HAW HAMBURG

BACHELORTHESIS
Hans Hartmann

Datenbasierte Zuordnung von Schadensfällen zu Sachverständigen in der Versicherungsbranche

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Hans Hartmann

Datenbasierte Zuordnung von Schadensfällen zu Sachverständigen in der Versicherungsbranche

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Bachelor of Science Wirtschaftsinformatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. Ulrike Steffens

Eingereicht am: 11. Oktober 2019

Hans Hartmann

Thema der Arbeit

Datenbasierte Zuordnung von Schadensfällen zu Sachverständigen in der Versicherungsbranche

Stichworte

Daten, Datenbasierter Algorithmus, Datenqualität, Prozess Automatisierung

Kurzzusammenfassung

Diese Arbeit beschäftigt sich mit der datenbasierten Zuordnung von einem Schadensfall zu einem Sachverständigen in der Versicherungsbranche. Unzählige Versicherungsunternehmen vollziehen diesen Schritt manuell. Da dies sehr zeitintesiv ist, soll dieser Prozess anhand von Daten und einem Algorithmus automatisiert werden. Die Daten, welche für diese Automatisierung benutzt werden, müssen zuerst gewonnen werden, danach integriert und dann muss die Datenqualität gesichert werden. Wenn diese drei Schritte vollzogen worden sind, kann mit den Daten gearbeitet werden. Einerseits dienen die Daten, um einen geeigneten Sachverständigen anhand der Distanz, Region oder seiner offenen Schadensfälle zu finden andererseits dienen die Daten dazu, die gefundenen Sachverständigen anhand Ihrer Erfahrungen und aktuellen Schadensfälle zu bewerten. Die bewerteten gefundenden Sachverständigen werden dem Mitarbeiter, welcher für die Zuordnung zuständig ist, angezeigt - dieser muss nur noch auf der Grundlage von den angezeigten Daten eine Entscheidung treffen.

Hans Hartmann

Title of Thesis

Data-based allocation of an expert to a given claim in the insurance industry

Keywords

Data, Data Algorithmus, Data Quality, Process automation

Abstract

This thesis is dealing with the allocation of a claim to an expert in the insurance business. Many insurance companies are allocating manually. That takes a lot of time. This is a reason to automate the process of the allocation. This process gets automated based on data and an algorithm. The data witch is used to automate the process needs to get cleaned, integrated and then the data quality need to get secured. After theses steps the data can be used to allocate an expert who fits to a claim. Experts get matched to a claim through their regions, in which they want to operate, the distance to the claim and the claims, the experts got in investigation. After finding possible experts the data is used to predict how the experts suits to a given claim. The possible experts get displayed to the employee, who is responsible for the allocation process. Now the employee just need to decide witch expert is the most suitable based on the data the employee gets displayed.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

In Versicherungsunternehmen werden tagtäglich viele Schadensfälle vielen Sachverständigen zugeordnet. Nach der Zuweisung des Schadenfalls an einen Sachverständigen macht dieser einen Termin mit dem Versicherten und guckt sich am Tag des Termins den Schaden an. Nachdem der Schaden besichtigt worden ist wird dieser vom Sachverständigen bewertet. Es gibt viele verschiedene Arten von Schadensfällen, unter anderem Leitungswasserschäden, Sturm, Hagel oder Brandschäden. Für die unterschiedlichen Arten von Schadensfällen gibt es jeweils Sachverständige, die sich in den einzelnen Sparten dieser Schadensart besonderes Wissen angeeignet haben und am besten geeignet wären diese zu bearbeiten. Jedoch zählt nicht nur die Expertise bei der Auswahl eines Sachverständigen. Ein Sachverständiger möchte effektiv arbeiten und daher möglichst kurze Wege zum Schadensfall haben. Die Sachverständigen würden am liebsten Termnine verbinden. Dadurch könnten Sie Termine kombinieren und erheblich an Zeit für Fahrten sparen.

Dies ist manuell nahezu unmöglich. Die Mitarbeiter, welche den Schadensfall einem Sachverständigen zuweisen, müssten sich alle Schadensfälle aller Sachverständigen angucken, deren Expertise herausfinden und gucken, wann und wo die Sachverständigen Termine haben. Dies ist zeitlich nicht für jeden einzelnden Schadensfall machbar. Eine Software, welche auf der Grundlage von Daten eine automatisierte Zuordnung von einem Schadensfall zu einem Sachverständigen vornimmt wäre wünschenswert. Daher ist eine datenbasierte Zuordnung eines Schadenfalles zu einem Sachverständigen das Thema meiner Bachelorarbeit.

Die Software sollte als Ergebnis eine Liste von Sachverständigen beeinhalten, welche anhand Ihrer Expertise und Ihrem Standort ausgewählt und bewertet werden. Der Mitarbeiter, welcher den Schadensfall zuweist, muss lediglich eine Entscheidung treffen. Ihm

sollen alle möglichen Sachverständigen, mit allen wichtigen Faktoren, die für den Schadensfall relevant sind, angezeigt werden und auf der Grundlage von den angezeigten Daten den Schaden zuweisen.

1.2 Aufgabe und Ziele

Ziel dieser Arbeit ist es den Prozess der Zuordnung eines Schadenfalls zu einem Sachverständigen zu automatiesieren. Ein weiteres Ziel ist es einen passenden Sachverständigen für den Schadensfall zu finden, daher sollen alle möglichen Sachverständigen bewertet werden und einen Wert erhalten, wie gut dieser zu dem Schadensfall passt.

1.3 Gliederung der Arbeit

Die Arbeit hat folgende Struktur: Im zweiten Kapitel werden die Grundlagen, welche für diese Arbeit relevant sind erläutert. Da es sich bei der Zuordnung eines Schadenfalls zu einem Sachverständigen um einen Prozess handelt, gehe ich auf die Prozessmodellierung ein. Die Auswahl eines Sachverständigen geschieht auf der Grundlage von Daten. Deswegen stelle ich die Datengewinnung, Datenintegration sowie Datenprofiling, Datenbereingung und Datenmonitoring in den Grundlagen vor.

Das dritte Kapitel beinhaltet das Konzept. Hier wird der aktuelle sowie automatisierte Prozess analysiert und modelliert. Anschließend werden die vorhandenen Daten analysiert und es wird geguckt, welche Daten vorhanden sind und welche Daten für diese Arbeit wünschenswert wären. Zudem ist in diesem Kapitel die Anforderungsanalyse der Software vorhanden. In diesem Bereich werden die User Stories, sowie die nichtfunktionalen Anforderungen vorgestellt. Der letzte Bereich des Konzeptes ist der Entwurf der Software. Hier wird auf die Architektur, die Schnittstelle, Benutzungsoberfläche und Tests eingegangen.

Das vierte Kapitel beschreibt die Datengewinnung, die Datenintegration, das Datenprofiling sowie die Datenbereingung.

Das fünfte Kapitel ist die Implementierung der Zuordnung von einem Schadensfall zu einem Sachverständigen. In diesem Kapitel wird zunächst über die allgemeinen Informa-

tionen der Software geschrieben und anschließend wird beschrieben, wie die User Stories und nicht-funktionalen Anforderungen implementiert worden sind.

Der Schluß der Bachelorarbeit bildet die Bewertung und die Zusammenfassung.

2 Grundlagen

2.1 Prozessmodellierung

Der Begriff Prozess leitet sich ursprünglich von dem lateinischen 'procedere' ab, übersetzt bedeutet dies vorgehen oder vorrücken. (Mittelstraß, 1995) Die Definition eines Prozesses ist jedoch spezifischer. Laut Schwarzer und Krcmar kann ein Prozess definiert werden, 'als die Transformation von Objekten durch vor und/oder nebengelagerte Aktivitäten eines oder mehrerer Menschen oder Maschinen in Raum und Zeit. Das Ziel ist dabei immer die Erreichung einer vorgegebenen Leistung.' (Schwarzer und Krcmar, 1995)

Schwarzer und Krcmar nennen spezifisch, dass es sich bei einem Prozess um eine Transformation von einem Objekt handelt, also dass durch spezifische Handlungen oder Aktivitäten eines Menschen oder einer Maschine eine Veränderugen des Objektes in Raum und Zeit entsteht. Das Ziel ist es eine 'vorgegeben Leistung' zu erzeugen, also einen output zu generieren.

Gaitanides definiert einen Prozess folgendermaßen: 'Prozesse sind inhaltlich abgeschlossene Erfüllungsvorgänge, die in einem logischen inneren Zusammenhang stehen. ' (Gaitanides, 1983)

Gaitanides geht speziell auf den logischen Zusammenhang der Aktivitäten ein, welche durchgeführt werden müssen, um einen inhaltlichen abgeschlossenen Erfüllungsvorgang zu erzeugen. Zudem definiert er das Ergebnis eines Prozesses, wie Schwarzer und Krcmar, als einen 'inhaltlich abgeschlossenen Erfüllungsvorgang', also eine vorgegebene Leistung.

Ein Prozess ist also eine chronologische Folge von Arbeitsschritten mit zuständigen Aktivitäten. Das Ziel eines Prozesses ist es, eine Aufgabe zu erfüllen.

Modelle besitzen verschiedene Eigenschaften bzw. Funktionen.

Eine Eigenschaft eines Modells ist die abbildende Funktion, sie dient der Darstellung eines Originals. (Hesse und Mayr, 2008) Modelle helfen visuell dabei, komplexe Abläufe zu verstehen. Ein Modell ist eine Abbildung oder Repräsentation eines Originals, jedoch kann das Original auch ein Modell sein. Dies bedeutet, dass ein Modell modelliert wird.

Des Weiteren sind Modelle vereinfachte Abbildungen. 'Modelle sollten stets vereinfachend bezüglich der eigentlichen Realität sein'. (Frank und Prasse, 1997) 'Das Modell kann nicht alle Eigenschaften bzw. Attribute des Originals aufweisen, sondern nur solche, die den jeweiligen Modellierern und/ oder Modellnutzern relevant erscheinen' (Hesse und Mayr, 2008) Eine Vereinfachung ist das Ziel einer Modellierung. Es werden nur relevante Informationen angegeben, die zum Beispiel dabei helfen einen Prozess zu verstehen.

Modelle sind zudem pragmatische Funktionen: 'Unter bestimmten Bedingungen und bzgl. einer bestimmten Fragestellung soll das Modell das Original ersetzen. Es stehen die Fragen nach dem für „wen, wann, wozu“ im Vordergrund, die durch den jeweiligen Kontext anwendungsbezogen und somit pragmatisch bestimmt werden' (Vom Brocke, 2003). Bei einem Prozessmodell in einem Logistikunternehmen ist es entscheidend zu wissen, welche Schritte von welchem Mitarbeiter chronologisch auszuführen sind (funktionaler, organisatorischer Aspekt). In einem Labor jedoch, ist es wichtig zu wissen, welche Messgeräte und welche Messwerte benutzt werden. (funktionalier, operationaler, datenorientierter Aspekt).

