# Abkürzung:

DSR Design Science Research

SD Schließdienst

MA Mitarbeiter

# 3. Methodik

## 3.1 Vorüberlegungen und Vorgehensweisen

Die Vorgehensweise bei der Entwicklung vom Software-Lösungsansatz in Sicherheitsdienst orientiert sich an Design Science Research (DSR).

Die ganze DSR Vorgehen bezieht sich hauptsächlich auf drei Zyklen, nämlich Relevanz Zyklus (auch Environmental Cycle genannt), Design Zyklus und Rigor Zyklus (vgl. Hevner, A. R. 2007, S. 88) (siehe auch Abbildung 3.1).

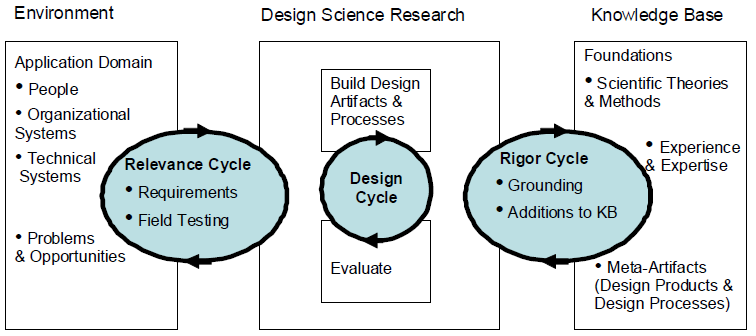


Abbildung 3.1: DSR-Zyklen (Hevner, A. R. 2007, S. 88)

Der Relevanz-zyklus enthält drei Phasen von DSR, das heißt die Problemstellung und Motivation, die Zielsetzung sowie die Evaluation im der Softwareentwicklung. Es gibt Probleme in Anwendungsdomäne, die mit Personal, Organisationssystem und technischem System eng verbunden sind. Aufgrund solcher Probleme wird man motiviert, eine neue Lösungsmöglichkeit zu finden und anschließend Ziele festzustellen. Außerdem sind die Ziele ausgehend von Anwendungsdomäne und Literaturen während einer Evaluation nach der Entwicklung zu berücksichtigen, (vgl. Hevner 2007, S. 6).

Darüber hinaus enthält der Design Zyklus die Artefakt Beschreibung, die den Entwurf, die Entwicklung und Demonstration der Artefakt umfasst (vgl. Hevner, A. R. 2007, S. 88).

Um die Ziele von Software vollständig zu erreichen, spielt der Rigor Zyklus eine wichtige Rolle. Dafür spricht, dass der Rigor Zyklus wie eine dynamisch wissenschaftliche Datenbank funktioniert. Einerseits werden die schon existierte Verfahren und Know-how von Forschungsergebnisse sowie die Artefakte von anderen Applikationen aus diese Datenbank herausgefunden. Andererseits führt die Erweiterung und neue Verfahren aus der neue softwarebasierte Lösung in den gleichen Bereich in die wissenschaftliche Datenbank ein. Die Datenbank bietet sowohl Grundlagen für Zielsetzung sowie Vergleichsbeispiele als auch Erweiterungsmöglichkeiten für eine weitere Diskussion an (vgl. Hevner et al. 2004, S. 94).

Demnach erfolgt eine methodische Aufteilung in folgenden sechs Phasen innerhalb den drei Zyklen (vgl. Peffers, Tuunanen et al. 2007, S. 67)(siehe auch Abbildung 3.2):

Abbildung 3.2: DSR-Phase innerhalb der Zyklen (vgl. Peffers, Tuunanen et al. 2007, S. 67)

Der DSR fängt mit Problemidentifikation an. Während dieser Phase wird die problematische Situation von Sicherheitsdienst in UMGf GmbH identifiziert. Daher wird die Motivation von der technischen Lösung aufgrund der Probleme weiterhin adressiert.

Anschließend lassen sich die quantitative und qualitative Ziele von der Softwareentwicklung aus Literatur in Verbindung mit Motivation konkret herleiten und einstellen (vgl. Vom Brocke, Buddendick (Ed.) 2006, S. 582). Quantitative Ziele bedeutet, in welchen Bereichen das neue System besser als das jetzige Schließdienstsystem im Szenario funktioniert, während qualitativer Ziele die Verbesserung der jetzige Schließdienst (Ist-Situation) in bisher noch nicht genannten Bereichen betonnen(vgl. Peffers, Tuunanen et al. 2007, S. 67). Die dritte Phase orientiert sich auf Artefakt Beschreibung, die eine der Hauptteile von der DSR-Methode entspricht. Innerhalb dieser Phase wird vor allem einen Entwurf der Lösung ausgehend von den Zielstellungen erläutert. In diesem Entwurf werden spezifische und Programmierung-relevante Anforderungen erstellt (vgl. Hofmann, Robert et al. 2013, S. 25). Dann wird die softwarebasierte Lösung nach Anforderungen der Entwurf entwickelt (vgl. Gregor, Hevner 2013, S. 350). Anschließend folgt eine Demonstration von wichtigen Screenshots, die durch die Applikation erzeugt.

