

Eingereicht von

**Maja Dusanic**

**Merisa Pargan**

**Anel Ljutic**

**Ivan Samardzic**

**Milos Tomic**

Angefertigt am

**Institut für Wirtschaftsinformatik – Information Engineering**



Betreuer / Betreuerin

**Dr. David Rückel**

**Dr. Barbara Krumay**

**Univ.-Prof. Dr. Stefan Koch**

Partner

**Institut für Wirtschaftsinformatik – Information Engineering**

06 2019



Seminararbeit

zur Erlangung des akademischen Grades

Bachelor of Science (MSc/BSc)

im Bachelorstudium

Wirtschaftsinformatik

Evaluierung einer CRm-lösung

**Institut für Wirtschaftsinformatik – Information Engineering**

Inhaltsverzeichnis

[1. Problem 3](#_Toc5123333)

[1.1. Einleitung 3](#_Toc5123334)

[1.2. Problembeschreibung und -nachweis 3](#_Toc5123335)

[1.3. Ziele 3](#_Toc5123336)

[1.4. Forschungsfragen 3](#_Toc5123337)

[2. Problemlösungsweg 3](#_Toc5123338)

[2.1. Customer-Relationship-Management 3](#_Toc5123339)

[2.2. Vorgehensmodell 4](#_Toc5123340)

[2.3. Phase 4](#_Toc5123341)

[2.4. Phase 4](#_Toc5123342)

[3. Ergebnisse 4](#_Toc5123343)

[4. Diskussion 4](#_Toc5123344)

[5. Tabellenverzeichnis 4](#_Toc5123345)

[6. Abbildungsverzeichnis 4](#_Toc5123346)

[7. Literaturverzeichnis 4](#_Toc5123347)

[8. Anhang 4](#_Toc5123348)

# Problem

## Einleitung

## Problembeschreibung und -nachweis

## Ziele

## Forschungsfragen

# Problemlösungsweg

## Customer-Relationship-Management

In den letzten Jahren wurde der Begriff CRM -Customer Relationship Management immer wichtiger für große sowie kleine Unternehmen. Mit der Entwicklung von CRM werden immer umfassendere Definitionen entwickelt, weshalb verschiedene Personen unter diesem Begriff eine unterschiedliche Bedeutung verstehen.   
Ein weit verbreitetes Verständnis ist die Verwendung von Wissen und Informationen, welches sich auf die Kunden bezieht, um relevante Dienstleistungen oder Produkte an Kunden zu liefern (Christopher, 2003). Eine andere Ansicht von CRM ist, dass es eine Technologielösung ist. Zur Verbesserung von soll CRM unter diesem Verständnis die separaten Datenbanken und Tools zur Automatisierung von Außendienstmitarbeitern erweitern, um die Vertriebs- und Marketingfunktionen zu überbrücken. Andere Organisationen betrachten CRM als ein Tool, das speziell für die Kundenkommunikation entwickelt wurde. Diese Aufgabe liegt ausschließlich in den Bereichen Vertrieb / Service, Call Center oder Marketing (Chen & Popovich, 2003).

Laut (Christopher, 2003) entwickelte sich CRM aus Geschäftsprozessen wie dem Beziehungsmarketing und der zunehmenden Betonung einer verbesserten Kundenbindung durch das effektive Management von Kundenbeziehungen. *Schlussendlich* soll man unter CRM eine funktionsübergreifende, kundenorientierte, technologieintegrierte Strategie für das Geschäftsprozessmanagement, die Beziehungen maximiert und die gesamte Organisation umfasst, verstehen (Chen & Popovich, 2003).

Eine CRM-Geschäftsstrategie verknüpft die Front-Office-Funktionen und Back-Office-Funktionen. Sie nutzt also Marketing, Betrieb, Vertrieb, Kundenservice, Personal, Forschung und Entwicklung, Finanzen sowie Informationstechnologie und Internet, um die Rentabilität der Kundeninteraktionen zu maximieren (Chen und Popovich, 2003).