Modelle sind ihren Original nicht eindeutig zuordenbar. Sie erfüllen unter Einschränkungen auf bestimmte Operationen zu einem bestimmten Zeitpunkt einen ihnen zugewiesenen Zweck. 'Das Ergebnis einer Modellierung ist eine vereinfachte und pragmatische Abbildung eines Realitätsausschnittes.' (Meerkamm, 2011)(s.58)

Um Prozesse zu veranschaulichen, können Prozessmodelle eingesetzt werden. Es gibt verschiedene Prozessmodellierungssprachen, um Prozzesse zu modellieren. BPMN: Business Process Modeling Notation und EPK: Ereignisgesteuerte Prozesskette wären dabei an erster Stelle zu nennen, da dies weitverbreitete Prozessmodellierungssprachen sind. (vgl. (Meerkamm, 2011)(S.212)). Auf die ereignisgesteuerte Prozessketten werde ich noch genauer eingehen.

Eine Prozessmodellierung bietet mehrere Vorteile, zum einen erhält man durch die visuelle Darstellung ein besseres Verständnis des Prozesses, zum anderen ist die visuelle

Darstellung des Prozesses auch eine Art Dokumentation der Abläufe von Arbeitsvorgängen. Zudem werden Modelle für Spezifikationen von prozessorientierten Systemen im Rahmen der Softwarenentwicklung verwendet. (vgl. (Danesh und Kock, 2005))

Die ereignisgesteuerte Prozesskette ist eine Prozessmodellierungssprache, welche die Prozesse als gerichteten Graphen darstellt. Ereignisgesteuerte Prozessketten bestehen aus Ereignisse, Funktionen und Konnektore. (vgl. ((Meerkamm, 2011)(s.73))

- Ereignisse: Ereignisse sind Zustände. Durch Zustände werden gewisse Aktionen also Funktionen ausgelöst. Ereignisgesteuerte Prozessketten starten und enden mit mindestens einem Ereignis.
- Funktionen: Funktionen sind Aufgaben, welche nach einem Ereignis ausgeführt werden.
- Konnektoren: Zum einen werden Konnektoren benutzt, um parallele Arbeitsabläufe zu ermöglichen (AND). Zudem kann der Kontrollfluss mit Konnektoren aufgespaltet oder zusammgeführt werden. (XOR, OR)

Die ereignisgesteuerte Prozessketten wurde durch die erweiterte ereignisgesteuerte Prozessketten erweitert. Die erweiterte ereignisgesteuerte Prozessketten können zudem Rollen, technische Ressourcen, Materialen und Leistungen (Produkte) abbilden.

- Organisatorische Einheiten werden direkt einer Funktion zugeordnet. Sie dienen der Darstellung der Gliderungsstruktur eines Unternehmens. (Rollen, Gruppen, Mitarbeiter)
- Informations- und Ressourceobjekte werden als In- bzw. Output einer Funktion angegeben. Sie sind eine Abbildung eines Gegenstandes aus der realen Welt.

Die ereignisgesteuerte Prozessketten wurden gezielt für die Modellierung von Geschäftsprozessen entwickelt. Dadurch entsteht keine Zweckentfremdung, wie zum Beispiel bei den Aktivitätsdiagrammen der Unified Modelling Language. Aufgrund der simplen Syntax und der abzählbaren Anzahl von Elementen ist es leicht diese Prozessmodellierungssprache zu lernen.

2.2 Datengewinnung

Es werden Daten benötigt, um diese dann integrieren, bereinigen, profilen zu können oder einfach nur für dessen Qualität zu sorgen.

Es gibt Primär- und Sekundärdaten. Primärdaten sind spezifische Daten, die speziell für ein Projekt durch eigene Erhebungsinstrumente erhoben worden sind. Dies hat den Vorteil, dass die Daten spezifisch auf das Problem zugeschnitten sind. Sekundärdaten sind Daten, welche schon vorhanden sind. Es können zum Beispeil Daten aus Studien sein, die in der Vergangenheit durchgeführt worden sind. Die Aktualität und die mangelnde Passung zur konkreten Fragestellung ist ein Nachteil, wenn man Sekundärdaten für seine Problemstellung benutzt. Jedoch ist die Erhebung von Primärdaten erheblich teurer und zeitintesiver.

Es gibt viele verschiedene Möglichkeiten, um Informationen zu erhalten. Primärdaten können durch Befragungen, Beobachtungen oder auch durch Experimente generiert werden. Sekundärdaten können aus firmeninternen Datenquellen, zum Beispiel einer Kundendatenbank, oder auch aus firmenexternen Datenquellen, wie einer amtlichen Statistik, stammen. Bei Versicherungsunternehmen ist es möglich anhand von Rechnungen für gewisse Schadensgrupppen Informationen über die Sachbeauftragen zu erhalten. Durch eine Analyse aller Rechnungen für eine gewisse Schadensgruppe lässt sich der Durchschnitt errechnen, wie viel im Schnitt für die gewisse Schadensgruppe in Rechnung gestellt worden ist. Wenn der Durchschnitt errechnet ist, kann nach Ausreißern in der Schadensgruppe gesucht werden. Dadurch erhält man Sachverständige, welche Rechnungsbeträge in Rechnung stellen, die erheblich für dieselbe Schadensgruppe vom errechneten Durchschnitt abweichen. Diese Informationen können genutzt werden, um ein weiteres Merkmal für die Bewertung eines Sachverständigen zu erhalten.

Für die Datengewinnung gibt es viele verschiedene Möglichkeiten. Viele Informationen lassen sich auch schon aus vorhandenen Daten erschließen.

2.3 Datenintegration

Datenintegration befasst sich mit dem Zusammenstellen von Daten aus verschiedenen Datenquellen. Verschiedene Fachabteilungen in Unternehmen nutzen und speichern Daten, die für Ihre jeweilige Domäne relevant sind. Jedoch gibt es Daten, die eine Schnitt-

menge aller Fachabteilungen aufweisen. Dieses Daten werden oft redundant in verschiedenen Datenquellen abgelegt. Das Marketing benötigt zum Beispiel für Mailingaktionen die Kundenadresse, ebenso die Kundenbetreung, die zum Beispiel Stammkunden mit einer Sonderaktion belohnen möchten. Die Logistik, die für die Lieferung der Ware zuständig ist und die Buchhaltung müssen ebenso die Kundenadresse kennen. Wenn Fachabteilungen nun jeweils eine isolierte Datenquelle benutzen, die nicht kontinuierlich miteinander synchronisiert werden, entstehen nicht nur redundate, sondern auch widersprüchliche Daten, die im Laufe der Zeit immer weiter auseinanderdriften und eigene Strukturen, Kodierungen, Formate und semantische Bedeutungen haben können. (vgl. (Rossak, 2013)(chap.2))

Datenintegration wird oft mit Data Warehouse verglichen, jedoch befasst sich die Datenintegration auch zum Beispiel mit der Migration von Daten aus Altsystemen, Replikation und Synchronisation von Daten in redundaten Systemen, dem Datenqualitätsmanagement oder dem Master Data Management. Die Hauptaufgabe in all diesen Bereichen ist die Konsolidierung der Daten, diese sollen einheitlich, vollständig und korrekt verfügbar sein. (vgl. (Rossak, 2013)(chap.2))

Die Konsolidierung kann in zwei Teilbereiche aufgeteilt werden. Der erste Teilbereich ist die Konsolidierung von Daten im operativen Bereich und der zweite ist die Konsolidierung von Daten im analytischen Bereich.

Im operativen Bereich wird die Datenintegration auf operative, in verteilten Datenquellen gespeicherte, Daten angewendet, die in einer Datenquelle vereinigt werden sollen. Ein gutes Beispiel hierfür wäre eine Fusionierung von zwei verschiedenen Unternehmen. Ein weiteres Beispiel ist die parallele Benutzung von einem Alt- und einem Neusystem. Durch die Parallelbenutzung kann es vorkommen, dass Daten an verschiedenen Stellen bearbeitet werden und dadurch die Gefahr der Datenkonsistenz besteht, da die Daten nicht über alle Systeme hinweg bearbeitet worden sind. Deshalb sollte bei der Einführung von Neusystemen direkt abgeklärt werden, wann das Alt- System abgeschaltet wird und wie mit den Daten aus dem Altsystem verfahren wird.

'Im analytischen Bereich steht die Auswertung von Daten im Vordergrund, mit dem Ziel, einen hohen Grad an Wissen aus den zur Verfügung stehenden Daten zu generieren und diese so ausfzubereiten und darzustellen, dass es schnellstmöglichst erfasst und für operative und strategische Entscheidungen genutzt werden kann'. (Rossak, 2013)(S.18)

Die Priorität im analytischen Bereich liegt darin, die Daten so zu transformieren, dass nützliches Wissen aus den Daten generiert werden kann. (vgl. (Rossak, 2013)(chap.2)) 'Um dieses Ziel zu erreichen, müssen Daten aus verschiedenen Quellen aufbereitet, angereichert, integriert und dargestellt werden.' (Rossak, 2013)(s.21) Für Analysen wird in der Regel ein Data Warehouse erstellt. Die in einem Data Warehouse gespeicherten, aufbereiteten Daten können nun mit Analysemethoden, wie Online Analytical Processing oder Data Mining, analysiert werden. Neben IST-Analysen, kann man die aufbereiten Datenbestände auch nutzen, um ein Forecasting durchzuführen, hierfür werden historische Daten ausgewertet und auf Grundlage dieser wird versucht zukünftige Entscheidunge vorherzusagen. Forecasting wird häufig für die Planung und Bereitstellung von neuen Produkten benutzt. (vgl. (Rossak, 2013)(chap.2)) Des Weiteren gibt es im analytischen Bereich noch Real Time Analysen. Bei dieser Analyse werden in Echtzeit Daten analysiert, so kann zum Beispiel ein Online Shop Inhaber in Realtime erkennen, wie stark ein gewisses Produkt nachgefragt wird. Dadurch ist es möglich, den Warenfluss in den Lieferketten in Echtzeit zu steuern. Zudem sind Realtime Analysen in der Qualitätskontrolle eine anwendungsbeliebte Methode So können durch Kontrollen von Produktproben eventuelle Qualitätsabweichungen in nahezu Echtzeit festgestellt werden und entsprechend schnell eine Reaktion erfolgen. (vgl. (Rossak, 2013)(chap.2))

Heterogenität (griechisch für verschiedenartig) ist oft das Hauptproblem der Datenintegration. In der Praxis wird versucht, Heterogenität durch Standards zu begrenzen oder gar zu vermeiden. Heterogenität wird in vier verschiedene Arten unterteilt, die sich gegenseitig bedingen. (vgl. (Naumann und Leser, 2006))

Die technische Heterogenität befasst sich mit den Unterschieden bezüglich der Protokolle, Austauschformate und Anfragesprache. Es gibt verschiedene Möglichkeiten Daten zu speichern und abzurufen. Die Daten können in einer relationalen Datenbank, XML-Datenbank oder zum Beispiel in einer CSV Datei abgelegt sein. Auf die gespeicherten Daten erhält man durch unterschiedliche Anfragesprachen und unterschiedliche Kommunikationsprotokolle Zugriff. Zudem sind die Daten im Austausch in unterschiedlichen Formaten. Die Tabellle verdeutlicht die Herausforderungen der technischen Heterogenität.

