group sizes remained sufficiently large for our thorough statistical analysis and were in line with other, comparable experimental IS studies (e.g., Hong et al. 2004) we encourage future research to increase sample size. Moreover, future research should also explore additional crucial control variables related to users' short term interests, such as propensity to resist change (Oreg 2003; Polites and

Karahanna 2012), stress (Maier et al. 2015) or habit (Polites and Karahanna 2013). Fifth, the positive effect of feature updates on users' CI was shown to work for an online service. However, the domain of online services the technical complexity of the update process and potential downsides in the user experience are largely hidden from the user. While we believe that this unobtrusiveness of updates applies to a wide and also increasing range of modern software products and services (web services, platforms for mobile devices and modern desktop operating systems) there might be types of software for which our results are not generalizable (e.g., legacy software). Future research is encouraged to show the same effect for other types of software.

2.6.4 Conclusion

Feature updates have become a pervasively used instrument of software vendors to maintain, alter and extend their products over time. Despite their prevalence in private and business IT usage contexts, their effects on crucial user reactions in the IS post-adoption context have remained largely unexplored. This study is among the first to demonstrate that feature updates have the potential to increase users' CI above and beyond a level generated by monolithic software packages that deliver the entire feature set at once. It also reveals the robustness of this effect by ruling out update size and users' construal level as potential boundary conditions to this phenomenon. Nonetheless, we identified users' valuations of feature updates to slightly diminish with increasing feature endowment of the updated software. Lastly, this study identified a positive disconfirmation of previous expectations as the underlying mechanism by which feature updates influence users' CI. In summary, this study represents an important first step towards a better understanding of the nature of feature updates and how they affect user reactions. It may therefore serve as a springboard for future studies on feature updates in the IS post-adoption context.

Chapter 3: Effects of Gains or Losses through Updates on Experts or Novices

Title:	Gains and Losses in Functionality – An Experimental Investigation of the Effect of Software Updates on Users' Continuance Intentions

Authors: Marvin Fleischmann, Ludwig-Maximilians-Universität München, Germany
Tillmann Grupp, Technische Universität Darmstadt, Germany
Miglena Amirpur, Technische Universität Darmstadt, Germany
Thomas Hess, Ludwig-Maximilians-Universität München, Germany
Alexander Benlian, Technische Universität Darmstadt, Germany

Published in: International Conference on Information Systems (ICIS 2015), Fort Worth, USA.

Abstract

Although software updates are ubiquitous in professional and private IS usage, their impact on user behaviors has received little attention in post-adoption research. Based on expectation-confirmation-theory and the IS continuance model, we investigate the effects of gaining and loosing features through updates on expert and novice users' continuance intentions (CI). In a vignette based experiment, we find that updates which add features to software after its release increase novices' CI above and beyond a level generated by a monolithic software package that contains the entire feature set from the beginning. With diminished CI, experts show a contrary reaction to the same update. Losing features through an update, on the other hand, severely diminishes CI for experts and novices alike. Mediation analysis reveals positive disconfirmation of previous expectations as psychological mechanism behind novices' counter-intuitive and somewhat non-rational responses to gaining features through an update. Implications for research and practice are derived.

Keywords: Software updates, IT features, IS expertise, expectation-confirmation theory, IS continuance model, IS post-adoption

3.1 Introduction

The software industry and its business models have changed over the last years. This particularly applies to the market for consumer software. Traditionally, software vendors developed discrete programs and sold them as pre-packaged software at fixed prices. During the time of its use, this software remained largely unchanged and the user eventually replaced it with a new generation of this software, once it became available. This new generation of software was again sold at fixed prices. The popular office suites from Microsoft are a typical example for this practice. Recently, however, many software vendors have adopted a different practice. Often, a first, rudimentary version of an application is developed and sold. Then, over the course of time, this initial application is frequently changed through software updates. This practice is often (but does not have to be) accompanied by subscription based or ad-based revenue models that require a large and active user base in order to generate reoccurring revenues for vendors from renewed subscriptions or ad sales. This has not only become common practice in the app economy for smart phones and tablet computers but has also been adopted in the more mature field of software for desktop computers and web services. For example, Microsoft recently announced that it planned to shift to this practice with the version "10" of its operating system Windows, constantly enhancing and extending the software through updates over time while their customers already use it (Myerson 2015). As this example shows, vendors usually update their software in order to enhance it by correcting flaws or extending its functionality (i.e. features). In practice, however, the quality of software is sometimes also diminished by updates. A vendor can do this intentionally, for strategic reasons or when licensing deals run out, for instance. One example for this is the mapping functionality on the iPhone. In 2012, Apple updated its iOS operating system and, amongst other changes, removed the access to Google's maps service (Apple 2015). Mapping functionality was replaced with a functional inferior, in-house developed service (Sherr 2012). Another example is an update to Google's Android operating system from 2013. It removed privacy features which had previously allowed users to control the degree of personal data that could be accessed by third party applications (Constantin 2013). In other cases, software functionality is sometimes lost or diminished through updates unintentionally, when faulty updates corrupt features or render them useless.

However, despite the ubiquitous use of software updates in practice and many vendors' dependency on their customers' loyalty (i.e. continued use), there is little research on the impact of updates on users' beliefs, attitudes and specifically their continuance intentions regarding the updated software (Hong et al. 2011; Claussen et al. 2013). Most of the existing research neglects the user perspective and explores software updates from a purely technical perspective. This includes research on software engineering (Sommerville 2010), software product lines (Clements and Northrop 2002), software release planning (Svahnberg et al. 2010) and software evolution and maintenance (Mens and Demeyer 2008). Because updates are the means by which the characteristics of software are changed over time, during use, they may have the potential to alter users' beliefs, attitudes and behaviors regarding this software in the post-adoption stage (Karahanna et al. 1999; Bhattacherjee 2001). A better understanding of software updates from a user's perspective thus has the potential to increase the explanatory and predictive power of existing post-adoption theory. However, researchers studying post-adoption phenomena often tend to conceptualize information systems (IS) as a monolithic and coarse-grained black box, rather than as a collection of specific and finer-grained features that are dynamic and alterable over time (Jasperson et al. 2005). Only few studies have explored IS usage at a feature level (Benlian 2015). These studies have considered that different users employ different feature sets (DeSanctis and Poole 1994; Leonardi 2013), value them differently (Hiltz and Turoff 1981) and that the breadth and depth of a feature set that is utilized may change over time (Kay and Thomas 1995; Sun 2012). Nonetheless this stream of research does usually not consider changes in the available feature set over time, during usage, such as the addition or removal of features through software updates. Understanding the granularity of software and its changes through software updates would help to explain how users' beliefs, attitudes, and behaviors fluctuate over time as a result of the flexible nature of information systems. Moreover, there are several calls for research from IS scholars who criticize the negligence of the IT artifact's role in IS research (Benbasat and Zmud 2003; Hevner et al. 2004; Orlikowski and Iacono 2001). They suggest focusing on changes in beliefs, attitudes and behaviors emanating from the IT artifact itself rather than from other IT-unrelated environmental stimuli. Another issue that arises from the increasing ubiquity of software (and consumer software specifically), is the potential diversity in a software's user base (Harrison and Klein 2007). As more and more people gain access to information technology, it is increasingly also used by late adopters and non-experts (Rogers 1995). This can be seen as a contrast to the usage in organizations, where information systems are often operated by specialists or employees who receive specific training. To theoretically account for these developments, it becomes increasingly important for IS research to explore the heterogeneity in different users' beliefs, attitudes and behaviors regarding IT. Past IS research has already accounted for this (e.g. Kim and Son 2009; Venkatesh et al. 2012) but when investigating new phenomena or use cases, this issue has to be considered consistently. This study therefore raises the following research questions:

RQ1: Does gaining or losing features through software updates impact users' continuance intentions?

RQ2: How and why do novices and experts differ in their responses to software updates?

Drawing on expectation-confirmation theory (Oliver 1980) which is embedded in the IS continuance model (Bhattacherjee 2001), we conducted a vignette based online experiment with 178 participants to answer these questions. This study contributes to prior research in three important ways. First, we increase the understanding of users' post-adoption behaviors by identifying differential reactions of novice and expert users to software updates. While experts show rational reactions, our findings regarding novices' responses are counter-intuitive and may be characterized as non-rational. We identify update type and user expertise as crucial moderators for explaining the use of agile information systems. By investigating the mediating role of disconfirmation of expectations, our second main contribution is shedding light on the explanatory mechanism behind these different responses to updates. This has not been explicated in such a nuanced way in previous continuance literature. Our third and overarching contribution lies in the extension of the predominant view of information systems in post-adoption literature from a mostly monolithic and static one to a finer-grained and more flexible perspective by showing how an alterable information system might influence users' attitudes and behaviors during use.

Software vendors can learn from this study's results that holding back functionality in the first release of a software to deliver it only later on through updates has the potential to please customers and increase their loyalty. This measure, however, may not work for expert users and even be counterproductive. Vendors should thus be well aware of their customer base's software specific expertise. Moreover, vendors should avoid removing features from software after its first release by any means. It may severely raise their customers' likelihood to stop using the software and switch to a competitor's product.

3.2 Theoretical Foundations

3.2.1 Software Updates

Software updates can be defined as self-contained modules of software that are provided to the user for free in order to modify or extend software after it has been rolled out and is already in use (e.g. Dunn 2004). Software updates are no discrete and stand-alone programs but rather integrate into the base software once they are applied to it. In practice, software updates are applied to different types of software (e.g. operating systems, drivers, office suites) and on different platforms (e.g. desktop computers, mobile devices). With varying terminology (e.g. update, upgrade, patch, bug fix, or hotfix), the concept of software updates is repeatedly addressed throughout the software engineering literature (Sommerville 2010), such as software release planning, software maintenance and evolution and software product lines (Weyns et al. 2011). In this context, software release planning or strategic release planning refers to the "*idea of selecting the optimum set of features or requirements to deliver in a release within given constraints*" (Svahnberg et al. 2010, p. 1). Following this definition, an update is the delivery of features after the first release of a software and also falls within the strategic considerations regarding when to deliver what type of functionality to the user. Literature on software evolution and maintenance addresses the later stages in the software lifecycle, where updates are utilized to adjust software to changing requirements or repair emerging flaws in the software while it is already in use (Shirabad et al. 2001). In contrast to this rich stream of technical literature dealing with software updates from the developers' perspective, the users' beliefs and attitudes regarding updates have so far been explored only sparsely. This reflects in few IS studies dealing with updates (Amirpur et al. 2015). Hong et al. (2011), for example, explore user's acceptance of information systems that change through the addition of new functionality. Benlian (2015), on the other hand, explores different IT feature repertoires and their impact on users' task performance, but does not consider changes in functionality through updates. This also applies to other studies at the feature level which have considered that different users employ different feature sets (DeSanctis and Poole 1994; Leonardi 2013), value individual features differently (Hiltz and Turoff 1981) and that the breadth and depth of the utilized feature set may change over the time of usage (Kay and Thomas 1995; Sun 2012). And while other IS studies have found updates to influence usage behaviors, they have often pushed them to the sidelines, treating them as control variables for investigating other phenomena (e.g. Claussen et al. 2013).

In the present study which investigates the user perspective, we distinguish two basic types of software updates: feature updates and non-feature updates. Feature updates change the core functionality of software to which they are applied. Functionality can be added to or removed from the original version of the software and refers to distinct, discernible features which are deliberately employed by the user in accomplishing the task for which he uses the software. The popular Facebook app for smartphones and tablet computers provides an example for this type of update. In a 2013 update, it received a comprehensive instant messaging feature (Etherington 2013). In contrast to feature updates, technical non-feature updates do not change the core functionality of software but only correct flaws (e.g. bug fixes) or change software properties that are not directly related to its core functionality (e.g. improvements in stability, security or performance) (Popović et al. 2001). Examples for this type of update are the prominent ‘hot fixes’ that Microsoft regularly distributes via its Windows Update service. Because they occur during the use of software and are usually recognized by users through notifications, required actions during installation and new or changed functionality, specifically feature updates have the potential to affect users’ post-adoption beliefs, attitudes and behaviors regarding software, including continuance intentions.

3.2.2 Information Systems Continuance

Together with research on users’ pre-adoption activities and the adoption decision, post-adoption research constitutes the research field IS usage—one of the most mature fields in IS (Jasperson et al. 2005). In the context of post-adoption research (Karahanna et al. 1999; Bhattacherjee 2001; Benlian et al. 2011), the term information systems continuance refers to the “*sustained use of an IT by individual users over the long-term after their initial acceptance*” (Bhattacherjee and Barfar 2011, p. 2). To explore IS users’ intentions to continue or discontinue using an IS, Bhattacherjee (2001) adopts expectation-confirmation theory (ECT) (Locke 1976; Oliver 1980, 1993; Anderson and Sullivan 1993). ECT puts customers’ repurchase intentions at the center of investigation. In Bhattacherjee’s (2001) model, repurchase intention is replaced by a user’s intention to continue using an IS (CI)—the core dependent variable in his model. Following ECT, the IS continuance model suggests that users compare their pre-usage expectations of an IS with their perception of the performance of this IS during actual usage (Bhattacherjee 2001). If perceived performance exceeds their initial expectations, users experience positive disconfirmation (DISC) which has a positive impact on their satisfaction regarding the IS. If perceived performance falls short of the initial expectations, negative disconfirmation occurs and users’ satisfaction with the IS is reduced (Bhattacherjee and Barfar 2011). Satisfied users intend to continue using the IS, while dissatisfied users discontinue its subsequent use (Oliver 1980; Bhattacherjee 2001).

The concept of positive (negative) disconfirmation thus has two prerequisites—unexpectedness and a positive (negative) experience. Moreover, ECT posits expectations as a relative, subjective reference point or baseline (i.e. not an absolute, objective value) upon which the user makes his comparative judgment (Oliver 1980). This idea of a subjective, relative reference point is based on Helson’s (1964) adaptation level theory. It proposes that human beings perceive stimuli relative to or as a deviation from an ‘adapted level’ or baseline stimulus level. “*This adapted level is determined by the nature of the stimulus, the psychological characteristics of the individual experiencing that stimulus, and situational context*” (Bhattacherjee 2001, p. 354).

The IS continuance model has made valuable contributions to post-adoption research (Bhattacherjee 2001). However, in its original form, the IS continuance model has a static perspective on the IS continuance setting, failing to account for changing user believes and attitudes during use. In response to this limitation, Bhattacherjee and Premkumar (2004) introduce a more dynamic perspective by showing that beliefs and attitudes do not only change from pre-usage to actual usage but also during the ongoing usage of an IS. Kim and Malhotra (2005), Kim and Son (2009), Ortiz de Guinea and Markus (2009) and Ortiz de Guinea and Webster (2013), for instance, have provided evidence that the IS itself can shape users’ beliefs, attitudes and even their affect regarding the IT in later usage stages. Following Bhattacherjee and Premkumar (2004), it is reasonable to assume that a change in the IT artifact can also induce changes in users’ beliefs and attitudes towards it. To investigate the changing nature of the IT artifact and its impact on users’ beliefs, attitudes and behaviors during post-adoption use, we explore software updates through the lens of the disconfirmation mechanism in ECT.

3.2.3 Information Systems Expertise

Due to superior knowledge and abilities regarding a subject matter, experts make better decisions and perform tasks more successfully than novices (Alba and Hutchinson 1987). Individuals’ expertise has been explored in various research fields such as auditing (Shanteau and Steward 1992) and political science (Voss et al. 1983). Expertise is, however, not a general trait but specific to a certain subject or domain (Anderson 1982). Chess experts, for instance, “*do not appear to be better general thinkers for their genius in chess*” (Nelson et al. 2000, p. 477). Research on consumer decision making, for example, has repeatedly identified an individual’s product related expertise to significantly influence product choices (e.g. Lynch et al. 1991) and the use of products (e.g. Blackler et al. 2010). One major finding of this stream of research is that past experience and knowledge about a product or class of products allows experts to make comparisons with previous evaluations when making decisions (Ghoshal et al. 2014). This can lead to more objective evaluations and make experts less prone to bias regarding product choice and use. In experiments, experts have been found to rely on extra experimental information retrieved from their memory to supplement the experimentally supplied information. Novices, on the other hand, are more stimulus-bound in their decision making (Lynch et al. 1991). Due to a lack of experience and knowledge regarding a product, novices' decisions are more bound to the immediate situation or product at hand. As a result, their decisions are often more subjective and they may fall prone to bias in their product related decision making more easily (Mishra et al. 1993).

Expertise has also been shown to affect beliefs, attitudes and behaviors in IS usage. Research has repeatedly found users' expertise with an information system to moderate the relationship between independent and dependent IS usage variables, significantly affecting their strength or direction (Venkatesh and Davis 1996, 2000; Szajna 1996; Venkatesh et al. 2003; Kim and Malhotra 2005). In IS research, there have been various conceptualizations of expertise, emphasizing its different dimensions such as knowledge or abilities. These conceptualizations include IS expertise (Nelson et al. 2000), IS competency (Huff et al. 1992; Munro et al. 1997; Marcolin et al. 2000; Eschenbrenner and Nah 2014) and computer self-efficacy (Marakas et al. 1998; 2007; Rhee et al. 2009). According to Munro et al. (1997, p. 45), user competence "*is composed of an individual's breadth and depth of knowledge of end user technologies, and his or her ability to creatively apply these technologies*". The concept of computer self-efficacy is related to expertise with an information system and has been found to be a strong predictor of end-user performance (Marakas et al. 2007). In particular, past use and the resulting user experience are known to play important roles as moderators in IS post-adoption phenomena (Venkatesh and Davis 2000; Jasperson et al. 2005; Kim and Malhotra 2005; Kim and Son 2009). In the case of continued use, a user's earlier evaluations of an information system affect later evaluations because knowledge gained from experience with an IS is utilized in the decision making on its continuation (Hogarth and Einhorn 1992, Bolton 1998; Kim and Malhotra 2005). Eschenbrenner and Nah (2014, p. 1366) moreover point out that "*competency in the domain of IS is unique considering IS are continuously evolving, in development, and periodically upgraded (i.e., being updated, replaced, and modified)*". However, despite its important role for understanding how users' beliefs, attitudes and behaviors might change over time, as the IT artifact's nature and composition evolves through software updates, user expertise has only been explored sparsely in post-adoption research so far. Especially in the consumer domain of IS usage, this constitutes a research gap, considering the abovementioned insights from consumer decision making research which highlight significant differences between experts' and novices' product related choice and use behaviors. This study thus addresses the moderating role of expertise in users' post-adoption perceptions of software updates and their potential impact on continuance intentions.

3.3 Hypothesis Development

In this section, we develop our hypotheses about how and under which conditions updates can influence users' beliefs and attitudes in post-adoption software usage. Specifically, we explore decisions on continued use or discontinuance in settings where use is not mandated, such as consumer software. We therefore focus on feature updates which are recognized by the user during usage through explicit notification and ignore updates that are implemented 'behind the scenes'. Within this scope, we further distinguish between feature updates that add functionality and feature updates that remove functionality. We also distinguish expert and novice users. In our theorizing, we assume updates to deliver common features with functional equivalence across the hypothesized conditions. We make this assumption to properly reflect the practice (free updates do usually not deliver uniquely extraordinary features) and because previous research has found that uncommon, unique features may bias decisions and thus interfere with our attempt to conceptually isolate the psychological mechanism through which software updates might influence users' continuance intentions (e.g. Dhar and Sherman 1996).

3.3.1 IS Novices' Response to Gaining a Feature through a Software Update

We argue that receiving feature updates during the post-adoption use of an IS can induce positive disconfirmation and increase a novices' CI (Bhattacherjee 2001). According to ECT, the occurrence of positive disconfirmation requires an *unexpected* and positive experience (Oliver 1980). Overall, the experience must constitute an unanticipated, relative improvement compared to a baseline, i.e. it must exceed an individual's subjective reference point (Helson 1964). In the context of software updates this means that a surprising update must lead to a perceived improvement in the functionality of a software compared to its pre-update state. Following research on product expertise, it is reasonable to assume that due to a lack of knowledge and past experiences (Alba and Hutchinson 1987), novices do usually not anticipate feature updates, making them surprising, unexpected experiences with the software. Even if a vendor provides release plans about future updates, in practice, novices are unlikely to follow such update plans in detail for each software product they use. Moreover, when assessing the value of gaining a feature through an update, novices simply compare a software's functionality after the update to the functionality before the update, using the software at hand as primary reference point. According to research on product expertise and IS user competence, novices' evaluations are bound to this immediate stimulus because they lack other reference points from domain specific knowledge or previous use experience (Lynch et al. 1991; Eschenbrenner and Nah 2014). Novices cannot assess if the received feature might be overdue, if it is maybe already available in competing software products, if the vendor has developed the feature long before and delivered it only later, with an intentional delay and if it took the vendor much effort to develop. In sum, it is thus likely that novices will perceive a feature update as *unexpected* and *positive* experience during use, inducing positive disconfirmation in the sense of ECT (Oliver 1980).

According to this logic, a software vendor should be able to create positive disconfirmation and thus increase IS novices' CI by applying the strategy of holding back features (functionality) in the first release of a software package and delivering this functionality only later on, through free software updates. Under this deferred feature delivery strategy, a feature-complete software package might be designed and developed by the vendor, but certain features might not be included in the initially shipped software version. As outlined above, the novice user is assumed to be unaware of the existence of these remaining features. Once these remaining features are subsequently delivered through updates, they likely elicit positive disconfirmation. Consistent with the IS continuance model, this could then lead to an increase in CI. This deferred feature delivery strategy is thus to be distinguished from an all-at-once feature delivery strategy under which all developed features are delivered in the first release[5]. To summarize, because of the subjective nature of the disconfirmation mechanism in ECT, which operates through an evaluation of relative instead of absolute change, and a lack of software specific knowledge and past experiences, novice users of software that receives functionality via feature updates will likely have a higher intention to continue using this software than novice users who received all these features right with the first release. We accordingly derive our first hypothesis:

H1a: IS novices have a higher continuance intention regarding software that receives features through updates compared to software that includes the complete and equivalent set of features right with the first release.

[5] Nonetheless, we assume that both feature delivery strategies overall comprise the same type and number of features. We also assume that under both strategies, the user's evaluation of the software regarding CI takes place at the same point in time, which is after the incremental feature delivery strategy has been executed (i.e. when users are endowed with the same set of features as if they had received them right with the first release).

3.3.2 IS Experts’ Response to Gaining a Feature through a Software Update

We moreover argue that while ECT also applies to IS experts, it implies a different response to receiving feature updates. Following again research on product expertise and IS user competence, in contrast to novices, IS experts have more knowledge and past use experience about the updated software or this class of software (Alba and Hutchinson 1987; Eschenbrenner and Nah 2014). First, experts are thus more likely to anticipate updates or follow release plans if available. This reduces the likelihood that experts are surprised by an update and perceive it as unexpected event. Second, even if experts are surprised by a feature update, when evaluating this gain of functionality, they will use a different baseline against which they compare the post-update state of the software. Due to their superior knowledge and past usage experience with the software or type of software, experts do not only compare the new functionality to the pre-update state of the immediate software at hand, but also consider information about other, competing or similar software products or general technological developments to assess the value of the added feature. Overall, compared to novices, experts’ evaluations of a feature update will be more objective, making them less subject to a biased perception of the new functionality (Lynch et al. 1991). Therefore, we argue that experts do not fall prey to a vendors’ deferred feature delivery strategy of holding back functionality in order to deliver it only later on and increase users’ CI as easy as novices would. In practice, experts may even show a negative response to such a strategy of deferring features, when they identify the delivered functionality as common feature that is not a true innovation by the vendor but was developed long before and only held back intentionally. We therefore derive the following hypothesis:

H1b: Experts have a lower continuance intention regarding software that receives features through updates compared to software that includes the complete and equivalent set of features right with the first release.

3.3.3 Novices’ and Experts’ Response to Losing a Feature through a Software Update

Hypotheses 1a and 1b proposed different user reactions to *gaining* a held back feature through an update for experts and novices due to different levels of experience and knowledge regarding the functionality of a software or class of software. We argue that this different reaction of experts and novices will, however, not be present when *losing* a feature through an update. When losing a feature through an update during the use of a software, the formation of CI will also be influenced by the ECT mechanism (Oliver 1980; Bhattacherjee 2001). However, in this case, the functional baseline against which the updated software with reduced (lost) functionality will be compared includes the removed (lost) feature for experts and novices (Kim and Malhotra 2005). In their pre-update use of the software, they both have experienced the feature and are thus assumed to be aware of its presence and helpfulness in task completion. When a feature is removed from the software through an update, this leads to a perceived deterioration of the software for experts *and* novices. The updated software then lacks a specific feature which may previously have served as a tool for accomplishing a certain task. Assuming that this task can still be accomplished using the deteriorated software, its completion should become more difficult or time consuming. The user might have to substitute the lost functionality with another feature in the software or compensate for the lost feature by conducting previously automated steps of his task manually. As a consequence of this loss of functionality, the updated software should be perceived as comparatively less valuable by experts *and* novices. This should subsequently reduce their satisfaction with the software and intention to continue using it. We thus propose the following joint hypothesis for experts and novices:

H2: Both, experts and novices, have a lower continuance intention regarding software that loses features through updates after the first release compared to software that keeps these features.

3.3.4 The Mediating Effect of Disconfirmation

As pointed out before, we argue that the difference in IS novices' and experts' responses regarding CI from gaining a feature through an update originates in their different evaluations of the software through the ECT mechanism (i.e. different subjective baselines or reference points). According to the continuance model, compared to losing a feature, the novice's positive response should thus be mediated by a positive disconfirmation of their subjective, previous expectations regarding the software, i.e. DISC (Bhattacherjee 2001). Moreover, the ECT mechanism also suggests that such a positive disconfirmation of previous expectations (DISC) would not directly affect CI but in turn be mediated by SAT, which ultimately leads to the proposed change in CI. Due to their different response to gaining a feature through an update, experts should not show this mediating effect. We thus propose the following mediation hypothesis:

H3: The positive response of novices to gaining a feature through an update compared to losing a feature is mediated by DISC and SAT.

3.4 Method

3.4.1 Experimental Design

With the goal to examine the effects of software updates on users' CI as suggested by our hypotheses, we opted for a vignette based online experiment. It allowed us to investigate and isolate the causal mechanisms that operate between software updates and attitudinal user reactions. We presented participants with carefully constructed textual scenarios (vignettes) that precisely described a person (user), task, software, software usage and a conditional update (see Figure 3-1). We opted for the experimental vignette methodology (EVM) because it provides consistent and identical treatments for all participants and reduces unwanted effects such social desirability bias (Aguinis and Bradley 2014). Even though this method comes with downsides such as a fictitious setting, it also allows for an accurate identification of the hypothesized effects. By being able to design a quasi-real scenario, the vignettes allowed us to ensure a high external validity, compared to a laboratory experiment. Nonetheless, researchers have shown that individuals respond quite similarly to hypothetical situations in vignettes compared to traditional laboratory experiments, making this method suitable for our needs (Rahman 1996; Shaw et al. 2003; De Cremer et al. 2007; Dennis et al. 2012).

We thus conducted a 1 x 3 between-subjects experiment (see Figure 3-1) with manipulations of update (no update vs. retained feature gained through update vs. feature lost through update). 178 participants from a large public university in Germany evaluated the impact of software updates on the user's continuance intentions. The participants read textual vignettes which described usage scenarios of a fictitious word-processing program ('xText') used by a fictitious student ('Tom') who had to write a term paper in group work together with classmates. Participants were randomly assigned to one of the three groups. Depending on the experimental condition, halfway during the described overall eight week use of the program, Tom received a feature through an update (group B) or lost a feature through an update (group C). In the control group, he used the software without any update (group A). Using a student sample was appropriate for this study, because students are likely to be familiar with word-processing programs, collaboration in group work tasks and software updates. They should also show similar attitudes and beliefs toward the treatments offered in our experiment compared to non-student samples (Jeong and Kwon 2012).

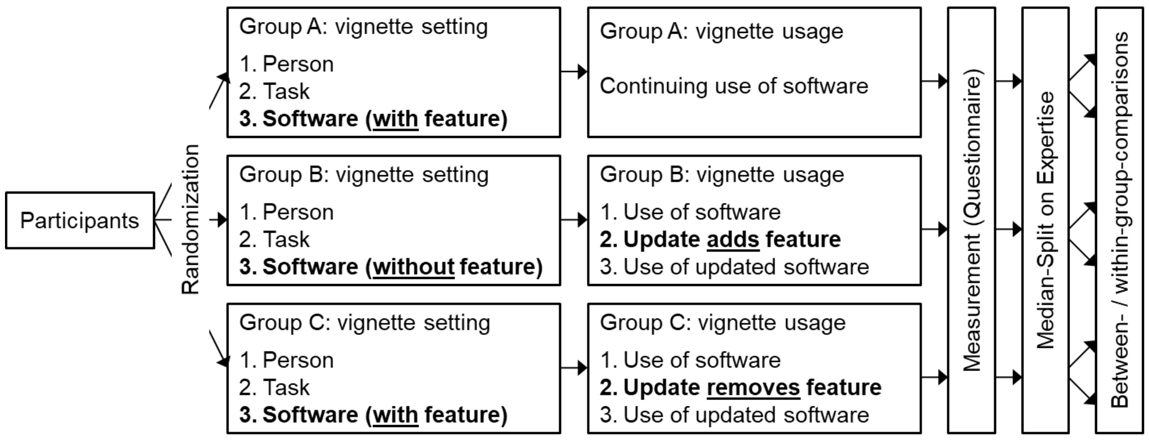

Participants
Randomization
Group A: vignette setting
1. Person
2. Task
3. Software (with feature)
Group B: vignette setting
1. Person
2. Task
3. Software (without feature)
Group C: vignette setting
1. Person
2. Task
3. Software (with feature)
Group A: vignette usage
Continuing use of software
Group B: vignette usage
1. Use of software
2. Update adds feature
3. Use of updated software
Group C: vignette usage
1. Use of software
2. Update removes feature
3. Use of updated software
Measurement (Questionnaire)
Median-Split on Expertise
Between- / within-group-comparisons

Figure 3-1: Experimental Setup, Groups, and Treatments

3.4.2 Manipulation of Independent Variables

In our experiment, we used a word-processing program for two reasons: We sought to ensure a basic familiarity with the program for all participants. Because nowadays almost any young person, especially students, needs to work with word-processing programs, we considered this criterion to be met. Second, for the update, we were looking for a software feature that was easily understandable through a textual description, preferably value-free and directly helpful in achieving the task but not indispensable so that the task could be completed also without the feature. Moreover, our hypothesis also required the update to deliver a feature that could technically be held back in the vendors' deferral strategy and was not an extraordinarily unique feature (Dhar and Sherman 1996). To identify this common feature for our treatment, we conducted a pre-study. In this pre-study, 52 subjects rated the relative importance of the 54 text editing features that are provided by the open source online text editor TinyMCE on seven-point-Likert scales (TinyMCE 2015). The subjects for this pre-study were recruited using WorkHub, a crowdsourcing platform similar to Amazon Mechanical Turk and participated online for a small payment (Paolacci et al. 2010). As a result, a feature for spell checking and grammar correction was selected. It met the requirements for our study best.