Unternehmen, die erfolgreich CRM implementieren, haben vielerlei Vorteile. Zu einem werden die Aussichten für die Rentabilität einer Organisation langfristig verbessert (Christopher, 2003). Zum anderem steigt die Wettbewerbsfähigkeit, was sich in höheren Einnahmen und niedrigeren Betriebskosten auswirkt. Durch die effektive und effiziente Verwaltung von Kundenbeziehungen in CRM Lösungen wird die Kundenzufriedenheit und die Kundenbindung erhöht (Chen & Popovich, 2003).

Kunden profitieren von Einfachheit und Komfort für die Durchführung von Transaktionen (Chen & Popovich, 2003), sparen Zeit und Geld und bekommen bessere Informationen (Christopher, 2003).

Eine erfolgreiche Implementierung ist jedoch für viele Unternehmen schwer umsetzbar, vor allem, weil sie nicht verstehen, dass CRM eine unternehmensweite, funktionsübergreifende, kundenorientierte Neugestaltung von Geschäftsprozessen erfordert. Mögliche Risiken wie Projektversagen, unzureichende Kapitalrendite, unglückliche Kunden, Verlust des Vertrauens der Mitarbeiter und fehlende Ressourcen müssen gut durchdacht sein (Chen & Popovich, 2003).

## Open Source Software

Unter einer Open Sorce Software versteht man eine Software, dessen Quellcode öffentlich und frei verfügbar ist. Sie ist nicht konkurrenzfähig und setzt eine urheberrechtliche Lizenz ein, um private Patentansprüche sowohl von Software-Innovatoren als auch von Software-Adoptern aus dem Weg zu räumen. Die Erstehung von dieser Art von Software ist in der Regel mit Open Source Softwareprojekten verbunden, die häufig von einer Einzelperson oder eine Gruppe initiiert werden, deren Ziel es ist eine Software zu entwickeln, die ihren eigenen Bedürfnissen entspricht. Heutzutage gibt es eine Vielzahl solcher Projekte, und sie produzieren Code für viele verschiedene Zwecke. (von Krogh & von Hippel, 2006)

Die Motivationen für die Beiträge zu diesen Projekten sind ein zentrales Thema bei der Erforschung der Open Source Software. Frühe empirische Arbeiten zu diesem Thema dokumentierten eine Reihe von Motiven für die Beteiligung von Projektbeteiligten, wie Spaß, Reputationsaufbau, Lernen und den privaten Gebrauchswert der Software, die entwickelt wird. (von Krogh & von Hippel, 2006)

Vorteile und Nachteile

Die Verwendung von Open Source Software ist mit verschiedenen Vorteilen bzw. Nachteilen verbunden.

Der wichtigste Vorteil sind die geringeren Anschaffungskosten. Diese haben in den meisten Fällen einen starken Einfluss auf den Entscheidungsprozess einer Non profit Organisation, wenn es um die Kaufentscheidung zwischen einer Open Source Software und einer kommerziellen Software geht.

Der nächste Vorteil ist die Möglichkeit den Code wiederzuverwenden. Die Wiederverwendung vom Code unterstützt die folgenden Szenarien (Haefliger, von Krogh, & Spaeth, 2008):

* Die Funktionalität soll schnell integriert werden
* Es wird zeitlich und technisch unter begrenzten Ressourcen gearbeitet
* Es sollen die Entwicklungskosten gesenkt werden

Zu den Vorteilen zählt auch die größere Flexibilität bei der Änderung und Verbesserung der Codebasis. Diese kann zu einer höheren Produktivität führen.