Ebene	Mögliche Ausprägungen
Anfragemöglichkeit	Anfragesprache, parametrisierte Funktionen, Formulare
Anfragesprache	SQL, XQuery, Volltextsuche
Austauschformat	Binärdaten, XML, HTML, tabellarisch
Kommunikationsprotokoll	HTTP, JDBC, SOAP

Tabelle 2.1: (Naumann und Leser, 2006)

Die syntaktische Heterogenität ist ein Ergebnis der technischen Heterogenität und bezieht sich auf die technischen Unterschiede in der Darstellung gleicher Sachverhalte, wie zum Beispiel verwendete Datentypen und -formate, Zeichensätze oder Dateistrukturen. (vgl. (Naumann und Leser, 2006)(chap.3))

Beispiele für syntaktische Unterschiede, die entstehen, wenn Datenbanken oder die Speicherung von Daten individuell entworfen worden sind:

- Unterschiedliche Datentypen (boolean vs. bit, float vs decimal etc.)
- Verwendung von verschiedenen Zeichensätzen (UTF-8, Unicode, ASCII)
- Verwendung von unterschiedlichen Trennzeichen (Komma, Semikolon, Tabulator)
- Verwendung von unterschiedlichen Dateiformaten (csv, xml, xls, txt)

(vgl. (Rossak, 2013)(chap.2))

Syntaktische Homogenität zu erhalten ist nicht sehr aufwändig. Zahlenformate lassen sich umrechnen, Zeichendarstellungen ineinander transformieren und Trennzeichen durch einfache Parser übersetzen. (vgl. (Naumann und Leser, 2006)(chap.3))

Von struktureller Heterogenität spricht man, wenn zum Beispiel zwei unterschiedliche Schemas denselben Sachverhalt beschreiben. Dies kann sich durch den verwendeten Modelltyp, den definierten Standardwerten und Formaten, den Integritätsbedingugen, dem Normalisierungsgrad oder der Tabellenstruktur ausdrücken.

Beispiele für strukturelle Heterogenität sind:

- Verwendung von unterschiedliche Datenmodellen oder Prozessmodellierungssprachen (ER, UML)

- Verwendung von unterschiedlichen Standardwerten und Formaten, wie z.B bei Telefonnummern oder Datumswerten
- Verwendung von unterschiedlichen Integritätsbedingugen (Schlüsselfelder)

(vgl. (Rossak, 2013)(chap.2))

Die Beseitigung dieser Probleme ist sehr aufwändig. Zum einen muss erstmal ermittelt werden, welche strukturellen Unterschiede bestehen und zum anderen müssen nun alle gefundene Probleme beseitigt werden. (vgl. (Rossak, 2013)(chap.2))

Die letzte Art der Unterteilung von den vier verschiedenen Arten der Heterogenität, ist die semantische Heterogenität. Die semantische Heterogenität bezieht sich auf die verschiedenen Kontexte, also Bedeutungen, der Daten in den verschiedene Datenquellen. Es muss geklärt werden, welche Bedeutungen Nullwerte haben, was für Skalen verwendet worden sind und was Kodes bedeuten. (vgl. (Rossak, 2013)(chap.2))

Beispiele für die semantische Heterogenität sind.

- Verwendung von unterschiedliche Benennungen für die gleichen Sachverhalte (Abkürzungen, Synonime, wie z.B IT oder EDV)
- Verwendung von gleichen Bennennungen für unterschiedliche Sachverhalte (Fälligkeit bei Rechnungen, Datum oder Betrag?)
- Verwendung von unterschiedlichen Kodes für die gleichen Werte (Geschlecht: m,w oder m,f)
- Verwendung von unterschiedlichen Skalen (Alter: Ab 6, Ab 14, Ab 21 21-30, 31-40, 41-50, ü.50, ü.70)
- Verwendung von unterschiedlichen Bemaßungen oder Skalen (Geschwindigkeit: km/h oder mp/h)

(vgl. (Rossak, 2013)(chap.2))

Diese Probleme zu lösen ist sehr zeitintesiv, da eine reine Analyse der Datenbank nicht ausreicht. Es muss Domänwissen aus den verschiedenen Fachabteilungen herangezogen werden, um die verschiedenen Bedeutungen der Daten zu vereinheitlichen. Eine Alternative zum Domänwissen ist es, die Meta Daten zu analysieren oder sich durch die Dokumentationen zu arbeiten und dadurch einen Einblick, wie die Daten zu interpretieren sind, zu erlangen. (vgl. (Rossak, 2013)(chap.2))

2.4 Datenqualität

Wenn die Datenqualität nicht den Anforderungen entspricht, kann es fatale Folgen haben.

Es gibt spektakuläre Beispiele für mögliche Folgen aus schlechter Datenqualität:

Eine Großbank verrechnet sich bei der Ermittlung ihrer Gewinne um 200 Millionen Schweizer Franken. Schuld war die mangelhafte Integration der Daten einer kurz vorher (1 Jahr) akquirierten Versicherung.(vgl. (Apel, Behme und Eberlein, 2010) (s.42))

US Finanzbehörde 1992: knapp 100.000 Steuererstattungsbescheide waren nicht zustellbar (vgl. (Apel, Behme und Eberlein, 2010) (s.42))

Diese Beispiele verdeutlichen wie hoch der Stellenwert einer guten Datenqualität ist.

'Datenqualität ist die Bewertung von Datenbeständen hinsichtlich ihrer Eignung, einen bestimmten Zweck zu erfüllen (»fitness for use«). Als Kriterien gelten dabei die Korrektheit, die Relevanz und die Verlässlichkeit der Daten, sowie ihre Konsistenz und Verfügbarkeit auf verschiedenen Systemen.' (Wandt, 2015)

'Datenqualität ist ein mehrdimensionales Maß für die Eignung von Daten, den an ihre Erfassung/Generierung gebundenen Zweck zu erfüllen. Diese Eignung kann sich über die Zeit ändern, wenn sich die Bedürfnisse ändern' (Apel, Behme und Eberlein, 2010)(s.19)

Die Gemeinsamkeit dieser Definitionen besteht in der Aussage, dass es sich nur auf einen bestimmten oder gebundenen Zweck bezieht. Dr. Holger Wandt bestimmt, dass es 5 Kriterien für die Datenqualität gibt: Korrektheit, Relevanz, Verlässlichkeit, Konsistenz und Verfügbarkeit. Apel, Behme und Eberlein gehen auf die Veränderung der Bedürfnisse ein. Bedürfnisse ändern sich mit der Zeit, dadurch ändert sich auch die gemessende Datenqualität.

Es gibt viele verschiedene Prozesse, welche die Datenqualität beeinträchtigen. Diese Prozesse lassen sich in drei verschiedene Rubriken einordnen.

Zum einen gibt es Prozesse, die externe Daten in die Datenbank integrieren. Dies geschieht bei der manuellen Dateneingabe, der Zusammführung von Systemen oder bei der Konvertierung von Daten.

Bei der Konsolidierung von zwei Systemen überlappen sich oft die Daten im konsolidierten System. Redundante Daten sowie Konflikte der Daten sind das Ergebnis der Konsolidierung. Dies kann mit einer Winner-Loser Matrix gelöst werden. Diese Matrix würde darüber entscheiden, welche Daten die korrekten sind, falls ein Konflikt auftritt. Diese Winner-Loser Matrix kann eine komplexe Hierarchie von Bedingungen sein. Zum Beispiel könnte man, falls Konflikte beim Namen auftreten, festlegen, dass zuerst der Name aus dem System A genommen wird, falls dieses System den Namen kennt, sonst von System B und falls System A und B beide keinen Namen haben, wird der Name aus System C genommen. Das Problem dieser Winner-Loser Matrix ist, dass sie so komplex werden kann, das die Verständlichkeit verloren geht. (vgl. (Maydanchik, 2007) (chap.1))

Bei der manuellen Eingabe der Daten kann es hinsichtlich der Datenqualität zu erheblichen Problemen kommen. Dass Personen bei der manuellen Dateneingabe Tippfehler passieren, ist die meist verbreiteste Fehlerquelle für inkorrekte Daten. Selbst wenn die Mitarbeiter darauf hingewiesen werden, bei der Überprüfung der Eingaben sehr aufmerksam und achtsam zu sein, bleiben 3% der Daten inkorrekt. Ein weiteres Problem der manuellen Dateneingabe ist es, das fehlende Werte einfach leer gelassen werden. Wenn das System jedoch kein leeres Feld erlaubt, werden oft unbedeutende Werte eingetragen. Einige der Probleme bei der manuellen Dateneingabe lassen sich durch gute Formulare und eine gute Anleitung vermeiden. (vgl. (Maydanchik, 2007) (chap.1))

Des Weiteren gibt es Prozesse, welche die Datenqualität "verfaulen" lassen. Dies könnte zum Beispiel sein, wenn Änderungen nicht erfasst werden, Systeme erweitert werden oder Prozesse automatisiert werden.

Änderungen von Daten müssen über alle Datenquellen hinweg synchron gehalten werden. Wenn Daten an einem Punkt geändert werden, muss die Änderung an allen anderen Stellen, wo die Daten abgelegt werden, migiriert werden. Wenn dies nicht erfolgt, können inkonsistene Daten entstehen. Es sollte also sicher gestellt werden, dass Änderungen von Daten überall migriert werden, wo diese Daten abgespeichert bzw. benutzt werden.