In the main study, the textual scenarios were presented to the participants in an online questionnaire that comprised several consecutive pages. On a first page, we described Tom, his task and the software with which he had to accomplish this task (see vignette setting, Figure 3-1). Tom had to write a term paper and work *"together with three classmates in a team. Their professor demands to write their term paper in English [which is not their native language]."* They had eight weeks to complete the paper. *"Because two team members are abroad during the entire working time, personal meetings are not possible"*. Therefore, they *“use the text editing program xText.”* The program only had *“a basic [yet sufficient] set of functionality but allows for collaboration in one text document by several users over the internet which is necessary…”.* Based on the information provided in the vignettes, the use of ‘xText’ was thus mandatory for this specific project. Depending on the experimental condition, the described software included—among other features which were listed in the vignette —the feature for English spell checking and grammar correction (groups A and C) or not (group B). The use of this spell checking and grammar correction feature, however, was not mandated. Like any other feature in ‘xText’ its utilization was optional but—if available—obviously helpful for achieving the task. On a second page, we described Tom’s experience with the software during the entire time of the task completion, i.e. from starting to work on the term paper to handing in the final paper (see vignette use, Figure 3-1). Depending on the experimental condition, the description included an update of the software that added (group B) or removed (group C) the feature for English spell checking and grammar correction or no update at all (group A). In group B, after four weeks of working on the paper, when opening the program, *“Tom is notified about an update that is automatically executed […] and adds [for free] a feature for spell checking and grammar correction to the program.”* The new feature is described to save time for Tom because *“now he does not need to check the text word for word.”* In group C, after four weeks of working on the paper, when opening the program, *“Tom is notified about an update that is automatically executed […] and removes [for free] the feature for spell checking and grammar correction from the program.”* As an explanation, it was stated that the vendor of ‘xText’ had only licensed this feature and it had to be removed because *“the licensing deal was not renewed“*. *“After the feature is removed, Tom has to check the text word for word for errors which costs time.”* Except for the description of the update, the usage of the program was described identically in group A. Except for the manipulated parts, we kept the scenario identical across the three groups. Each vignette ended with the group handing in the term paper after eight weeks. After this second page, participants started to answer the questionnaire. This included their evaluation of the protagonist’s intention to continue using ‘xText’ for future term papers when its use would no longer be mandated. Participants could only proceed to the next page when all questions were answered and returning to previous pages, including the vignettes, was not possible.

A pilot test with six subjects was conducted to ensure that the treatments were manipulated according to the experimental design (Perdue and Summers 1986). Specifically, subjects were asked about the comprehensiveness of the instructions, the vignettes and the questions in the following questionnaire. Suggestions were obtained from the participants and the vignettes and the questionnaire were revised accordingly for the main experiment.

3.4.3 Measures

Dependent Variables

We used validated scales with minor wording changes for all constructs. Measures for CI were adapted from Bhattacherjee (2001). Participants were asked to evaluate what they thought Tom would do, if after the completion of this term paper, he would have to write another paper in the future: ci1. *Tom intends to continue using xText rather than discontinue its use;* ci2. *Tom's intentions are to continue using xText than use any alternative means;* ci3. *If Tom could, he would like to discontinue his use of xText* (reverse coded). DISC was also adopted from Bhattacherjee (2001): disc1. *Tom's experience with using the word-processing program xText was better than what he expected;* disc2. *The functionality provided by the word-processing program xText was better than what Tom expected;* disc3. *Overall, most of Tom's expectations from using the word-processing program xText were confirmed.* Measures for SAT were based on Kim and Son (2009): sat1. *Tom is content with the features provided by the word-processing program xText;* sat2. *Tom is satisfied with the features provided by the word-processing program xText;* sat3. *What Tom gets from using the features of the word-processing program xText meets what he expects for this type of programs.* Because constructs were measured with multiple items, summated scales based on the average scores of the multi-items were used in group comparisons (Zhu et al. 2012). Unless stated otherwise, the questionnaire items were measured on seven-point-Likert-scales anchored at (1)=strongly disagree and (7)=strongly agree.

Control Variables

In our study, we examined participant's *motivation to process information* with one item (Suri and Monroe 2003), because this variable may also influence the response behavior of the participants and, thus, the validity of the results. Moreover, after conducting the experimental task, participants were asked to what extent they had understood *the items' formulation*, to what extent they were *able to put themselves in the hypothetical setting described in the vignette*, if *the setting in the described story was realistic* and if they knew *what the goals of this survey were*. We included these control variables as well as the subjects' demographics as covariates to isolate the effects of the manipulated variables. The participants' expertise regarding word-processing programs was captured on an established four item, seven-point semantic differential scale with the items *know very little about/know very much about,* *inexperienced/experienced, uniformed/informed, novice buyer/expert buyer* (Mishra et al. 1993).

3.4.4 Participants, Incentives and Procedures

Participants for the final experiment were members of a large, public university in Germany. In December 2014, 6039 members of the university received an email, inviting them to participate in "an online survey about software usage". The email contained a link to the online experiment and stated that ten Amazon vouchers worth 10 € and one Amazon voucher worth 50 € were drawn in a lottery among all participants, once the study had been completed. Overall, 254 subjects started the experiment. 76 participants did not complete the experiment. They were excluded from our analysis. We thus used a sample of 178 subjects in the following analysis. Of these 178 subjects, 60 were males. The participants' average age was 25.12 (σ=6.80). 148 participants were students, 27 were employees or self-employed and three were seeking work. The educational backgrounds of the participants were diverse, including management, medical science, law, education, biology, physics, philosophy etc. Across the four seven-point semantic differential items, the mean score of the self-stated expertise with word-processing software was 3.96 (σ=0.45) on average. Based on this mean value across the four items for each participant, a median split was performed to classify subjects as experts and novices for the later hypothesis testing regarding expertise (Lynch et al. 1991). This resulted in the following group sizes: group A, no update, n=57 (30 experts, 27 novices); group B, feature gained, n=63 (42 experts, 21 novices); group C, feature lost, n=58 (31 experts, 27 novices). Across all groups, the participants indicated that they were able to put themselves in the hypothetical setting described in the vignette ($\bar{x}$=5.40, σ=1.57) and that they thought the described setting was realistic ($\bar{x}$=5.23, σ=1.49). Participants also indicated a high motivation to process information ($\bar{x}$=6.42, σ=1.03) and understood the questionnaire items well ($\bar{x}$=6.11, σ=1.36). On average, they stated that they did not know what the goals of this survey was ($\bar{x}$=3.37, σ=1.75). This indicates that we were successful in designing the experiment according to its purpose.

3.5 Data Analysis and Results

3.5.1 Control Variables

Based on the results of Fisher's exact tests, it can be concluded that there was no significant difference across the three experimental conditions in terms of gender (p>0.1) and profession (p>0.1). Furthermore, based on ANOVA tests, no significant differences were found across the six experimental conditions regarding age (F=0.14, p>0.1), and Mishra et al.'s (1993) self-evaluation of expertise on the seven-point semantic differentials (F=0.88, p>0.1). Furthermore, there was no significant difference across the three experimental conditions regarding the task-relevant control variables motivation to process information (F=0.15, p>0.1), the extent to which subjects were able to put themselves in the hypothetical situation described in the experimental task (F=0.47, p>0.1), the evaluation of the vignette's realism (F=1.83, p>0.1), the comprehensiveness of the items' phrasing (F=0.74, p>0.1), and knowing what the goals of the survey were (F=1.11, p>0.1). It is therefore reasonable to conclude that participants' demographics and task-relevant controls were homogeneous across the three conditions and did not confound the effects of our experimental manipulations.

3.5.2 Measurement Validation

Because we adopted established constructs for our measurement, confirmatory factor analysis (CFA) was conducted to test the instrument's convergent and discriminant validity (Levine 2005). Table 3-1 reports the CFA results using SmartPLS version 3.0 (Chin et al. 2003; Ringle et al. 2014) for the core constructs.

Table 3-1: Results of Confirmatory Factor Analysis for Core Variables

Latent construct	Number of Indicators	Range of Standardized Factor Loadings*	Cronbach's Alpha	Composite Reliability (ρ_c)	Average Variance Extracted (AVE)
Continuance Intention (CI)	3	0.792-0.906	0.833	0.901	0.753
Satisfaction (SAT)	3	0.805-0.951	0.885	0.930	0.816
Disconfirmation (DISC)	3	0.782-0.920	0.844	0.907	0.766
Note: *All factor loadings are significant at least at the p<0.01 level					

All items loaded on the target factors and scored above the threshold of 0.7, indicating proper construct validity (Cook and Campbell 1979; Bartholomew et al. 2008). AVE values for each construct ranged from 0.753 to 0.818, exceeding the variance due to measurement error for that construct (AVEs exceeded 0.5). The constructs were assessed for reliability using Cronbach's alpha (Cronbach 1951). A value of at least 0.7 is suggested to indicate adequate reliability (Nunnally et al. 1994). The alphas for all constructs were well above 0.8. The composite reliability of all constructs exceeded 0.7, which is considered the minimum threshold (Hair et al. 2011). Thus, all of the constructs met the norms for convergent validity. For satisfactory discriminant validity, the square root of AVE from the construct should be greater than the variance shared between the construct and other constructs in the model

(Fornell and Larcker 1981). As seen from the factor correlation matrix in Table 3-2, all square roots of AVE exceeded inter-construct correlations, providing strong evidence for discriminant validity. Hence, the constructs in our study are both theoretically and empirically distinguishable.

Table 3-2: Means, Standard Deviations, and Correlation Matrix for Core Variables

Latent construct	M	SD	1	2	3
(1) Continuance Intention (CI)	4.118	1.626	**0.868**		
(2) Satisfaction (SAT)	4.642	1.537	0.512***	**0.875**	
(3) Disconfirmation (DISC)	4.541	1.525	0.564***	0.756***	**0.903**
Note: Bolded diagonal elements are the square root of AVE. These values should exceed inter-construct correlations (off-diagonal elements) for adequate discriminant validity; ***p<0.01, **p<0.05, *p<0.1.					

3.5.3 Hypotheses Testing

In order to test our hypotheses, we conducted one-way analyses of variance (ANOVA) with contrast analyses using StataCorp Stata 12. Continuance intention (CI) was analyzed as function of update and expertise. There was a significant main effect for update (F=25.94, p<0.01) but not for expertise (F=2.01, p>0.1). However, the interaction between expertise and update had a significant effect on CI (F=2.73, p<0.05). Contrast analysis revealed that experts and novices showed different reactions to gaining a feature. Novices showed a significant *higher* CI when gaining the feature ($\bar{x}$'s = 5.24 vs. 4.44, p<0.05). This supports our hypothesis 1a. Experts, on the other hand exhibited a significant *lower* CI when gaining the same feature through an update ($\bar{x}$'s = 4.31 vs. 4.77, p<0.1), supporting our hypothesis 1b. When losing a feature during use, both novices and experts had a significant lower CI ($\bar{x}$'s = 3.21 vs. 4.44, p<0.01 and $\bar{x}$'s = 2.88 vs. 4.77, p<0.01). This supports our hypothesis 2. Table 3-3 provides an overview over the effects of different update types and expertise on CI. Figure 3-2 visualizes the different user reactions to software updates, indicating mean values of CI for experts and novices across groups.

Table 3-3: Means, Mean Differences and Significance Levels for Continuance Intention

Expertise with Software	Mean Values for Groups			Mean Differences and Significance Levels	
Experts / Novices	No Update (A) n=57	Feature Gained through Update (B) n=63	Feature Lost through Update (C) n=58	B-A	C-A
Experts	4.77	4.31	2.88	**-0.46***	**-1.89*****
Novices	4.44	5.24	3.21	**0.80****	**-1.23*****
Note: *** *p*<0.01, ** *p*<0.05, * *p*<0.1 (one-sided); ANOVA-tests with contrast analyses					

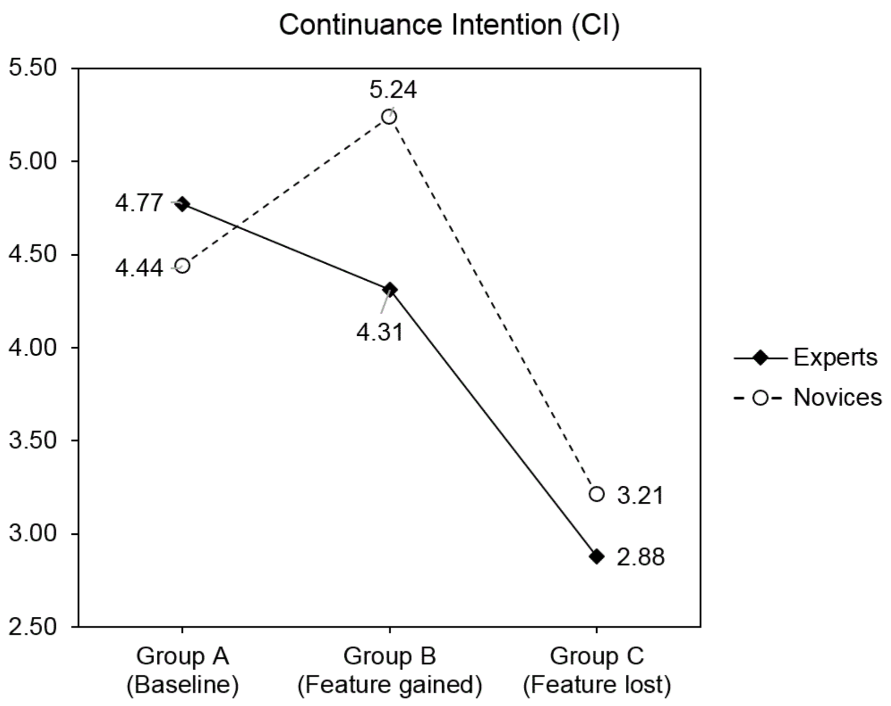
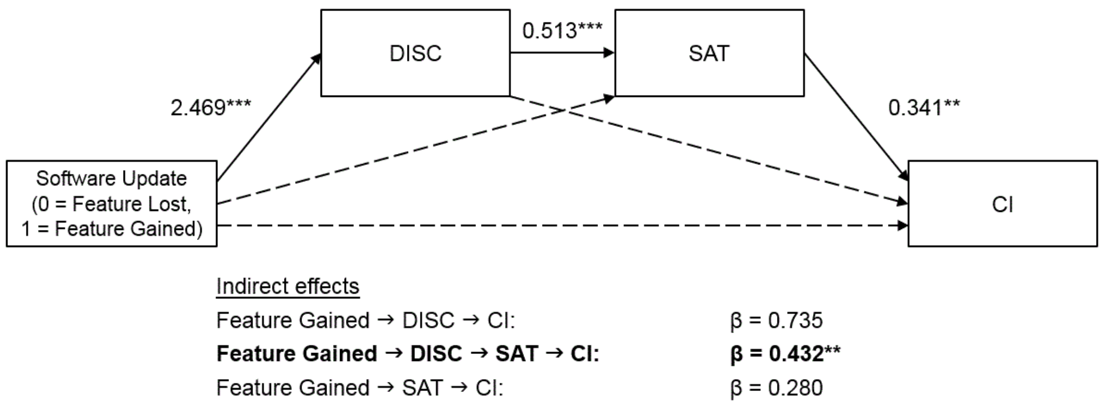


Figure 3-2: Expert and Novice Responses to Gaining and Loosing Features from an Update

In order to investigate hypothesis 3 and explore the psychological mechanism behind the novices' different responses to gaining and losing a feature, a mediation analysis of the continuance model's core variables (Bhattacherjee 2001) was performed for novices in groups B (gaining a feature) and C (losing a feature). To analyze the mediating effects of DISC and SAT, we used PROCESS, a regression-based approach developed by Hayes (2013). PROCESS uses bootstrapping procedures for estimating direct and indirect effects. Figure 3-3 provides an overview of the analyzed conceptual model with direct and indirect paths. As recommended by Hayes (2013), path coefficients are unstandardized because the independent variable (software update) is dichotomous.


DISC
0.513***
SAT
2.469***
0.341**
Software Update (0 = Feature Lost, 1 = Feature Gained)
CI
Indirect effects
Feature Gained → DISC → CI: β = 0.735
Feature Gained → DISC → SAT → CI: β = 0.432**
Feature Gained → SAT → CI: β = 0.280

Note: Dashed lines indicate non-significant paths; *** p<0.01, ** p<0.05, * p<0.1

Figure 3-3: Mediation Mechanism Behind Novices' Positive Response to Gaining a Feature through an Update

The results from bootstrapping analysis in Table 3-4 revealed that only the (unstandardized) indirect effect path (2) from gaining a feature through an update via DISC and SAT to CI was significant. Moreover, the direct effect of gaining a feature though an update on users' CI became insignificant after including DISC and SAT, suggesting full mediation (Hayes 2013). This mediation analysis was also performed separately for experts. As also expected from hypothesis 3, due to their different response to gaining a feature through an update, this mediation was not found for experts. Hence, hypothesis 3 is supported.

Table 3-4: Results from Serial Multiple Mediation Analysis of Novices in Groups B and C (Bootstrapping Results for Indirect Paths)

Indirect effect paths	Effect z	Boot SE	LLCI	ULCI
(1) Feature Gained → DISC → CI	0.735	0.596	-0.210	2.211
(2) Feature Gained → DISC → SAT → CI	**0.432**	**0.257**	**0.093**	**1.207**
(3) Feature Gained → SAT → CI	0.280	0.319	-0.133	1.086
Note: Inferential tests for indirect effect paths based on 1.000 bootstrap samples generating 95% bias-corrected bootstrap confidence intervals (LLCI=Lower Limit/ULCI=Upper Limit of Confidence Interval)				

3.6 Discussion

This study sought to achieve three main objectives: (1) to examine the effects of different types of software updates on users' intentions to continue using an information system compared to monolithic software (i.e. whether there are discernible effects from gaining or losing features through updates), (2) to investigate the moderating role of IS expertise (i.e. if novices perceive updates differently than experts) and (3) to unravel the explanatory mechanism behind such different responses to updates (i.e. how and why such an effect from updates occurs). To achieve these objectives, we drew on the IS continuance model, the underlying expectation-confirmation theory and theory on IS user expertise and investigated our hypotheses based on a vignette based online experiment with 178 participants.

Drawing on the advantages of the experimental method, which allows to isolate the effects of manipulated stimuli on user responses from other confounding variables and thus to unveil causal relationships, we found that expert and novice users showed different reactions to updates. In the case of experts, any type of update led to a decrease in CI (groups B and C). Not only losing a feature through an update (group C) but even gaining a feature (group B) significantly lowered their intention to continue using the software. The response to losing a feature is comprehensible. Halfway through task completion, the user is deprived of a helpful functionality in the utilized program. This reduction in functionality makes his present task more difficult and the program less valuable for any future use. Consequently, the user's intention to continue using the program beyond the current project (CI) is diminished. The experts' response to gaining a feature, on the other hand, may seem surprising at first, because it seemingly increases the value of the program to the user. However, the gained feature was artificially held back and intentionally delivered only later on, through an update. As suggested in hypothesis 1b, theory on product expertise (Alba and Hutchinson 1987) and information systems expertise (Eschenbrenner and Nah 2014) implies that expert users are likely to identify the delivered functionality as common feature that is not a true innovation by the vendor but was developed before and only held back on purpose. In line with this reasoning, experts in group B did not fall prey to the deferred feature delivery strategy, overall showing a rational behavior.

Novices on the other hand showed different reactions. While they also had a lower CI when losing a feature through an update (group C), their CI was significantly higher in the positive update condition (group B) than in the non-update condition (group A). This increase of novices' CI in group B compared to group A can be interpreted as being a somewhat counter-intuitive finding because the user described in the vignette with a feature gained through an update (group B) was objectively disadvantaged compared to the user who had all functionalities right with the first release (group A): during the time span of usage as described in the vignette, the user in group B had in sum fewer features per time to accomplish his text-formatting task compared to group A. Despite this objective disadvantage, novice participants in group B showed significantly higher scores in CI. This suggests the presence of a somewhat non-rational effect (Fleischmann et al. 2014). When comparing the absolute values of the novices’ responses to gaining and losing a feature, their evaluations seem even less rational. Considering the non-update condition (group A) as reference point, the perceived loss from removing a feature from the software through an update (mean difference between responses by novices in group A and C) was higher in magnitude than the perceived gain from receiving the exact same feature through an update (mean difference between responses by novices in group A and B). This suggests the presence of loss aversion in novices (Kahneman and Tversky 1979). As such, both findings of novices’ responses to updates challenge the idea of a ‘rational user’ in the IS continuance literature (Ortiz de Guinea and Markus 2009; Bhattacherjee and Barfar 2011; Ortiz de Guinea and Webster 2013).

Finally, we could demonstrate that the positive response to gaining a feature through an update regarding CI (novices in group B) is fully mediated by the ECT core variables DISC and SAT. Due to a lack of experience and outside knowledge, novices seem to be unable to objectively evaluate the gain of a retained feature from an update. In terms of ECT, novices only use their immediate, subjective perception of the software’s functionality before the update as reference point. Exceeding this subjective reference point induces positive disconfirmation of previous expectations (DISC) which initiates a psychological process by which increases in SAT eventually lead to a higher CI.

3.6.1 Implications for Research

The paper makes three main contributions to the literature. First, we identify different user reactions to software updates. These responses crucially depend on the type of update and the users’ expertise regarding the updated software. Losing a feature through an update decreases CI for experts and novices. Gaining a retained feature through an update, on the other hand, induces a positive reaction in novices. This has even a stronger and more positive impact on novices’ continuance intentions compared to situations in which the entire feature set is provided at once and with the first release. Expert users, however, do not show this positive response. The gain of a feature can therefore be seen as necessary and the lack of expertise with the software as sufficient condition for this positive response to software updates that deliver features which have been held back at the initial release of software. Conceptually, update type and expertise regarding the updated software thus seem to moderate the effect of updates on CI. This interaction emphasizes the importance of a *joint* consideration of IT artifacts’ and the users’ characteristics when investigating usage behaviors. Our second main contribution is shedding light on the explanatory mechanism behind the identified positive effect of updates on CI for IS novices, which could not be ascertained for IS experts.

Specifically, we find that this positive effect for IS novices is fully mediated by a positive disconfirmation of previous expectations regarding the software due to the update (DISC) and SAT. This finding once again highlights the pivotal role of ECT within the IS continuance model. Our third and overarching contribution lies in showing how a malleable information system might influence users' attitudes and behaviors during post-adoption use. This answers the calls of several IS researchers by extending the still predominant view of post-adoption literature on the IT artifact as a static and monolithic block to a more flexible and finer-grained perspective which considers information systems as a modular composition of features that may change over time (Jasperson et al. 2005; Benbasat and Barki 2007 etc.). We complement existing IS post-adoption literature through the notion that users' beliefs and attitudes might fluctuate over time, in conjunction with changes in the used information system.

3.6.2 Implications for Practice

Our results also have important implications for practice. First, despite the extensive use of software updates by vendors to maintain, alter and extend their products after they have already been rolled out, it is surprising to find that insights on how these updates are perceived and evaluated by users are still scarce. This leaves practitioners without guidance. From the results of our experimental study we can conclude that vendors should avoid removing features from software after its release. This also includes well-intentioned updates which unintentionally damage the software and render certain features useless. When vendors remove functionality from their software, they significantly increase their customers' likelihood to discontinue using their product (and perhaps switch to a competitor's product). In the already highly competitive market for consumer software, vendors may want to avoid this by any means.

Adding helpful features through free updates, on the other hand, might seem as a straightforward measure for vendors to please customers and increase their loyalty (i.e. CI). More specifically, our findings suggest, that it could even be advisable for vendors to hold back software functionality and distribute it over time via updates, instead of delivering all features right with the first release of a software. Feature updates have the potential to increase users' CI above and beyond a level generated by software packages that are delivered with the entire feature set at once. However, the findings of this study reveal that this effect seems to work only for novice users. Software vendors can learn from this study's results that they should be well aware of their customer base and its expertise regarding the software.

Utilizing customer data or conducting market research can be helpful in this regard. It should also be noted that vendors should not overdraw the holding back of functionality when applying the deferred feature delivery strategy. Starting out with a too small feature set might render the first release of a software almost useless and lead to discontinuation before the program can be updated or even prohibit the adoption in the first place. Especially vendors who face direct competition from other, similar software products should carefully evaluate what type and number of features they can afford to hold back under this strategy and which ones ought to be provided immediately in order to win or retain customers. In practice, each vendor will have to determine this sufficient amount of features for his own, specific case.

Finally, when maintaining their software after its first release, software vendors should not only focus on their own product but also keep track of connected or compatible programs from other vendors. In today's interconnected but quickly changing software industry, many programs rely on interoperability through interfaces, plug-ins and compatibility. When other connected or compatible software is changed through updates, the own interfaces and plug-ins may stop working and compatibility may vanish, rendering some features useless. In order to avoid losing customers' from such a loss in functionality (even if only temporary), vendors should closely monitor the integrity and functionality of these interfaces, plug-ins and compatibility and quickly respond to restore or repair them if necessary.

3.6.3 Limitations and Future Research

Four limitations of this study are noteworthy and provide avenues for future research. First, our experiment utilized textual vignettes to describe software usage scenarios. While this is a proven methodology, it also has some limitations (Aguinis and Bradley 2014). Our constructed setting was fictitious and it required subjects to put themselves in the position of the scenario's protagonist. Moreover, because the study was conducted online, there was no instructor who could have answered any questions regarding the described vignette scenario. We thus controlled for motivation to process information, perceived realism of the scenario and how well participants understood the questions and thought that they were able to put themselves in the hypothetical setting. Based on the results regarding these measures, we are confident that our vignettes worked as intended and our study's implications are applicable to real usage settings. Nonetheless, future studies could investigate actual usage experiences with real software to validate our findings. Second, we identified update type and user expertise as crucial moderators for the effect of updates on users' CI. Future studies are encouraged to further differentiate update types (e.g. several features in one update) and explore additional user characteristics (e.g. different cultural backgrounds). Furthermore, complementary qualitative studies (e.g. thought-listing) could further substantiate our theoretical reasoning behind the identified moderators e.g., why experts disliked the deferred delivery of features through an update (Ma and Roese 2014). Third, the demonstrated effects of updates on users' CI were shown to work for productivity (word-processing) software. Future research could show whether the same effects also occur for hedonic (e.g. entertainment) software. Finally, we conducted a controlled experiment with the purpose of presenting results with a high internal validity. This required some reasonable but strict assumptions, such as exploring a common feature, an identical and linear course of events for all users and ex-post measurement of variables. Future studies are encouraged to complement the findings of this study by investigating different types of features (e.g. extraordinary features) and conducting longitudinal field experiments, to advance the external validity of our findings. Also settings with repeated updates over longer time spans with participants evaluations measured at several points in time could provide additional evidence for the robustness of our findings. Specifically, a field experiment using an online service similar to Google Docs or Microsoft Office Online would be well suited to collect panel data from real usage over a longer period of time.

3.6.4 Conclusion

Software updates have become a pervasively used instrument for vendors to maintain, alter and extend their products over time. Despite this prevalence, their effects on crucial post-adoption user reactions have remained largely unexplored. This study's diverse findings highlight the importance of a profound understanding of updates for both researchers and practitioners. Updates that add features to a software after its first release, while it is already in use, have the potential to increase users' CI above and beyond a level generated by a monolithic software package that is released with the entire feature set at once. However, this only applies for novice users but not for experts. Losing a feature through an update, on the other hand, severely diminishes CI and raises a user's likelihood of switching to a competitor's product. Furthermore, this study explains the psychological mechanism behind the different user responses to updates. It works through disconfirmation of previous expectations regarding the updated software.

Chapter 4: Updates and the Role of Update Frequency and Update Type

Title: The Role of Software Updates in Information Systems Continuance – An Experimental Study from a User Perspective

Authors: Marvin Fleischmann, Ludwig-Maximilians-Universität München, Germany
Miglena Amirpur, Technische Universität Darmstadt, Germany
Tillmann Grupp, Technische Universität Darmstadt, Germany
Alexander Benlian, Technische Universität Darmstadt, Germany
Thomas Hess, Ludwig-Maximilians-Universität München, Germany

Published in: Decision Support Systems (83), pp. 83-96.

Abstract

Although software updates are a ubiquitous phenomenon in professional and private IT usage, they have to date received little attention in the IS post-adoption literature. Drawing on expectation-confirmation theory and the IS continuance literature, we investigate whether, when and how software updates affect users' continuance intentions (CI). Based on a controlled laboratory experiment, we find a positive effect of feature updates on users' CI. According to this effect, software vendors can increase their users' CI by delivering features through updates after a software has been released and is already used by customers. We also find that users prefer frequent feature updates over less frequent update packages that bundle several features in one update. However, the positive effect from updates occurs only with functional feature updates and not with technical non-feature updates, disclosing update frequency and update type as crucial moderators to this effect. Furthermore, we unveil that this beneficial effect of feature updates operates through positive disconfirmation of expectations, resulting in increased perceived usefulness and satisfaction. Implications for research and practice as well as directions for future research are discussed.

Keywords: Software updates, IT features, IS continuance, IS post-adoption, Expectation-confirmation theory

4.1 Introduction

In recent years, software vendors have increasingly leveraged software updates as a measure to modify and enhance their software products, while they are already being used by their customers. This phenomenon is particularly prevalent in the area of mobile applications and operating systems, but updates have also been used long before in the desktop space. Apple iPhone users, for instance, regularly receive updates for their apps. On the desktop, web browsers such as Google Chrome and Mozilla Firefox continuously receive updates, which extend their functionalities. Other examples include Microsoft Windows, the Adobe Reader and Sun’s Java platform which all regularly receive updates to close security gaps or fix minor flaws.