Zu den Nachteilen gehört auf der ersten Stelle die Tatsache, dass die technische Unterstützung nicht garantiert wird. Einige Benutzer bieten zwar Hilfe bei der Erstellung von Handbüchern oder der Beantwortung von Benutzerfragen an, es gibt jedoch keine Garantie dafür, dass die Frage eines Benutzers jemals beantwortet wird, da Benutzer keinen Servicevertrag mit einem Anbieter haben können. Zu den weiteren Nachteilen gehören Sicherheitsbedenken (z.B. Sicherheit der Daten) und die fehlenden vertraglichen Beziehungen. Die Kombination dieser Nachteile kann dazu führen, dass Benutzer einer Open Source Software Zeit für die Problembehebung ihrer Software verschwenden, anstatt an der Verbesserung der Software zu arbeiten. Dies kann sich in einigen Fällen negative auf die Produktivität des Unternehmens auswirken. (Nagle, 2015)

## Make vs Buy

Bei der Entwicklung eines Informationssystems stehen Unternehmen vor einer grundlegenden Entscheidung: Ein vorentwickeltes Paket zu kaufen (möglicherweise anzupassen oder die Geschäftsprozesse des betreffenden Unternehmens an das Paket anzupassen) oder das System selbst zu erstellen. Es wurde festgestellt, dass eine Reihe von Kernfaktoren die Kauf- und Baukosten kritisch beeinflussen(Hung & Low, 2008):

* Strategie
* Rohstoff- und Wettbewerbsvorteil
* Reifegrad
* Kosten
* Anforderungen

Zu erwähnen sind auch Faktoren, die die Kauf- und Bauskoten weniger beeinflussen, dennoch aber wichtig für den Entscheidungsprozess wichtig sind: (Hung & Low, 2008):

* Risiko
* Zeit
* politische Faktoren
* laufende Unterstützung

## Non Profit Organisationen (NPO)

NPO sind Organisationen, die durch folgende Merkmale gekennzeichnet sind:

* Organisation als eine Institution
* Sind privat und kein Teil der Regierung
* Selbstverwaltung ihrer Aktivitäten und Verfahren
* Keine Verteilung vom Gewinn

Diese Organisationen müssen sich verschiedenen Herausforderungen stellen, die in ihrem organisatorischen Kontext einzigartig sind. Hiermit sind unter anderem monetäre Herausforderungen gemeint (z.B. reduzierte staatliche Mittel), der zunehmende Wettbewerb, die Herausforderung Freiwillige zu rekrutieren bzw. zu binden und die Notwendigkeit wirksamer Programme zur Kontaktaufnahme mit den Sponsoren. Diese Herausforderungen grenzen die allgemeine Wirksamkeit und Effizienz der NPO ein. (Rathi & Given, 2017)

NPO Technologielandschaft

Um sich den oben genannten Herausforderungen zu stellen benutzen die NPO verschiedene Technologien. Nach dem Zusammenfügen dieser Technologien ergeben sich folgende allgemeine Kategorien: (Rathi & Given, 2017)

* Low-cost/no-cost Cloud basierte Dienste
* Öffentliche Webseiten
* Kommerzielle Produktivitätssoftware
* Private Webseiten
* Open-source content management Software
* Low-cost/no-cost Produktivitätssoftware
* Enterprise content management Software

## Anforderungskatalog

Unter Systemanforderungen versteht man Anforderungen an das System und die Umgebung, in der sich das System befindet. Durch die Beschreibung der genauen Anforderungen für das Systemverhalten und die Architektur wird die Grundlage für die Entwicklung dieses Systems gelegt. Die sich daraus ergebenden Anforderungsspezifikationen stellen dabei die Grundlage für die Analyse der Anforderungen dar. Sie geben an, dass sie tatsächlich das sind, was die Interessengruppen wollen, definieren, was Konstrukteure schaffen müssen, und vergewissern sich, dass sie dies bei der Lieferung der Ergebnisse richtig gemacht haben.