Danach folgt eine Evaluation von anfertigten Prototyp je nach festgesetzte quantitativen Zielen. Ziel dieser Phase ist zu entscheiden, ob die Entwicklung der softwarebasierten Lösung fortsetzen kann. Falls der Prototyp die quantitativen Ziele vollständig erreicht, kann der sich weiter in Diskussionsphase einführen lassen. Ansonsten muss der wieder zu die dritte Schritt rückschreiten, um die Entwurf des Artefakt zu modifizieren (vgl. Peffers, Tuunanen et al. 2007, S. 68).

Nach der Evaluation wird eine Diskussion über Interpretation des Artefakts durchgeführt. Dabei geht diese Phase um ein Rigor Zyklus. Das heißt, die schon existierte Kenntnisse mit dem neuen Verfahren von neu entwickeltem Artefakt verglichen werden. Aus der Vergleichung ergeben sich die Potenziale, Limitationen und Erweiterung von dem neuen Artefakt zu analysieren(vgl. Gregor, Hevner 2013, S. 351).

Zu guter Letzt werden die erlernte Verfahren sowie neue Beiträge zu den Bereich von CAFM-System sowie FM-System zusammengefasst (vgl. Hevner et al. 2004, S. 94).

## 3.2 Anwendung der DSR Framework

#### Anwendung in Relevanz Zyklus

Um den Prozess des Szenarios in UMGf GmbH zu modellieren, wurde ein Use-Case-Diagramm erstellt. Dafür wurden folgende Arbeitsschritte durchgeführt (siehe auch Abbildung 3.3):

Abbildung 3.3: Arbeitsschritte von Use-Case-Diagramm

Um die Ist-Arbeitsprozesse, Ist-Beziehungen zwischen Prozesse und relevante Personen von Anwendungsdomäne auszukennen, wurde zuerst eine Feldforschung durchgeführt. Umfassend waren die Besichtigung der Sicherheitsdienst in UMGf GmbH und 3 kleine Interview mit 3 Kollegen von Sicherheitsdienst (SD). Anschließend lassen sich die quantitative und qualitative Ziele von der Softwareentwicklung aus Literatur mit den Stichwörtern *FM-System* und *CAFM-System* herleiten. Ziel dieser Schritt ist, konkrete Ziele im Sinne der Soll-Prozess sowie deren Soll-Beziehungen dazwischen festzusetzen. Abschließend wurde ein Use-Case-Diagramm mit Soll-Prozesse, -Beziehungen und Akteure abgebildet.

#### Anwendung in Design Zyklus

Um die Ziele zu realisieren, wurde web-basierte Applikation als ein Artefakt entwickelt. Dafür wurden folgende Arbeitsschritte durchgeführt (siehe auch Abbildung 3.3):

Abbildung 3.4: Arbeitsschritte von Artefakt Beschreibung

Design Zyklus enthält drei Phasen. Zuerst wurde einen Entwurf von Applikation erledigt, damit nicht nur die Zielsetzung im Relevanz Zyklus sondern auch spezifische Anforderungen für Programmieren berücksichtigt wurden. Danach war ein Modeling Framework in Form des Prototyps zu repräsentieren. Zum Schluss war der anfertigte Prototyp durch die Screenshots präsentieren. Solange der Prototyp die Anforderungen von Applikation nicht vollständig erreicht, kehrt der Prototyp zu der Entwicklungsphase zurück, um die Fehler in Model oder in Quellcode zu korrigieren (vgl. Vom Brocke, Buddendick (Ed.) 2006, S. 591).

#### Anwendung in Rigor Zyklus

Aus neuem Artefakt die erweitere Wissen in Wissensdatenbank einführen

In Wissensdatenbank nach existierten Wissen suchen

**Artefakt**

**Wissens-**

**Datenbank**

Abbildung 3.5: Arbeitsschritte von Erfassung der Grundlagen und Beiträge des neuen Artefakts (vgl. Gregor, Hevner 2013, S. 350)

Innerhalb des Rigor Zyklus sind einerseits Grundlagen für Zielsetzung der Softwarebasierte Lösung aus der wissenschaftlichen Datenbank herauszufinden. Andererseits tragt die neue Softwarebasierte Lösung zur wissenschaftlichen Datenbank bei, damit die erweitere Wissen und Verfahren aus dem Artefakt in Wissensdatenbank einführt.

In dem nächsten Kapitel ergeben sich die Ergebnisse von DSR-Methode.

# Literaturverzeichnis

Alan R. Hevner, Salvatore T. March, Jinsoo Park, Sudha Ram 2004. “Design Science in Information Systems Research,” MIS quarterly (28:1), pp. 75–106.

Gregor, S., and Hevner, A. R. 2013. “Positioning and presenting design science research for maximum impact,” MIS quarterly (37:2), pp. 337–355.

Hevner, A. R. 2007. “A three cycle view of design science research,” Scandinavian journal of information systems (19:2), pp. 87–92.

Hofmann, M., Robert, K., and Frieder, J. (eds.) 2013. Entwicklung eines Software-Prototyps zur automatischen Erstellung nutzerspezifischer ETL-Dokumentation: Ein Fallbeispiel für gestaltungsorientierte, problemzentrierte Forschung.

Huber, S. 2014. Informationsintegration in dynamischen Unternehmensnetzwerken: Architektur, methode und anwendung, Wiesbaden: Springer Gabler.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. 2007. “A Design Science Research Methodology for Information Systems Research,” Journal of Management Information Systems (24:3), pp. 45–77.

Vom Brocke, J., Buddendick, C. (eds.) 2006. *Reusable conceptual models–requirements based on the design science research paradigm,* Citeseer.