This ubiquitous use of updates by software vendors in practice reflects in a large body of research on the technical design of software, its maintenance and management. Research on software engineering (Sommerville 2010), including software product lines (Clements and Northrop 2002), software release planning (Svahnberg et al. 2010) and software evolution and maintenance (Mens and Demeyer 2008) explores how and when software functionality should be developed and delivered in order to maintain the technical integrity of the software and optimize the vendor’s production process. While this stream of research does account for customer needs, its primary focus lies on the supply side, exploring technical design aspects of software. There is as yet, however, little understanding of the user’s perspective on software updates—the demand side. In particular, the behavioral dimension, i.e., how updates are perceived by users is still an under-explored area that has so far received only minimal research attention (Hong et al. 2011; Sandberg and Alvesson 2011). Investigating the effect of software updates on users’ beliefs, attitudes and behaviors regarding an information system (IS), however, might be beneficial for software vendors and of particular interest in the postadoption context, because users’ continuance decisions (i.e., customer loyalty) are strongly influenced by their experiences made during actual IS use (Bhattacherjee and Barfar 2011). For software vendors, shedding light on the role of software updates for the IS continuance decision can thus result in a better understanding of how to deliver updates to users in order to achieve desirable performance outcomes such as higher user loyalty and sustained revenue streams.

From a research perspective, a better understanding of software updates from a user’s perspective has the potential to increase the explanatory and predictive power of existing postadoption theory. In conjunction with pre-adoption and adoption, post-adoption research constitutes IS usage, one of the most mature fields in IS (Jasperson et al. 2005). However, compared to research on pre-adoption and adoption decisions, post-adoption studies still remain sparse. Many scholars have thus called for studies that explicitly focus on post-adoptive phenomena (e.g., Benbasat and Barki 2007). Furthermore, researchers studying IS post-adoption phenomena often tend to conceptualize information systems as a monolithic and coarse-grained black box, rather than as collection of specific and finer-grained features that are dynamic and alterable over time. However, understanding the granularity of software and its changes through software updates would help explain how users' beliefs, attitudes, and behaviors fluctuate over time as a result of the dynamic nature of information systems. In addition, the focus on changes in beliefs, attitudes and behaviors, emanating from the IT artifact itself rather than from other IT-unrelated environmental stimuli, is a response to several calls for research from IS scholars who criticize the negligence of the IT artifact's role in IS research (Benbasat and Zmud 2003; Hevner et al. 2004; Orlikowski and Iacono 2001). From a theoretical perspective, it is not only important to explore *whether* software updates have an effect on users' beliefs, attitudes and behaviors towards the software and their continuance intentions (CI) in particular. It is equally important to examine *when* and *how* these effects might occur, thus providing a profound theoretical explanation as well as the possibility to predict user reactions towards software updates. Against this backdrop, our objective is to study software updates as a measure by which a vendor can provide maintenance for or extend the functionality of its software over time, while it is already being used by customers. To the best of our knowledge, software updates and their effects on users' IS continuance decisions are thus far still underexplored in the IS post-adoption context. We therefore seek to address this research gap by examining the questions of *whether, when and how software updates influence users' IS continuance intentions.*

In line with the mentioned research gaps, we contribute to prior research in three important ways. First, our overarching contribution is to advance the predominant view of information systems in post-adoption literature from a mostly monolithic and static to a finer-grained and more dynamic perspective by showing how a functionally malleable information system might influence users' beliefs, attitudes and behaviors over time. As such, we also accentuate the changing nature of the IT artifact for users' CI and thus explicitly consider the software product lifecycle in our theorizing. Second, we identify substantially different user reactions to different update types and modes of delivery. While feature updates increase users' continuance intentions, technical non-feature updates (e.g. bug fixes) have no effect on the intention to continue using the software. Moreover, we find that users prefer features to be delivered in individual updates over a delivery of features in larger but less frequent update packages comprising several features. Update type and frequency thus seem to moderate the effect of software updates on users' continuance intentions. Third, we not only investigate the direct effect of software updates on CI; we also open up the theoretical black box of *how* software updates influence IS continuance intention by highlighting the complementary roles of cognition and affect. From a practitioner's perspective, our study offers implications for software vendors on how to deliver software updates in order to increase their customers' loyalty (i.e., CI). We not only provide guidelines on which actions to take, but also on which measures to avoid in order to benefit from the positive effect of feature updates on users' CI.

4.2 Theoretical Foundations

4.2.1 Software Updates

Consistent with previous research (e.g., Dunn 2004), we consider software updates to be self-contained modules of software that are provided to the user for free in order to modify or extend software after it has been rolled out and is already in use. Software updates are thus not discrete and stand-alone programs but rather integrate into the base software once they are applied to it. In practice, software updates are applied to different types of software, such as system software (e.g., operating systems, drivers) or application software (e.g., office suites) and on different platforms (e.g., desktop computers, mobile devices). With varying terminology (e.g. update, upgrade, patch, bug fix, or hotfix), the concept of software updates is repeatedly addressed throughout the software engineering literature (Sommerville 2010), such as software release planning, software maintenance and evolution and software product lines (Svahnberg et al. 2010; Shirabad et al. 2001; Weyns et al. 2011).

In contrast to this rich stream of technical literature dealing with software updates from the developers' perspective, the customer perspective has received less attention (Morgan and Ngwenyama 2015). Specifically users' perceptions of updates have so far been explored only sparsely. This reflects in few IS studies dealing with updates. Hong et al. (2011), for example, explore user's acceptance of information systems that change through the addition of new functionality. Benlian (2015), on the other hand, explores different IT feature repertoires and their impact on users' task performance, but does not consider changes in functionality through updates. Other IS studies that found updates to influence usage behaviors, have often pushed them to the sidelines, treating them as control variables for investigating other phenomena (e.g., Claussen et al. 2013). Existing IS research has, however, not explored the specific impact of updates on users' beliefs and attitudes regarding an IS. Specifically, the impact of different modes of delivery (e.g., frequency of updates) and different update types have so far not been explored.

Concerning the present study, we distinguish between two basic types of software updates: feature updates and non-feature updates (e.g., Microsoft 2015a). Feature updates change the core functionality of software to which they are applied. Functionality can be added to or removed from the original version of the software and refers to distinct, discernible features which are deliberately employed by the user in accomplishing the task for which he uses the software. The Facebook app for smartphones and tablet computers provides an example for this type of update. In a 2013 update, it received a comprehensive instant messaging feature (Etherington 2013). An example from the desktop space example is the ‘tab sync’ functionality, which was added to the browser Google Chrome in 2012 via a feature update. It enabled users to synchronize websites (tabs) across different computers and mobile devices to seamlessly continue browsing when switching devices (Mathias 2012). In contrast to feature updates, technical non-feature updates do not change the core functionality of software but only correct flaws (e.g., bug fixes) or change software properties that are not directly related to its core functionality (e.g., improvements in stability, security or performance) (Popović et al. 2001). Thus non-feature updates usually do not directly affect the user’s interaction with the software and therefore the changes in the software are often not even evident to the user. Moreover, non-feature updates often fix problems that concern only a small number of users, use cases or setups but have no consequence for the majority of users. Examples for this type of update are the ‘hot fixes’ that Microsoft regularly distributes via its Windows Update service.

4.2.2 Information Systems Continuance

Together with research on users’ pre-adoption activities and the adoption decision, postadoption research constitutes the research field IS usage—one of the most mature fields in IS (Jasperson et al. 2005). Post-adoption research explores users’ beliefs, attitudes, and behaviors around the continued use of an IS (Karahanna et al. 1999; Bhattacherjee 2001). In this regard, the term information systems continuance refers to “*sustained use of an IT by individual users over the long-term after their initial acceptance*” (Bhattacherjee and Barfar 2011, p. 2). To explore users’ intentions to continue or discontinue using an IS, Bhattacherjee (2001) adopts expectation-confirmation theory (ECT) (Locke 1976, Oliver 1980, 1993, Anderson and Sullivan 1993). In Bhattacherjee’s (2001) model, a user’s intention to continue using an IS (CI) is the core dependent variable. It is positively influenced by satisfaction

(SAT) and perceived usefulness (PU). PU captures the expectations about future benefits from IS usage (Bhattacherjee and Barfar 2011) and has a positive impact on SAT and CI (Bhattacherjee 2001). While SAT represents the affective part of the continuance model, PU rather represents the cognitive one. The concept of PU has been carried over from adoption theory (Davis et al. 1989). Perceived ease of use (PEoU), which is the second main driver of technology adoption is, however, not part of the IS continuance model. While ease of use is an important determinant of individual technology adoption decisions (i.e., at earlier stages of use), research has found ambiguous results regarding its effect on continuance decisions (Davis et al. 1989; Bhattacherjee 2001; Hong et al. 2006). Studies even suggest that its influence on usage decisions disappears in later stages of use, once users gain experience with the information system (Karahanna et al. 1999).

The IS continuance model moreover suggests that users compare their pre-usage expectations of an IS with their perception of the performance of this IS during actual usage (Bhattacherjee 2001). If perceived performance exceeds their initial expectations, users experience positive disconfirmation which increases their PU and SAT. If perceived performance falls short of the initial expectations, negative disconfirmation occurs and users’ PU and SAT are reduced (Bhattacherjee and Barfar 2011). The concept of positive (negative) disconfirmation thus has two prerequisites—unexpectedness and a positive (negative) experience (Oliver 1980; Bhattacherjee 2001). ECT moreover posits expectations as a relative, subjective reference point or baseline (i.e., not an absolute, objective value) upon which the user makes his comparative judgment (Helson 1964; Oliver 1980).

In its original form, the IS continuance model (Bhattacherjee 2001) has a static perspective on the IS continuance setting, failing to account for a change in user believes and attitudes over time. In response to this limitation, Bhattacherjee and Premkumar (2004) introduced a more dynamic perspective, showing that beliefs and attitudes do not only change from pre usage to actual usage but also during the ongoing usage of an IS (Kim and Malhotra 2005). While this dynamic perspective already provides valuable insights into the drivers of post-adoption behavior, it still neglects the IT artifact’s changing and malleable nature. Evidence from practice shows, however, that information systems are constantly modified over time, for example, when vendors update and change their software or introduce new software generations. Considering the fact that beliefs and attitudes change over time during the ongoing use as a result of users’ experience with the IT (Bhattacherjee and Premkumar 2004), it is reasonable to assume that a change in the IT artifact may also induce a change in users’ beliefs and attitudes toward it. Kim and Malhotra (2005), Kim (2009), Ortiz de Guinea and Markus (2009) and Ortiz de Guinea and Webster (2013), for instance, have provided strong evidence that external factors such as IS-related tasks as well as the IS itself can shape users' beliefs, attitudes and even their affect regarding the IT in later usage stages. In order to investigate the changing nature of the IT artifact and its effect on users' beliefs, attitudes and behaviors during post-adoption use, we explore software updates through the lens of the disconfirmation mechanism in ECT.

4.3 Hypotheses Development

In this section, we develop our hypotheses about how and under which conditions updates can influence users' beliefs and attitudes in post-adoption software usage. Specifically, we explore decisions on continued use or discontinuance in settings where use is not mandated, such as consumer software. To this end, we focus on software updates which are recognized by the user during usage through explicit notification and ignore software updates that are implemented 'behind the scenes'. Within this scope, we further distinguish between two different types of software updates (feature updates and non-feature updates) and two modes of delivery (low and high frequency).

4.3.1 The Effect of Feature Updates on Users' Continuance Intentions

Research on information system characteristics in post-adoption user behavior has repeatedly identified system design features to affect users' beliefs and attitudes regarding an information system (Saeed and Abdinnour-Helm 2008; Nicolaou and McKnight 2011). We thus argue that a change in information systems characteristics has the potential to also affect a user's beliefs and attitudes regarding this information system. Specifically, we suggest that receiving a free feature update that provides additional functionality which directly serves users in accomplishing their IS-based tasks will be perceived as a positive experience with the software (Goodhue and Thompson 1995; Larsen et al. 2009).

Furthermore, it is reasonable to assume that feature updates are usually not anticipated by users and can thus be perceived as unexpected experiences with the software. Even if a software vendor does provide release plans about future feature updates, we suggest that in practice, most users—and especially consumers—are unlikely to follow such update plans in detail for each and every individual software product they have in use. If feature updates are perceived as unexpected and positive experiences during usage, according to ECT, they should induce perceived positive disconfirmation (Oliver 1980). Drawing on ECT and the IS continuance model (Bhattacherjee 2001), it is thus plausible that this perceived positive disconfirmation will increase users' CI regarding the updated software.

Regarding our assumptions about feature updates, we acknowledge that in practice, there might be cases, where feature updates are perceived negatively by users. For example, if features are intentionally removed (e.g. because of expired licensing deals), software functionality is unintentionally impaired or updates bring major changes to the software which necessitate users to learn and adjust. Nevertheless, we argue that in most cases, feature updates are intended to enhance the software, help users and are thus perceived positively.

We also acknowledge that receiving feature updates might lead to interruptions in the workflow through notifications or required installations. While previous research on IT events in post-adoption use (Tyre and Orlikowski 1994; Ortiz de Guinea and Webster 2013) and interruptions in human computer interaction (Hodgetts and Jones 2007; Sykes 2011) has found negative impacts from update notifications on users' workflow and their beliefs and attitudes towards the updated system, we suggest that vendors are aware of this and deliberately try to minimize these inconveniences. Moreover, even if updates result in undesired interruptions of workflow, these are one-time events that should be are outweighed by the benefits of receiving new, helpful features and their repeated use and contribution to task accomplishment. We thus derive our first hypothesis:

H1a: Receiving functionality through feature updates after the first release of a software increases users' continuance intentions.

4.3.2 The Role of Frequency in the Delivery of Feature Updates

New features are often the result of subsequent, incremental software development. When vendors want to deliver new features to their users through updates, they can often choose between different delivery-strategies. A vendor may deliver each individual feature in a separate update, once the feature is developed. Another option is to accumulate a certain number of features and deliver them bundled together in a larger update-package. (Under the latter strategy, the user is assumed to be unaware when individual features are developed and that they might be held back some time until delivery.) Over the course of time, the former option would result in a high update frequency, while the latter results in a low update frequency. Nonetheless, under both strategies, the same amount of features is delivered to users.

While both feature delivery strategies ultimately lead to the same feature endowment for the user, theory implies that these strategies might be perceived differently by the users. More specifically, ECT implies that the positive disconfirmation from a feature update depends on a relative change in functionality compared to a user's subjective reference point (i.e., the pre-update configuration of the software) rather than an absolute change (Helson 1964; Oliver 1980).

Once features are subsequently delivered through updates, each update is likely to elicit positive disconfirmation. Following Adaptation Level Theory (Helson 1964) and ECT (Oliver 1980) which build the basis for the IS continuance model (Bhattacherjee 2001), this high-frequency feature delivery strategy could then lead to a higher level of CI than the low frequency delivery strategy which provides users with the same type and amount of features but bundled in larger update packages. Moreover, if features are delivered through individual updates, they may 'stick out' more than if they are one among many, bundled in a larger update package. The positive contribution of an individual feature may thus be highlighted more and increase CI even further.

A drawback of the high-frequency delivery strategy is that it is accompanied by more frequent interruptions in the workflow by the previously outlined update notifications and installations, for example. However, we suggest, that in practice, the benefits from receiving features outweigh the drawbacks from the interrupted workflow even under the high-frequency delivery strategy where features are delivered individually, accompanied by notifications and other associated drawbacks.[6]

To summarize, because of the nature of the disconfirmation mechanism in ECT, which operates through an evaluation of relative instead of absolute change, users of software that receive functionality via incremental feature updates under a high-frequency update delivery strategy will likely have a higher intention to continue using this software than under a low

[6] We acknowledge that once frequency increases to a certain point, updates may no longer be perceived beneficial. In this extreme case, a decreasing marginal utility from additional features (Nowlis and Simonson 1996) in combination with overly frequent workflow interruptions from notifications and installations, may outweigh the benefits from the feature updates and no longer increase CI or even diminish it. However, the update frequencies which can usually be observed in practice should not reach this point.

frequency delivery strategy even though users receive the same set of features under both strategies.[7] We thus hypothesize:

H1b: Users have a higher continuance intention regarding software that receives features in individual updates compared to software that receives the same set of features in one update package.

4.3.3 The Effect of Non-Feature Updates on Users' Continuance Intentions

In addition to unexpectedness, the second key component that is required for the positive effect of software updates to occur is the positive experience from an increase in functionality of the software. While non-feature updates are also unexpected events during usage (see hypothesis 1a), they lack the added functionality of their feature update counterparts and are thus unlikely to exert similar positive effects on CI. While such non-feature updates technically alter the software through bug fixes or security improvements, these changes do not directly serve users in accomplishing their IS-based tasks by offering useful functionality. In terms of ECT, this means that non-feature updates do not lead to the necessary perceived relative change in functionality compared to the reference point (i.e., the pre-update configuration of the software) (Helson 1964; Oliver 1980). In sum, we argue that software that receives non-feature updates instead of feature updates will not exert positive disconfirmation. This will, in turn, result in a lower CI compared to the scenarios suggested in hypothesis 1a and 1b. Furthermore, non-feature updates do not only fail to deliver functionality that directly serves users in accomplishing their IS-based tasks. They may even be perceived as unsolicited interruptions in the workflow without being accompanied by any direct benefit for accomplishing the immediate IS-based task (i.e., without additional helpful functionality). This might even diminish CI. We thus hypothesize:

H2a: Receiving software fixes through non-feature updates after the first release of a software does not increase users' continuance intentions.

[7] In our theorizing regarding hypothesis 1b and 2b, we assume software updates of one type to deliver common (non-)features with equivalence regarding their content across the hypothesized conditions. We make this assumption to properly reflect the practice (free updates do usually not deliver uniquely extraordinary content) and because previous research has found that uncommon, unique product attributes may bias consumer decisions and thus interfere with our attempt to conceptually isolate the psychological mechanism through which software updates might influence users' continuance intentions (e.g. Dhar and Sherman 1996).

4.3.4 The Role of Frequency in the Delivery of Non-Feature Updates

Following the logic as outlined above, non-feature updates should not increase CI, independent from their frequency of delivery. Moreover, non-feature updates which are delivered with high frequency may even diminish CI since they interrupt users' workflow even more frequently without any direct and immediate benefit. However, we argue that the delivery of updates has nowadays become mostly seamless, minimizing the interruptions in workflow and other downsides from applying updates. Therefore, we suggest that unless non-feature updates reach extreme levels of frequency, the will not affect users' CI. We thus hypothesize:

H2b: Users have the same continuance intention regarding software that receives fixes in individual updates as regarding software that receives the same set of fixes in one update package.

4.3.5 The Mediating Roles of Disconfirmation, Perceived Usefulness and Satisfaction

As outlined in the theoretical foundations, ECT (Oliver 1980) applied to the context of the IS continuance model (Bhattacherjee 2001) implies that unexpected feature updates should be perceived by users as helpful 'gifts' from the vendor that exceed their expectations regarding the software. Feature updates thus lead to positive disconfirmation (DISC). Due to their lack of directly helpful content, non-feature updates, however, fail to exceed the users' expectations. The mediating effect of DISC on CI from receiving updates during use is thus conditional to the type of the received update. The relationship between software updates, positive disconfirmation and continuance intentions is therefore one of a moderated mediation where DISC is only increased by updates that contain features (Hayes 2013). Furthermore, according to the IS continuance model, the conditionally increased DISC from feature updates subsequently leads to higher PU and SAT.

PU, which represents the cognitive component of the IS continuance model is a forward-looking construct and captures the future benefits from using the software (Bhattacherjee and Barfar 2011). Feature updates increase PU because they provide a relative improvement of the software by extending its functionality compared to the pre-update state. After the disconfirming feature update, the software thus becomes more useful to achieve present and future tasks. Consequently, this will increase users' intentions to continue using the updated software (CI).

Being a welcomed and surprising ‘gift’ from the vendor, the positive disconfirmation from feature updates will also reflect in the affective component of the IS continuance model. Users who receive a free update that improves the software with which they work will be more satisfied (SAT) than users who do not receive such a pleasant update (Bhattacherjee and Barfar 2011). These higher levels of satisfaction will also make it more likely that users will return to the updated software for future tasks (CI).

The previously discussed PEoU should, however, not be involved in this mediation mechanism. While additional features from updates extend the functionality of a software and thus increase its usefulness, added features do usually not change the user interface or the overall interaction with the program. They are thus not expected to affect the ease of use of the updated program (Karahanna et al. 1999). To summarize, software updates affect users’ continuance intentions (CI) through a causal chain of effects that conditionally originates from the positive disconfirmation of unexpectedly receiving additional functionality during usage (DISC) and is subsequently mediated by perceived usefulness (PU) and satisfaction (SAT). We thus derive our moderated mediation-hypotheses:

H3a: Software updates increase continuance intentions because they positively disconfirm users’ expectations regarding the software only when they deliver additional functionality.

H3b: Positive disconfirmation from receiving additional features through updates leads to higher continuance intentions by increasing perceived usefulness and satisfaction.

Our theorizing about the impact of software updates on users’ continuance intentions is summarized by the moderated multiple-mediation model shown in Figure 4-1.

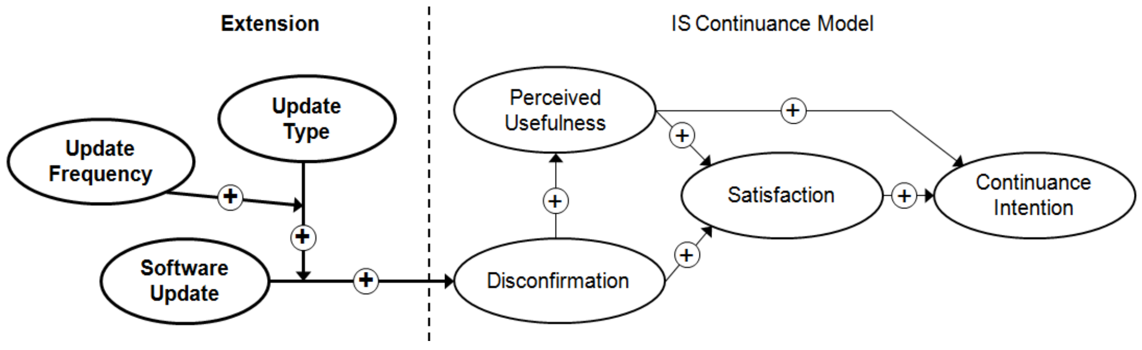


Figure 4-1: Research Model

4.4 Method

4.4.1 Experimental Design

With the goal to examine the effects of software updates on users' CI as suggested by our hypotheses, we opted for a laboratory experiment that allowed us to investigate and isolate the causal mechanisms that operate between software updates and attitudinal user reactions. Even though this laboratory setting comes with the downsides of a simplified experimental task and a limited time span of observable usage, it also allows for an accurate identification of the hypothesized effects which we consider as crucial given that this study is the first to explore the effect of software updates on users' continuance intentions. A second reason for choosing an experiment was the indication from theory that, working through affect, the core mechanism behind our proposed effect of feature updates might be outside of their awareness, which made a cross-sectional survey with self-reported measures less suitable. Third, the experimental setting enabled us to account for the claims of numerous continuance researchers to put the IT artifact more at the center of investigation in post-adoption research by using an IS as basis for manipulations.

We thus conducted a posttest-only 2x2 full-factorial between-subjects laboratory experiment with manipulations of update type (feature update vs. non-feature update), update frequency (low frequency vs. high frequency) and a hanging control group (no update) (Malaga 2000; Irmak et al. 2005; Hoffmann and Broekhuizen 2009). 135 participants were recruited at a large public university in Germany to evaluate the impact of software updates on the user's DISC, PU, SAT and CI. The participants used a word-processing program ('eWrite') with a simplified user interface that was developed and tailored to the purposes of this experiment to complete a text formatting task. All experimental groups started with the same software configuration including one feature. The hanging control group (group A) did not receive any updates during the time of the experiment. The first treatment group (B) received three non-features in one update package in the same time span. The second treatment group (C) received three features in one update package. The third treatment group (D) received the same non-features as group B, only spread out over the experimental time span in three individual updates. Lastly, the fourth treatment group (E) received three features in three individual updates spread out over the experimental time span. Figure 4-2 illustrates the experimental implementation.

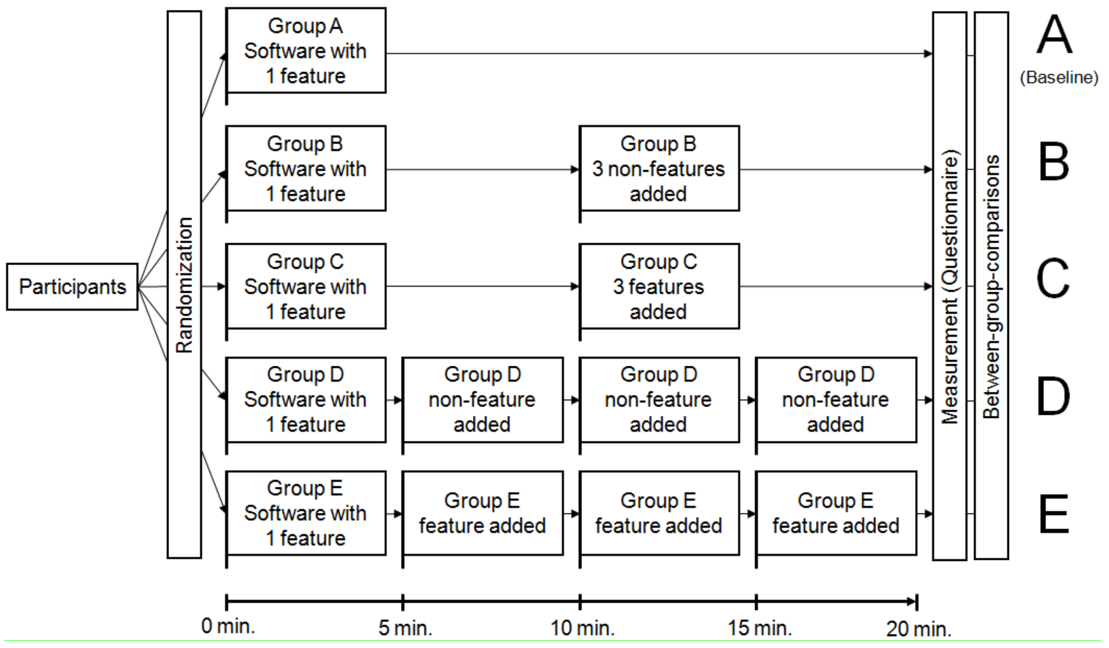

Participants
Randomization
Group A Software with 1 feature
Group B Software with 1 feature
Group B 3 non-features added
Group C Software with 1 feature
Group C 3 features added
Group D Software with 1 feature
Group D non-feature added
Group D non-feature added
Group D non-feature added
Group E Software with 1 feature
Group E feature added
Group E feature added
Group E feature added
Measurement (Questionnaire)
Between-group-comparisons
A
(Baseline)
B
C
D
E
0 min.
5 min.
10 min.
15 min.
20 min.

Figure 4-2: Experimental Setup, Groups, and Treatments

4.4.2 Manipulation of Independent Variables

In our experiment, we used a word-processing program for two reasons: Our first criterion was ensuring a basic familiarity with the program of choice for all participants. Because nowadays almost any young person, especially students, needs to work with word-processing programs, we considered this criterion to be met.[8] Second, to minimize unwanted variance in our response data, we were looking for software features that are preferably value-free, equivalent[9], and independent (i.e., modular). We used a total of four text formatting features in our word-processing system context: 1) font size, 2) font style, 3) font, and 4) text alignment, and three non-feature updates: 1) improvement of program stability, 2) elimination of a security gap in the program, and 3) improvement of program speed. By adding new text-formatting functionalities the feature updates were directly related to the experimental task. In contrast, the non-feature updates were not related to the task. They did not change the program at all but only consisted of a notification explaining their alleged content. This implementation was chosen to properly resemble the experience that many users have in

[8] Section 4.4.4 shows that this assumption is clearly met in our sample, as the vast majority of our participants indicated a regular use of word-processing programs and reported high levels competence in the use of word-processing programs.

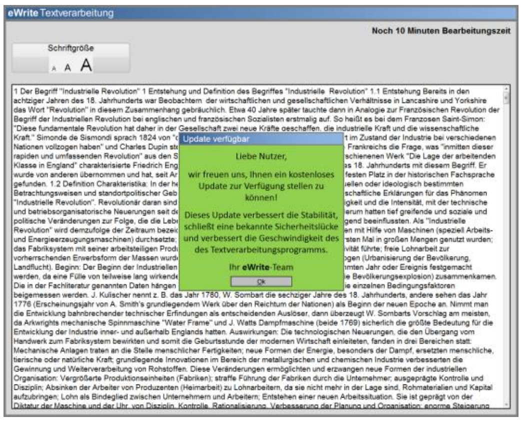
[9] The scope and importance of the four text formatting functionalities in groups A, C and E for completing the experimental task were held constant in order to avoid potential confounding effects emerging from the nature of the updates' contents. The functional equivalence of the individual feature updates for the text formatting task had been validated in a pre-test study with 52 subjects that were recruited using WorkHub, a crowdsourcing platform similar to Amazon Mechanical Turk (Paolacci et al. 2010). The subjects participated online for a small payment. No significant differences emerged among the four text-formatting features (all t < 1).

practice when receiving non-feature updates (section 4.2.1). The available time for task completion was 20 minutes. In condition B, participants simultaneously received the three non-features in one update ten minutes after having started to work on the task. In condition C, participants simultaneously received features 2, 3, and 4 after ten minutes. In the condition D, participants received the first non-feature update after five minutes, the second non-feature update after ten minutes and the third non-feature update after fifteen minutes. In the condition E, participants received the first feature update (with feature 2) after five minutes, the second feature update (with feature 3) after ten minutes and the third feature update (with feature 4) after fifteen minutes. Participants in each group were informed about updates via a pop-up notification window at the center of the screen. It contained a brief explanation of the update's content and required them to confirm the update by clicking an 'Ok' button before they could proceed with their experimental task. After confirming the notification, participants in the feature-update conditions (C and E) could immediately use the new features. The notification had been included in order to ensure awareness of the software update. Figure 4-3 provides examples of the user interface.

Non-feature Update

Group B (1 Update with 3 Non-features) after 10 min.