Es gibt viele Ansätze bzw. Techniken, die man verwenden kann, um Anforderungen einer Software zu spezifizieren, analysieren und validieren. Einer dieser Ansätze ist IEEE 830-1998-Standard. Dieser schlägt eine Vorlage vor, die mehrere Anforderungsartefakte identifiziert. Die wichtigsten werden in einem Abschnitt mit der Bezeichnung "Spezifische Anforderungen" beschrieben. Diese Artefakte umfassen: "Externe Schnittstellen" für Benutzer, Hardware, Software und / oder Kommunikation, "Funktionen" der Software, "Einschränkungen" für Software; "Design Constraints" usw. (McIlroy & Stanton, 2012; Navarro, Letelier, Mocholi, & Ramos, 2018)

### Anforderungsanalyse

Im Folgenden wird eine Anforderungsanalyse der Interview-Ergebnisse durchgeführt, die durch die Befragung folgender Personen zu Stande gekommen ist: Frau Angela Dorfer vom Institut Information Engineering (JKU), Frau Dr. Barbara Krumay vom Institut Information Engineering (JKU), Herr Dr. David Christoph Rückel vom Institut Information Engineering (JKU) und Herr Michael Holoubek vom Institut Information Engineering (JKU).

Als Ansatz für die Anforderungsanalyse wird IEEE 830-1998-Standard verwendet. Es werden dabei nicht alle Punkte durchgenommen, weil die Ergebnisse der Interviews nicht alle Punkte behandeln. (Elliott, Sr. & Allen, 2013; Standard, 2010)

#### Zweck

Zweck des Programms ist es grundlegende Funktionen einer CRM-Lösung für Kontakt- bzw. Eventmanagement zu bieten. Dies beinhaltet u.a. Informationen über Kunden zu sammeln, Einladungen bezüglich verschiedener Events herauszuschicken usw.

#### Allgemeine Beschreibung

Im weiteren Verlauf werden die allgemeinen Eigenschaften des Programms beschrieben.

#### *Allgemeine Funktionalität*

Die allgemeine Funktionalität des Programms besteht darin Informationen über Kunden zu sammeln, Kontakte zu organisieren, Einladungen herauszuschicken und Daten in gewünschter Form zu darstellen.

#### *System und Hardware Einschränkungen*

* Das Programm soll auf folgenden operativen Systemen lauffähig sein: Windows, Linux und OSX (need to have)
* Es sollen keine cloudbasiertes System sein (need to have)
* Das Programm soll auf mobilen Endgeräten lauffähig sein (nice to have)

#### *Rechtliche Rahmenbedingungen*

* Bezüglich der Benutzung sollen keine Lizenzprobleme auftreten (need to have)

#### *Sonstige Einschränkungen*

* Das Programm soll Open Source sein (need to have)
* Das Programm soll ausbaubar sein (need to have)

#### *Benutzerzielgruppen*

Insgesamt sollen die folgenden Rollen von Benutzern existieren:

* Superadministrator
* Administrator
* Editor (Studentischer Mitarbeiter)
* Studentische Hilfskraft

#### Spezifische Anforderungen

#### Im Folgenden werden die Funktionalitäten, welche das Programm haben soll, behandelt.

***Funktionale Anforderungen***

* Funktionelle Anforderung 1

Das Programm soll in der Lage sein, Kontakte zu hinzufügen. Diesbezüglich werden folgende Informationen eingefügt: „Titel“, „Vorname“, „Nachname“, „Email“, „Nummer“, „Adresse“, „DSGVO“

-need to have

* Funktionelle Anforderung 2

Das Programm soll in der Lage sein, doppelte Kontakte zu erkennen.

-nice to have

* Funktionelle Anforderung 3

Das Programm soll in der Lage sein, alle Kontakte auf einer Seite anzeigen zu können.

-nice to have

* Funktionelle Anforderung 4

Das Programm soll in der Lage sein, Zusatzinformationen zu den Kontakten bzw. Einladungen zu erstellen.

-need to have

* Funktionelle Anforderung 5

Das Programm soll in der Lage sein, die Informationen über die Kunden schnell ändern zu können. Bzw. man sollte nicht mehrere Fenster öffnen müssen um dies zu tun.