Ein weiterer Prozess ist der Verlust von Kompetenzen. Mitarbeiter, die seit langer Zeit in einem Unternehmen arbeiten und wissen, wo welche Daten abgelegt sind und wie man mit diesen Daten arbeiten sollte, verlassen irgendwann das Unternehmen. Wenn diese Prozesse nicht gut dokumentiert sind, können neue Mitarbeiter, ohne eine Einarbeitung, nicht wissen wofür, warum, und wieso die Daten so abgespeichert sind, wie sie sind. Dadurch werden die Daten möglicherweise falsch benutzt und es entstehen Datenqua-

litätsprobleme. Dieses Problem könnte durch eine detailreiche Dokumentation in einem Meta Daten Repository gelöst werden.

Durch die Automatisierung von Prozessen entstehen Datenqualitätsprobleme. Ein Mensch würde die erhaltenen Daten erst einmal validieren, bevor er diese benutzt, ein Computer jedoch nimmt die Daten so, wie er sie bekommt und kann nicht darüber urteilen, ob die Daten valide sind oder nicht. Es ist möglich Validierungsfunktionen in den Automatisierungsprozess zu integrieren, jedoch ist es nahezu unmöglich alle ungewöhnlichen Eigenschaften der Daten zu erfassen. (vgl. (Maydanchik, 2007) (chap.1))

Die letzte Rubrik von Prozessen, welche die Datenqualität beeinflussen, sind die Vorgänge, welche Daten ändern. Ein Beispiel hierfür wäre Datenbereinigung oder auch Datenintegration.

Datenbereingung kann mehr Probleme hervorrufen, als ursprünglich vorhanden waren. Früher wurde die Datenbereingung manuell durchgeführt, wodurch eine bessere Datenqualität in der Regel das Ergebnis war. Datenbereinigung wird heutzutage jedoch in der Regel durch automatisierte Algorithmen durchgeführt. Diese Algorithmen können Bugs haben, die tausende von Einträgen in der Datenbank beeinflussen und in Folge könnte dadurch eine falsche Bereinigung durchgeführt werden. Datenbereiningung ist ein zweitseitiges Schwert, mit welchem man sich schwer verletzten kann, wenn man dieses nicht aufmerksam benutzt. (vgl. (Maydanchik, 2007) (chap.1;s.16))

'Datenqualität ist kein Berg, auf den man klettern kann, seine Flagge hisst und danach immer glücklich lebt. Datenqualität ist ein Garten, auf den man durchgängig achten sollte. Mit Mühe und Geduld kann der "Garten" wunderschön werden, jedoch lässt man den "Garten" für eine kurze Zeit nur außer Acht, wächst das Unkraut schon wieder aus dem Boden.' (Maydanchik, 2007) (chap.1;s.4)

2.4.1 Datenprofiling

Der erste Schritt zur Sicherung der Datenqualität ist es, die Daten genauer zu untersuchen, also Datenprofiling zu betreiben. Beim Datenprofiling oder Data-profiling werden die Daten aus Datenquellen anhand von Analysen, speziellen Regeln oder Kriterien untersucht und analysiert. Die einzelnen Analysen liefern jedoch nur einzelne Mosaiksteinchen vom Bild der Datenqualität - erst die richtige Kombination dieser Analy-

sen ergibt ein vollständiges Bild der Datenqualität. (vgl. (Apel, Behme und Eberlein, 2010)(chap.2;s.114))

Attribut-Analyse: Bei der Attribut-Analyse versucht man detaillierte Informationen über den Inhalt und der Struktur einer Spalte oder eines bestimmten Attributs herauszufinden.

Attributnamen-Analyse: Diese Analyse untersucht die Attributbezeichnung. Die Attributbezeichnung ermöglicht das Wissen über den Datentyp sowie den Inhalt der Daten. Zum Beispiel ist ein Attribut, welches Name heißt typischerweise eine Zeichenkette und beinhaltet den Namen. Ein Attribut mit dem Namen claim_ID sollte ein numerischer Wert und ein Unique Identifier sein. Enthält ein Attributname einen Hinweis auf einen Datentyp (zum Beispiel Einkommen_NR) oder die Funktion (zum Beispiel claim_ID)so sollte diese auch richtig sein. Beispielsweise können mit ID bezeichnete Attribute, aber nicht eindeutige Attribute genauso zu einem falschen Verständnis führen, wie solche mit NR und nichtnumerischen Inhalt. Ein Beispiel hierfür wäre ein Attribut mit dem Namen Berufsgruppen_NR, dessen Werte nur zu 72.5 % numerisch sind. Zusätlich wird analysiert, ob die Namen eindeutig sind oder Synonyme existieren. (vgl. (Apel, Behme und Eberlein, 2010)(chap.2)

Datentyp-Analyse: Ziel dieser Analyse ist es, den Datentyp herauszufinden. Der Datentyp wird anhand der Meta Daten festgestellt. Danach werden alle zu dem jeweiligen Datentyp gehörenden Attributwerte analysiert. Das Ergebnis dieser Analyse ist der tatsächliche, korrekte physikalische Datentyp. Zusätzlich wird der Datentyp hinsichtlich der Länge und Genauigkeit (Nachkommastellen bei Zahlen) sowie die davon abweichenden Werte analysiert. (vgl. (Apel, Behme und Eberlein, 2010)(chap.2;s.116)) Weicht die dokumentierte Länge sehr stark von der maximalen oder der durchschnittlichen Länge ab, so liegt vermutlich eine Zweckentfremdung dieses Attributes vor. 'Ist zum Beispiel das Attribut 'Langbeschreibung' eines Artikels mit einer Länge von 100 Zeichen dokumentiert, hat die Analyse hingegen eine maximale Länge von drei Zeichen ergeben, so ist das sehr verdächtig. Diese Werte sind dann näher zu betrachten und mit dem Fachbereich zu diskutieren.' (Apel, Behme und Eberlein, 2010)(s.114)

Wertebereich-Analyse: Hier werden statistische Kennzahlen, wie Minimum, Maximum, Mittelwert oder die Standardabweichung eingesetzt. Durch diese Kennzahlen kann ein Werteausschlussbereich festlegen werden, so dass alle Kunden, die zum Beispiel älter als 150 Jahre sind, gelöscht werden.

Null-Werte-Analyse: Mit Hilfe dieser Analyse werden alle Null-Werte identifiziert.

Duplikat-Analyse: Ziel der Duplikatsanalyse ist es, Duplikate im Datensatz zu finden. Duplikate, genau wie Null-Werte, sind für die Auswertungen Fehlerquellen.

Muster-Analyse: Die Muster der Daten müssen auch untersucht werden. Gerade bei Daten, die Telefonnummern oder ein Datum enthalten, ist das Muster extrem wichtig. Ein Datum kann unterschiedliche Formate annehmen. Daher ist es wichtig, dass alle Datumsfelder dasselbe Format aufweisen. Zudem wird bei der Musteranalyse nach versteckten Zeichen im Datensatz gesucht. Versteckte Zeichen können z.B Leerzeichen am Anfang oder am Ende der Zeichenkette sein. 'Solche versteckten Zeichen führen bei Auswertungen häufig zu Fehlern, da sie bei Abfragen unbemerkt nicht berücksichtigt werden. Die Muster-Analyse identifiziert solche Fälle zielsicher und macht sie sichtbar, sodass diese Werte anschließend korrigiert werden können.' (Apel, Behme und Eberlein, 2010)(s.121)

Domänen-Analyse: Nur zulässige Attributelemente werden identifiziert. Ausgehend davon, dass im Datensatz eines Onlineshops die Sparte "Farbe" enthalten ist, würde bei der Analyse dieser Sparte festgestellt werden, dass nur Zeichenketten zulässig sind, welche eine Farbe repräsentieren. Mithilfe dieser Analyse können verschiedene Spezifikationen und Regeln abgeleitet werden, welche die Korrektheit der Daten erheblich steigern.

Eindeutigkeits-Analyse: 'Die Eindeutigkeits-Analyse bestimmt für die Attribute den Grad der Eindeutigkeit der enthaltenen Werte. Nur eindeutige Attribute können als Schlüsselattribute verwendet werden und unterschiedliche Tabellen verknüpfen.' (Apel, Behme und Eberlein, 2010)(s.123)

Datenprofiling beeinhaltet nicht nur eine Analyse von Attributen, sondern auch die Analyse von ganzen Datensätzen. Diese Analyse wird benutzt, um funktionale Abhängigkeiten zu identifizieren.

'Bei der Analyse funktionaler Abhängigkeiten sucht man nach Abhängigkeiten zwischen den Attributen eines Datensatzes. Das Ziel ist es, die Existenz von Beziehungen und Korrelationen zwischen den Werten zu entdecken. Dabei unterscheidet man zwischen uni- und bidirektionalen Zusammenhängen.' (Apel, Behme und Eberlein, 2010)(s.133)

'Unidirektionale Zusammenhänge sind Zusammenhänge, bei denen man von einem Wert auf den anderen schließen kann, jedoch nicht wie bei den bidirektionalen Zusammenhängen, auch zurück.' (Apel, Behme und Eberlein, 2010)(s.133)

Ein Beispiel für einen bidirektionalen Zusammenhang wäre die Sozialversicherungsnummer eines Mitarbeiters und der Mitarbeiter selbst. Man kann anhand der Sozialversicherungsnummer den Mitarbeiter finden und anhand des Mitarbeiters auch seine Sozialversicherungsnummer. Ein Kunde gehört zu einer Kundengruppe und anhand des Kunden kann man seine Kundengrupppe herausfinden, jedoch ist es nicht möglich anhand einer Kundengruppe einen Kunden zu identifizieren; dies ist ein Beispiel für einen unidirektionale Zusammenhang.