Low Update Frequency



Noch 10 Minuten Bearbeitungszeit

1 Der Begriff "Industrielle Revolution" 1 Entstehung und Definition des Begriffes "Industrielle Revolution" 1.1 Entstehung Bereits in den
achtziger Jahren des 18. Jahrhunderts war Beobachtern der wirtschaftlichen und gesellschaftlichen Verhältnisse in Lancashire und Yorkshire
das Wort "Revolution" in diesem Zusammenhang gebräuchlich. Etwa 40 Jahre später tauchte dann in Analogie zur Französischen Revolution der
Begriff der Industriellen Revolution bei englischen und französischen Sozialisten erstmalig auf. So heißt es bei dem Franzosen Saint-Simon:
"Diese fundamentale Revolution hat daher in der Gesellschaft zwei neue Kräfte geschaffen, die industrielle Kraft und die wissenschaftliche
Kraft." Simonde de Sismondi sprach 1824 von "c t im Zustand der Industrie bei verschiedenen
Nationen vollzogen haben" und Charles Dupin ste Frankreichs die Frage, was "inmitten dieser
rapiden und umfassenden Revolution" aus den S rschienenen Werk "Die Lage der arbeitenden
Klasse in England" charakterisierte Friedrich Eng as 18. Jahrhunderts mit diesem Begriff. Er
wurde von anderen übernommen und hat, seit Ar festen Platz in der historischen Fachsprache
gefunden. 1.2 Definition Charakteristika: In der he uellen oder ideologisch bestimmten
Betrachtungsweisen und standortpolitischer Geb schaftliche Erklärungen für das Phänomen
"Industrielle Revolution". Revolutionär daran sind igkeit und die Intensität, mit der technische
und betriebsorganisatorische Neuerungen seit de lerum hatten tief greifende und soziale und
politische Veränderungen zur Folge, die die Lebe gend beeinflussten. Als "Industrielle
Revolution" wird demzufolge der Zeitraum bezei en mit Hilfe von Maschinen (speziell Arbeits-
und Energieerzeugungsmaschinen) durchsetzte; rsten Mal in großen Mengen genutzt wurden;
das Fabriksystem mit seiner arbeitsteiligen Produ ivität führte; freie Lohnarbeit zur
vorherrschenden Erwerbsform der Massen wurd ogen (Urbanisierung der Bevölkerung,
Landflucht). Beginn: Der Beginn der Industriellen mten Jahr oder Ereignis festgemacht
werden, da eine Fülle von teilweise lang wirkende ie Bevölkerungsexplosion) zusammenkamen.
Die in der Fachliteratur genannten Daten hängen ie einzelnen Bedingungsfaktoren
beigemessen werden. J. Kulischer nennt z. B. das Jahr 1780, W. Sombart die sechziger Jahre des 18. Jahrhunderts, andere sehen das Jahr
1776 (Erscheinungsjahr von A. Smith's grundlegendem Werk über den Reichtum der Nationen) als Beginn der neuen Epoche an. Nimmt man
die Entwicklung bahnbrechender technischer Erfindungen als entscheidenden Auslöser, dann überzeugt W. Sombarts Vorschlag am meisten,
da Arkwrights mechanische Spinnmaschine "Water Frame" und J. Watts Dampfmaschine (beide 1769) sicherlich die größte Bedeutung für die
Entwicklung der Industrie inner- und außerhalb Englands hatten. Auswirkungen: Die technologischen Neuerungen, die den Übergang vom
Handwerk zum Fabriksystem bewirkten und somit die Geburtsstunde der modernen Wirtschaft einleiteten, fanden in drei Bereichen statt:
Mechanische Anlagen traten an die Stelle menschlicher Fertigkeiten; neue Formen der Energie, besonders der Dampf, ersetzten menschliche,
tierische oder natürliche Kraft; grundlegende Innovationen im Bereich der metallurgischen und chemischen Industrie verbesserten die
Gewinnung und Weiterverarbeitung von Rohstoffen. Diese Veränderungen ermöglichten und erzwangen neue Formen der industriellen
Organisation: Vergrößerte Produktionseinheiten (Fabriken); straffe Führung der Fabriken durch die Unternehmer; ausgeprägte Kontrolle und
Disziplin; Absinken der Arbeiter von Produzenten (Heimarbeit) zu Lohnarbeitern, da sie nicht mehr in der Lage sind, Rohmaterialien und Kapital
aufzubringen; Lohn als Bindeglied zwischen Unternehmern und Arbeitern; Entstehen einer neuen Arbeitssituation. Sie ist geprägt von der
Diktatur der Maschine und der Uhr, von Disziplin, Kontrolle, Rationalisierung, Verbesserung der Planung und Organisation; enorme Steigerung

Liebe Nutzer,

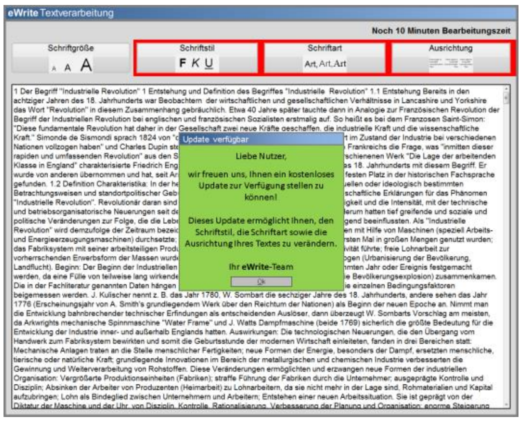
wir freuen uns, Ihnen ein kostenloses Update zur Verfügung stellen zu können!

Dieses Update verbessert die Stabilität, schließt eine bekannte Sicherheitslücke und verbessert die Geschwindigkeit des des Textverarbeitungsprogramms.

Ihr **eWrite**-Team

Feature Update

Group C (1 Update with 3 Features) after 10 min.



Noch 10 Minuten Bearbeitungszeit

1 Der Begriff "Industrielle Revolution" 1 Entstehung und Definition des Begriffes "Industrielle Revolution" 1.1 Entstehung Bereits in den
achtziger Jahren des 18. Jahrhunderts war Beobachtern der wirtschaftlichen und gesellschaftlichen Verhältnisse in Lancashire und Yorkshire
das Wort "Revolution" in diesem Zusammenhang gebräuchlich. Etwa 40 Jahre später tauchte dann in Analogie zur Französischen Revolution der
Begriff der Industriellen Revolution bei englischen und französischen Sozialisten erstmalig auf. So heißt es bei dem Franzosen Saint-Simon:
"Diese fundamentale Revolution hat daher in der Gesellschaft zwei neue Kräfte geschaffen, die industrielle Kraft und die wissenschaftliche
Kraft." Simonde de Sismondi sprach 1824 von "c t im Zustand der Industrie bei verschiedenen
Nationen vollzogen haben" und Charles Dupin ste Frankreichs die Frage, was "inmitten dieser
rapiden und umfassenden Revolution" aus den S rschienenen Werk "Die Lage der arbeitenden
Klasse in England" charakterisierte Friedrich Eng as 18. Jahrhunderts mit diesem Begriff. Er
wurde von anderen übernommen und hat, seit Ar festen Platz in der historischen Fachsprache
gefunden. 1.2 Definition Charakteristika: In der he uellen oder ideologisch bestimmten
Betrachtungsweisen und standortpolitischer Geb schaftliche Erklärungen für das Phänomen
"Industrielle Revolution". Revolutionär daran sind igkeit und die Intensität, mit der technische
und betriebsorganisatorische Neuerungen seit de lerum hatten tief greifende und soziale und
politische Veränderungen zur Folge, die die Lebe gend beeinflussten. Als "Industrielle
Revolution" wird demzufolge der Zeitraum bezei en mit Hilfe von Maschinen (speziell Arbeits-
und Energieerzeugungsmaschinen) durchsetzte; rsten Mal in großen Mengen genutzt wurden;
das Fabriksystem mit seiner arbeitsteiligen Produ ivität führte; freie Lohnarbeit zur
vorherrschenden Erwerbsform der Massen wurd ogen (Urbanisierung der Bevölkerung,
Landflucht). Beginn: Der Beginn der Industriellen mten Jahr oder Ereignis festgemacht
werden, da eine Fülle von teilweise lang wirkende ie Bevölkerungsexplosion) zusammenkamen.
Die in der Fachliteratur genannten Daten hängen ie einzelnen Bedingungsfaktoren
beigemessen werden. J. Kulischer nennt z. B. das Jahr 1780, W. Sombart die sechziger Jahre des 18. Jahrhunderts, andere sehen das Jahr
1776 (Erscheinungsjahr von A. Smith's grundlegendem Werk über den Reichtum der Nationen) als Beginn der neuen Epoche an. Nimmt man
die Entwicklung bahnbrechender technischer Erfindungen als entscheidenden Auslöser, dann überzeugt W. Sombarts Vorschlag am meisten,
da Arkwrights mechanische Spinnmaschine "Water Frame" und J. Watts Dampfmaschine (beide 1769) sicherlich die größte Bedeutung für die
Entwicklung der Industrie inner- und außerhalb Englands hatten. Auswirkungen: Die technologischen Neuerungen, die den Übergang vom
Handwerk zum Fabriksystem bewirkten und somit die Geburtsstunde der modernen Wirtschaft einleiteten, fanden in drei Bereichen statt:
Mechanische Anlagen traten an die Stelle menschlicher Fertigkeiten; neue Formen der Energie, besonders der Dampf, ersetzten menschliche,
tierische oder natürliche Kraft; grundlegende Innovationen im Bereich der metallurgischen und chemischen Industrie verbesserten die
Gewinnung und Weiterverarbeitung von Rohstoffen. Diese Veränderungen ermöglichten und erzwangen neue Formen der industriellen
Organisation: Vergrößerte Produktionseinheiten (Fabriken); straffe Führung der Fabriken durch die Unternehmer; ausgeprägte Kontrolle und
Disziplin; Absinken der Arbeiter von Produzenten (Heimarbeit) zu Lohnarbeitern, da sie nicht mehr in der Lage sind, Rohmaterialien und Kapital
aufzubringen; Lohn als Bindeglied zwischen Unternehmern und Arbeitern; Entstehen einer neuen Arbeitssituation. Sie ist geprägt von der
Diktatur der Maschine und der Uhr, von Disziplin, Kontrolle, Rationalisierung, Verbesserung der Planung und Organisation; enorme Steigerung

Liebe Nutzer,

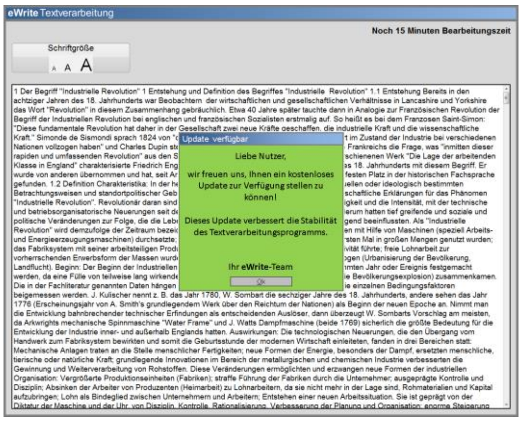
wir freuen uns, Ihnen ein kostenloses Update zur Verfügung stellen zu können!

Dieses Update ermöglicht Ihnen, den Schriftstil, die Schriftart sowie die Ausrichtung Ihres Textes zu verändern.

Ihr **eWrite**-Team

High Update Frequency

Group D (3 Non-Feature Updates) after 5 min.



eWrite Textverarbeitung

Noch 15 Minuten Bearbeitungszeit

Schriftgröße

A A A

1 Der Begriff "Industrielle Revolution" 1 Entstehung und Definition des Begriffes "Industrielle Revolution" 1.1 Entstehung Bereits in den
achtziger Jahren des 18. Jahrhunderts war Beobachtern der wirtschaftlichen und gesellschaftlichen Verhältnisse in Lancashire und Yorkshire
das Wort "Revolution" in diesem Zusammenhang gebräuchlich. Etwa 40 Jahre später tauchte dann in Analogie zur Französischen Revolution der
Begriff der Industriellen Revolution bei englischen und französischen Sozialisten erstmalig auf. So heißt es bei dem Franzosen Saint-Simon:
"Diese fundamentale Revolution hat daher in der Gesellschaft zwei neue Kräfte geschaffen, die industrielle Kraft und die wissenschaftliche
Kraft." Simonde de Sismondi sprach 1824 von "c … t im Zustand der Industrie bei verschiedenen
Nationen vollzogen haben" und Charles Dupin st … Frankreichs die Frage, was "inmitten dieser
rapiden und umfassenden Revolution" aus den S … schienenen Werk "Die Lage der arbeitenden
Klasse in England" charakterisierte Friedrich Eng … as 18. Jahrhunderts mit diesem Begriff. Er
wurde von anderen übernommen und hat, seit Ar … festen Platz in der historischen Fachsprache
gefunden. 1.2 Definition Charakteristika: In der he … uellen oder ideologisch bestimmten
Betrachtungsweisen und standortpolitischer Geb … schaftliche Erklärungen für das Phänomen
"Industrielle Revolution". Revolutionär daran sind … igkeit und die Intensität, mit der technische
und betriebsorganisatorische Neuerungen seit de … ierum hatten tief greifende und soziale und
politische Veränderungen zur Folge, die die Lebe … gend beeinflussten. Als "Industrielle
Revolution" wird demzufolge der Zeitraum bezei … en mit Hilfe von Maschinen (speziell Arbeits-
und Energieerzeugungsmaschinen) durchsetzte; … rsten Mal in großen Mengen genutzt wurden;
das Fabriksystem mit seiner arbeitsteiligen Produ … ivität führte; freie Lohnarbeit zur
vorherrschenden Erwerbsform der Massen wurd … ogen (Urbanisierung der Bevölkerung,
Landflucht). Beginn: Der Beginn der Industriellen … mmten Jahr oder Ereignis festgemacht
werden, da eine Fülle von teilweise lang wirkende … ie Bevölkerungsexplosion) zusammenkamen.
Die in der Fachliteratur genannten Daten hängen … ie einzelnen Bedingungsfaktoren
beigemessen werden. J. Kulischer nennt z. B. das Jahr 1780, W. Sombart die sechziger Jahre des 18. Jahrhunderts, andere sehen das Jahr
1776 (Erscheinungsjahr von A. Smith's grundlegendem Werk über den Reichtum der Nationen) als Beginn der neuen Epoche an. Nimmt man
die Entwicklung bahnbrechender technischer Erfindungen als entscheidenden Auslöser, dann überzeugt W. Sombarts Vorschlag am meisten,
da Arkwrights mechanische Spinnmaschine "Water Frame" und J. Watts Dampfmaschine (beide 1769) sicherlich die größte Bedeutung für die
Entwicklung der Industrie inner- und außerhalb Englands hatten. Auswirkungen: Die technologischen Neuerungen, die den Übergang vom
Handwerk zum Fabriksystem bewirkten und somit die Geburtsstunde der modernen Wirtschaft einleiteten, fanden in drei Bereichen statt:
Mechanische Anlagen traten an die Stelle menschlicher Fertigkeiten; neue Formen der Energie, besonders der Dampf, ersetzten menschliche,
tierische oder natürliche Kraft; grundlegende Innovationen im Bereich der metallurgischen und chemischen Industrie verbesserten die
Gewinnung und Weiterverarbeitung von Rohstoffen. Diese Veränderungen ermöglichten und erzwangen neue Formen der industriellen
Organisation: Vergrößerte Produktionseinheiten (Fabriken); straffe Führung der Fabriken durch die Unternehmer; ausgeprägte Kontrolle und
Disziplin; Absinken der Arbeiter von Produzenten (Heimarbeit) zu Lohnarbeitern, da sie nicht mehr in der Lage sind, Rohmaterialien und Kapital
aufzubringen; Lohn als Bindeglied zwischen Unternehmern und Arbeitern; Entstehen einer neuen Arbeitssituation. Sie ist geprägt von der
Diktatur der Maschine und der Uhr, von Disziplin, Kontrolle, Rationalisierung, Verbesserung der Planung und Organisation; enorme Steigerung

Update verfügbar

Liebe Nutzer,

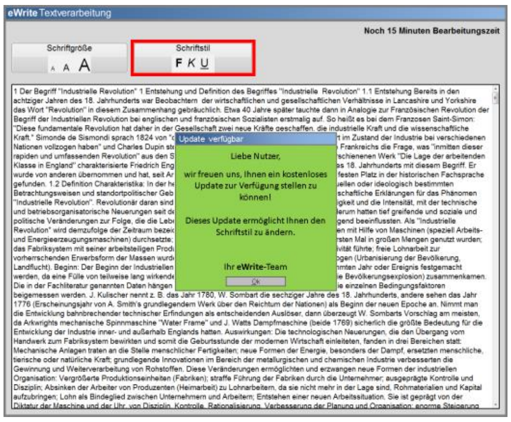
wir freuen uns, Ihnen ein kostenloses Update zur Verfügung stellen zu können!

Dieses Update verbessert die Stabilität des Textverarbeitungsprogramms.

Ihr eWrite-Team

Ok

Group E (3 Feature Updates) after 5 min.



eWrite Textverarbeitung

Noch 15 Minuten Bearbeitungszeit

Schriftgröße

A A A

Schriftstil

F K U

1 Der Begriff "Industrielle Revolution" 1 Entstehung und Definition des Begriffes "Industrielle Revolution" 1.1 Entstehung Bereits in den
achtziger Jahren des 18. Jahrhunderts war Beobachtern der wirtschaftlichen und gesellschaftlichen Verhältnisse in Lancashire und Yorkshire
das Wort "Revolution" in diesem Zusammenhang gebräuchlich. Etwa 40 Jahre später tauchte dann in Analogie zur Französischen Revolution der
Begriff der Industriellen Revolution bei englischen und französischen Sozialisten erstmalig auf. So heißt es bei dem Franzosen Saint-Simon:
"Diese fundamentale Revolution hat daher in der Gesellschaft zwei neue Kräfte geschaffen, die industrielle Kraft und die wissenschaftliche
Kraft." Simonde de Sismondi sprach 1824 von "c … t im Zustand der Industrie bei verschiedenen
Nationen vollzogen haben" und Charles Dupin st … Frankreichs die Frage, was "inmitten dieser
rapiden und umfassenden Revolution" aus den S … schienenen Werk "Die Lage der arbeitenden
Klasse in England" charakterisierte Friedrich Eng … as 18. Jahrhunderts mit diesem Begriff. Er
wurde von anderen übernommen und hat, seit Ar … festen Platz in der historischen Fachsprache
gefunden. 1.2 Definition Charakteristika: In der he … uellen oder ideologisch bestimmten
Betrachtungsweisen und standortpolitischer Geb … schaftliche Erklärungen für das Phänomen
"Industrielle Revolution". Revolutionär daran sind … igkeit und die Intensität, mit der technische
und betriebsorganisatorische Neuerungen seit de … ierum hatten tief greifende und soziale und
politische Veränderungen zur Folge, die die Lebe … gend beeinflussten. Als "Industrielle
Revolution" wird demzufolge der Zeitraum bezei … en mit Hilfe von Maschinen (speziell Arbeits-
und Energieerzeugungsmaschinen) durchsetzte; … rsten Mal in großen Mengen genutzt wurden;
das Fabriksystem mit seiner arbeitsteiligen Produ … ivität führte; freie Lohnarbeit zur
vorherrschenden Erwerbsform der Massen wurd … ogen (Urbanisierung der Bevölkerung,
Landflucht). Beginn: Der Beginn der Industriellen … mmten Jahr oder Ereignis festgemacht
werden, da eine Fülle von teilweise lang wirkende … ie Bevölkerungsexplosion) zusammenkamen.
Die in der Fachliteratur genannten Daten hängen … ie einzelnen Bedingungsfaktoren
beigemessen werden. J. Kulischer nennt z. B. das Jahr 1780, W. Sombart die sechziger Jahre des 18. Jahrhunderts, andere sehen das Jahr
1776 (Erscheinungsjahr von A. Smith's grundlegendem Werk über den Reichtum der Nationen) als Beginn der neuen Epoche an. Nimmt man
die Entwicklung bahnbrechender technischer Erfindungen als entscheidenden Auslöser, dann überzeugt W. Sombarts Vorschlag am meisten,
da Arkwrights mechanische Spinnmaschine "Water Frame" und J. Watts Dampfmaschine (beide 1769) sicherlich die größte Bedeutung für die
Entwicklung der Industrie inner- und außerhalb Englands hatten. Auswirkungen: Die technologischen Neuerungen, die den Übergang vom
Handwerk zum Fabriksystem bewirkten und somit die Geburtsstunde der modernen Wirtschaft einleiteten, fanden in drei Bereichen statt:
Mechanische Anlagen traten an die Stelle menschlicher Fertigkeiten; neue Formen der Energie, besonders der Dampf, ersetzten menschliche,
tierische oder natürliche Kraft; grundlegende Innovationen im Bereich der metallurgischen und chemischen Industrie verbesserten die
Gewinnung und Weiterverarbeitung von Rohstoffen. Diese Veränderungen ermöglichten und erzwangen neue Formen der industriellen
Organisation: Vergrößerte Produktionseinheiten (Fabriken); straffe Führung der Fabriken durch die Unternehmer; ausgeprägte Kontrolle und
Disziplin; Absinken der Arbeiter von Produzenten (Heimarbeit) zu Lohnarbeitern, da sie nicht mehr in der Lage sind, Rohmaterialien und Kapital
aufzubringen; Lohn als Bindeglied zwischen Unternehmern und Arbeitern; Entstehen einer neuen Arbeitssituation. Sie ist geprägt von der
Diktatur der Maschine und der Uhr, von Disziplin, Kontrolle, Rationalisierung, Verbesserung der Planung und Organisation; enorme Steigerung

Update verfügbar

Liebe Nutzer,

wir freuen uns, Ihnen ein kostenloses Update zur Verfügung stellen zu können!

Dieses Update ermöglicht Ihnen den Schriftstil zu ändern.

Ihr eWrite-Team

Ok

Figure 4-3: Sample Screenshots of Text Editor.

The simplifications in functionality and user interface of our experimental software were made on purpose and followed similar IS studies (e.g., Murray and Häubl 2011). This simplified setting enabled us to establish a controlled environment and unmistakably ascribe any observed changes in the dependent variables (DISC, PU, SAT, CI) directly to our experimental treatments. Nonetheless, such simplifications might also have some downsides. In our case, the participants' evaluations of the experimental word-processing program might have been diminished by associations with widely known, real programs such as Microsoft Word, which are much more refined and feature-rich. In order to mitigate this unwanted effect, we confronted the participants with a hypothetical scenario. Participants were asked to imagine that they were in 1980 and only word-processing programs with similar, basic functionalities were available. To support participants' imagination of this hypothetical scenario, an image of an old computer was positioned below the instructions, since images attract attention and are remembered better than just text (Levin 1981).[10]

The text which had to be formatted in the experimental task was a historical text about the Industrial Revolution. We consider this type of text, just like the program features, to be a 'neutral', objective one, compared for example to a newspaper article about a current event, which is often an emotive one. Furthermore, the text was long enough—as a pilot test showed—to keep the participants busy throughout the entire twenty minutes. Thus, we ensured that the participants could not complete their task too quickly and might have had to wait, which could have confounded our results. The participants were also instructed that they

[10] As the experiment's results show (see 4.5.1), the application of this vignette-like scenario seems to have been successful because the majority of subjects reported that they were able to put themselves into this hypothetical setting.

did not need to format the entire text, but to focus on the formatting *quality*, which in turn fostered the comprehensive use of *all* available program features.

A pilot test with 12 subjects was conducted to ensure that all of the treatments were manipulated according to the experimental design (Perdue et al. 1986). Specifically, subjects were asked about the functional equivalence of the individual updates, ease of use of the text-formatting editor and comprehensibility of instructions and items. Feedback and suggestions were obtained from participants after they had completed the pre-test experiment. The word-processing program and the questionnaire were accordingly revised for the main test.

4.4.3 Measures

Dependent Variables

We used validated scales with minor wording changes for all constructs, capturing the core part of the IS continuance model (DISC, PU, SAT, CI) (Bhattacherjee 2001). Measures for CI and DISC were adapted from Bhattacherjee (2001). Measures for PU and SAT were based on Kim and Son (2009). The questionnaire items are provided in Appendix A. Because constructs were measured with multiple items, summated scales based on the average scores of the multi-items were used in group comparisons (Zhu et al. 2012). Unless stated otherwise, the questionnaire items were measured on 7-point-Likert-scales anchored at (1)=strongly disagree and (7)=strongly agree.

To better understand the nature of disconfirmation from receiving the software updates in the four experimental conditions, we additionally applied a qualitative approach. This was done, in order to understand not only if expectations regarding software updates were confirmed or disconfirmed, but also for what reason. We asked participants in group B, C, D and E to first describe (i.e., to typewrite) how they felt when they received updates and, second, what they thought at that moment. We consider this combination of quantitative and qualitative measurement in this initial experimental study important to get a more complete picture of how updates may influence users' DISC, PU, SAT and CI by using the advantages of both measurement types (Venkatesh et al. 2013).

Control Variables

In our study, we included a set of control variables as well as the subjects' demographics as covariates to isolate the effects of the manipulated variables. Specifically, we controlled for the impact of *usage intensity of word-processing programs* in real life, *frequency of updates* in real life for productivity software/entertainment software and desktop computer/smartphone/tablet and *computer self-efficacy* (Marakas et al. 2007) on CI. We did this because previous research has repeatedly shown that past experiences and expertise with an information system can affect post-adoption beliefs, attitudes and behaviors (Venkatesh and Davis 2000; Jasperson et al. 2005; Kim and Malhotra 2005; Kim and Son 2009) and we wanted to avoid cofounding effects to our results from this. We also controlled for PEoU. As outlined before, PEoU has been identified as major driver of usage intentions but should lose its impact in the later stages of usage which we investigate (post-adoption). Nonetheless we sought to ensure that none of our results were cofounded by this variable. Furthermore, we examined participant's *motivation to process information* with one item (Suri and Monroe 2003), because this variable may also influence the response behavior of the participants and, thus, the validity of the results. After conducting the experimental task, participants were asked to what extent they had understood the *items' formulation* and to what extent they were *able to put themselves in the hypothetical situation described in the experimental task*. Finally, we included three control questions about the experimental treatments (Appendix B).

4.4.4 Participants, Incentives and Procedures

135 participants were recruited from the campus of a large public university at Germany. Each subject received 5€ for participating in the lab experiment. In order to align their motivations to properly fulfil the experimental task, 3 x 50€ Amazon vouchers and an iPad Mini were announced as rewards for the four most appealingly edited texts. Three participants were excluded from the sample based on the control questions. We therefore used a sample of 132 subjects in the following analysis. Of the 132 subjects, 70 were females. The participants' age ranged from 19 to 56, with an average value of 23.47 (σ=4.20). 125 participants were university students, five were employees and one was self-employed. One participant refused to state his occupation. The educational backgrounds of the participants were diverse, including physics, education, journalism, law, medical science, biology, engineering, sociology etc. 6.1% of the subjects (n=8) use word-processing programs less than one hour per month, 31.8% from one up to five hours (n=42), 40.9% between five and 30 hours (n=54), and 20.5% more than 30 hours per month (n=27). One participant refused to state his word-processing program usage.

When participants arrived at the laboratory, they were randomly assigned to a treatment/control group. All instructions were presented on the computer screen in order to reduce the interaction with the supervisor of the experiment. In order to ensure comparable initial conditions, participants were further presented with a program tutorial (a program screen similar to that of the actual experimental task). In this tutorial, the initially available features (depending on the experimental condition) were presented and each one was explained in a text bubble. Before they could proceed, all participants had to try out each available feature at least once by formatting a short sample text, ensuring that each participant had understood the program's functionality. On the next two screens, the actual experimental scenario and task, the time available to complete the task, and the results-based incentives were introduced. After having read these instructions, the participants could manually start the actual experimental task by clicking on a button. After having worked 20 minutes on the experimental task, participants had to complete a paper based questionnaire, which contained the measurement of all dependent variables (quantitative and qualitative), all control variables such as motivation to process information and perceived ease of use, the control questions and demographic variables such as gender and age. Finally, they were compensated for their participation and debriefed.

4.5 Data Analysis and Results

4.5.1 Control Variables and Manipulation Check

Based on the results of a series of Fisher's exact tests, we could conclude that there was no significant difference across the four experimental conditions and the hanging control group in terms of gender (p>0.1), age (p>0.1), intensity of using word-processing programs (p>0.1), as well as frequencies of the received updates (desktop/productive: p>0.1; desktop/entertainment: p>0.1; smartphone/productive: p>0.1; smartphone/entertainment: p>0.1; tablet/productive: p>0.1; tablet/entertainment: p>0.1). Furthermore, based on a series of ANOVA tests, we found no significant differences across the four experimental conditions and the control group regarding the task-relevant control variables perceived ease of use (F=1.395, p>0.1), motivation to process information (F=1.233, p>0.1) and items' formulations (F=0.783, p>0.1), the extent to which subjects were able to put themselves in the hypothetical situation described in the experimental task (F=0.382, p>0.1), understanding of the goals of the experiment (F=0.998, p>0.1) and liking of the utilized text (F=0.603, p>0.1). It is therefore reasonable to conclude that participants' demographics and task-relevant controls were homogeneous across the four conditions and the control group and thus did not confound the effects of our experimental manipulations.

To examine whether our experimental treatments worked as intended, a separate manipulation check study was performed with 27 other participants from the same population (Shu and Carlson 2014; Zhang et al. 2014). The subjects performed the identical experimental task as the participants of the main study, but answered questions regarding the manipulations instead of the questionnaire of the main study (Yin et al. 2014; Appendix C). Participants in the frequent update conditions indicated significantly higher levels of perceived frequency (M_{high}=5.272) than in the low update frequency conditions (M_{low}=2.500; F=16.204, p<0.01). Moreover, participants in the feature update conditions indicated significantly higher levels of perceived helpfulness for task completion (M_{very}=5.000) than in the non-feature update condition (M_{not}=1.364; F=44.693, p<0.01). Overall, the results from our manipulation checks suggest that our experimental treatments were successful.

Prior to testing the hypotheses, we also evaluated the control questions of the main study. As mentioned above, in three observations wrong conditions were stated. This led to the exclusion of those cases from the final sample (one subject had wrongly ticked all control questions, one subject had stated the wrong frequency of updates and one subject claimed to have received an update despite being in a group that did not receive any updates).

4.5.2 Measurement Validation

Because we adopted established constructs for our measurement, confirmatory factor analysis (CFA) was conducted to test the instrument's convergent and discriminant validity for the dependent variables (Levine 2005). Table 4-1 reports the CFA results using SmartPLS, version 2.0 M3 (Chin et al. 2003; Ringle et al. 2005) for the core constructs.[11]

[11] For brevity, we omitted items and/or detailed scale characteristics for computer self-efficacy and other control variables. These scales also satisfied the criteria regarding Cronbach's Alpha, AVE and Cross Loadings. Items and respective scale specifications can be obtained from the authors upon request.