-need to have

* Funktionelle Anforderung 6

Das Programm soll in der Lage sein, Event zu hinzufügen. Diesbezüglich werden folgende Informationen eingefügt: „Name“, „Datum“, „Ort“, „Veranstalter“, „Kommentar“.

-need to have

* Funktionelle Anforderung 7

Das Programm soll in der Lage sein, bei dem hinzufügen von Events einen Kalender zu vorlegen, um das Datum leichter zu selektieren.

* Funktionelle Anforderung 8

Das Programm soll in der Lage sein, Mails bzw. Einladungen zu verschicken.

-need to have

* Funktionelle Anforderung 9

Das Programm soll in der Lage sein, ein Template (z.B. JKU-Daten) bei der Erstellung der Mails zu bieten. Die Auswahl vom Template ergibt sich aus den benutzten Templates der letzten sechs Monate.

* Funktionelle Anforderung 9

Das Programm soll in der Lage sein, ein default Template (z.B. JKU-Daten) bei der Erstellung der Mails zu zeigen.

-nice to have

* Funktionelle Anforderung 9

Das Programm soll in der Lage sein, Events bei dem Verschicken von den Einladungen automatisch in die Facebook-Gruppe zu posten.

-nice to have

* Funktionelle Anforderung 10

Das Programm soll in der Lage sein, vollautomatisiert Einladungen zu verschicken.

-need to have

* Funktionelle Anforderung 11

Das Programm soll in der Lage sein, Serienbriefe (Massenmail) zu verschicken.

-need to have

* Funktionelle Anforderung 12

Das Programm soll in der Lage sein bei jedem Kunden einen Status erstellen zu können, wo man sieht, ob der Kunde die Einladung zu einem Event akzeptiert bzw. abgelehnt hat oder nicht geantwortet hat. Diesen Status soll man einfach umändern können.

-need to have

* Funktionelle Anforderung 13

Das Programm soll in der Lage sein, die Kunden nach dem Status der Einladung (bzw. Akzeptiert/Abgelehnt) zu sortieren.

-need to have

* Funktionelle Anforderung 14

Das Programm soll in der Lage sein, die Kunden nach dem Status der Einladung (bzw. Akzeptiert/Abgelehnt) zu exportieren.

-need to have

* Funktionelle Anforderung 15

Das Programm soll in der Lage sein, die Kunden anhand vom Status der Einladung (bzw. Akzeptiert/Abgelehnt) zu zählen.

-need to have

* Funktionelle Anforderung 16

Das Programm soll in der Lage sein, Kunden, die nicht auf eine Einladung geantwortet haben, einen „Reminder“ zu schicken.

-nice to have

* Funktionelle Anforderung 17

Das Programm soll in der Lage sein, Daten in verschiedenen Formaten (z.B. CSV, Excel, PDF, Word, JPG) zu exportieren.

-need to have

* Funktionelle Anforderung 18

Das Programm soll in der Lage sein, die Daten in einer übersichtlichen Form (z.B. Grafisch) darstellen zu können

-need to have

* Funktionelle Anforderung 19

Das Programm soll eine Suchmöglichkeit enthalten.

-need to have

* Funktionelle Anforderung 20

Das Programm soll in der Lage sein, die Suche zu spezifizieren (z.B. Suche nur in den Kontakten und nicht in den Mails).

-nice to have

* Funktionelle Anforderung 21

Das Programm soll in der Lage sein, eine Benachrichtigung anzuzeigen, wenn eine Mail versendet wird bzw. wenn eine neue Mail kommt.

-nice to have

* Funktionelle Anforderung 22

Das Programm soll in der Lage sein, einen Kalender bei der Terminverwaltung vorzuzeigen.

-nice to have

* Funktionelle Anforderung 23

Das Programm soll in der Lage sein, sich an den letzten Login zu erinnern bzw. die Daten sollte man nicht wieder beim Login eingeben müssen, weil sie automatisch erstellt werden.

-nice to have

***Performance Anforderungen***

* Performance Anforderung 1

Das Programm soll in der Lage sein, bis zu 500 Kunden zu verwalten.