Analyse auf Schlüsselattribute: 'Die Schlüsselattribute identifizieren eindeutig einen Datensatz innerhalb einer Tabelle und besitzen ebenfalls eine funktionale Abhängigkeit zu allen anderen Attributen dieses Datensatzes. In der Praxis findet man häufig neben den technischen Schlüsselattributen auch alternative, fachliche Schlüssel.' (Apel, Behme und Eberlein, 2010)(s.131)

'Die technischen Schlüssel bestehen in der Mehrzahl aus eindeutigen, künstlich generierten, numerischen Werten und beschränken sich auf ein Attribut (z.B. das Attribut KUNDEN_ID in der Tabelle KUNDEN).' (Apel, Behme und Eberlein, 2010)(s.131)

'Die fachlichen Schlüssel bestehen meist aus einer Kombination vorhandener, natürlicher Attribute (z.B. NAME, VORNAME, GEBURTSDATUM, ADRESSE). Eindeutig identifizieren diese einen Datensatz und darüber das zugehörige, reale Geschäftsobjekt (z.B. KUNDE, BESTELLUNG). Es gibt aber auch fachliche Schlüssel, die nur aus einem Attribut bestehen. Beispiel hierfür ist die Rentenversicherungsnummer.' (Apel, Behme und Eberlein, 2010)(s.131)

Die identifizierten möglichen Schlüssel werden anhand der Daten getestet. Am Ende bleiben nur die möglichen Kandidaten übrig, für die im Datensatz eine hinreichende Eindeutigkeit gefunden worden ist. (vgl. (Apel, Behme und Eberlein, 2010)(chap.2;s.114))

'Technische und fachliche Schlüssel aus nur einem Attribut lassen sich mit der Eindeutigkeitsanalyse auf Attributebene (siehe oben) sehr leicht identifizieren. Technische und fachliche Schlüssel aus mehr als einem Attribut werden hingegen mit der Analyse auf Schlüsselattribute gefunden.' (Apel, Behme und Eberlein, 2010)(s.135)

Analyse auf abgeleitete Attribute: Bei dieser Analyse werden abgeleitete Attribute überprüft. Es gibt mathematische sowie regelbasierte Beziehungen. Ein Beispiel für eine mathematische Beziehung ist zum Beipsiel das Gesamtvolumen der Einkäufe eines Kunden. Hierfür werden alle Bestellungen eines Kunden summiert und abgespeichert. Die daraus ergebene Formel definiert die mathematische Beziehung. Eine regelbasierte

Beziehung könnte zum Beispiel sein, dass bestimmte Berufsgruppen innerhalb eines Unternehmens, dieselbe Bezahlung bekommen würden. Die Regel wäre also, falls du zu der Gruppe Entwickler gehörst, ist dein Gehalt x €.

Zudem gibt es noch **Analysen auf Tabellenebene.** 'Bei der Analyse auf Tabellenebene wird versucht, referenzielle Abhängigkeiten innerhalb von Tabellen zu identifizieren.' (Apel, Behme und Eberlein, 2010)(S.138) Durch diese Analyse werden Beziehungen zwischen Attributen in verschieden Tabellen oder zwischen Datensätzen innerhalb einer Tabelle ermittelt. Zudem werden die Kardinalitäten der Beziehungen erfasst, dadurch ist es möglich Hierachien und Eltern-Kind Beziehungen zu entdecken sowie Waisen und Eltern ohne Kinder. 'Die referenzielle Integrität garantiert die Existenz einer Verknüpfung zwischen den verbundenen Tabellen und dass für jede Referenz in der einen auch ein zugehöriger Datensatz in der anderen Tabelle vorhanden ist. Nur damit ist eine Verknüpfung dieser Tabellen (zum Beispiel für eine Auswertung) möglich.' (Apel, Behme und Eberlein, 2010)(s.138)

2.4.2 Datenbereinigung

Datenbereinigung befasst sich mit der Beseitigung von Fehlern, der Beseitigung von Inkonsistenten und dem korrigieren von unlogischen oder unplausiblen Daten in einem Datensatz. IBM Data Analytics schrieb, dass Data Scientiest 80 Prozent ihrer Zeit nutzen, um Fehler in den Daten zu finden und dieses zu beseitigen. Die restlichen 20 Prozent werden lediglich genutzt, um eine Datenanalyse durchzuführen. (vgl. (IBM))

Die Daten sollen helfen, Entscheidungen treffen zu könnnen. Fehlerhafte Daten oder nicht vorhandene Daten sind nicht förderlich für die Entscheidung, da sie die Grundlage für die Entscheidung sind. "Algorithmen, die auf der Grundlage von Daten basieren, können nur so gut sein, wie Ihre Daten selbst." (Ki Hyun Tae, 2019)

Datenbereinigung ist ein Prozess, welcher in drei verschiedene Teilprozesse gegliedert werden kann. Als erstes wird definiert und bestimmt, was ein Fehler ist. Der zweite Schritt ist es, die Fehler im Datensatz zu identifizieren. Der letzte Teilprozess besteht darin, die Fehler zu beseitigen.

Einer der Hauptaufgaben der Datenbereingung ist es Fehler zu definieren. Es ist essentiell, dass man versteht, wo der Großteil der Fehler seinen Ursprung hat. Domänwissen ist hier sehr wichtig. Es muss zum Beispiel bestimmt werden, wie mit Null-Werten oder

Duplikaten verfahren wird. Des Weiteren muss bestimmt werden, wie mit inkonsistenten Daten umgegangen wird. Wenn zum Beispiel ein Kunde zwei verschiedene Kunden IDs besitzt, muss entschieden werden, welche ID die aktuelle ist. Nachdem dies entschieden ist, kann festlegen werden, dass die alte ID mit der neuen ID ausgetauscht wird.

Eine Möglichkeit Fehler zu identifizieren ist die Anwendung von Dataprofiling. Hier werden, wie im Abschnitt Datenprofiling beschrieben, die Daten genauer untersucht. So können z.B Null-Werte oder Ausreißer durch eine Wertebereichsanalyse bestimmt werden.

Es gibt verschiedene Möglichkeiten Fehler zu beseitigen. Eine wäre die manuelle Bereinigung, die jedoch sehr zeitaufwändig und in großen Datensätzen nahezu unmöglich ist. Eine andere Verfahrensmöglichkeit zur Datenbereingung ist die Anwendung von Software, so zum Beispiel das Open Source Tool OpenRefine. Diese Software unterstützt die Identifikation von Fehlern, um diese dann zu beseitigen. OpenRefine bietet die Möglichkeit nach Duplikaten oder Null-Werten zu suchen, die dann anschließend transformiert oder gelöscht werden können.

Nachdem man die Fehler des Datensatzes identifiziert und bereinigt hat, sollte der Datensatz auf Korrektheit, bezüglich der definierten Regeln im ersten Teilprozess, überprüft werden.

Zudem ist es nützlich einen Bericht zu erstellen. Dabei ist es vorteilhaft auf folgende Fragestellungen einzugehen:

- Welche Typen von Rauschen traten in den Daten auf?
- Welche Ansätze haben Sie verfolgt, um das Rauschen zu entfernen? Welche Techniken waren erfolgreich?
- Gab es Fälle oder Attribute, die nicht verwertet werden konnten? Vergessen Sie nicht, Daten zu vermerken, die aufgrund von Rauschen ausgeschlossen wurden.

(vgl. (IBM))

Dieser Bericht liefert eine Dokumentation über den Ablauf der Datenbereinigung.

2.4.3 Datenmonitoring

Die kontinuierliche Überwachung der Datenqualität wird Datenmonitoring genannt. Ziel des Datenmonitoring ist es, eine verlässliche Aussage über die aktuelle Datenqualität treffen zu können.

Dies kann einerseits mittels definierter Kennzahlen und Berichte durchgeführt werden, andererseits lässt sich die Wirksamkeit von Maßnahmen zur Verbesserung der Datenqualität auch durch regelmäßiges Dataprofiling verifizieren. Zwar kann zu Projektbeginn die Datenqualität evaluiert werden; ob sich die Datenqualität aber wirklich permanent verbessert, entscheidet sich erst im laufenden Betrieb. Dem Datenmonitoring kommt somit eine wichtige Rolle bei der nachhaltigen Realisierung einer guten Datenqualität zu. (vgl. (Apel, Behme und Eberlein, 2010) (chap.2))

Wichtig ist es, dass Fehler frühzeitig erkannt und Gegenmaßnahmen eingeleitet werden, falls definierte Grenzwerte überschritten werden. Je mehr Fehler unbemerkt in die Datenquellen integriert werden, desto schwerer und umfangreicher wird es, Konsistenz und Korrektheit sicherzustellen.

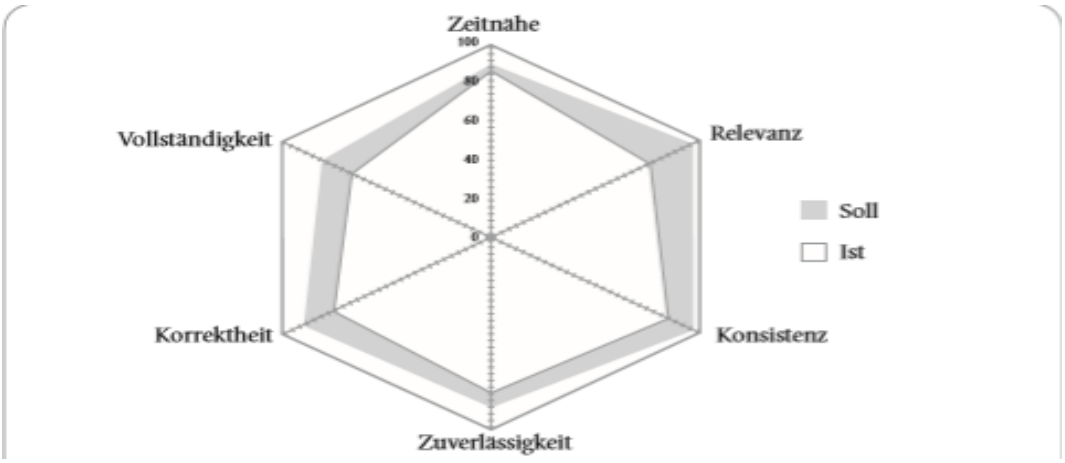
'Mit einer kontinuierlichen Überprüfung der Datenqualität wird sichergestellt, dass qualitativ hochwertige Daten konsistent, korrekt und zuverlässig bleiben. Die Datenmonitoring-Ergebnisse können in Datenqualitätsberichten dargestellt werden.' (Apel, Behme und Eberlein, 2010)(s.228)

Den Kreislauf des Datenmonitoring könnte man mit Erkennen, Kommunizierren und Lösen beschreiben. Zuerst werden Fehler in den Daten durch Regeln und Bedingungen erkannt. Nun müssen die erkannten Fehler an alle relevanten Personen, die diese Daten benutzen, kommuniziert werden. Dies kann mittels einer Betroffenheitsmatrix erfolgen. Diese Matrix umfasst alle Stakeholder, die beim Auftreten von Datenqualitätsproblemen zu benachrichtigen sind. Handelt es sich um besonders kritische Daten, sollten die Daten Stewards benachrichtigt werden. Nun wissen alle relevanten Personen über das Datenqualitätsproblem Bescheid und es ist möglich das Datenqualitätsproblem zu lösen. Nun beginnt erneut der Kreislauf um Datenqualitätsprobleme zu erkennen.

Die laufende Überwachung der Daten kann visuell dargestellt werden. Hierfür gibt es verschiedene Visualisierungsformen, die ich nachfolgend erläutern möchte.