Table 4-1: Results of Confirmatory Factor Analysis for Core Variables

Variables	Number of Indicators	Range of Standardized Factor Loadings*	Cronbach's Alpha	Composite Reliability (ρ_c)	Average Variance Extracted (AVE)
Continuance Intention (CI)	3	0.826 – 0.904	0.850	0.909	0.770
Satisfaction (SAT)	3	0.920 – 0.965	0.937	0.960	0.889
Perceived Usefulness (PU)	3	0.910 – 0.916	0.902	0.938	0.835
Disconfirmation (DISC)	3	0.837 – 0.887	0.823	0.894	0.738
Perceived Ease of Use (PEOU)	3	0.673 – 0.866	0.736	0.840	0.640
Note: * All factor loadings are significant at least at the p<0.01 level					

The constructs were assessed for reliability using Cronbach's alpha (Cronbach 1951). A value of at least 0.7 is suggested to indicate adequate reliability (Nunnally et al. 1994). The alphas for all constructs were well above 0.7. Moreover, the composite reliability of all constructs exceeded 0.7, which is considered the minimum threshold (Hair et al. 2011). Values for AVEs for each construct ranged from 0.738 to 0.889, exceeding the variance due to measurement error for that construct (that is, AVE exceeded 0.5). Thus, all of the constructs met the norms for convergent validity. In addition, for satisfactory discriminant validity, the square root of average variance extracted (AVE) from the construct should be greater than the variance shared between the construct and other constructs in the model (Fornell and Larcker 1981).

As seen from the factor correlation matrix in Table 4-2, all square roots of AVE exceeded inter-construct correlations, providing strong evidence of discriminant validity. Hence, the constructs in our study represent concepts that are both theoretically and empirically distinguishable.

Table 4-2: Means, Standard Deviations, and Correlation Matrix for Core Variables

Latent construct	M	SD	1	2	3	4	5
(1) Continuance Intention (CI)	5.690	1.448	**0.877**				
(2) Satisfaction (SAT)	4.112	1.829	0.499***	**0.888**			
(3) Perceived Usefulness (PU)	4.130	1.569	0.495***	0.741***	**0.835**		
(4) Disconfirmation (DISC)	3.822	1.450	0.471***	0.630***	0.673***	**0.859**	
(5) Perceived Ease of Use (PEoU)	5.631	1.364	0.327***	0.461***	0.617***	0.361***	**0.800**
Note: Bolded diagonal elements are the square root of AVE. These values should exceed inter-construct correlations (off-diagonal elements) for adequate discriminant validity; ***p<0.01, **p<0.05, *p<0.1.							

4.5.3 Hypotheses Testing

In order to test our hypotheses, we conducted one-way ANOVAs with planned contrast analyses with IBM SPSS Statistics 23. Table 4-3 presents the mean values of the dependent variables for groups A, B, C, D and E.

Table 4-3: Mean Values for Dependent Variables

	No Update, One Feature (A), n=26 (Control)	One Non-Feature Update (B), n=26	One Feature Update (C), n=27	Three Non-feature Updates (D), n=26	Three Feature Updates (E), n=27
DISC	3.141	3.295	4.383	3.269	4.852
PU	3.603	3.731	4.321	3.795	5.062
SAT	3.718	2.923	4.716	3.500	5.506
CI	5.256	5.141	5.876	5.795	6.395

Table 4-4 presents the deviations of the mean values of these dependent variables from the hanging control group (A), which received no update during usage.

Table 4-4: Mean Differences from Baseline (No Updates, Control Group A) and Significance Levels

	B-A	C-A	D-A	E-A
DISC	0.154	**1.242*****	0.128	**1.711*****
PU	0.128	**0.719****	0.192	**1.459*****
SAT	**-0.795***	**0.998****	-0.218	**1.788*****
CI	-0.115	**0.620***	0.539	**1.139*****

Table 4-5 provides the mean differences between feature and non-feature update treatment groups with low update frequency (C-B) and high update frequency (E-D).

Table 4-5: Direct Comparisons of Update Types

	C-B	E-D
DISC	**1.088*****	**1.583*****
PU	**0.590***	**1.267*****
SAT	**1.793*****	**2.006*****
CI	**0.735****	**0.600****

Correspondingly, Table 4-6 presents the mean differences between low-frequency updates and high-frequency updates for feature updates (E-C) and non-feature updates (D-B).

Table 4-6: Direct Comparisons of Update Frequencies

	E-C	D-B
DISC	**0.469***	-0.026
PU	**0.741***	0.064
SAT	**0.790****	0.577
CI	**0.519***	**0.654****

Because participants were randomly assigned to one of the experimental groups and everything except the treatment was held constant across the groups, any of the observed differences between the groups regarding the dependent variables can be ascribed to our update treatments. In hypothesis 1a, we claimed that software that receives additional functionality via feature updates will induce higher user CI compared to software that does not receive updates. The experiment's results indicate that on average, participants' CI in groups C (one feature update) and E (three feature updates) was significantly higher than participants' CI in group A (no updates). This can be seen from Table 4-4 (C-A, E-A). This result is further supported by the significant differences between the different update types found from the comparisons between groups B and C (C-B) as well as D and E (E-D). Table 4-5 shows these. Hence, hypothesis 1a is supported.

Moreover, hypothesis 1b posits that users prefer a high-frequency delivery of feature updates over a low-frequency delivery. As hypothesized, our results in Table 4-6 (see E-C) show that a high update frequency (i.e., three individual feature updates in the given timeframe; group E) was perceived more positively than the low update frequency condition (i.e., group C with one update comprising three features) in terms of CI. Hence, hypothesis 1b is supported.

With hypothesis 2a, we addressed the impact of non-feature updates on CI, claiming that users in these conditions (groups B and D) would not have a significantly higher CI compared to users in the no update condition (group A). In support of hypothesis 2a, the experiment's results in Table 4-4 indicate that on average, participants' CI in groups B and D was not significantly different from group A (B-A, D-A).

Hypothesis 2b claims that there is no difference in the users' perception between low-frequency and high-frequency non-feature updates terms of CI. Contrary to hypothesis 2b, participants showed on average higher levels of CI in the high-frequency non-feature update condition, compared to the corresponding low-frequency non-feature update condition (Table 4-6, D-B). It should however be noted that this does not mean that high-frequency non-feature updates have an overall positive effect (see supported hypothesis 2a). Moreover, other mean differences in CI that were found significant (Tables 4-6) were accompanied by significant changes in DISC, PU and SAT. This is not the case here (Table 4-6, D-B).

In order to test our mediation hypotheses (hypothesis 3a and 3b) a serial multiple mediator analysis (Hayes 2013) was performed on a sub-sample that comprised groups A and E[12] (n=53). To analyze the mediating effects of DISC, PU and SAT, we used PROCESS, a regression-based approach developed by Hayes (2013). PROCESS uses bootstrapping procedures for estimating direct and indirect effects. Figure 4-4 and Table 4-7 provide an overview of the analyzed conceptual model with direct and indirect paths. As recommended

[12] Group E was selected for analysis over group B because the condition (with three updates) better resembles a real world situation of repeatedly and frequently updated software.

by Hayes (2013), path coefficients are unstandardized because the independent variable (feature updates) is dichotomous. The results reveal that only the two indirect effect paths (1, 4) from high-frequency feature updates via DISC to CI and via DISC, PU and SAT to CI were significant. Moreover, the direct effect of feature updates on users' CI became insignificant after inclusion of the complete path, suggesting at least partial mediation (Hayes 2013). Hence, hypothesis 3a is fully supported. The significant effects of PU and SAT moreover support hypothesis 3b. The existence of path 1 (i.e., the direct connection between DISC and CI) was, however, not predicted by theory.

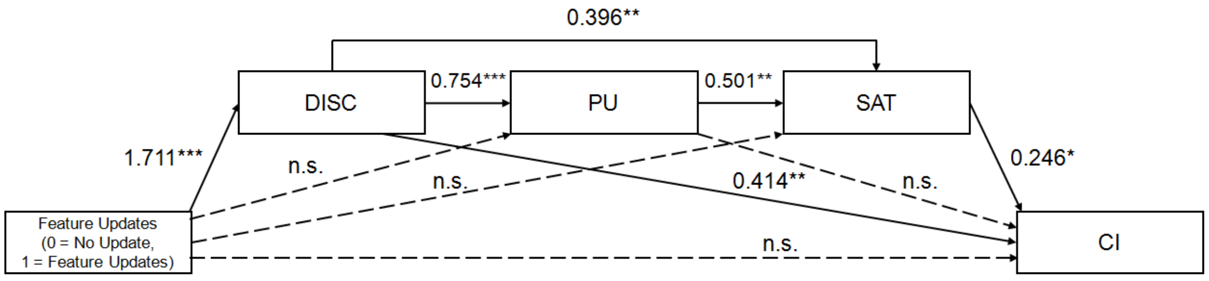


Figure 4-4: Mediation Analysis for Groups A and E

Table 4-7: Results from Serial Multiple Mediation Analysis, Groups A and E

Indirect effect paths	Effect z	Boot SE	LLCI	ULCI
(1) Feature Updates → DISC → CI	**0.709**	**0.444**	**0.053**	**1.885**
(2) Feature Updates → DISC → PU → CI	0.014	0.266	-0.451	0.657
(3) Feature Updates → DISC → SAT → CI	0.166	0.159	-0.012	0.660
(4) Feature Updates → DISC → PU → SAT → CI	**0.159**	**0.126**	**0.037**	**0.555**
(5) Feature Updates → PU → CI	0.002	0.074	-0.145	0.186
(6) Feature Updates → PU → SAT → CI	0.021	0.056	-0.053	0.217
(7) Feature Updates → SAT → CI	0.094	0.144	-0.076	0.564
Note: Bootstrapping results for indirect paths; We conducted inferential tests for the indirect effect paths based on 1.000 bootstrap samples generating 95% bias-corrected bootstrap confidence intervals (LLCI=Lower Limit/ULCI=Upper Limit of Confidence Interval), n=53.				

Finally, and complementary to the quantitative data, results from the collected qualitative data revealed that participants in group B reported the following feelings: "*I was confused and felt unsure. I did not know what to do*", "*I was confused because the update did not bring evident changes*", while participants in group D reported the following: *"[…] At first I was surprised and happy, but then every time I hoped for new features. That was very disappointing then*", "*surprised, annoyed and disturbed*". In contrast participants in group C and E felt "*pleasantly surprised*", "*happy, that now more options are available to edit the text*", and also "*confused, delighted, overstrained, satisfied*", as well as "*surprised, because of unexpectedness*". This difference in the perception of updates between the treatments is also reflected in what participants thought. While participants' predominant statements in group B were mirrored by the following statements: *"[...] Bug fixing is mostly not evident to me as a user. Therefore the question of meaningfulness rises. Was the update necessary?*" and in group D by the following: "*They interrupted my work and only security issues were fixed. No new functionality was added*". A different opinion tendency could be observed in group C and E: "*The use of new features provides better results, but requires somewhat more time*" and "*Now I can better structure the text, what will be the next update?*" These qualitative findings confirm and further illustrate the reported quantitative results regarding the positive effect of feature updates on DISC, PU and SAT and the disturbing effect of non-feature updates that fail to deliver useful functionality. Such updates seem to leave participants confused, particularly in low frequency settings. These participants' statements can be considered as representative for groups B, C, D and E respectively, as our detailed analysis of all statements has revealed.

4.6 Discussion

This study sought to achieve three main objectives: (1) to examine the effects of software updates on users' intentions to continue using an information system (i.e., *whether* there is a discernible effect from updates), (2) to investigate crucial moderators of this effect (i.e., *when* there is an effect from updates and when not), and (3) to unravel the explanatory mechanism through which such an effect occurs (i.e., *how* such an effect from updates operates). To achieve these objectives, we drew on the IS continuance model that is embedded in the expectation-confirmation theory and investigated our hypotheses based on a controlled lab experiment.

Drawing on the advantages of the experimental method, which allows to isolate the effects of manipulated stimuli on user responses from other confounding variables and thus to unveil causal relationships, we found that receiving software updates during usage can significantly alter users' intentions to continue or discontinue using an IS—a finding that complements existing post-adoption research that has previously often assumed monolithic IS which remain static over time (Bhattacherjee and Premkumar 2004; Kim and Malhotra 2005). However, our analysis also revealed that not all software updates exert this effect. Only in the feature-update conditions (groups C and E) CI was significantly higher than in the non-update condition (control group A). Non-feature updates (groups B and D) could not increase users' CI compared to the no-update condition (control group A). This significant increase in CI in groups C and E also persisted when compared to the non-feature update conditions (groups B and D), identifying update type as a distinct and crucial moderator to the effect of software updates on CI.

Receiving a helpful feature through an update was viewed by participants in groups C and E as direct benefit, enabling them to better accomplish their text formatting task. This positive response persisted despite the drawbacks which were associated with the updates. Update notifications interrupted the participants in their workflow and since they received these additional features only during use (5, 10 or 15 minutes after they had started their text-formatting task), some of the formatting work which they had done prior to the update had to be redone to apply the new features. Since they were unrelated to the text-formatting task and did not have any direct or immediate relevance, non-feature updates were not viewed as beneficial by participants.

Furthermore, our experiment also found significant differences between the two feature-update conditions (groups C and E), identifying update frequency as second crucial moderator to the effect of software updates on users' CI. Participants in group E showed significant higher scores of CI compared to group C, despite the fact that both groups received the same type and amount of features through updates. This particular finding seems counter intuitive at first. Even though participants in group E received the first additional feature 5 minutes earlier than group C, they received their third additional feature 5 minutes later than group C, eradicating any advantage from earlier access to some functionality. Participants in group E were even interrupted in their workflow more often (three times, i.e. every 5 minutes) than group C (only once, i.e. after 10 minutes) and additionally had to repeatedly cope with changes in the text-editing software (three times, i.e. every 5 minutes).

In our further analysis of the participants' positive response to feature updates, we could demonstrate that this effect was mediated by user's DISC, PU and SAT. Groups C and E seemingly perceived the feature updates as unexpected, positive events during their usage, which exerted a positive disconfirmation of their initial expectations regarding the used text-editing software. These additional features subsequently also lead to a higher perceived usefulness. This in turn increased user satisfaction and ultimately concluded this causal chain of effects by leading to higher intentions to continue using the program for future text-formatting tasks. Considering the previously discussed roles of update type and update frequency, we thus identified a moderated mediation mechanism through which updates that deliver additional features increase users' continuance intentions. Our mediation analysis confirms the explanatory power of Bhattacherjee's (2001) IS continuance model—even in complex post-adoption settings where users' beliefs and attitudes fluctuate over time alongside changes in the system characteristics of the employed IS.

4.6.1 Implications for Research

The paper makes three primary contributions to the literature. First, our overarching contribution lies in the extension of the predominant view of information systems in postadoption literature from a mostly monolithic and static one to a finer-grained and more dynamic perspective by showing how an alterable and malleable information system might influence users' attitudes and behaviors over time. In doing so, we answer several calls of IS scholars (e.g., Jasperson 2005; Benbasat and Barki 2007 etc.) to consider the granularity of information systems in research studies and how IS evolve over time. As such our study offers a novel complement to the existing IS post-adoption literature by showing that user attitudes and behaviors change, as the IT artifact's nature and composition evolves over time through software updates. Our second main contribution lies in the detection of a positive user reaction to software updates. Specifically, delivering software features to users through updates during usage can increase their intentions to continue using the information system. We investigate this effect in great detail by identifying update type and update frequency as crucial moderators. Regarding update type, our findings imply that only feature updates can exert this effect. Due to their insufficient level of usefulness for task completion, non-feature updates cannot induce a similar positive user response. Aside from update type, we found that update frequency is a crucial moderator to the identified positive effect of feature updates such that users prefer the frequent delivery of individual features over bundling them in larger update packages and delivering them less frequently. Our third contribution consists in shedding light on the explanatory mechanism behind the identified effect of software updates on CI. Specifically, we found that the positive effect of feature updates on CI involves both, the cognitive (PU) and the affective component (SAT) of the IS continuance model and originates from a positive disconfirmation of expectations (DISC). DISC, which starts this causal mediation chain, furthermore consists of two crucial components: unexpectedness and a positive experience. While unexpectedness is the necessary condition, its occurrence alone is not enough for DISC to occur (see non-feature update conditions, groups B and D). In order to initiate the mediation chain which leads to an increase in CI, software updates need to be perceived as helpful by the users (see feature update conditions, groups C and E). This makes a positive experience the second crucial component of DISC and identifies it as the sufficient condition for initiating this mediation mechanism.

4.6.2 Implications for Practice

Our results also have important implications for practice. First, despite the extensive use of software updates by vendors to maintain, alter and extend their products after they have already been rolled out, it is surprising to find that insights on how these updates are perceived and evaluated by users are still scarce. This apparently leaves practitioners puzzled and without guidance. From the results of our experimental study we can conclude that it might be advisable for vendors to distribute software functionality over time via updates, because feature updates can induce a positive state of surprise, which, in turn, increases users' CI. For vendors, users with a high CI are a particularly desirable goal because these are the loyal, returning customers who ensure the long term profitability of their businesses in the highly competitive software industry. Moreover, a high CI is particularly important for the increasing share of subscription-based business models in the software industry (Veit et al., 2014). However, while the identified positive effect of feature updates seems to be a useful measure for software vendors to keep their customers satisfied and 'on board', it also needs to be well understood and correctly applied in order to achieve the desired outcomes. Software vendors should be aware of the fact that the discussed positive effect of updates can only be achieved with feature updates. Updates must deliver actual useful functionality for users. Non-feature updates may even have the potential to diminish CI, when they are perceived as unsolicited interruptions in the workflow. Vendors should therefore have a clear understanding which updates are perceived as really useful by users and which ones not. The findings of this study also reveal that vendors should spread the delivery of features over several individual updates instead of bundling them in one larger update package that delivers them all at once. Each individual update that delights users with new functionality can induce its own unexpected, positive experience. In sum, these individual experiences seem to supersede the impact of a larger update package containing the same set of features. Finally, for vendors, our findings highlight an additional benefit from using a modular architecture for their software. Aside from flexibility in the development, a modular architecture is beneficial, because features that are encapsulated in discrete modules are technically easier to deliver as updates and can be integrated easily in existing systems that are already being used.

4.6.3 Limitations and Future Research

Four limitations of this study are noteworthy and provide avenues for future research. First, in our experiment, we utilized a self-developed, simplified word-processing program with homogeneous and functionally equivalent features and a single measurement at the end of a predefined usage time in order to reduce confounding effects and isolate the impact of updates. Nevertheless, research settings with repeated updates and participants' evaluations measured at several points in time could help to understand the identified user reactions even better. Moreover, to increase generalizability and to better resemble real-world update practices of software vendors, future studies could investigate more complex word-processing programs and specify the identified moderators (e.g., tipping points in frequency) even more precisely. They could further distinguish between different types of feature updates (e.g., common features, extraordinary features), different types of update notifications (e.g., no notification, unobtrusive notifications, obtrusive notifications), different initial feature endowments, if information about updates already plays a role in the software selection decision (e.g., before usage vs. after usage) or what effect update packages consisting of features and non-features and the specific composition of such bundles could have. Second, to avoid that an existing positive effect of feature updates on CI might remain undiscovered due to our experimental program's simplified feature set, we put participants in the hypothetical situation of a market where feature-rich and refined programs such as Microsoft Word or Open Office were not available. Although our subjects could reportedly put themselves well into this scenario, future research should replicate our findings by using a research design without a hypothetical scenario.[13] Third, the positive effect of feature updates on users' CI was shown to work for productivity software (word-processing). Future research is encouraged to show whether the same effect occurs also for hedonic (e.g., entertainment) software. Because this positive effect of feature updates occurred in software with a low affective quality (word-processing), we are confident that it might have an even stronger impact on CI for entertainment software, which is per se more emotionally charged. Finally, we conducted a controlled laboratory experiment with the purpose to make a first step towards exploring the causal effect of software updates for information systems continuance, thus presenting results with a high internal validity. Future studies are encouraged to

[13] It should, however, also be noted that in the case of this study, these simplifications with regard to task and time are not necessarily a disadvantage for the generalizability of its results: As participants showed the hypothesized positive responses to updates even in our artificial setting, they might be even more likely to show these responses in a real world usage scenario.

complement the initial findings of this study by conducting longitudinal field experiments or case studies, in order to advance the external validity of our findings.

4.6.4 Conclusion

Software updates have become a pervasively used instrument of vendors to maintain, alter and extend their products over time. However, despite their prevalence in private and business IT usage contexts, software updates' effects on user reactions in the IS post-adoption context have remained largely unexplored. This study is not only the first to demonstrate that software updates have the potential to increase users' CI; it also reveals update type and update frequency as crucial moderators. Specifically, the identified positive effect on CI can be elicited only by functional feature updates and users prefer a high update frequency. Furthermore, this study explains the underlying mechanism of why and how software updates influence users' CI. In summary, it represents an important first step towards better understanding how software updates affect user reactions over time and may therefore serve as a springboard for future studies on software updates in the context of IS post-adoption research.

Chapter 5: Updates and the Role of Delivery Strategy and Update Type

Title: Seamless Updates – How Security and Feature Update Delivery Strategies Affect Continuance Intentions with Digital Applications

Authors: Tillmann Grupp, Technische Universität Darmstadt, Germany

Schneider, David, Technische Universität Darmstadt, Germany

Published in: European Conference on Information Systems (ECIS 2017), Guimarães, Portugal.

Abstract

Although updates have become the rule rather than the exception in modern digital ecosystems, to date they have received little attention in the IS post-adoption literature. We therefore draw on the IS continuance literature and expectation-confirmation theory to investigate, how different delivery strategies of security and feature updates impact users' continuance intentions (CI). Based on an online-experiment with 282 participants, we find a positive effect of security updates on users' CI only if users are notified after successful implementation. Feature updates, in contrast, elicit a positive effect on users' CI if they are at least announced before or after successful implementation. We also find that this positive effect of ex-ante announced feature updates diminishes if users have the choice to consume the update or not. In essence, our findings contribute to IS research by extending the mostly monolithic view of information systems by showing how an alterable information system might influence users' attitudes and behaviors during use. For practitioners, we show that it seems to be beneficial to inform users about updates, even though a silent integration has become possible with modern digital ecosystems, and that updates should be applied consistently. Directions for further research are discussed.

Keywords: Feature Updates, Security Updates, Delivery Strategies, IS continuance, IS post-adoption, Expectation-Confirmation Theory

5.1 Introduction

In most modern software ecosystems, where updates have become the rule rather than the exception, providers have strived for making the update process as integrated and unobtrusive as possible. Recently, with its newest release of the multi-device platform Android, Google has even announced to introduce ‘seamless updates’ (Samat, 2016). This is an update strategy, where updates are downloaded and installed completely in the background, without affecting application usage. Software updates, in this context, are no discrete and standalone programs themselves but are rather integrated into the base software to modify, extend or alter it, once they are applied to it (e.g., Dunn, 2004). From a user’s perspective, two major update types delivering either additional functionality or security enhancements may be distinguished. Feature updates deliver additional functionality that extends the software with respect to its core purpose and are thus noticeable by users. Security updates remove potential vulnerabilities or enhance the software’s security and only indirectly and unobservable add value to the software (Ng et al., 2009). Fostering and maintaining secure behavior is a major topic in IS (Steinbart et al., 2016; Liang and Xue, 2010), which includes promoting the application of such updates. If updates are rolled out to users, developers of applications or platforms have various options to make them available to users. Updates may be applied consistently or only optional and they may be announced before or after successful implementation. In the near future, they may even be completely implemented in the background. From a software provider perspective, it thus becomes crucial to understand how their users perceive such distinct update delivery strategies.

Though updates are ubiquitously used and digital businesses heavily depend on their customers’ loyalty (i.e., continued use), there is little research on the impact of their delivery strategies on users’ beliefs, attitudes, and specifically continuance intentions regarding the updated software (Hong et al., 2011; Claussen et al., 2013). This understanding is essential to fully grasp individual behaviors in digital ecosystems (e.g., Carillo et al., 2014; Liu et al., 2016). Current research often neglects the user perspective and explores software updates mostly from a technical perspective. This includes research on software engineering (Sommerville, 2010), software product lines (Clements and Northrop, 2002), release planning (Svahnberg et al., 2010), and software maintenance (Mens and Demeyer, 2008). Updates may change the software during use and over time, and therefore may have the potential to alter users’ beliefs, attitudes, and behaviors in the post-adoption stage (Karahanna et al., 1999; Bhattacherjee, 2001). Increasing the understanding of updates and their delivery strategies from a user’s perspective has the potential to significantly increase the body of knowledge of existing post-adoption theory.

However, existing research often tends to conceptualize information systems as monolithic black boxes, rather than as a collection of functionalities and characteristics that are alterable over time (Jasperson et al., 2005; Benlian, 2015). Moreover, there are several calls for research from IS scholars who criticize the negligence of the IT artifact’s role in IS research and suggest focusing on changes in beliefs, attitudes, and behaviors emanating from the IT artifact itself rather than from other IT-unrelated environmental stimuli (Benbasat and Zmud, 2003; Hevner et al., 2004; Orlikowski and Iacono, 2001). Understanding the granularity of software, the changes triggered by updates and the effects of distinct strategies of delivering such updates to users, would help to explain how beliefs, attitudes, and behaviors may fluctuate over time because of the evolving nature of information systems that may be permanently advanced by providers. This study therefore raises the following two research questions:

RQ1: Does the delivery strategy affect an update’s impact on users’ continuance intentions?

RQ2: Do potential effects of delivery strategies differ between feature and security updates?

Drawing on the expectation-confirmation theory (Oliver, 1980), that is embedded in the IS continuance model (Bhattacherjee, 2001), we conducted an online experiment with 282 participants to answer these questions. This study thereby contributes to prior research in three important ways. First, we find somewhat different user reactions to major update delivery strategies for security and feature updates. Thereby, we identify update type and notification strategy as crucial moderators for explaining the ongoing use of agile information systems. Our second main contribution is shedding light on the effects of a non-mandatory delivery of updates on the identified effect of updates on users’ CI. The finding of a diminished positive effect in the feature update case highlights the pivotal role of ECT and its central effect on IS continuance. Our third and overarching contribution lies in the extension of the predominant view of information systems in post-adoption literature. Here we show how an alterable information system might influence users’ attitudes and behaviors during use. Software application developers and platforms may also benefit from this study’s results in practice. We find that in most cases, users should be notified of updates (for security updates only after successful implementation), even though a seamless and silent integration of updates has become possible with modern digital eco-systems. Moreover, in situations where the user is involved in accomplishing a task, software providers should avoid rolling out non-mandatory updates. Doing so may wipe out any positive effects and may leave the software in a vulnerable or inferior state.

The remainder of the paper is organized as follows. First, we review relevant literature and develop our hypotheses. We then discuss our research methodology and outline the operationalization of our study. We subsequently present empirical results of our analysis. Finally, we conclude and discuss limitations of this research.

5.2 Theoretical Foundations

5.2.1 Feature Updates and Security Updates

Consistent with previous research (e.g., Dunn, 2004), software updates can be defined as self-contained modules of software that are provided to the user for free, to modify or extend software after it has been rolled out and is already in use. With various terms, software updates have been the subject throughout software engineering literature from a technical perspective (Shirabad et al., 2001; Svahnberg et al., 2010; Weyns et al., 2011). In this context, software release planning refers to the "*idea of selecting the optimum set of features or requirements to deliver in a release within given constraints*" (Svahnberg et al., 2010, p. 1), thus falling within the strategic considerations of a service provider on how and when to deliver which software enhancements to users. In contrast to this rich stream of technical literature dealing with software updates, research on users' beliefs and attitudes regarding updates has so far been very limited (e.g., Fleischmann et al., 2016). Specifically, essential characteristics of the update's delivery process such as update notifications or consumption choices in context with different types of updates have so far not been explored.

For this study, we distinguish two basic types of software updates for which user perceptions are quite different (Dinev and Hu, 2007), namely feature and security updates. Feature updates change the core functionality of a software by adding distinct features that are deliberately utilized by users to accomplish the task for which the software is used. In contrast, security updates, falling in the broader category of non-feature updates, do not change the core functionality of software and cannot be directly observed by users, but enhance the software's protective powers or close vulnerabilities (Ng et al., 2009). Because the user's interaction with the software may change when the software's perceived value changes, updates have the potential to influence users' beliefs, attitudes, and behaviors in the post-adoption stage of IS usage. This may even affect users' decisions on continued use.

5.2.2 Information Systems Continuance

In the context of post-adoption research (Karahanna et al., 1999; Bhattacherjee, 2001), the term information systems continuance refers to the "*sustained use of an IT by individual users over the long-term after their initial acceptance*" (Bhattacherjee and Barfar, 2011, p. 2). Bhattacherjee (2001) has adopted the expectation-confirmation theory (ECT) (Locke, 1976; Oliver, 1980, 1993; Anderson and Sullivan, 1993) to explore IS users' intentions to continue or discontinue using an IS. ECT posits, that customers compare their initial expectations with perceived product performance. The discrepancy determines their level of satisfaction. The level of satisfaction further impacts repurchase intention (Oliver, 1980, 1993). Bhattacherjee (2001) has replaced repurchase intention of the ECT model by users' intention to continue using an IS (CI), suggesting that users compare pre-usage expectations with their experience during IS usage. If perceived performance exceeds (falls short) initial expectations, users experience positive (negative) disconfirmation (DISC), which has a positive impact on their satisfaction (SAT) regarding the IS (Bhattacherjee and Barfar, 2011). Satisfied users intend to continue using the IS, while dissatisfied users discontinue its subsequent use (Oliver, 1980; Bhattacherjee, 2001). Perceived usefulness (PU) captures the expectations about future benefits from IS usage (Bhattacherjee and Barfar, 2011) and has a positive impact on both SAT and on CI (Bhattacherjee, 2001).

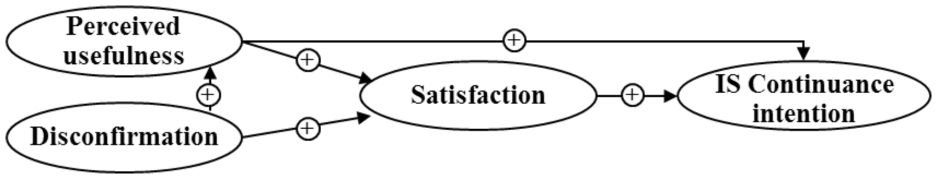


Figure 5-1: IS Continuance Model (Following Bhattacherjee, 2001)

While the IS continuance model has made valuable contributions to post-adoption research (Bhattacherjee, 2001) it has a static perspective on the IS continuance setting, failing to account for changing user beliefs and attitudes during use. In response to this limitation, several authors have introduced a more dynamic perspective, showing that beliefs and attitudes change from pre-usage to actual usage and during the ongoing usage of an IS (Bhattacherjee and Premkumar, 2004; Kim and Malhotra, 2005; Kim and Son, 2009; Ortiz de Guinea and Markus, 2009; Ortiz de Guinea and Webster, 2013). To investigate this changing nature of the IT artifact and its impact on users' beliefs, attitudes, and behaviors during post-adoption use, we therefore explore software updates and their delivery strategies through the lens of the disconfirmation mechanism in ECT and the IS continuance model.