-need to have

***Logische Datenbankanforderungen***

* Zugriffsrechte

Der Superadministrator soll für den reibungslosen Betrieb des Systems zuständig sein, sowie für die Verteilung der Rollen. Er soll in der Lage sein Benutzer zu löschen, Events zu erstellen, Kontakte zu erstellen und alle weiteren Funktionalitäten des Programms benutzen zu können.

Der Administrator erhält die gleichen Rechte wie der Superadministrator, außer der Möglichkeit die Einstellungen vom Programm zu ändern. Bzw. er kann Rollen verteilen, Benutzer löschen, Events erstellen usw.

Der Editor (studentischer Mitarbeiter) erhält die gleichen Rechte wie der Administrator, außer der Möglichkeit Rollen zu verteilen, Benutzer zu löschen und Events zu erstellen. Bzw. er kann Kontakte erstellen, Einladungen verschicken usw.

Die studentische Hilfskraft erhält nur Leserechte im Programm. Bzw. sie erhält Einsicht in die Kontaktelisten, die verschickten Einladungen usw.

* Zugriffshäufigkeit

Das Programm bzw. die Datenbanken sollten jederzeit verfügbar sein.

* Datenintegrität

Es sollen zwei Datenbanken existieren, die für die verschiedenen Rollengruppen eingeteilt werden. (nice to have)

## Vorgehensmodell

Unter einem Vorgehensmodell wird eine Folge von Aktivitäten und Ergebnissen verstanden, die zur erfolgreichen Durchführung eines Projekts erforderlich sind. Ergänzt wird ein Vorgehensmodell durch Prinzipien, Methoden, Techniken und Werkzeuge. Hauptziel eines Vorgehensmodells ist es, eine komplexe Problemstellung in leichter lösbare Teilaufgaben zu reduzieren und diese phasenweise zu strukturieren, um somit schneller und verlässlicher die angestrebten Projektziele zu erreichen.

Des Weiteren erlauben Vorgehensmodelle Erfahrungswissen zu explizieren und so eine weitgehende Personenunabhängigkeit zu erreichen. Die Modelle dürfen aber nicht zum Selbstzweck werden und sollten nicht im Widerspruch zu Intuition und Kreativität stehen. Vielmehr sind sie als Leitfaden für die Lösung praktischer Probleme zu verstehen.

## Planung der Umsetzung

Im Folgendem werden die Schritte für die Planung der Umsetzung vorgestellt.

Auswahl der CRM-Software

In diesem Schritt wird die CRM-Software ausgewählt. Die Auswahl ist von den oben angeführten Anforderungen abhängig bzw. es wird geschaut wie sehr die CRM-Software diese Anforderungen erfüllt.

Set-Up

In diesem Schritt mach man sich mit der CRM-Software, durch das Durchlesen vom Benutzerhandbuch und den verschiedenen Tutorials, vertraut.

Konfiguration nach den Anforderungen

In diesem Schritt wird die CRM-Software nach den need-to-have Anforderungen angepasst. Wenn möglich werden auch die nice-to-have Anforderungen angepasst.

Design

In diesem Schritt wird das Design der CRM-Software (wenn möglich) angepasst, um wichtige Funktionen leichter auffindbar zu machen.

Testen

In diesem Schritt werden alle Funktionen der CRM-Software getestet.

Einschulung

In diesem Schritt werden die Zielpersonen eingeschult.

Installation

In diesem Schritt wird die CRM-Software am Zielort installiert. Hier werden auch die benötigten Daten (Kontaktdaten usw.) integriert.