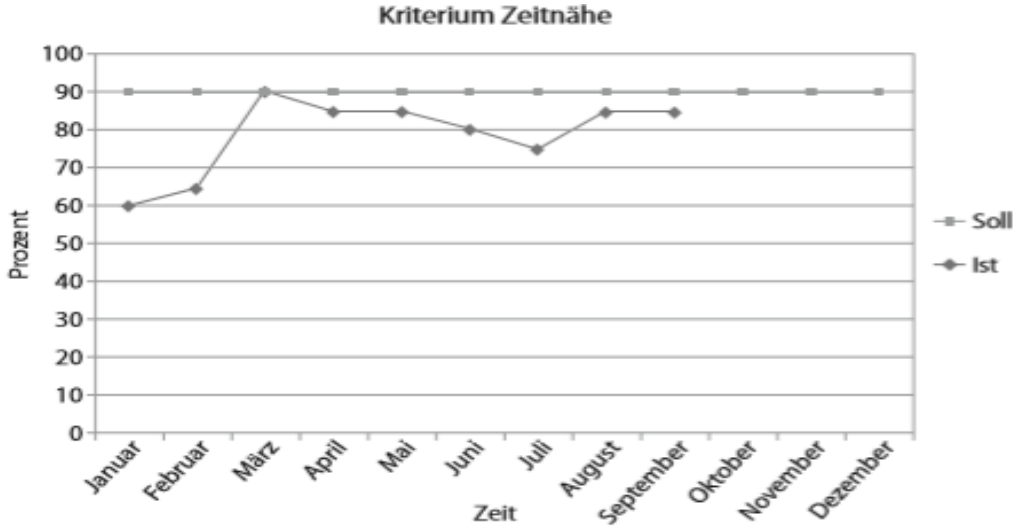
Eine Visualisierungsform ist die Spinnengrafik. Dies ist eine übersichtliche Darstellung der Datenqualitätskriterien. Bei der Spinnengrafik wird der Ist-Zustand mit dem Soll-Zustand verglichen. Siehe Abbildung:

Abbildung 2.1: Spinnengrafik (vgl. (Apel, Behme und Eberlein, 2010) (chap.2))



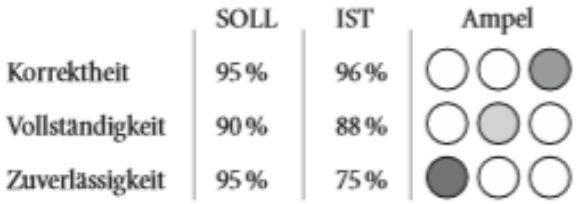
Historienreihen haben zum Soll- und Ist-Zustand noch eine zeitliche Komponente. Durch die zeitliche Komponente, kann erkannt werden wie die Datenqualität sich mit der Zeit entwickelt. Siehe Abbildung:

Abbildung 2.2: Historiereihe (vgl. (Apel, Behme und Eberlein, 2010) (chap.2))



Eine weitere Visualisierungsform sind Ampelgrafiken. Diese zeigen Zustände in Farben an. Hierfür werden Wertebereiche festgelegt. Rot bedeutet zum Beispiel das der IST-Zustand unter 85% für das jeweilige Kriterium liegt. Gelb liegt zum Beispiel bei 85% - 89% und Grün wäre alles über 90%. Durch diese farbliche Gestaltung ist auf den ersten Blick ein Eindruck, wie gut die Datenqualität gerade ist, möglich. Siehe Abbildung:

Abbildung 2.3: Ampelgrafik (vgl. (Apel, Behme und Eberlein, 2010) (chap.2))



'Ein Dashboard visualisiert große Mengen von meist verteilten Informationen, um die wesentlichen Kennzahlen, nach denen ein Prozess gesteuert wird, übersichtlich darzustellen. Dashboards können sich dabei aus unterschiedlichen Grafiken und Tabellen zusammensetzen und werden damit zum zentralen Kontrollpunkt in Sachen Datenqualität.' (Apel, Behme und Eberlein, 2010)(s.234)

Ohne Datenmonitoring würde es keine laufende Überwachung der Datenqualität geben. Daher ist es ein wesentlicher Bestandteil jedes Datenqualitätsmanagementprogramms. Durch die Überwachung erhält man Aussagen über den Zustand und die Entwicklung der Datenqualität sowie die Wirksamkeit von Maßnahmen zur Verbesserung der Datenqualität.

Verantwortliche Personen sowie Ziele und Regeln sind klar zu definieren. Das Monitoring lässt sich mit verschiedenen Werkzeugen realisieren. Wesentlich dabei ist es, dass die Konsistenz und die Nachvollziehbarkeit gewahrt bleiben.

'Abschließend ist nochmals festzuhalten, dass nur jene Bereiche eine Verbesserung erfahren werden, die sich auch messen und überwachen lassen' (Apel, Behme und Eberlein, 2010)(S.235)

3 Konzept

In diesem Kapitel werde ich das Konzept meiner Implementierung zur datengetriebenen Zuordnung von Schadensfällen zu Sachverständigen in der Versicherungsbranche vorstellen.

3.1 Prozesse

Der Prozess der Zuordnung von Schadensfällen zu Sachverständigen ist durchaus umfangreich. Versicherungsunternehmen benutzen verschiedenen Methoden, um einen Sachverständigen einen Schadensfall zuzuweisen. Sachverständige können durch die Schadenssumme des Schadensfalls oder ihrer Expertise zum Schaden zugeordnet werden. Eine weitere Möglichkeit der Zuordnung besteht darin die von den Sachverständigen angegebene Region, in der diese operieren möchten, zu berücksichtigen. Da manche Sachverständige die räumliche Distanz begrenzt haben wollen, zum Beispiel nicht mehr als 100 km fahren wollen, kann diese Information auch als Grundlage für die Zuordnung bedeutsam sein. Es gibt daher die verschiedensten Schritte, um einen Schadensfall einem passenden Sachverständigen zuzuordnen.

In diesem Abschnitt werde ich den aktuellen manuellen Prozess, sowie den automatisierten Prozess der Zuordnung von einem Schadensfall zu einem Sachverständigen kurz erläutern und modellieren.

3.1.1 Aktuelle Prozessanalyse

Bei claimsforce, dem StartUp, in welchem ich meine Bachelorarbeit erarbeitet habe, werden Schadensfälle hauptsächlich anhand der Postleitzahlengebiete der Sachverständigen zugeordnet. Um dies umzusetzten muss derjenige, der den Schaden zuweisen möchte sich alle Experten und deren Regionen anzeigen lassen und dann anhand der Daten einen

geeigneten Sachverständigen finden, der in der Region des Schadens Schäden bearbeiten möchte. Falls ein geeigneter Sachverständiger gefunden worden ist, kann anhand der Service Level Agreements überprüft werden, ob der Sachverständige für den Schadensfall geeignet ist. Wenn kein Sachverständiger in einer gewissen Region gefunden werden kann, probiert der Mitarbeiter, der den Schadensfall zuweisen möchte, einen Sachverständigen in einer gewissen Nähe zum Schaden zu finden. Dafür muss der Mitarbeiter sich sehr gut in Deutschland auskennen und übliche Distanzen zwischen Städten abschätzen können oder er muss ein Service wie Googlemaps nutzen, um die genaue Distanz zwischen der Adresse des Schadensfalles und der Adresse des Sachverständigen zu erhalten und auszuwerten. Die manuelle Überprüfung der Distanz ist sehr zeitinsiv und bei steigender Zahl an Sachverständigen nicht mehr umsetzbar. Falls ein Sachverständiger mit einer gewissen Nähe zum Schaden gefunden wird, muss wieder mit Hilfe der Service Level Agreements überprüft werden, ob dieser geeignet ist. Wenn der Mitarbeiter nach diesen beiden Zuordnungsverfahren immer noch keinen passenden Sachverständigen gefunden hat, versucht der Mitarbeiter einen Sachverständigen anhand von Erfahrungswerten zu finden. Ein relevanter Erfahrungswert wäre zum Beispiel, ob der Sachverständige in der Vergangenheit bereit war, längere Strecken zu fahren. Unter der Vorraussetzung, dass die Service Level Agreements des Sachverständigen passen, kann der Schadensfall dem Sachverständigen zugeordnet werden. Sollte kein passender Sachverständiger ermittelt werden, werden die Ansprüche an die Service Level Agreements dekrementiert.

Meine Aufgabe für diese Bachelorarbeit ist es nun diesen Prozess zu verbessern, zu automatisieren und zu implementieren.

3.1.2 Automatisierte Prozessanalyse

Der automatisierte und datengetriebene Prozess der Zuordnung von einem Schadensfall zu einem Sachverständigen beeinhaltet die Zuordnung von Sachverständigen zu einem Schaden mittels der Postleitzahlengebiete, der offenen Schadensfälle in der Region, wenn der Termin in der Zukunft liegt und der Distanz zwischen dem Sachverständigen und der Addresse des Schadens. Anschließend werden die gefundenen Sachverständigen mit allen relevanten Informationen bewertet, nach der Bewertung sortiert und an den Mitarbeiter übergeben.

Die Zuordnung via offenen Schadensfällen in der Region, mit einem Termin in der Zukunft und die Zuordnung via Distanz ist mit steigender Zahl von Sachverständigen und

Schadensfällen manuell nicht umsetzbar.

Der Mitarbeiter, welcher die Schadensfälle zuweist, muss sich alle Schadensfälle der Sachverständigen angucken, danach überprüfen, ob einer der Schäden, welcher der Sachverständige in Untersuchung hat in der gleichen Region wie der Schaden liegt und danach muss noch überprüft werden, ob der Termin schon stattgefunden hat.