5.3 Hypotheses Development

In the following section, we will develop our hypotheses on how different update types and delivery strategies in software ecosystems might influence users' post-adoption beliefs and attitudes in non-mandatory or individual use settings. To isolate the core effects, we will focus on a seamless update experience, setting aside notable downsides like download and installation delays. In doing so, we limit ourselves to feasible delivery strategies in modern digital platforms, with either a 'silent' update or a notification given either before or after the update is run. Moreover, we distinguish between the most prevalent and important update types from a user's perspective, those that provide either additional functionality or security enhancements, setting aside minor stability fixes. Finally, to complete our hypotheses, we will posit whether an option to consume an update should be given to the user or not.

5.3.1 Effects of Notifications for Security Updates

We argue that receiving software updates during post-adoption use can induce positive disconfirmation and increase users' CI (Bhattacherjee, 2001; Hong et al., 2011). According to ECT, the occurrence of positive disconfirmation requires a positive experience compared to prior expectations, i.e. a relative improvement compared to a baseline (Helson, 1964; Oliver, 1980). In the context of software updates, this baseline is formed by the software's pre-update state. An update must therefore exceed this subjective reference point to increase users' CI by leading to a *perceived improvement* of the software (Hong et al., 2011).

Following research on IT security, it is reasonable to assume that users' awareness of the 'protective enhancements' provided by an update plays a major role (i.e. the user has to be aware that something has changed in his favor to feel positively about it). Security updates manifest themselves quite differently than updates providing additional functionality (Ng et al., 2009). They only "*contribute to the wellbeing of their users indirectly and subtly*" (Dinev and Hu, 2007, p. 387). The benefits resulting from security updates cannot be observed directly within the software, as such updates do not add any usable features. Consequently, their value may be derived only from the information provided on the update's intent. A notification of added benefits may therefore substitute the users' experience of an actual change in the software, that the user may otherwise not be aware of (Darby and Karni, 1973).

Such information about an update's intent may be provided to users through notifications either before or after the update is successfully implemented. A notification *before* the execution of the update, however, leaves the user in considerable doubt, as to whether or not the security update was indeed successfully applied (Hoxmeier, 2000; Hong et al., 2011). There is no actual confirming experience on the software's enhancement. Therefore, due to the absent information on successful completion, it will most likely not be perceived as an actual improvement, failing to induce positive disconfirmation and to increase users' CI. In contrast, a notification *after* the successful application of an update clearly conveys the message that the update was completed successfully and that the software therefore, compared to its status quo, may indeed have improved (Hoch and Ha, 1986). Therefore, it is likely that a security update, if announced after successful implementation, will be *perceived* as an *improvement* during use, inducing positive disconfirmation in the sense of ECT (Oliver, 1980). Through an increase in SAT and PU it will thereby increase users' CI eventually (Bhattacherjee, 2001; Hong et al. 2011). We accordingly derive our first two hypotheses:

H 1.1: Users who receive a notification before the implementation of a security update will exhibit similar continuance intentions compared to users who did not receive the security update.

H 1.2: Users who receive a notification after the successful implementation of a security update will exhibit higher continuance intentions compared to users who did not receive the security update.

5.3.2 Effects of Notifications for Feature Updates

Moreover, we argue that ECT also applies to the potential effects of feature updates. As reasoned above, to induce positive disconfirmation and to increase CI, an update must lead to a perceived *improvement* of the software. Feature updates can directly contribute to the productivity of the user, and thus elicit a positive experience compared to the software's un-updated state (Hong et al. 2011). However, although feature updates deliver such functionality that directly improves the software with respect to its core purpose, users are often unaware of newly delivered functionality available in the software (Alba and Hutchinson, 1987; Brucks, 1985; Jasperson et al., 2005; Sun, 2012; Benlian, 2015). The user's capacity of attention is limited, and the user's main task and other interferences will compete for the user's cognitive processing capacity necessary to perceive all available functionality (Kahnemann, 1973; Norman and Bobrow, 1975; Van der Heijden, 1992). Hence, the functionality gains through feature updates may remain unnoticed, if not explicitly presented to users (Sun, 2012).

However, again, the newly available functionality can be made more apparent to users by providing notifications, either before or after the update's successful implementation. In the case of a feature update, though, we posit that the announcement of additional functionality before the update can be confirmed by actual experiences of the specific software enhancements afterwards (Hoxmeier, 2000; Hong et al., 2011). Therefore, notifications before the implementation do not leave users in doubt about the update's success, and thus also have the potential to facilitate a positive experience deriving from the additional new functionality. Summing up, feature updates that are not explicitly announced, may not be recognized by users and therefore may fail to induce positive disconfirmation and eventually increase users' CI. In contrast, feature updates that *are* announced either *before or after* successful implementation will be *perceived as improvements* during use, inducing positive disconfirmation in the sense of ECT (Oliver, 1980). Thereby, in this case, through an increase in SAT and PU, users' CI will increase eventually (Bhattacherjee, 2001; Hong et al. 2011). Accordingly, we derive the following three hypotheses:

H 2.1: Users who receive additional functionality through a feature update without notification will exhibit similar continuance intentions compared to users who did not receive the feature update.

H 2.2 Users who receive a notification before the implementation of a feature update will exhibit higher continuance intentions compared to users who did not receive the feature update.

H 2.3 Users who receive a notification after the successful implementation of a feature update will exhibit higher continuance intentions compared to users who did not receive the feature update.

5.3.3 Effects of Non-Mandatory Security and Feature Updates

From conventional practice, one could think that in addition to a pre-update notification, it might be beneficial to provide the option to users on whether to consume an update or not, because doing so would offer more control over the process (Iyengar and Lepper, 2000; Scheibehenne et al., 2010). However, we argue that such a strategy will most likely foster a different result in our case. When users are engaged in using the software to complete a task (Jenkins et al., 2016), the update seems to provide appropriate benefits and, due to our assumption of seamless integration, the update comes with no or only very few downsides, the option to consume an update increases necessary efforts and weakens the potential positive perception of benefits received from an update (Iyengar and Lepper, 2000; Jenkins et al., 2016). Not enhancing the software in the first place, but questioning the update's necessity may leave users in doubt of the update's advantages. As a result, the choice to update may be deferred and the update may therefore fail to exceed prior expectations, as compared to situations where the update is always applied (Jenkins et al., 2016). An obligatory choice can thereby make an update fail to elicit positive disconfirmation through the mechanisms of ECT (Oliver, 1980) and fail to increase users' CI as outlined in our hypothesizing above (Bhattacherjee, 2001; Hong et al. 2011). Summing up, we argue that providing a consumption choice for an update will impair a *perceived improvement* resulting from the functionality gains in cases where the update would otherwise increase users' CI (as argued in hypotheses H2.2). By weakening the update's necessity, a choice to either consume or to dismiss an update will diminish users' potential positive experiences emanating from the update's content. However, in cases where the update does not elicit positive disconfirmation (as argued in our hypotheses H1.2), providing a choice does not harm users' CI. We therefore hypothesize:

H 3.1 Users who have the choice to optionally consume a security update before its conditional implementation will exhibit similar continuance intentions compared to users who consistently receive the update with a notification given beforehand.

H 3.2 Users who have the choice to optionally consume a feature update before its conditional implementation will exhibit lower continuance intentions compared to users who consistently receive the update with a notification given beforehand.

5.4 Method

5.4.1 Experimental Design

With the goal to examine the effects of security and feature updates and their delivery strategies on users' CI, we conducted a 2 x 4 between-subjects online-experiment with manipulations of update type (security update vs. feature update) and delivery strategy (no update notification vs. post-update notification vs. pre-update notification vs. pre-update notification and update consumption choice). The design may also be considered as a combined 2 x 3 (update type vs. notification and timing) and 2 x 2 (update type vs. choice) experiment. We carefully developed this design, because an update consumption choice can only be provided by simultaneously notifying users about the upcoming update. However, the chosen design allowed us to both separate the effects of the two factors and to subsequently put them into relation. We opted for an online experiment because it allowed us to investigate and clearly isolate the causal mechanisms that operate between delivery strategies, update types and changes in user attitudes, beliefs, and intentions. We consider this as crucial given that this study is one of the first to explore the effect of different update delivery strategies on users' CI. It also enabled us to account for the claims of numerous researchers to put the IT artifact more at the center of investigation of post-adoption research by using actual changes in an IS as basis for manipulations. The software and the task for which the software had to be used were held constant across all conditions.

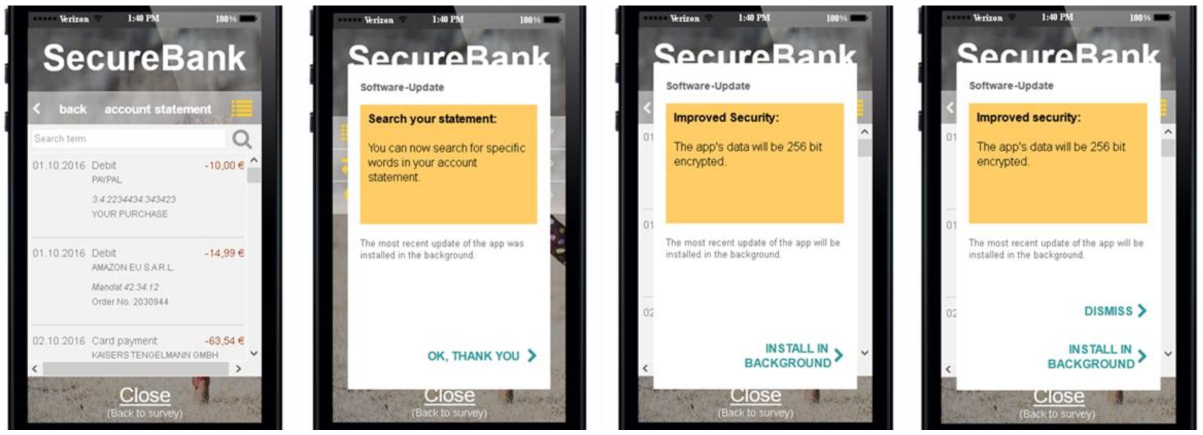
The experiment proceeded in four major steps: First, subjects were randomly assigned to one of the eight groups. Second, subjects were instructed to make use of a banking app to check for an outstanding bank transfer (i.e., our cover story) and were then transferred to a fully functional click dummy of a banking app. The app provided an account statement that listed several realistic but random payments, but did not contain the transfer in question. Third, subjects were told, that, on the next day, they would reuse the app to once more check for the outstanding transfer and were then forwarded to the banking app again. In this second usage period the transfer in question was contained towards the end of the list (to equally engage the user in the app). According to the experimental group (see Table 5-1), for a security update, the app was kept constant in both usage periods (because the security update does not manifest in the user interface) (Group A), only in the second period a 'successfully updated' notification was given (Group B), in the first period an update announcement was given (Group C), and in the first period an update announcement including the option to either dismiss or to install the update in the background was given (Group D). For a feature update, in the second period a feature was added (Group E), in the second period a feature was added including a 'successfully updated' notification (Group F), in the first period and update announcement was given and then a feature was added in the second period (Group G), and in the first period an update announcement including the option to either dismiss or to install the update in the background was given and then, according to the user's choice, a feature was conditionally added to the app in the second period (Group H). Subsequently, after the two usage periods of the banking app, a post-experimental survey was conducted to assess the subjects' CI with respect to the software and all further variables (see Measures).

Table 5-1: Experimental design and experimental groups (N: notification, F: feature added)

Update type:	Security update			Feature update		
Delivery strategy:		*Usage period 1*	*Usage period 2*		*Usage period 1*	*Usage period 2*
No notification	**A**	-	-	**E**	-	F
Post-notification	**B**	-	N	**F**	-	N, F
Pre-notification	**C**	N	-	**G**	N	F
Pre-not. and choice	**D**	N (choice)	-	**H**	N (choice)	F (conditionally)

5.4.2 Manipulation of Independent Variables

To realize our manipulations, we opted for the software context of a banking app running on a mobile application platform to ensure that subjects had previous usage experience and that both security and feature updates would provide relevant value. By choosing a mobile software ecosystem, we could realistically mimic the forthcoming behavior of such platforms (Samat, 2016) and separate the effects of receiving updates from interfering factors like performance or technical issues (Sykes, 2011; Tyre and Orlikowski, 1994). While such downsides have been traditionally associated with updates, however, we argue that modern platforms integrate software updates increasingly frictionless and we are thus confident that we can develop viable implications for many contemporary software ecosystems.



01.10.2016	Debit PAYPAL *3.4.2234434.343423* YOUR PURCHASE		-10,00 €
01.10.2016	Debit AMAZON EU S.A.R.L. *Mandat 42.34.12* Order No. 2030944		-14,99 €
02.10.2016	Card payment KAISERS TENGELMANN GMBH		-63,54 €

Search your statement:

You can now search for specific words in your account statement.

The most recent update of the app was installed in the background.

Improved Security:

The app's data will be 256 bit encrypted.

The most recent update of the app will be installed in the background.

Improved security:

The app's data will be 256 bit encrypted.

The most recent update of the app will be installed in the background.

Figure 5-2: Sample screens of app with no, post-, pre-notification, and additional choice (l.t.r.)

Manipulations of the update type were realized as follows: for the subject of the security and feature update, we first asked 49 participants to rate a list of distinct features of banking apps on perceived importance, which we had compiled through interviews and desk research. Given these insights, we subsequently established the feature 'search account statement' and the security enhancement '256 Bit encryption' as subjects for the corresponding feature and security updates. Because enhancements of security do not directly manifest in the user interface other than by a conditional notification (the experimental group without notification may thereby serve as a control group), only the feature update would also actually add a distinct functionality to the software by providing a search slot above the account statement. Manipulations of the delivery strategy (i.e., notification and conditional choice) were implemented by (1) providing no notification, (2) providing a confirmation layer that describes the successfully installed update and its content after the user revisits the app, (3) providing an announcement layer that describes the pending update and its content when the user first visits the app (the layer comes up with several seconds delay), and (4) providing the aforementioned layer and additionally giving the option to either accept or to defer the update's installation (see Figure 5-2).

A qualitative pilot test with five subjects was conducted to ensure that the treatments were manipulated according to the experimental design, that participants would assess the setting as realistic, and that they would understand it well (Perdue and Summers, 1986). Specifically, subjects were asked about the comprehensiveness of the instructions, the effects of the manipulations through the app and the questions in the following questionnaire. In an additional pre-study (n=48), we confirmed the successful manipulation based on measures of control questions. Suggestions were obtained from the participants and the app and the questionnaire were accordingly revised for the main experiment.

5.4.3 Dependent variables, Control Variables and Manipulation Checks

We used validated scales with minor wording changes for all constructs. Measures for CI and DISC were adapted from Bhattacherjee (2001): CI1. *I intend to continue using the app rather than discontinue its use;* CI2. *My intentions are to continue using the app than use any alternative means (traditional banking);* CI3. *If I could, I would like to discontinue my use of the app* (reverse coded). DISC1. *My experience with using the app was better than what I expected;* DISC2. *The functionality provided by the app was better than what I expected;* DISC3. *Overall, most of my expectations from using the app were confirmed.* Measures for PU and SAT were based on Kim and Son (2009): PU1. *Using the app enhanced my effectiveness in completing the task;* PU2. *Using the app enhanced my productivity in completing the task;* PU3. *Using the app improved my performance in completing the task.* SAT1. *I am content with the features provided by app;* SAT2. *I am satisfied with the features provided by the app;* SAT3. *What I get from using the app meets what I expect for this type of programs.* Because constructs were measured with multiple items, summated scales based on the average scores of the multi-items were used in group comparisons (Zhu et al., 2012). Unless stated otherwise, the questionnaire items were measured on a seven-point-Likert-scale anchored at (1)=strongly disagree and (7)=strongly agree. To ensure successful manipulations we captured whether participants thought that *they had received an update*, what the *subject of the update* was, and if they had been *notified before or after*. Also, in groups with non-mandatory updates, we measure actual *confirmations* and *dismissals* of updates and the participants' *intentions* to *install* or to *not install* such an update. Participants were further asked to what extent they had understood *the items' formulation*, whether they were *able to put themselves in the given situation*, if *the scenario was realistic*, and if they knew *what the goals of the survey were*. The participants' *expertise regarding online banking* was captured on an established four item scale developed by Mishra et al. (1993). We included this control variable as well as the participant's *online banking usage intensity, perceived common update frequency*, and finally the subjects' *demographics* (age, gender, profession), to isolate the effects from other possible covariates.

5.4.4 Participants, Incentives and Procedures

Participants were recruited over Clickworker, a German crowdsourcing platform similar to Amazon Mechanical Turk (Paolacci et al., 2010). We offered a small payment for the participation in our online experiment. Overall, 312 subjects started the experiment. The rate of completion was 96%, i.e., a total number of 301 subjects completed the questionnaire. We excluded 19 participants from our final analysis because they did not pass our quality check questions. The average time needed for the completion of the experiment and questionnaire was 9.10 minutes. Of the 282 remaining German speaking participants used in the following analyses, 142 were females and 140 were males. Subjects' average age was 36.42 (σ=11.28) years. On average, in one month, 14% of the subjects use online banking up to one time, 20% up to four times, 21% up to eight times, and 45% more than eight times. The average reported expertise with online banking was 5.42 (σ=1.36) on a seven-point semantic differential scale. More than 40% of the subjects were employees, 21% self-employed, 13% students, and the remainder had various or no occupation. The educational backgrounds of the participants were diverse, including psychology, law, educational sciences, chemistry, computer science, economics, design, agriculture and marketing.

5.5 Data Analysis and Results

5.5.1 Control Variables and Manipulation Check

To confirm a successful randomization, we first searched for differences of the control variables between groups. However, the results of a one-way MANOVA showed no significant differences between groups (λ=0.83, F[49,1365]=1.03, p>0.05). Neither of the control variables were significant: age (F=1.42, df=7, p>0.05), gender (F=0.61, df=7, p>0.05), profession (F=0.87, df=7, p>0.05), usage intensity (F=1.62, df=7, p>0.05), update frequency (F=1.26, df=7, p>0.05), and product expertise (F=1.39, df=7, p>0.05). Hence, we concluded that participants' demographics and relevant controls were homogeneous across conditions and did not confound the effects of our manipulations. Finally, we confirmed successful manipulations by performing a Fisher's exact test finding significant differences between conditions in terms of the reported software delivery design type (p<0.01) and the reported subject of the update (p<0.01). As indicators for the external validity of our findings, we further reviewed participants' answers regarding the realism and adaption of the scenario. For both measures, participants reported high levels on a seven-point-Likert-scale (realism $\overline{x}$=6.32; σ=1.09; adaption $\overline{x}$=6.35; σ=1.04). It is therefore reasonable to assume that our manipulations worked as intended, that participants acted typically, and that the setting was realistic.

5.5.2 Measurement Validation

Because we adopted established constructs for our measurement, a confirmatory factor analysis (CFA) was conducted to test the instruments' convergent and discriminant validity (Levine, 2005), using SmartPLS 2 (Chin et al., 2003; Ringle et al., 2005). Table 5-2 reports the results for the core constructs.

Table 5-2: Results of confirmatory factor analysis for core variables

Latent construct	Number of indicators	Range: standardized factor loadings*	Cronbach's alpha	Comp. reliability (ρc)	Avg. variance extracted (AVE)
Disconfirmation (DISC)	3	0.869-0.922	0.889	0.931	0.818
Perceived Usefulness (PU)	3	0.930-0.953	0.939	0.961	0.892
Satisfaction (SAT)	3	0.934-0.966	0.948	0.967	0.906
Continuance Intention (CI)	3	0.783-0.936	0.847	0.908	0.768
Note: *All factor loadings are significant at least at the p<0.01 level					

All items loaded on the target factors and scored above the threshold of 0.7, indicating proper construct validity (Cook and Campbell, 1979; Bartholomew et al., 2008). AVE values for each construct ranged from 0.768 to 0.906, exceeding the variance due to error (0.5). The constructs were also assessed for reliability using Cronbach's alpha (Cronbach, 1951). A value of at least 0.7 is suggested to indicate adequate reliability which we could confirm for all constructs (Nunnally et al., 1994). Furthermore, the composite reliability of all constructs exceeded 0.7, which is considered the minimum threshold (Hair et al., 2011). Thus, all constructs met the norms for convergent validity. For satisfactory discriminant validity, the square root of the constructs' AVE should be greater than the variance shared between the constructs in the model (Fornell and Larcker, 1981). All square roots of AVE exceeded inter-construct correlations, indicating proper discriminant validity. Hence, the constructs in our study are theoretically and empirically distinguishable.

5.5.3 Hypotheses Testing

In order to test our hypotheses, we conducted a one-way ANOVA with planned contrast analyses. We found significant differences between groups for DISC (F=4.023, p<0.01), PU (F=3.349, p<0.01), SAT (F=2.959, p<0.01), and CI (F=2.511, p<0.05). Figure 5-3 summarizes the results for our main dependent variable CI for both security and feature updates.

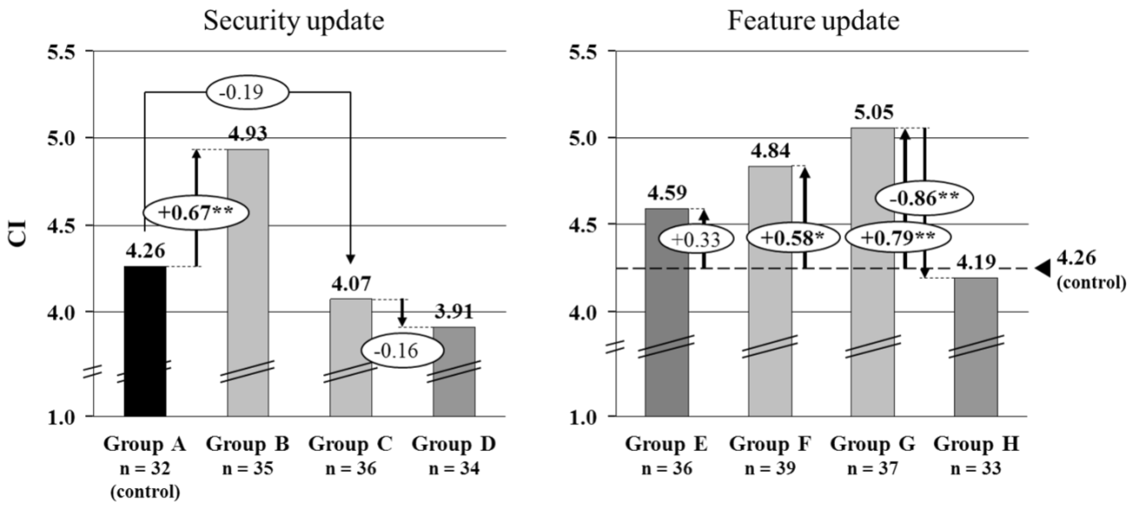


Figure 5-3: Mean values, differences and significance levels for CI between groups

Regarding security updates, the contrast analysis revealed that users who received a pre-update notification for the security update showed indifferent reactions in terms of DISC, PU, SAT, and CI compared to users who did not receive any notification (CI: $\bar{x}$'s = 4.07 vs. 4.26, p>0.1) (see Table 5-3). This supports our hypothesis 1.1. However, users who received a post-update notification on a security update exhibited significantly higher DISC, PU, SAT, and CI compared to users who did not receive any notification, supporting our hypothesis 1.2 (CI: $\bar{x}$'s = 4.93 vs. 4.26, p<0.05).

Table 5-3: Mean values, differences and significance levels for security update groups

	Security update delivery strategy (n)	DISC	PU	SAT	CI
Groups	A. No notification / control (32)	4.59	5.00	4.88	4.26
	B. Post-notification (35)	5.19	5.63	5.48	4.93
	C. Pre-notification (36)	4.48	4.61	4.78	4.07
	D. Pre-notification and choice (34)	4.59	4.91	5.05	3.91
Diff.	B-A.[14]	**0.60****	**0.63****	**0.60****	**0.67****
	C-A.[14]	-0.11	-0.39	-0.10	-0.19
	D-C.[14]	0.11	0.30	0.27	-0.16

Investigating feature updates, the contrast analysis revealed that users who received the update but were not notified at all showed *indifferent* reactions in terms of DISC, PU, SAT, and CI compared to users who did not receive any update (CI: $\bar{x}$'s = 4.59 vs. 4.26, p>0.1) (see Table 5-4). This supports our hypothesis 2.1. However, users who received a pre-update notification on the feature update exhibited significantly *higher* DISC, PU, SAT, and CI compared to users who did not receive any update, supporting our hypothesis 2.2 (CI: $\bar{x}$'s = 5.05 vs. 4.26, p<0.05). Likewise, users provided with a post-update notification exhibited significantly *higher* DISC, PU, SAT, and CI compared to users who did not receive any update, which supports our hypothesis 2.3 (CI: $\bar{x}$'s = 4.84 vs. 4.26, p<0.1).

[14] ANOVA-tests with planned contrast analyses; *** p<0.01, ** p<0.05, * p<0.1 (one-sided).

Table 5-4: Mean values, differences and significance levels for feature update groups

	Feature update delivery strategy (n)	DISC	PU	SAT	CI
Groups	E. No notification (36)	4.84	4.78	5.05	4.59
	F. Post-notification (39)	5.61	5.50	5.74	4.84
	G. Pre-notification (37)	5.62	5.68	5.79	5.05
	H. Pre-notification and choice (33)	4.65	4.91	4.91	4.19
Diff.	E-A.[14]	0.25	-0.22	0.17	0.33
	F-A.[14]	**1.01*****	**0.50***	**0.89*****	**0.58***
	G-A.[14]	**1.03*****	**0.68****	**0.92*****	**0.79****
	H-G.[14]	**-0.98*****	**-0.78*****	**-0.88*****	**-0.86****

Finally, regarding non-mandatory updates, the results of the contrast analysis revealed that for security update there was *no significant difference* in terms of DISC, PU, SAT, and CI (Group D-C) between users who had the choice to either consume the security update or not, compared to users who received the security update in any case (CI: $\bar{x}$'s = 3.91 vs. 4.07, $p>0.1$). This supports our hypothesis H 3.1. On the contrary, users who had the choice to either consume the feature update or not (Group H-G) exhibited significantly *lower* DISC, PU, SAT, and CI, compared to users who received the feature update in any case, as predicted by our hypothesis H 3.2. Further inspecting the actual decisions of update installations, based on a chi-square test, we could not find a significant difference between security updates (Confirmed vs. dismissed: 11 vs. 23) and feature updates (Confirmed vs. dismissed: 10 vs. 23) (χ^2=0.033, $p>0.1$). However, in the reported intentions to dismiss or confirm such an update, we could find a difference between security updates (Confirmed vs. dismissed: 65 vs. 1) and feature updates (Confirmed vs. dismissed: 52 vs. 14) (χ^2=12.711, $p<0.001$).

5.6 Discussion

This study sought to achieve two main objectives: (1) to examine the effects of different software update delivery strategies on users' continuance intentions, and (2) to investigate potential distinctions between the natures of security and feature updates. To achieve these two objectives, we drew on the IS continuance model and we investigated our hypotheses based on an online-experiment with 282 participants in the context of a banking app, operated on a mobile platform.

Our results reveal that users who receive a security update show divergent reactions to being notified of the update before or after its successful implementation. In the case of a post-update notification (Group B), users showed a significantly higher CI. This finding strengthens the notion that for security updates a notification on the update's successful implementation may serve as a proxy for its actual realization (which is not observable from a user's perspective). However, in the case of an ex-ante notification (Group C), no significant change in CI could be observed. Given our first finding, this may seem somewhat counter-intuitive at first. However, it may be explained by the fact that the results of a security update are not physically observable in the software (Ng et al., 2009). Thus, users are being left in vagueness about the update's actual implementation. Regarding feature updates, users receiving additional new functionality without further notification (Group E), did not show a significant increase in CI, despite this increased value provided by the software. This somewhat unexpected result may be explained by the users' attention bound to the task users had to accomplish (Kahnemann, 1973), leaving the additional functionality unnoticed. Only in both cases when the feature update was announced before or after successful implementation (Group F and G), we found a significant increase in users' CI. In those cases, the noticeable 'gift' of additional functionality was then able to elicit positive disconfirmation, thereby increasing users' CI.

In addition, we could evidence that updates that are delivered with a non-mandatory strategy do not increase users' CI. In case of a security update (Group D), providing the update to users as an optional alternative did not increase users' CI, compared to the ex-ante announcement and a mandatory rollout. In case of a feature update (Group H), a consumption choice even perhaps significantly decreased users' CI compared to an ex-ante announcement and a mandatory installation. Probably by questioning the necessity of an update and thereby preventing the consumption in many cases, such an option inhibited a potential positive experience. Inspecting the numbers of actual confirmations and dismissals for both update types, surprisingly, we could observe that they were more often dismissed than consumed with a rate that did not differ significantly between the two types. On the contrary, the intention of users to install security updates was significantly higher than for feature updates, which stresses the users' perception of importance of security updates. This finding again highlights a gap between intentions and actual behavior and thereby provides avenues for further research (Jenkins et al., 2016).

5.6.1 Implications for Research

The paper makes three main contributions to the literature. First, we identify update type and delivery strategy as crucial moderators for the positive effect of an update on users' CI. We find that providing a security update increases users' CI by disconfirming previous expectations only if it is announced after successful implementation. A feature update, on the other hand, induces a positive reaction in all situations in which it is announced in addition to its rollout (i.e., before or after implementation), while it does not have such a potential if it is silently implemented in the background. This interaction emphasizes the importance of a joint consideration of the IT artifacts' and the update's characteristics when investigating user behavior. Our second main contribution is shedding light on the effects of a non-mandatory update on the identified effect of updates on users' CI. Specifically, we find that a positive effect of feature updates on CI, by positively disconfirming previous expectations, is diminished when the update is provided only optionally. Nevertheless, CI remains unaffected for security updates in this case. These findings once again highlight the pivotal role of ECT and its central effect on IS continuance compared to other factors. Both findings add to the body of knowledge on software updates. Our third and overarching contribution lies in showing how a malleable information system might influence users' attitudes and behaviors during post-adoption use. We answer the calls of several IS researchers by extending the still predominant view of post-adoption literature on the IT artifact as a monolithic block to a more flexible perspective that considers information systems as a modular composition of functionality that may change over time (Jasperson et al., 2005; Benbasat and Barki, 2007; etc.). We complement existing IS post-adoption literature and research on digital ecosystems (Carillo et al., 2014; Liu et al., 2016) through the notion that users' beliefs and attitudes might change with the advancement of the system.