Erweiterung

In diesem Schritt wird (wenn möglich) die CRM-Software nach Wunsch erweitert. (Al-Mudimigh, Zairi, & Al-Mashari, 2001; Ehie & Madsen, 2005; Parr & Shanks, 2000; Peslak & Subramanian, 2008; Waring & Maddocks, 2005)

## Phase

## Phase

# Ergebnisse

# Diskussion

# Tabellenverzeichnis

# Abbildungsverzeichnis

# Literaturverzeichnis

Al-Mudimigh, A., Zairi, M., & Al-Mashari, M. (2001). ERP software implementation: An integrative framework. *European Journal of Information Systems*, *10*(4), 216–226. https://doi.org/10.1057/palgrave.ejis.3000406

Chen, I. J., & Popovich, K. (2003). Understanding customer relationship management (CRM): People, process and technology. *Business Process Management Journal*, *9*(5), 672–688. https://doi.org/10.1108/14637150310496758

Christopher, B. (2003). Strategic issues in customer relationship management (CRM) implementation. *Business Process Management Journal*, *9*(5), 592. Retrieved from http://proquest.umi.com/pqdweb?did=436780121&Fmt=7&clientId=43922&RQT=309&VName=PQD

Ehie, I. C., & Madsen, M. (2005). Identifying critical issues in enterprise resource planning (ERP) implementation. *Computers in Industry*, *56*(6), 545–557. https://doi.org/10.1016/j.compind.2005.02.006

Elliott, Sr., R. a., & Allen, E. B. (2013). A Methodology for Creating an IEEE Standard 830-1998 Software Requirements Specification Document. *Journal of Computing Sciences in Colleges*, *29*, 123–131. Retrieved from http://dl.acm.org/citation.cfm?id=2535418.2535437%5Cnhttp://dl.acm.org/ft\_gateway.cfm?id=2535437&type=pdf

Haefliger, S., von Krogh, G., & Spaeth, S. (2008). Code Reuse in Open Source Software. *Management Science*, *54*(1), 180–193. https://doi.org/10.1287/mnsc.1070.0748

Hung, P., & Low, G. C. (2008). Factors affecting the buy vs build decision in large Australian organisations. *Journal of Information Technology*, *23*(2), 118–131. https://doi.org/10.1057/palgrave.jit.2000098

McIlroy, R. C., & Stanton, N. A. (2012). Specifying the requirements for requirements specification: The case for Work Domain and Worker Competencies Analyses. *Theoretical Issues in Ergonomics Science*, *13*(4), 450–471. https://doi.org/10.1080/1463922X.2010.539287

Nagle, F. (2015). Open Source Software and Firm Productivity. *Ssrn*, (March 2019). https://doi.org/10.2139/ssrn.2559957

Navarro, E., Letelier, P., Mocholi, J. A., & Ramos, I. (2018). A Metamodeling Approach for Requirements Specification. *Journal of Computer Information Systems*, *46*(5), 67–77. https://doi.org/10.1080/08874417.2006.11645925

Parr, A., & Shanks, G. (2000). A model of ERP project implementation. *Journal of Information Technology*, *15*(4), 289–303. https://doi.org/10.1080/02683960010009051

Peslak, A., & Subramanian, G. (2008). The phases of ERP software implementation and maintenance: A model for predicting preferred ERP use. *Journal of Computer*, *48*(2), 25–33. https://doi.org/10.1080/08874417.2008.11646006

Rathi, D., & Given, L. M. (2017). Non-profit organizations’ use of tools and technologies for knowledge management: a comparative study. *Journal of Knowledge Management*, *21*(4), 718–740. https://doi.org/10.1108/JKM-06-2016-0229

Standard, I. (2010). Systems and software engineering — Vocabulary INTERNATIONAL STANDARD ISO / IEC / IEEE 24765:2010, *2010*.

von Krogh, G., & von Hippel, E. (2006). The Promise of Research on Open Source Software. *Management Science*, *52*(7), 975–983. https://doi.org/10.1287/mnsc.1060.0560

Waring, T., & Maddocks, P. (2005). Open Source Software implementation in the UK public sector: Evidence from the field and implications for the future. *International Journal of Information Management*, *25*(5), 411–428. https://doi.org/10.1016/j.ijinfomgt.2005.06.002

# Anhang