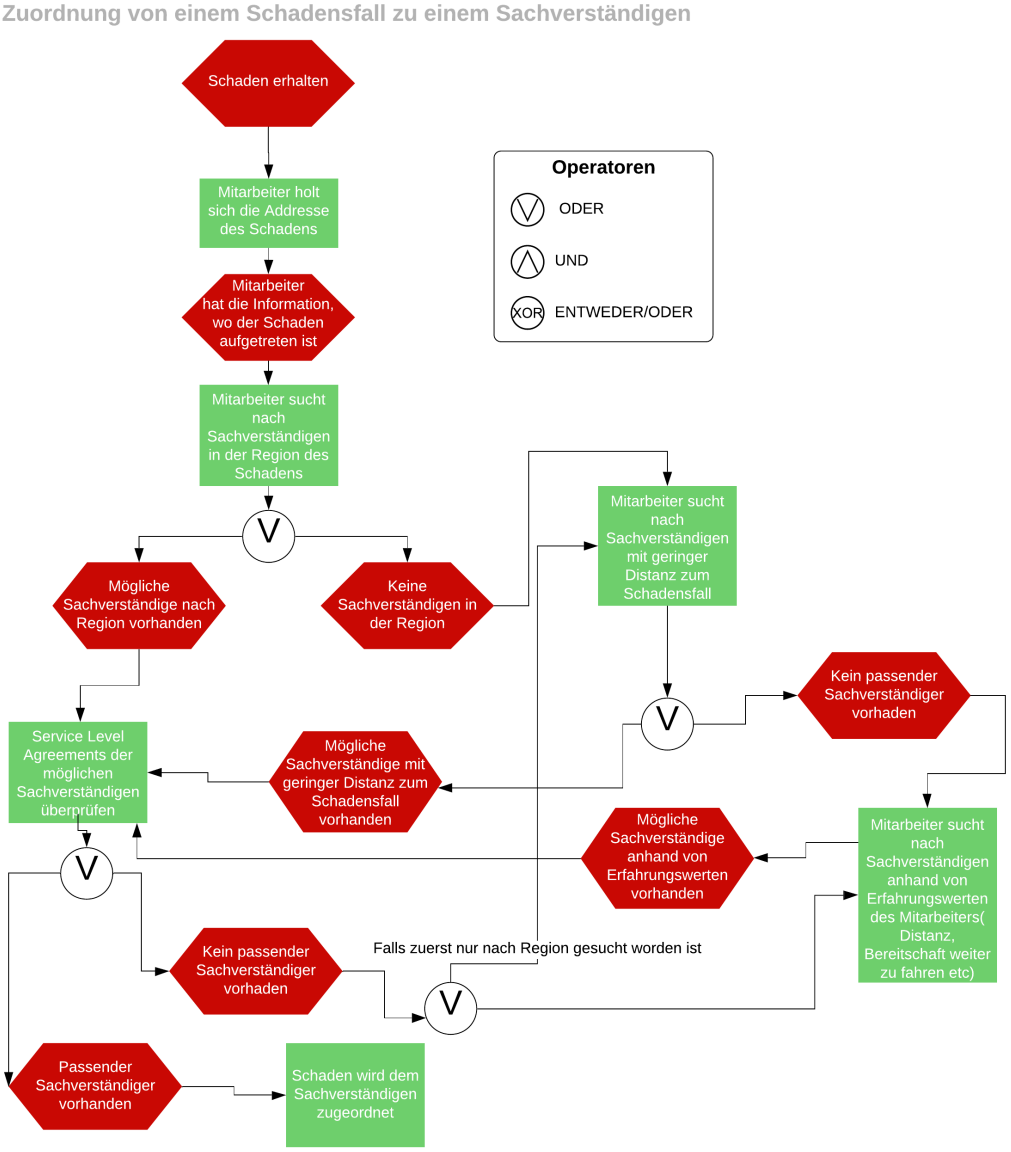
Eine manuelle Überprüfung der Distanz zwischen dem Schadensfall und den Sachverständigen ist sehr zeitinsiv. Der Mitarbeiter müsste Distanzen zwischen den Adressen der Sachverständigen und dem Schadensort ermitteln und diese Daten auswerten. Da es sehr umfangreich ist diese Schritte manuell zu erledigen, ist dieser Prozess automatisiert worden. Mit Hilfe von Python werden die Daten automatisiert ausgelesen und analysiert.

Durch eine Automatisierung des Prozesses der Zuordnung von Schadensfällen zu Sachverständigen ist es möglich, die neuen Kriterien der Zuordnung zu erfüllen.

3.1.3 Aktuelle Prozessmodellierung

Um den Prozess zu modellieren, habe ich mich für die Ereignisgesteuerte Prozesskette entschieden. Dieses Prozessmodell wurde zu einer bessern Übersicht, zum Verständnis und zur Dokumentation erstellt.

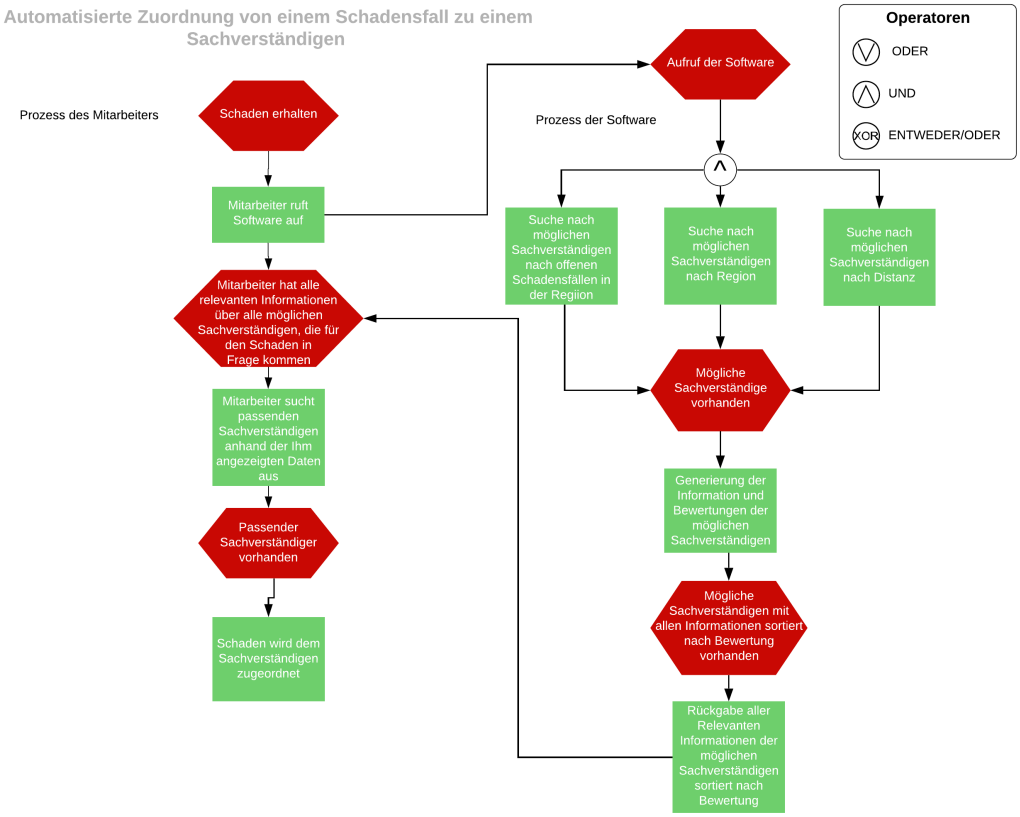
Abbildung 3.1: Ereignisgesteuerte Prozesskette des manuellen Prozesses



3.1.4 Automatisierte Prozessmodellierung

Um den automatisierten Prozess zu modellieren habe ich mich ebenso für die Ereignisgesteuerte Prozesskette entschieden.

Abbildung 3.2: Ereignisgesteuerte Prozesskette des automatisierten Prozesses



3.2 Daten Analyse

Claimsforce ist ein junges StartUp, deshalb sind keine großen Mengen an Daten zu Schadensfällen und Sachverständigen vorhanden. Es wurden lediglich ca. 2000 Schadensfälle über claimsforce bearbeitet und es gibt ca. 60 Sachverständige, die aktiv für claimsforce arbeiten.

Bei claimsforce habe ich folgende Schadens- und Sachverständigendaten, um einem Sachverständigen einen Schaden zuordnen zu können:

- Service Level Agreements der Sachverständigen
- Postleitzahlengebiete der Sachverständigen, falls angegeben

- Art des Schadens
- Vergangene Schadensfälle des Sachverständigen, welche bei claimsforce bearbeitet wurden
- Zeitachse des Schadens
- Adresse des Schadens
- Termin des Schadens
- Schätzung der Höhe des Schadens

Diese Daten sind ausreichend, um einen geeigneten Sachverständigen zu einem Schadensfall zu erfassen. Wünschenswert jedoch wäre es, wenn eine größere Auswahl und Anzahl an Daten zur Verfügung stehen würde.

Service Level Agreements ergeben sich aus der Zeitachse des Schadensfalls. Aus der Berechnung, wie viele Tage und Stunden zwischen dem Datum der Zuweisung des Schadensfalls und dem Datum, an dem der Bericht an das zuständige Versicherungsunternehmen abgesendet worden ist, ergibt sich der Wert 'time to invoice'. Nachfolgend werde ich die Service Level Agreements als TTI oder time to invoice bezeichnen.

Folgende Daten wären wünschenswert und würden bei einer genaueren Bewertung und Auswahl der Sachverständigen helfen:

- Kilometergenaue Angabe, wie weit die Sachvertändigen bereit sind zu fahren.
- Alle vergangenen Schadensfälle des Sachverständigen, um:
  - die gesamte Erfahrung des Sachverständigen dokumentiert zu haben
  - die Expertise in Teilbereichen, wie Leitungswasser, in Erfahrung bringen zu können
  - die gesamte Leistung, 'time to invoice', des Sachverständige zu erhalten
  - die Leistung in Teilbereichen des Sachverständigen zu erhalten
- Alle Schadensfälle, die ein Sachverständiger aktuell bearbeitet und nicht nur die, die bei claimsforce aktuell in Bearbeitung sind, um:
  - Routen für die Sachverständigen planen zu können

  - Termine für die Sachverständigen zu terminieren
  - die Auslastung zu prüfen
- Bewertungen über die Qualität der Berichte, welche die Sachverständigen verfasst haben

Die Einsicht, dass die Bewertung der Berichtsqualität von Sachverständigen wichtig ist, hat claimsforce dazu bewogen, entsprechende Daten zu erfassen. Der Mitarbeiter, welcher den Bericht vom Sachverständigen erhält und diesen an das entsprechende Versicherungsunternehmen weiterleitet, kann seit jüngster Zeit, nachdem der Bericht abgesendet worden ist, diesen bewerten.

Durch eine größere Anzahl und Auswahl an Daten wäre es möglich, passendere Sachverständigen zu finden und diese genauer bewerten zu können.