5.6.2 Implications for Practice

Our results have important and viable implications for practice, particularly for contemporary software ecosystem settings, where updates are integrated increasingly frictionless. First, despite the extensive use of updates by organizations to enhance and progress their services on digital platforms, it is surprising that insights on how these updates and their delivery are perceived by users are still scarce. This leaves practitioners without guidance. From the results of our experimental study we can conclude that developers of applications and platforms should rather announce feature and security enhancements instead of implementing them silently. However, for security enhancements, the only helpful measure for developers in terms of the user's loyalty (i.e., CI) is to announce such updates only after the successful implementation. More specifically, our findings suggest, that only in cases when the user is notified after successful implementation of a security update, it has the potential to increase users' CI above and beyond a level generated by software where the security update was communicated before implementation or not communicated at all. With respect to feature updates, developers can learn from this study’s results that they can increase their users’ loyalty by announcing them before or after successful implementation. Both strategies should be preferred over not at all communicating such enhancements, as updates won’t be always noticed by users in the software itself.

Finally, it is not advisable for developers of applications and platforms to provide users the option to either consume or to defer an update. It is better to apply updates consistently. Providing such an option may not only diminish an update’s positive effect, but may leave the software in an inferior state. In today’s interconnected and quickly changing multi-device and multi-platform environments, users heavily rely on security and on a comparable feature set with respect to competitors’ solutions. To avoid losing customers from vulnerabilities or major disadvantages (even if only temporary), platform and application providers should thus quickly respond to such needs and roll out according changes consistently. It should be noted, however, that these findings only apply to situations where the update process does not come with major downsides and the update’s contents are unquestionably helpful.

5.6.3 Conclusion, Limitations, and Future Research

In modern digital ecosystems, software updates have become a pervasively used instrument for businesses to enhance their digital services over time. Despite this prevalence, the effects of update delivery strategies on crucial post-adoption user reactions have remained largely unexplored. This study’s diverse findings highlight the importance of a profound understanding of update delivery strategies in evolving software ecosystems for both researchers and practitioners. Security updates have the potential to increase users’ CI only if they are communicated after implementation, while feature updates have such a potential if at least communicated at any time. Providing an option to defer an update however seems to be unfavorable or even harmful, as it may diminish any positive effects elicited by an update, and in the end, because users tend to dismiss them considerably, may leave the application in an inferior or even vulnerable state.

Three limitations of this study are noteworthy and provide avenues for future research. First, in our experiment, we utilized a self-developed, simplified click dummy of a banking app with a homogeneous feature set. This quasi-realistic setting of a digital ecosystem’s software required subjects to adapt to the software and setting. Hence, we controlled for adaption and perceived realism of the scenario. Based on the convincing results for these controls, we are confident that our study’s implications are applicable to real usage settings. Nonetheless, future studies could investigate actual usage experiences with real software to validate our findings. Second, we identified security and feature updates in the banking context as crucial update types for examining the effects of update delivery strategies on users' CI. Also, subjects were recruited in Germany. Since security plays a major role in the banking context and attitudes towards security might differ between countries, future studies are encouraged to validate our findings in different contexts and cultural settings. Furthermore, complementary qualitative studies (e.g., thought-listing) could substantiate our theoretical reasoning and could uncover additional mediating mechanisms. Finally, we conducted a controlled experiment with the purpose of obtaining results with a high internal validity. This required some reasonable but strict assumptions, such as a limited observation period, an identical and linear course of events, a determined task and ex-post measurement of variables. Future studies are encouraged to complement our findings by conducting longitudinal field experiments to advance the external validity of our findings over longer timespans and to account for learning effects. Also, settings with repeated updates with participants' evaluations measured at several points in time could provide additional evidence for the robustness of our findings. In the further course, research should seek to deepen the understanding of how dynamic software ecosystems need to be shaped to both satisfy and protect users by considering individual behaviors.

Chapter 6: Thesis Conclusion and Contributions

This thesis was motivated by three primary concerns regarding the current understanding of users and their relationships to IT Artifacts and Agile IS. First, despite the prevalence of Agile IS in today's software landscapes the understanding of user perspectives of Agile IS is still very limited. Second, despite the recent shift in focus to the user, little is known on how updates that deliver changes in Agile IS to users may affect users' loyalty. Lastly, the current predominant scholarly view of IT Artifacts is still that of monolithic and static systems, which is not reflective of the prevailingly dynamic and malleable nature of most IS contexts today. Conversely, considering the rapid advances in information technology and the barriers to users switching between IT systems that are diminished by the widespread diffusion of digital technologies, understanding users' loyalty to Agile IS has become increasingly important. Surprisingly, only few studies of the subject have been conducted is this field of research (e.g., Hong et al. 2011), leaving many central questions unanswered. Against this backdrop, this thesis sought to answer two overarching questions: First, whether Agile IS can positively affect users' loyalty and how a potential effect occurs. Secondly, which factors moderate a potential effect, and as a consequence how Agile IS can be designed to foster positive outcomes. To address these questions and to obtain empirical answers, four major studies were conducted, each investigating distinct subjects of the questions to refine our knowledge and isolate any relevant factors.

The results of the first and the second study demonstrated that in terms of an increase in continuance intentions users generally prefer Agile IS over monolithic software, which is somewhat irrational. Users appear to prefer software with a limited feature set that delivers additional features incrementally than having feature-complete software right from the outset. However, this effect only exists if users are not too knowledgeable regarding the software; experts devalue software that receives features later on (compared to feature-complete software). In addition we could demonstrate that the loss of features decreases continuance intentions. Surprisingly, the absolute magnitude of the effect when losing a feature exceeded the increase related to a gain of exactly the same feature. Also we found that the magnitude of the positive effect of additional functionality provided by an update diminishes, if the software is already feature rich. The update size however, does not appear to change the magnitude of the effect significantly. In the first, second and third study we could further confirm that the effect of software updates on user's continuance intentions is mediated by a mechanism of disconfirmation, perceived usefulness, and satisfaction as hypothesized before. These findings address the first research question.

The results of the third and fourth study further revealed that the frequency of updates moderates the positive effect of feature updates on users. In particular, more frequent updates will increase the positive effect. This should be considered together with the finding of the first study that the update size does not change the magnitude of the effect significantly. Moreover, in this setting, we found that for non-feature updates a positive effect does not emerge (i.e. performance improvements, bug fixes etc.). Finally, it could be shown that an update only has the potential to increase user's continuance intentions if it is announced either before or after the update's implementation in the case of a feature update, and only if it is announced after successful implementation in the case of a security update. Moreover, if a choice of installation was provided to users of the software, the positive effect was canceled out. This uncovers further crucial moderator for the effect of Agile IS on users. These additional moderators and partially malleable factors answer our second research question.

Overall, by answering the research questions, the limited understanding of software updates including non-rational responses of users to changes in the feature level composition of IT Artifacts is extended. Considered jointly, the results of the studies thereby contribute to a better understanding of Agile IS and provide several crucial theoretical and practical contributions that are outlined in the following section.

6.1 Theoretical Contributions

The thesis makes three main contributions to the literature that highlight whether, how, and why Agile IS affect users' continuance intentions and – more generally speaking – loyalty. First, understanding Agile IS from a user's perspective is crucially important both to build a comprehensive theoretical foundation on Agile IS and to put the user and his needs more at the center of all investigations (Brenner et al. 2014). Only few studies have explored this perspective thus far (e.g., Hong et al. 2016). While most of the studies in this field have pushed the user to the sidelines (e.g., Chan and Thong 2009), this thesis clearly contributes to a better understanding of how modifications of software compositions may change users' perceptions. In particular, we could demonstrate that Agile IS with a limited feature-set at the first release have the potential to increase user's continuance intentions through successive feature releases compared to monolithic and static IS. We could replicate and confirm this effect in several studies. These findings provide evidence that the user should be thoroughly considered in research on Agile IS. However, the results do not only underline the central role of the user in IS research. Also they suggest that some user responses to changes in software appear non-rational. Considering that users in most cases respond with higher continuance intentions to Agile IS despite being deprived of features some way through the usage period (compared to users of a feature-complete and monolithic software), the results may be interpreted as possible empirical evidence for a reference point dependency (Kahneman and Tversky 1979). This adds to the notion that users in IS research do not always act fully rational, but may be prone to heuristics and biases (Fleischmann et al. 2014). Finally, our results once again confirm the pivotal role of ECT and the IS Continuance Model in IS research. The results of our studies repeatedly demonstrate a mediation of the positive effect of updates on users' continuance intentions through the mediation mechanism of disconfirmation with previous expectations, satisfaction, and perceived usefulness. The discovered effect therefore requires an unexpected and positive surprise compared to a previous baseline. This subjective evaluation is clearly in line with theory on bounded rationality of users and therefore again questions the concept of a rational user in IS.

Second, our findings provide evidence in support of the necessity of a fine-grained understanding of IT Artifacts and the joint consideration of users in IS research. This answers the call of several researchers to put the IT Artifact more at the focus of IS research (Benbasat and Zmud 2003). Our findings show that the particular composition of features, changes in the feature-set, and the characteristics of the change have the potential to affect user responses and must be considered when theorizing on IT Artifacts. For example, our results indicate, that the previous endowment of software in terms of the available number of features seems to act as a moderator on the effect of updates on users, by diminishing its magnitude. Also, we could demonstrate that the removal of a specific feature is valued more in absolute (negative) magnitude than the equivalent acquisition of the same feature. This indicates a possible loss-aversion (Kahneman and Tversky 1979). In this setting, we found evidence that the positive effect of updates is only elicited, if users have less expertise regarding the software (i.e., they are novices). These findings again underline the somewhat non-rational nature of user responses in IS research, although experts come closer to rational actors. However, the findings also emphasize the important role of specific feature level compositions of IT Artifacts and changes in it to increase the predictive power of IS theory (Benlian 2015).

Adding to this finding, we could demonstrate that the size of an update (i.e., number of features contained in the update) does not play a significant role for the magnitude of the effect. However, the frequency of updates does: more frequent updates stimulate an even stronger positive response than less frequent updates do. As a result, features spread over several distinct and thus more frequent updates increase the positive effect even further. Finally, we could show that deliberate design of the update delivery process has the potential to moderate the effect of updates on users. In the case of feature updates, a positive effect was only established when the user was notified of the update before or after the update. In the case of a security update, only the notification after successful installation established a positive effect. Providing a choice to only optionally consume the update diminished any positive effect. Considering all these findings collectively, the results of our studies add to the predominantly monolithic understanding of software by providing a more modular understanding of software as specific compositions of features at a certain point in time that may be subject to change. Moreover, in sum, our findings highlight the necessity to join consideration of the malleable nature of IT Artifacts and the characteristics of its users to fully understand potential consequences.

Third, our overarching contribution lies in the extension of the predominant view of IS in post-adoption literature from a mostly static to a more dynamic perspective by showing how an evolving IS might change users' attitudes and behaviors over time. Thereby we complement existing IS post-adoption literature and research on digital ecosystems (e.g., Carillo et al., 2014; Liu et al., 2016) through the notion that users' beliefs and attitudes may change with the advancement of a system. With this finding, we answer the call of several IS scholars to consider the evolution of IS more thoroughly (e.g., Jasperson et al. 2005; Benbasat and Barki 2007). In particular, our results show how changes in IS due to updates, induce changes in users' beliefs and attitudes towards a system. This result confirms the previous findings of other scholars that an IT Artifact itself can affect users' beliefs and attitudes during use in later usage stages (e.g., Kim and Malhotra 2005; Kim and Son 2009; Ortiz de Guinea and Markus 2009; Ortiz de Guinea and Webster 2013). Hence our thesis highlights the consequences of the evolving nature of IS. Next, after outlining our theoretical contributions, we continue by highlighting practical and managerial contributions.

6.2 Practical Contributions

Our results also have important implications for IS practice. Particularly for firms that are discussing the implementation of Agile IS and are looking for empirically backed rationales to inform management decisions can benefit from our findings. But also firms that are already providing Agile IS to users and are looking for guidance on strategic or design considerations can benefit from our findings.

First, despite the prevalence of updates in modern software ecosystems to roll out increments of Agile IS, it is surprising that there is little managerial guidance on how firms can benefit from the potential for such updates to increase users’ loyalty. Our results offer strategic advice on when and how to deliver which functionality to users. From the results of our experimental studies we can generally conclude that firms should plan to deliver additional features to users successively, after the initial release of a software instead of providing a feature-complete software package right from the beginning. This will increase users’ continuance intentions above and beyond levels generated by a monolithic software. This effect is particularly strong for software that has a small feature-set and diminishes somewhat for software with many features. However, firms should not overdraw holding back features, as this might result in the software being discontinued before a positive effect can even be elicited. In particular, firms must also consider their customer base. Experts will not fall prey to held back features. They are more likely to know common feature sets and are more aware of the available feature-set in software. Thus, firms should first investigate the sociodemographic structure of their customer base, e.g. through market research, before implementing an appropriate strategy. Finally, firms should under no circumstances remove features from software if not absolutely necessary. Users will perceive the loss in functionality significantly more negatively than a comparable gain in functionality.

Second, the manifold results of our studies offer many specifics for practitioners on how to design and communicate increments of Agile IS. Our findings suggest that the update’s size does not play a significant role in its reception, however the frequency of updates does. Therefore, firms should roll out features distributed across more frequent but smaller update packages. Such smaller packages will delight the user each with a positive and unexpected surprise of additional functionality, inducing higher intentions to continue using the Agile IS. Nonetheless, users should always be notified about updates. In the case of feature updates either before or after the successful implementation, and in the case of security updates only after the successful implementation. This is because the update is only a necessary condition for a positive effect, while a notification is the sufficient condition that helps the update to be noticed which establishes the positive effect. However, in no case should updates be offered to users as voluntary choices. This will not only discard any positive effect, but also reduce the installation rate drastically. In the case of security updates this could increase the risk of users being victims of security attacks and should be avoided in any case (if for example the update closes a critical security breach). In sum, our studies provide many details on how to design Agile IS for the benefit of both users and also firms.

Third, our studies provide an overarching and strong rationale for practitioners in favor of establishing Agile IS from the user's perspective. Many practitioners still struggle internally with the deployment and acceptance of Agile IS. Often, despite clear indications of better and more effective development results, less wasted resources, and increased agility; in many cases firms are still not able to adapt to agile methods. Reasons often lie within unsuitable established cultures, fixed processes, and planning horizons, and in regulatory requirements. However, markets are still becoming increasingly more customer centric. The findings of our studies show that even if not required out of an internal perspective, Agile IS have the potential to increase user's loyalty. In times of strong competition and with the growing number of business models that are based on reoccurring revenue streams from customers and their engagement, this characteristic gains vital importance. This applies even more so, because an increase in agility drastically decreases the firm's time-to-market. Practitioners can therefore benefit from this thesis by adding the evident gains in value-to-customer through Agile IS to their cost-benefit analysis when assessing agile methods and Agile IS.

6.3 Limitations and Future Research

Three limitations of this thesis are noteworthy and provide avenues for future research. First, in our studies, we conducted either controlled laboratory experiments or online experiments. The manipulations of the experiments were realized through self-developed, fully-functional software mock-ups, comprehensive click dummies, and textual scenarios. This allowed us to identify causal relations, while accurately controlling for potential cofounding variables. Hence we obtained not only results with a high internal validity, even more, with the help of these studies on the effect of Agile IS on users, we could clearly isolate the primary effects and mechanisms. However, the settings are only approximations of real-world usage scenarios and, for the sake of clarity, reduced to capture only all relevant core aspects. Taking this into account, we deliberately controlled for the participants' perceptions of the setting as realistic and other control variables, and carefully pre-tested our experiments in several cycles. Based on the results derived from these measures, we are confident that our experiments worked as intended and that our implications are applicable to real-usage settings. And since we could confirm our results in several usage contexts and with different manipulation methods, even more, we have reasons to believe that they are relevant to a wide range of settings.

Nonetheless, future studies could investigate actual usage experiences with real software to validate our findings. Moreover, such studies could employ software with more diverse compositions of features and a higher complexity of interaction, to extend and validate our findings even further.

Second, for this fundamental study of Agile IS from the user's perspective, we imposed a linear and uniform course of events and a limited observed timespan to obtain a feasible setting. Though this allows us to compare monolithic, full-featured software to Agile IS and derive precise implications for both strategies, nevertheless, the external validity of our findings can be improved by increasing the duration of observations timespans and increasing the complexity of the scenarios. In our studies, we considered these aspects by carefully constructing the scenarios in a way that each task, the procedure, and the observed timespan were evaluated as natural by participants. However, future studies are encouraged to complement the findings of our work by conducting longitudinal field experiments, to advance the external validity of our findings. Additionally, settings with repeated updates over longer time spans with participants' evaluations measured at several points in time could provide additional evidence for the robustness of our findings and address any questions of the consistency of the effect on users over long timespans.

Finally, IS theory suggests that due to limited cognitive resources and the interfering nature of updates, there might be a boundary condition for the amount of updates that are perceived as beneficial by users leading to a potential saturation of the effect. Although we have found the first indications that allow for assumptions in this direction (i.e., by thought-listing in the third study), we could not show an explicit saturation or even a negative effect caused by an overload of changes in Agile IS through updates to users. Admittedly, modern Agile IS implements update processes that are as unobtrusive to the user as possible, which generally decreases the interference of updates. Such new seamless processes and their consequences were investigated in study four. However, future studies should investigate potential downsides of excessive changes in IT Artifacts due to updates from the user's perspective. Only by gaining a better understanding of the range of potential negatively influencing factors, can IS theory help firms to develop increasingly seamless update delivery processes and thereby increase the acceptance of Agile IS and the perceived value of Agile IS to the end user.

In conclusion, Agile IS have become an integral part of today's IT landscapes. Although their characteristics have been widely studied from a firm's perspective, their nature and consequences from a user’s perspective have remained underexplored so far. Although this thesis is only a first step to extend the understanding from this perspective, we were able to demonstrate in four distinct empirical experiments that the effects of Agile IS on users are salient, diverse, and shapeable and thus cannot be neglected. Only with an integrated, human focused, and finer-grained view of Agile IS we can fully understand its value-to-customers and build a solid theoretical foundation of user experience of IT systems. Thereby we can empower companies with the knowledge of how to increase customer loyalty and ultimately the perceived value to their customer base. Following this notion, we hope that this substantial perspective shift and our somewhat surprising results will foster further research by other IS scholars into this direction.

References

Ackermann, T., and Buxmann, P. 2010. “Quantifying Risks in Service Networks: Using Probability Distributions for the Evaluation of Optimal Security Levels,” *Sixteenth Americas Conference on Information Systems (AMCIS)*, Lima, Peru.

Ågerfalk, P. J., Fitzgerald, B., and Slaughter, S. A. 2009. “Introduction to the Special Issue—Flexible and Distributed Information Systems Development: State of the Art and Research Challenges,” *Information Systems Research* (20:3), pp. 317-328.

Aguinis, H., and Bradley, K. J. 2014. “Best Practice Recommendations for Designing and Implementing Experimental Vignette Methodology Studies,” *Organizational Research Methods* (17:4), pp. 351-371.

Alba, J. W., and Hutchinson, J. W. 1987. “Dimensions of Consumer Expertise,” *Journal of Consumer Research* (13:4), pp. 411-454.

Amirpur, M., Fleischmann, M., Benlian, A., and Hess, T. 2015. “Keeping Software Users on Board—Increasing Continuance Intention Through Incremental Feature Updates,” *Proceedings of the 23rd European Conference on Information Systems*, Münster, Germany, pp. 1-16.

Anderson, E. W., and Sullivan, M. W. 1993. “The Antecedents and Consequences of Customer Satisfaction for Firms,” *Marketing Science* (12:2), pp. 125-143.

Anderson, J. R. 1982. “Acquisition of Cognitive Skill,” *Psychological Review* (89:4), pp. 369-406.

Apple. 2015. *iOS 6 Software-Update*, https://support.apple.com/kb/DL1578, 01.05.2015.

Bartholomew, D. J., Steele, F., Moustaki, I., and Galbraith, J. 2008. *Analysis of Multivariate Social Science Data* (2nd ed.), London: CRC Press.

Beck, K. 1999. “Embracing Change with Extreme Programming,” *IEEE Computer* (32:10), pp. 70-77.

Benartzi, S., Thaler, R. H. 1995. “Myopic Loss Aversion and the Equity Premium Puzzle,” *The Quarterly Journal of Economics* (110:1), pp. 73-92.

Benbasat, I., and Barki, H. 2007. “Quo vadis, TAM?,” *Journal of the Association for Information Systems* (8:4), pp. 211-218.

Benbasat, I., and Zmud, R. W. 2003. “The Identity Crisis within the IS Discipline: Defining and Communicating the Discipline's Core Properties,” *MIS Quarterly* (27:2), pp. 183-194.

Benlian, A. 2013a. “Are We Aligned… Enough? The Effects of Perceptual Congruence Between Service Teams and Their Leaders on Team Performance,” *Journal of Service Research* (17:2), pp. 212-228.

Benlian, A. 2013b. “Effect Mechanisms of Perceptual Congruence Between Information Systems Professionals and Users on Satisfaction with Service,” *Journal of Management Information Systems* (29:4), pp. 63-96.

Benlian, A. 2015a. “IT Feature Use Over Time and its Impact on Individual Task Performance,” *Journal of the Association for Information Systems* (16:3), pp. 144-173.

Benlian, A. 2015b. “Web Personalization Cues and Their Differential Effects on User Assessments of Website Value,” *Journal of Management Information Systems* (32:1), pp. 225-260.

Benlian, A., and Haffke, I. 2016. “Does Mutuality Matter? Examining the Bilateral Nature and Effects of CEO–CIO Mutual Understanding,” *The Journal of Strategic Information Systems* (25:2), pp. 104-126.

Benlian, A., and Hess, T. 2011a. “Opportunities and Risks of Software-as-a-service: Findings from a Survey of IT Executives,” *Decision Support Systems* (52), pp. 232-246.

Benlian, A., and Hess, T. 2011b. “The Signaling Role of IT Features in Influencing Trust and Participation in Online Communities,” *International Journal of Electronic Commerce* (15:4), pp. 7-56.

Benlian, A., Koufaris, M., and 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), pp. 85-126.

Bhattacherjee, A. 2001. “Understanding Information Systems Continuance: An Expectation Confirmation Model,” *MIS Quarterly* (25:3), pp. 351-370.

Bhattacherjee, A., and Barfar, A. 2011. “Information Technology Continuance Research: Current State and Future Directions,” *Asia Pacific Journal of Information Systems* (21:2), pp. 1-18.

Bhattacherjee, A., and Premkumar, G. 2004. “Understanding Changes in Belief and Attitude toward Information Technology Usage: A Theoretical Model and Longitudinal Test,” *MIS Quarterly* (28:2), pp. 229-254.

Blackler, A., Popovic, V., and Mahar, D. 2010. “Investigating Users’ Intuitive Interaction with Complex Artefacts,” *Applied Ergonomics* (41:1), pp. 72-92.

Bolton, R. N. 1998. “A Dynamic Model of the Duration of the Customer's Relationship with a Continuous Service Provider: The Role of Satisfaction,” *Marketing Science* (17:1), pp. 45-65.

Bowden, Z. 2017. *How Microsoft improved Windows Update in the Creators Update*, https://www.windowscentral.com/windows-update-less-pain-butt-creators-update, 04.03.2018.

Brenner, W., Österle, H., Petrie, C., Uebernickel, F., Winter, R., Karagiannis, D., Kolbe, L., Krüger, J., Leifer, L., Lamberti, H-J., Leimeister, J., , Schwabe, G., and Zarnekow, R. 2014. “User, Use & Utility Research,” *Business & Information Systems Engineering* (2014:1), pp. 55-61.

Brucks, M. 1985. “The Effects of Product Class Knowledge on Information Search Behavior,” *Journal of Consumer Research.* (12:2), pp. 1-16.

Carillo, K. D. A., Scornavacca, E., and Za, S. 2014. “An Investigation of the Role of Dependency in Predicting Continuance Intention to Use Ubiquitous Media Systems: Combining a Media System Perspective with Expectation-Confirmation Theories,” *ECIS 2014 Proceedings*. Münster: Germany.

Chan, F. K. Y., and Thong, J. Y. L. 2009. “Acceptance of Agile Methodologies: a Critical Review and Conceptual Framework,” *Decision Support Systems* (46:4), pp. 803-814.

Chin, W. W., Marcolin, B. L., and Newsted, P. R. 2003. “A Partial Least Squares Latent Variable Modeling Approach for Measuring Interaction Effects: Results from a Monte Carlo Simulation Study and an Electronic-Mail Emotion/Adoption Study,” *Information Systems Research* (14:2), pp. 189-217.

Claussen, J., Kretschmer, T., and Mayrhofer, P. 2013. “The Effects of Rewarding User Engagement: The Case of Facebook Apps,” *Information Systems Research* (24:1), pp. 186-200.

Clements, P., and Northrop, L. 2002. *Software Product Lines: Practices and Patterns* (3rd ed.), Boston, MA: Addison-Wesley Reading.

Conboy, K. 2009. "Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development," *Information Systems Research* (20:3), pp. 329-354.

Constantin, L. 2013. *EFF criticizes Google for removing 'vital privacy feature' with Android 4.4.2 | PCWorld*, http://www.pcworld.com/article/2080241/eff-criticizes-google-for-removing-vital-privacy-feature-with-android-442.html, 04.09.2015.

Constine, J. 2016. *Instagram launches "Stories," a Snapchatty feature for imperfect sharing*, https://techcrunch.com/2016/08/02/instagram-stories/, 04.02.2018.

Cook, T. D., and Campbell, D. T. 1979. *Quasi-Experimentation: Design & Analysis Issues for Field Settings*, Boston, MA: Houghton Mifflin.

Cronbach, L. J. 1951. "Coefficient Alpha and the Internal Structure of Tests," *Psychometrika* (16:3), pp. 297-334.

Darby, M. R., and Karni, E. 1973. "Free Competition and the Optimal Amount of Fraud," *Journal of Law and Economics* (16:1), pp. 67-88.

Davis, F. D. 1989. "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly* (13:3), pp. 319-340.

De Cremer, D. D., Van Dijke, M. V., and Bos, A. E. R. 2007. "When Leaders Are Seen as Transformational: The Effects of Organizational Justice," *Journal of Applied Social Psychology* (37:8), pp. 1797–1816.

Denning, S. 2015a. *Surprise: Microsoft Is Agile*, https://www.forbes.com/sites/stevedenning/2015/10/27/surprise-microsoft-is-agile/, 03.02.2018.

Denning, S. 2015b. *Why Do Managers Hate Agile?*, https://www.forbes.com/sites/stevedenning/2015/01/26/why-do-managers-hate-agile/, 03.02.2018.

Dennis, A. R., Robert Jr, L. P., Curtis, A. M., Kowalczyk, S. T., and Hasty, B. K. 2012. "Research Note-Trust Is in the Eye of the Beholder: A Vignette Study of Postevent Behavioral Controls' Effects on Individual Trust in Virtual Teams," *Information Systems Research* (23:2), pp. 546-558.

DeSanctis, G., and Poole, M. S. 1994. "Capturing the Complexity in Advanced Technology Use: Adaptive Structuration Theory," *Organization science* (5:2), pp. 121-147.

Dhar, R., and Sherman, S. J. 1996. "The Effect of Common and Unique Features in Consumer Choice," *Journal of Consumer Research* (23:3), pp. 193-203.

Dinev, T., and Hu, Q. 2007. "The Centrality of Awareness in the Formation of User Behavioral Intention toward Protective Information Technologies," *Journal of the Association for Information Systems* (8:7), pp. 386-408.

Dunn, K. 2004. "Automatic Update Risks: Can Patching Let a Hacker in?," *Network Security 2004* (7), pp. 5-8.

Eschenbrenner, B., and Nah, F. F.-H. 2014. "Information Systems User Competency: A Conceptual Foundation," *Communications of the Association for Information Systems* (34:1), pp. 1363-1378.

Etherington, D. 2013. *Chat Heads Coming To iOS Facebook App Via Update Pushing Out Anytime Now*, http://techcrunch.com/2013/04/16/chat-heads-coming-to-ios-facebook-app-via-update-pushing-out-anytime-now/, 01.05.2014.

Evans, J. S. 2006. "The Heuristic-analytic Theory of Reasoning: Extension and Evaluation," *Psychonomic Bulletin & Review* (13:3), pp. 378-395.

Evans, J. S. 2008. "Dual-processing Accounts of Reasoning, Judgment, and Social Cognition," *Annu. Rev. Psychol.* (59), pp. 255-278.

Facebook 2015. Facebook Product News, http://newsroom.fb.com/news/category/product-news/, 23.04.2015.

Fleischmann, M., Amirpur, M., Benlian, A., and Hess, T. 2014. "Cognitive Biases in Information Systems Research: A Scientometric Analysis," *Proceedings of the 22nd European Conference on Information Systems (ECIS)*, Tel Aviv, Isreal.

Fleischmann, M., Amirpur, M., Grupp, T., Benlian A., and Hess, T. 2016. "The Role of Software Updates in Information Systems Continuance – An Experimental Study from a User Perspective," *Decision Support Systems* (83), pp. 83-96.

Fornell, C., and Larcker, D. F. 1981. "Evaluating Structural Equation Models with Unobservable Variables and Measurement Error," *Journal of Marketing Research* (18:1), pp. 39-50.

Fowler, M., and Highsmith, J. 2001. “Agile Methodologists Agree on Something,” *Software Development* (9), pp. 28-32.

Fruhling, A., and Vreede, G.-J. 2006. “Field Experiences with Extreme Programming: Developing an Emergency Response System,” *Journal of Management Information Systems* (22:4), pp. 39-68.

Fujita, K., Trope, Y., Liberman, N., and Levin-Sagi, M. 2006. “Construal Levels and Self-Control,” *Journal of Personality and Social Psychology* (90:3), pp. 351–367.

Ghoshal, T., Yorkston, E., Nunes, J. C., and Boatwright, P. 2014. “Multiple Reference Points in Sequential Hedonic Evaluation: An Empirical Analysis,” *Journal of Marketing Research* (51:5), pp. 563-577.

Goldbach, T., Benlian, A., and Buxmann, P. 2017. “Differential effects of formal and self-control in mobile platform ecosystems: Multi-method findings on third-party developers’ continuance intentions and application quality,” *Information & Management*.

Goodhue, D. L., and Thompson, R. L. 1995. “Task-technology fit and individual performance,” MIS Quarterly (19:2), pp. 213–236.

Gupta, S., and Kim, H.-W. 2007. “The Moderating Effect of Transaction Experience on the Decision Calculus in On-Line Repurchase,” *International Journal of Electronic Commerce* (12:1), pp. 127-158.

Haffke, I., Kalgovas, B., and Benlian, A. 2017. “Options for Transforming the IT Function Using Bimodal IT,” *MIS Quarterly Executive* (16:2), pp. 101-120.

Hair, J. F., Hult, G. T. M., Ringle, C. M., and Sarstedt, M. 2014. *A Primer on Partial Least Squares Structural Equation Modeling (PLS-SEM),* Thousand Oaks, CA: Sage Publications.

Hair, J. F., Ringle, C. M., and Sarstedt, M. 2011. “PLS-SEM: Indeed a Silver Bullet,” Journal *of Marketing Theory and Practice* (19:2), pp. 139-151.

Harnisch, S., and Buxmann, P. 2013. *Evaluating Cloud Services Using Methods of Supplier Selection.* In: Business Information Systems. BIS 2013. Heidelberg: Springer.

Harris, M. L., Collins, R. W., and Hevner, A. R. 2009. “Control of Flexible Software Development Under Uncertainty,” *Information Systems Research* (20:3), pp. 400-419.

Harrison, D. A., and Klein, K. J. 2007. “What's the difference? Diversity constructs as separation, variety, or disparity in organizations,” *Academy of Management Review* (32:4), pp. 1199-1228.

Hayes, A. F. 2013. *Introduction to Mediation, Moderation, and Conditional Process Analysis: A Regression-Based Approach*, New York, NY: Guilford Publications, Inc.

Helson, H. 1964. *Adaptation-Level Theory: An Experimental and Systematic Approach to Behavior,* New York: Harper & Row.

Hevner, A. R., March, S. T., Park, J., and Ram, S. 2004. “Design Science in Information Systems Research,” *MIS Quarterly* (28:1), pp. 75-105.

Highsmith, J., and Cockburn, A. 2001. “Agile Software Development: The Business of Innovation,” *IEEE Computer* (34:9), pp. 120-122.

Hiltz, S. R., and Turoff, M. 1981. “The Evolution of User Behavior in a Computerized Conferencing System,” *Communications of the ACM* (24:11), pp. 739-751.

Hirotaka, T., and Nonaka, I. 1986. *The New New Product Development Game,* https://hbr.org/1986/01/the-new-new-product-development-game, 03.02.2018.

Hoch, S. J., and Ha, Y.-W. 1986. “Consumer Learning: Advertising and the Ambiguity of Product Experience,” *Journal of Consumer Research* (13:3), pp. 221-233.

Hodgetts, H. M., and Jones, D. M. 2007. “Reminders, alerts and pop-ups: The cost of computer-initiated interruptions,” in *Human–computer interaction*, J. Jacko (ed.). Berlin/Heidelberg: Springer-Verlag, pp. 818-826.

Hoffmann, A. O., and Broekhuizen, T. L. 2009. “Susceptibility to and impact of interpersonal influence in an investment context,” *Journal of the Academy of Marketing Science* (37:4), pp. 488-503.

Hogarth, R. M., and Einhorn, H. J. 1992. “Order Effects in Belief Updating: The Belief-Adjustment Model,” *Cognitive Psychology* (24:1), pp. 1-55.

Hong, S., Thong, J. Y., and Tam, K. Y. 2006. “Understanding continued information technology usage behavior: A comparison of three models in the context of mobile internet,” *Decision Support Systems* (42:3), pp. 1819-1834.

Hong, W., Thong, J. Y. L., and Tam, K. Y. 2004. “Does Animation Attract Online Users’ Attention? The Effects of Flash on Information Search Performance and Perceptions,” *Information Systems Research* (15:1), pp. 60-86.

Hong, W., Thong, J. Y., Chasalow, L. C., and Dhillon, G. 2011. “User Acceptance of Agile Information Systems: A Model and Empirical Test,” *Journal of Management Information Systems* (28:1), pp. 235-272.

Hoxmeier, J. A. 2000. “Software Preannouncements and Their Impact on Customers' Perceptions and Vendor Reputation,” *Journal of Management Information Systems* (17:1), pp. 115-139.

Huff, S. L., Munro, M. C., and Marcolin, B. 1992. “Modelling and measuring end user sophistication,” *Proceedings of the 1992 ACM SIGCPR Conference*, Cincinnati: OH, pp. 1-10.

Hummel, M. 2014. “State-of-the-Art: A Systematic Literature Review on Agile Information Systems Development,” *47th Hawaii International Conference on System Science (HICSS)*, Hawaii.

Hummel, M., Rosenkranz, C., and Holten, R. 2013. “The Role of Communication in Agile Systems Development – an Analysis of the State of the Art,” *Business & Information Systems Engineering* (5), pp. 343-355.

Irmak, C., Block, L. G., and Fitzsimons, G. J. 2005. “The placebo effect in marketing: Sometimes you just have to want it to work,” *Journal of Marketing Research* (42:4), pp. 406-409.

Iyengar, S. S., and Lepper, M. R. 2000. “When Choice is Demotivating: Can One Desire Too Much of a Good Thing?” *Journal of Personality and Social Psychology* (79:6), pp. 995-1006.

Jasperson, J. S., Carter, P. E., and Zmud, R. W. 2005. “A Comprehensive Conceptualization of Post-Adoptive Behaviors Associated with Information Technology Enabled Work Systems,” *MIS Quarterly* (29:3), pp. 525-557.

Jenkins, L., Anderson, B. B., Vance, A., Kirwan, C. B., and Eargle, D. 2016. “More Harm Than Good? How Messages That Interrupt Can Make Us Vulnerable,” *Information Systems Research* (27:4), pp. 880-896.

Jeong, H. J., and Kwon, K.-N. 2012. “The Effectiveness of Two Online Persuasion Claims: Limited Product Availability and Product Popularity,” *Journal of Promotion Management* (18:1), pp. 83-99.

Juner, C., and Benlian, A. 2017. “Praxisbasierte Capability-Modelle Für DevOps-Einsätze in Unternehmen," *HMD Praxis der Wirtschaftsinformatik* (54:2), pp. 230-243.

Kahneman, D., and Tversky, A. 1979. “Prospect Theory: An Analysis of Decision Under Risk,” *Econometrica: Journal of The Econometric Society* (47:2), pp. 263-291.

Kahneman, D., Knetsch, J. L., and Thaler, R. H. 1991. “Anomalies: The endowment effect, loss aversion, and status quo bias,” *The journal of economic perspectives* (5:1), pp. 193-206.

Kahnemann, D. 1973. *Attention and effort.* Englewood Cliffs: Prentice-Hall.

Karahanna, E., Straub, D. W., and Chervany, N. L. 1999. “Information Technology Adoption Across Time: A Cross-Sectional Comparison of Pre-Adoption and Post-Adoption Beliefs,” *MIS Quarterly* (23:2), pp. 183-213.

Kay, J., and Thomas, R. C. 1995. “Studying Long-Term System Use,” *Communications of the ACM* (38:7), pp. 61-69.

Kim, S. S. 2009. “The integrative framework of technology use: an extension and test,” *MIS Quarterly* (33:3), pp. 513-537.

Kim, S. S., and N. K. Malhotra 2005. “A Longitudinal Model of Continued IS Use: An Integrative View of Four Mechanisms Underlying Postadoption Phenomena,” *Management Science* (51:5), pp. 741-755.

Kim, S. S., and Son, J.-Y. 2009. “Out of Dedication or Constraint? A Dual Model of Post-Adoption Phenomena and its Empirical Test in the Context of Online Services,” *MIS Quarterly* (33:1), pp. 49-70.

Larsen, T. J., Sørebø, A. M., and Sørebø, Ø. 2009. “The role of task-technology fit as users’ motivation to continue information system use,” *Computers in Human Behavior* (25), pp. 778-784.

Lee, G., and Xia, W. 2010. “Toward Agile: an Integrated Analysis of Quantitative and Qualitative Field Data on Software Development Agility,” *MIS Quarterly* (34:1), pp. 87-114.

Leonardi, P. M. 2013. “When does Technology Use Enable Network Change in Organizations? A Comparative Study of Feature Use and Shared Affordances,” *MIS Quarterly* (37:3), pp. 749-775.

Levin, J. R. 1981. “On Functions of Pictures in Prose,” in *Neuropsychological and Cognitive Processes in Reading*, P. F. J and W. M. C (eds.). New York: Academic Press, p. 344.

Levine, T. R. 2005. “Confirmatory Factor Analysis and Scale Validation in Communication Research,” *Communication Research Reports* (22:4), pp. 335-338.

Liang, H., and Xue, Y. 2010. “Understanding Security Behaviors in Personal Computer Usage: A Threat Avoidance Perspective,” *Journal of the Association for Information Systems* (11:7), pp. 394-413.

Liu, J., Abhishek, V., and Li, B. 2016. “The Impact of Mobile Adoption on Customer Omni-Channel Banking Behavior,” *ICIS 2016 Proceedings*. Dublin: Irland.

Locke, E. A. 1976. “The Nature and Causes of Job Satisfaction,” in *Handbook of Industrial and Organizational Psychology*, Ed. by M.D. Dunnette. Chicago: Rand McNally College Pub. Co., pp. 1297-1349.

Lu, Y., and Ramamurthy, K. 2011. “Understanding the Link Between Information Technology Capability and Organizational Agility: An Empirical Examination,” *MIS Quarterly* (35:4), pp. 931-954.

Lynch Jr, J. G., Chakravarti, D., and Mitra, A. 1991. “Contrast Effects in Consumer Judgments: Changes in Mental Representations or in the Anchoring of Rating Scales?,” *Journal of Consumer Research* (18:3), pp. 284-297.

Ma, J., and Roese, N. J. 2014. “The maximizing mind-set,” *Journal of Consumer Research* (41:1), pp. 71-92.

Maier, C., Laumer, S., Weinert, C., and Weitzel, T. 2015. “Resistance to Change: Developing an Individual Differences Measure,” *Information Systems Journal* (25), pp. 275-308.

Malaga, R. A. 2000. “The effect of stimulus modes and associative distance in individual creativity support systems,” *Decision Support Systems* (29:2), pp. 125-141.

Marakas, G. M., Johnson, R. D., and Clay, P. F. 2007. “The Evolving Nature of the Computer Self-Efficacy Construct: An Empirical Investigation of Measurement Construction, Validity, Reliability and Stability Over Time,” *Journal of the Association for Information Systems* (8:1), pp. 16-46.

Marakas, G. M., Yi, M. Y., and Johnson, R. D. 1998. “The Multilevel and Multifaceted Character of Computer Self-Efficacy: Toward Clarification of the Construct and an

Integrative Framework for Research," *Information Systems Research* (9:2), pp. 126-163.

Marcolin, B. L., Compeau, D. R., Munro, M. C., and Huff, S. L. 2000. "Assessing User Competence: Conceptualization and Measurement," *Information Systems Research* (11:1), pp. 37-60.

Maruping, L. M., Venkatesh, V., and Agarwal, R. 2009. "A Control Theory Perspective on Agile Methodology Use and Changing User Requirements," *Information Systems Research* (20:3), pp. 377-399.

Mathias, R. 2012. "Google Chrome Blog: Keeping tabs on your tabs," http://chrome.blogspot.de/2012/05/keeping-tabs-on-your-tabs.html, 01.05.2014.

Mehta, R., Zhu, R., and Meyers-Levy, J. 2014. "When Does a Higher Construal Level Increase or Decrease Indulgence? Resolving the Myopia versus Hyperopia Puzzle," *Journal of Consumer Research* (41), pp. 475-488.

Mens, T., and Demeyer, S. 2008. *Software Evolution*. Berlin and Heidelberg: Springer.

Microsoft 2015a. *Windows lifecycle fact sheet – Windows Help,* http://windows.microsoft.com/en-us/windows/lifecycle, 21.07.2015.

Microsoft 2015b. *Office 365 Roadmap,* http://roadmap.office.com/, 23.04.2015.

Mishra, D., Mishra A., and Ostrovska, S. 2012. "Impact of Physical Ambiance on Communication, Collaboration and Coordination in Agile Software Development: an Empirical Evaluation," *Information Software Technology* (54:10), pp. 1067-1078.

Mishra, S., Umesh, U. N., and Stem, D. E. J. 1993. "Antecedents of the Attraction Effect: An Information-Processing Approach," *Journal of Marketing Research* (30:3), pp. 331-349.

Morgan, H. M., and Ngwenyama, O. 2015. "Real Options, Learning Cost and Timing Software Upgrades: Towards an Integrative Model for Enterprise Software Upgrade Decision Analysis," International Journal of Production Economics (168), pp. 211-223.

Mukherjee, A., and Hoyer, W. D. 2001. "The effect of novel attributes on product evaluation," *Journal of Consumer Research* (28:3), pp. 462-472.

Munro, M. C., Huff, S. L., Marcolin, B. L., and Compeau, D. R. 1997. "Understanding and measuring user competence," *Information & Management* (33:1), pp. 45-57.

Murray, K. B., and Häubl, G. 2011. "Freedom of Choice, Ease of Use, and the Formation of Interface Preferences," MIS Quarterly (35:4), pp. 955-976.

Myerson, T. 2015. *The next generation of Windows: Windows 10,* http://blogs.windows.com/bloggingwindows/2015/01/21/the-next-generation-of-windows-windows-10/, 01.05.2015.

Nelson, K. M., Nadkarni, S., Narayanan, V., and Ghods, M. 2000. "Understanding Software Operations Support Expertise: A Revealed Causal Mapping Approach," *MIS Quarterly* (24:3), pp. 475-507.

Ng, B.-Y., Kankanhalli, A., and Xu, Y. 2009. "Studying users' computer security behavior: A health belief perspective,“ *Decision Support Systems* (46), pp. 815-825.

Nicolaou, A. I., and McKnight, D. H. 2011. "System design features and repeated use of electronic data exchanges," *Journal of Management Information Systems* (28:2), pp. 269-304.

Norman, D. A., and Bobrow, D. G. 1975. "On Data-limited and Resource-limited Process," *Cognitive Psychology* (7:1), pp. 44-64.

Nowlis, S. M., and Simonson, I. 1996. "The Effect of New Product Features on Brand Choice," *Journal of Marketing Research* (33), pp. 36-46.

Nunnally, J. C. 1994. *Psychometric Theory,* (3rd ed.). New York, NY: McGraw-Hill.

Ohno, T. 1988. *Toyota Production System - beyond large-scale production,* New York, NA: Productivity Press.

Oliver, R. L. 1980. "A Cognitive Model of the Antecedents and Consequences of Satisfaction Decisions," *Journal of Marketing Research* (17:4), pp. 460-469.

Oliver, R. L. 1993. "Cognitive, Affective, and Attribute Bases of the Satisfaction Response," *Journal of Consumer Research* (20:3), pp. 418-430.

Oreg, S. 2003. "Resistance to Change: Developing an Individual Differences Measure," *Journal of Applied Psychology* (88:4), pp. 680-693.

Orlikowski, W. J., and Iacono, C. S. 2001. "Research Commentary: Desperately Seeking the 'IT' in IT Research - A Call to Theorizing the IT Artifact," *Information Systems Research* (12:2), pp. 121-134.

Ortiz de Guinea, A., and Markus, M. L. 2009. “Why Break the Habit of a Lifetime? Rethinking the Roles of Intention, Habit, and Emotion in Continuing Information Technology Use,” *MIS Quarterly* (33:3), pp. 433-444.

Ortiz de Guinea, A., and Webster, J. 2013. “An Investigation of Information Systems Use Patterns: Technological Events as Triggers, the Effect of Time, and Consequences for Performance,” *MIS Quarterly* (37:4), pp. 1165-1188.

Paolacci, G., Chandler, J., and Ipeirotis, P. G. 2010. “Running Experiments on Amazon Mechanical Turk,” *Judgment and Decision Making* (5:5), pp. 411-419.

Perdue, B. C., and Summers, J. O. 1986. “Checking the Success of Manipulations in Marketing Experiments,” *Journal of Marketing Research* (23:4), pp. 317-326.

Polites, G. L., and Karahanna, E. 2012. “Shackled to the Status Quo: the Inhibiting Effects of Incumbent System Habit, Switching Costs, and Inertia on New System Acceptance,” *MIS Quarterly* (36:1), pp. 21-42.

Polites, G. L., and Karahanna, E. 2013. “The Embeddedness of Information Systems Habits in Organizational and Individual Level Routines: Development and Disruption,” *MIS Quarterly* (37:1), pp. 221-246.

Popović, M., Atlagić, B., and Kovačević, V. 2001. “Case Study: a Maintenance Practice Used with Real-Time Telecommunications Software,” *Journal of Software Maintenance and Evolution: Research and Practice* (13:2), pp. 97-126.

Poppendeick, M. 2001. “Lean Programming,” *Software Development* (9), pp. 71-75.

Rafaeli, S., and Raban, D. R. 2003. “Experimental Investigation of the Subjective Value of Information in Trading,” *Journal of the Association for Information Systems* (4:2003), pp. 119-139.

Rahman, N. 1996. “Caregivers’ Sensitivity to Conflict: The Use of Vignette Methodology,” *Journal of Elder Abuse & Neglect* (8:1), pp. 35-47.

Rhee, H.-S., Kim, C., and Ryu, Y. U. 2009. “Self-efficacy in information security: Its influence on end users' information security practice behavior,” *Computers & Security* (28:8), pp. 816-826.

Ringle, C. M., Wende, S., and Becker, J.-M. 2014. *SmartPLS 3.0*, Hamburg: SmartPLS GmbH (http://www.smartpls.com).

Ringle, C. M., Wende, S., and Will, A. 2005. “Smart PLS 2.0 (M3),” University of Hamburg, Hamburg, http://www.smartpls.de.

Rising, L., and Janoff, N. S. 2000. “The Scrum software development process for small teams,” *IEEE Software* (17:4), pp. 26-32.

Rogers, E. M. 1995. *Diffusion of Innovations* (4th ed.), New York: Free Press.

Ruiz-Hopper, M. 2017. *What’s new in the Windows 10 Creators Update*, https://blogs.windows.com/windowsexperience/2017/04/11/whats-new-in-the-windows-10-creators-update/#Duc0u5I7ugceijYY.97, 03.02.2018.

Saeed, K. A., and Abdinnour-Helm, S. 2008. “Examining the effects of information system characteristics and perceived usefulness on post adoption usage of information systems,” *Information & Management* (45:6), pp. 376-386.

Salmela, H., Tapanainen, T., Baiyere, A., Hallanoro, M., and Galliers, R. 2015. “IS Agility Research: An Assessment and Future Directions,” *23nd European Conference on Information Systems (ECIS)*, Münster, Germany.

Samat, S. (2016). *Android 7.0 Nougat: a more powerful OS, made for you*, https://blog.google/products/android/android-70-nougat-more-powerful-os-made/, 14.11.2016.

Sandberg, J., and Alvesson, M. 2011. “Ways of Constructing Research Questions: Gap-spotting or Problematization?,” *Organization* (18:1), pp. 23-44.

Scheibehenne, B., Greifeneder, R., and Todd, P. M. 2010. “Can There Ever Be Too Many Options? A Meta-Analytic Review of Choice Overload,” *Journal of Consumer Research* (37:3), pp. 409-425.

Schneider, D., Lins, S., Grupp, T., Benlian, A., and Sunyaev, A. 2017. “Nudging Users Into Online Verification: The Case of Carsharing Platforms,” *Thirty Eighth International Conference on Information Systems (ICIS)*, Seoul, South Korea.

Schwaber, K., and Beedle, M. 2002. *Agile Software Development with Scrum*, New Jersey: Prentice-Hall.

Sen, S., and Morwitz, V. G. 1996. “Is It Better to Have Loved and Lost Than Never to Have Loved at All? The Effect of Changes in Product Features over Time,” *Marketing Letters* (7:3), pp. 225-235.

Shanteau, J., and Stewart, T. R. 1992. "Why Study Expert Decision Making? Some Historical Perspectives and Comments," *Organizational Behavior and Human Decision Processes* (53:2), pp. 95-106.

Shaw, J. C., Wild, R. E., and Colquitt, J. A. 2003. "To justify or Excuse? A Meta-Analytic Review of the Effects of Explanations," *Journal of Applied Psychology* (88:3), pp. 444-458.

Shenoy, N. 2018. Firmware Updates and Initial Performance Data for Data Center Systems, https://newsroom.intel.com/news/firmware-updates-and-initial-performance-data-for-data-center-systems/, 04.02.2018.

Sherr, I. 2012. *Apple Makes a Wrong Turn as Users Blast Map Switch – WSJ*, http://www.wsj.com/articles/SB10000872396390443890304578008712527187512, 01.05.2015.

Shirabad, J. S., Lethbridge, T. C., and Matwin, S. 2001. "Supporting Software Maintenance by Mining Software Update Records," *Proceedings of the IEEE International Conference on Software Maintenance*, Florence, Italy, pp. 22-31.

Shu, S. B. and Carlson K. A. 2014. "When Three Charms but Four Alarms: Identifying the Optimal Number of Claims in Persuasion Settings," *Journal of Marketing* (78:1), pp. 127-139.

Simon, H. 1959. "Theories of Decision-Making in Economics and Behavioral Science," *The American Economic Review* (49:3), pp. 253-283.

Sommerville, I. 2010. *Software Engineering: International Version* (9th ed.), Boston, MA: Pearson.

Steinbart, P. J., Keith, M. J., and Babb, J. 2016. "Examining the Continuance of Secure Behavior: A Longitudinal Field Study of Mobile Device Authentication," *Information Systems Research* (27:2), pp. 219-239.

Sun, H. 2012. "Understanding User Revisions When Using Information System Features: Adaptive System Use and Triggers," *MIS Quarterly* (36:2), pp. 453-478.

Suri, R., and Monroe, K. B. 2003. "The Effects of Time Constraints on Consumers' Judgments of Prices and Products," *Journal of Consumer Research* (30:1), pp. 92-104.

Svahnberg, M., Gorschek, T., Feldt, R., Torkar, R., Saleem, S. B., and Shafique, M. U. 2010. "A Systematic Review on Strategic Release Planning Models," *Information and Software Technology* (52:3), pp. 237-248.

Sykes, E. R. 2011. "Interruptions in the workplace: A case study to reduce their effects," *International Journal of Information Management* (31:4), pp. 385-394.

Szajna, B. 1996. "Empirical Evaluation of the Revised Technology Acceptance Model," *Management Science* (42:1), pp. 85-92.

Tallon, P., and Pinsonneault, A. 2011. "Competing Perspectives on the Link Between Strategic Information Technology Alignment and Organizational Agility: Insights From a Mediation Model," *MIS Quarterly* (35:2), pp. 463-486.

Thaler, R. 1979. "Towards a Positive Theory of Consumer Choice," *Journal of Economic Behavior and Organization* (l), pp. 39-60.

Thaler, R. H. 2016. *Misbehaving: The Making of Behavioral Economics,* New York: W. W. Norton & Company.

Thaler, R. H., and Sunstein, C. R. 2008. *Nudge: Improving Decisions About Health, Wealth, and Happiness,* New Haven: Yale University Press.

TinyMCE. 2015. *TinyMCE – Home,* http://www.tinymce.com/, 01.05.2015.

Trope, Y., and Liberman, N. 2003. "Temporal Construal," *Psychological Review* (110:3), pp. 403-421.

Tversky, A., and Kahneman, D. 1973. "Availability: A Heuristic for Judging Frequency and Probability," *Cognitive Psychology* (5:2), pp. 207-232.

Tversky, A., and Kahneman, D. 1974. "Judgment Under Uncertainty: Heuristics and Biases," *Science* (185:4157), pp. 1124-1131.

Tversky, A., and Kahneman, D. 1992. "Advances in Prospect Theory: Cumulative Representation of Uncertainty," *Journal of Risk and Uncertainty* (5:4), pp. 297-323.

Tyre, M. J., and Orlikowski, W. J. 1994. "Windows of Opportunity: Temporal Patterns of Technological Adaptation in Organizations," *Organization Science* (5:1), pp. 98-118.

Van der Heijden, A. H. C. 1992. *Selective Attention in Vision.* New York: Routledge.

Veit, D., Clemons, E., Benlian, A., Buxmann, P., Hess, T., Kundisch, D., Leimeister, J. M., Loos, P., and Spann, M. 2014. "Business Models - An Information Systems Research Agenda," *Business & Information Systems Engineering* (6:1), pp. 45-53.

Venkatesh, V. 2000. "Determinants of Perceived Ease of Use: Integrating Control, Intrinsic Motivation, and Emotion into the Technology Acceptance Model," *Information Systems Research* (11:4), pp. 342-365.

Venkatesh, V., and Davis, F. D. 1996. "A Model of the Antecedents of Perceived Ease of Use: Development and Test," *Decision Sciences* (27:3), pp. 451-481.

Venkatesh, V., and Davis, F. D. 2000. "A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies," *Management Science* (46:2), pp. 186-204.

Venkatesh, V., Brown, S. A., and Bala, H. 2013. "Bridging the Qualitative-Quantitative Divide: Guidelines for Conducting Mixed Methods Research in Information Systems," *MIS Quarterly* (37:1), pp. 21-54.

Venkatesh, V., Morris, M. G., Davis, G. B., and Davis, F. D. 2003. "User Acceptance of Information Technology: Toward a Unified View," *MIS Quarterly* (27:3), pp. 425-478.

Venkatesh, V., Thong, J. Y., and Xu, X. 2012. "Consumer Acceptance and Use of Information Technology: Extending the Unified Theory of Acceptance and Use of Technology," *MIS Quarterly* (36:1), pp. 157-178.

Versionone 2017. *11th Annual State of Agile Report*, http://stateofagile.versionone.com/, 11.11.2018.

Vetter, J., Benlian, A. and Hess, T. 2011. "Setting Targets Right! How Non-Rational Biases Affect the Risk Preference of IT-Outsourcing Decision Makers-An Empirical Investigation," *Proceedings of 19th the European Conference of Information Systems*, Helsinki, Finnland.

Vidgen, R., and Wang, X. 2009. "Coevolving Systems and the Organization of Agile Software Development," *Information Systems Research* (20:3), pp. 355-376.

Voss, J. F., Greene, T. R., Post, T. A., and Penner, B. C. 1983. "Problem-Solving Skill in the Social Sciences," in *The Psychology of Learning and Motivation, Vol. 17,* G.H. Bower (ed.), New York, NY: Academic Press, pp. 165-213.

Wan, E. W., and Agrawal, N. 2011. “Carryover Effects of Self-Control on Decision Making: A Construal-Level Perspective,” *Journal of Consumer Research* (38:1), pp. 199-214.

Weinmann, M., Schneider, C., and vom Brocke, J. 2016. “Digital Nudging,” *Business & Information Systems Engineering* (58:6), pp. 433-436.

Weyns, D., Michalik, B., Helleboogh, A., and Boucke, N. 2011. “An Architectural Approach to Support Online Updates of Software Product Lines,” *Proceedings of the 9th Working IEEE/IFIP Conference on Software Architecture*, Boulder, CO, pp. 204-213.

Weyns, D., Michalik, B., Helleboogh, A., and Boucke, N. 2011. “An Architectural Approach to Support Online Updates of Software Product Lines,” *WICSA 2011*, Boulder, CO, pp. 204-213.

Yin, D., Bond, S. D. and Zhang, H. 2014. “Anxious or Angry? Effects of Discrete Emotions on the Perceived Helpfulness of Online Reviews,” *MIS Quarterly* (38:2), pp. 539-560.

Zhang, Y., Feick, L., and Mittal, V. 2013. “How Males and Females Differ in Their Likelihood of Transmitting Negative Word of Mouth,” *Journal of Consumer Research* (40:6), pp. 1097-1108.

Zhu, M., Billeter, D. M., and Inman, J. J. 2012. “The Double-Edged Sword of Signaling Effectiveness: When Salient Cues Curb Postpurchase Consumption,” *Journal of Marketing Research* (49:1), pp. 26-38.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbstständig angefertigt habe. Sämtliche aus fremden Quellen direkt und indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher nicht zu Prüfungszwecken verwendet und noch nicht veröffentlicht.

Tillmann Grupp

Appendix

Appendix 4.A

Continuance Intention (7-point Likert scale adapted and modified from Bhattacherjee 2001)

CI1 I intend to continue using eWrite rather than discontinue its use.

CI2 My intentions are to continue using eWrite than use any alternative means.

CI3 If I could, I would like to discontinue my use of eWrite. (reverse coded)

Satisfaction (7-point Likert scale adapted and modified from Kim and Son 2009)

SAT1 I am content with the features provided by the word-processing program eWrite.

SAT2 I am satisfied with the features provided by the word-processing program eWrite.

SAT3 What I get from using the word-processing program eWrite meets what I expect for this type of programs.

Perceived Usefulness (7-point Likert scale adapted and modified from Kim and Son 2009)

PU1 Using eWrite enhanced my effectiveness in completing the task.

PU2 Using eWrite enhanced my productivity in completing the task.

PU3 Using eWrite improved my performance in completing the task.

Perceived Ease of Use (7-point Likert scale adapted and modified from Kim and Son 2009)

PEoU1 Interacting with eWrite does not require a lot of mental effort.

PEoU2 I find it easy to get eWrite to do what I want it to do.

PEoU3 I find eWrite easy to use.

Disconfirmation (7-point Likert scale adapted and modified from Bhattacherjee 2001)

DISC1 My experience with using eWrite was better than what I expected.

DISC2 The service level provided by eWrite was better than what I expected.

DISC3 Overall, most of my expectations from using eWrite were confirmed.

Appendix 4.B

Control Questions (Self developed)

1) What was the experimental task? (To format the entire text; to format the text as appealingly as possible)
2) How many updates did you receive during the experiment? (no updates; one update containing three features; three updates each containing one feature; one update containing three non-features; three updates each containing one non-feature)
3) How many features did you have at the end of completion time? (one feature; four features)

Appendix 4.C

Questions for Manipulation Check Study (Self developed)

1) As how frequent did you perceive the updates that you received during the experiment? (7-point Likert scale; 1=not frequent at all, 7=very frequent, I did not receive any updates)
2) As how helpful did you perceive the updates that you received during the experiment? (7-point Likert scale; 1=not helpful at all, 7=very helpful, I did not receive any updates)