3.3 Anforderungsanalyse

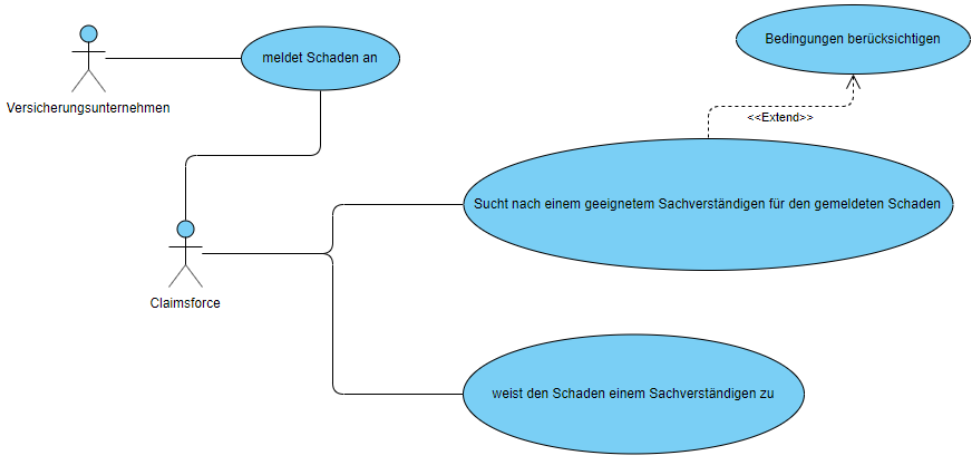
Zur Ermittlung der Anforderungen an die datengetriebene Zuordnung von Schadensfällen zu Sachverständigen in der Versicherungsbranche wurde der Prozess der Zuordnung von Schadensfällen zu Sachverständigen analysiert und mit einem Mitarbeiter, welcher die Schäden zuweist, besprochen. Die Anforderungen an die Software wurden in User Stories aufgeteilt. User Stories beschreiben die Anforderungen an ein Produkt oder Service aus Sicht des Nutzers.

3.3.1 Anforderungen an die Software

Die Software soll anhand einer Schadens ID alle möglichen Sachverständige für diesen Schaden finden und diese anschließend bewerten. Wenn ein neuer Schaden Claimsforce gemeldet wird, versucht ein Mitarbeiter, der für die Zuordnung verantwortlich ist, einen geeignetn Sachverständigen für diesen Schadensfall, unter bestimmten Bedingungen, zu finden.

Anwendungsfall der Software als Use-Case Diagramm:

Abbildung 3.3: Use-Case Diagramm



3.3.2 User Stories

Anhand den User Stories werden die gewünschten Funktionalitäten der Software beschrieben.

1. **User Story: Zuordnung via Region**

   Die Software soll Sachverständige anhand der Region des Schadens und der angegebenen Region des Sachverständigen finden. Jeder Sachverständige hat Regionen, in welchen er gerne Schäden bearbeiten möchte.

2. **User Story: Zuordnung via Schäden in Bearbeitung**

   Die entwickelte Software soll Sachverständige finden, die Schäden in Bearbeitung haben, welche in der Region stattfinden, in der der Schaden aufgetreten ist. Zusätlich muss hier überprüft werden, ob der Termin der Schadensbegutachtung in der Zukunft liegt oder schon stattgefunden hat. Falls der Termin für die Schadensbegutachtung in der Zukunft liegt, ist dieser Schaden sehr relevant, da es möglich wäre eine Route für den Sachverständigen zu erstellen. Routen sind bei Sachverständigen sehr beliebt, da man mehrere Schäden auf einem Fahrtweg erledigen kann und dadurch erheblich an Zeit für Fahrtwege spart.

3. **User Story: Zuordnung via der Entfernung vom Sachverständigen zum Schadensfall**

   Sachverständige sollen mittels der Entfernung zum Schadensfall gefunden werden. Falls die Entfernung zwischen dem Sachverständigen und dem Schadensfall weniger als 100 Kilometer beträgt, sollte dieser als möglicher Sachverständige vorgeschlagen werden.

4. **User Story: Bewertung der möglichen Experten**

   Nachdem alle möglichen Experten via den verschiedenen Methoden gefunden worden sind, sollen diese anhand Ihrer Daten bewertet werden und dann sortiert an die Schnittstelle übergeben werden. Die zu bewertenen Eigenschaften eines Sachverständigen sind:

   - Entfernung zum Schadensfall
   - Sachverständigen 'time to invoice' (Die Zeit, die ein Sachverständiger braucht um seinen Bericht zu erstellen)
   - Sachverständigen 'time to invoice' für die entsprechende Schadensgruppe (Die Zeit die ein Sachverständiger braucht um seinen Bericht in einer gewissen Schadensgruppe zu erstellen)
   - Ist dies die gewünschte Region des Sachverständigen?
   - Hat der Sachverständige Schadensfälle in der gewissen Region, welche noch nicht stattgefunden haben? (Möglicherweise ist eine zusammenhängende Route von meherere Schadensfällen möglich)
   - Anzahl aller Schadensfälle, die der Sachverständige in Bearbeitung hat
   - Anzahl aller Schadensfälle, die der Sachverständige bearbeitet hat

5. **User Story: Ergebnisse als JSON bereitstellen**

   Die Ergebnisse der Software sollen sortiert nach ihrere Bewertung im JSON Format bereitgestellt werden.

3.3.3 Nicht-funktionale Anforderungen

Nicht-Funktionale Anforderungen beschreiben, mit welchem Umfang die Funktionen implementiert werden sollen. Dadurch haben sie eine große Bedeutung auf die Entwicklung und Wartung des Systems. Zusätzlich dienen diese Anforderungen dazu, die Akzeptanz des Systems gegenüber dem Anwender zu erhöhen (Martin, 2000)

Funktionalität

Alle geforderten User Stories sollen implementiert und ausführbar sein.

Zuverlässigkeit

Die Software soll zuverlässig sein, da sonst durch falsche Vorschläge, Schadensfälle eine nicht optimale Zuweisung erhalten. Zudem soll die Software fehlerfrei laufen, da der Mitarbeiter sonst erheblich länger für den Prozess benötigt. Im Falle eines Fehlers soll es für den Mitarbeiter ersichtlich sein und ihm die Möglichkeit geben, die Suche nach einem geeignetem Sachverständigen noch einmal zu starten.

Benutzbarkeit

Die Software soll benutzerfreundlich sein. Der Mitarbeiter soll in seiner Anzeige, wenn er den Schaden zuweisen möchte, lediglich auf den Knopf “Schaden verteilen” drücken, danach soll der Mitarbeiter eine Übersicht über alle möglichen Sachverständigen erhalten. Nun soll es für jeden Sachverständigen einen Knopf mit SSchaden zuweisen"geben. Mit einem Knopfdruck auf “Schaden zuweisen”, wird der Sachverständige ausgewählt, der in der Zeile des Knopfes angezeigt wird.

Effizienz

Die Software soll möglichst effizient Anfragen bearbeiten können. Die Verarbeitung und Berechnung der Daten darf nicht länger als 10 Sekunden dauern.

Änderbarkeit

Die Software muss auf Grundlage der sich immer wechselnden Anforderungen erweiterbar sein. Neue Geschäftsregeln und Anwendugsfälle für unterschiedliche Versicherungsunternehmen müssen möglich und schnell umsetzbar sein.

Übertragbarkeit

Die Software soll die berechneten Daten der möglichen Sachverständigen an eine Schnittstelle übergeben. Auf diese Schnittstelle kann jedes System zugreifen.

Tracking

Es soll verfolgt werden können, welchen Sachverständigen der Mitarbeiter für den Schadensfall ausgewählt hat. Durch die Verfolgung der Vorschläge und Auswahl der Sachverständigen durch den Mitarbeiter soll analysiert werden, wie gut die Vorschläge zu den wirklichen Zuweisungen passen. Es wäre ideal, wenn die Software einen Sachverständigen an der ersten Stelle vorschlägt und dieser den Schaden annimmt und bearbeitet. Falls dies nicht der Fall ist, soll eine Analyse durchgeführt werden können, inwiefern die Bewertung für einen möglichen Sachverständigen angepasst werden kann.

3.4 Entwurf

In diesem Kapitel wird der Entwurf der Implementierung beschrieben.

3.4.1 Architektur

Die Software soll als Clean Code Architektur implementiert werden. Dadurch ist es einfacher, die Software zu warten und zu testen. Um ein großes Projekt pflegen zu können ist es von Vorteil Klassen in getrennten Komponenten zu entwickeln, wodurch diese unabhängig von den anderen Komponenten ausgetauscht werden können. Sollte sich zum Beispiel in der Infrastruktur die Datenbank ändern, muss nur eine Klasse angepasst werden. Dies erleichtert die Wartung und Pflege der Software. Des Weiteren lässt sich durch

dieses Unterteilung der Code einfacher testen. Da die Komponenten nicht voneinander abhängig sind ist es möglich diese einzelnd zu testen. (vgl. (Martin, 2018) (p.196))

Clean Code lässt sich in drei Teilbereiche aufteilen, die Infrastruktur, die Applikation und die Domain.

1. Die Domain beeinhaltet Entititäten, welche nichts über andere Ebenen wissen. Diese Entitäten sollten keine Abhängigkeiten besitzen. Eine Entität in der Zuordnungssoftware von einem Schadensfall zu einem Sachverständigen wäre zum Beispiel die Klasse "Schaden" oder "Sachverständiger". Entitäten sollten von verschiedene Applikationen im Unternehmen genutzt werden können. Zudem werden diese nicht von externen Änderungen beeinflusst. Die Entitäten ändern sich nicht, sollten Ansprüche auf Änderungen bezüglich der Sicherheit oder des Datenbanksystems bestehen. (vgl. (Martin, 2018) (p.197))
2. Die Applikationschicht behandelt die Geschäftsregeln der Applikation, auch Use-Cases genannt. Die Applikationsschicht interagiert mit der Domainschicht, jedoch weiß die Applikationsschicht nichts über die äußeren Schichten. Der Applikationsschicht soll es egal sein, ob die Daten in der Cloud oder in einer Dynamo DB abgelegt werden. Des Weiteren organisiert die Applikationsschicht den Fluss der Daten von den Entitäten und zu den Entitäten. (vgl. (Martin, 2018) (p.197))
3. Die äußerste Schicht, die Infrastrukturschicht, ist die Schicht, in der die Benutzungsoberfläche, die Datenbanken, Frameworks und Geräte definiert werden. Da sich diese Technologien oder Oberflächen häufig ändern könnne, hält man diese so weit wie möglich weg von den anderen stabileren Schichten, welche sich selten ändern. (vgl. (Martin, 2018) (p.198))

Domainschicht

In der Domainschicht benötigt man alle Entitäten. Für meine Software habe ich folgende Entitäten bestimmt:

- Sachverständigen (Expert)
- Schaden (Claim)
- Adresse (Addresse)

- AdressenMitDistanz(AddressWithDistance)- Diese Klasse erbt von der Klasse Adresse.
- Termin(Appointment)
- Bewertung(Prediction)
- Status - Aufzählung von den verschiedenen Status der Schäden

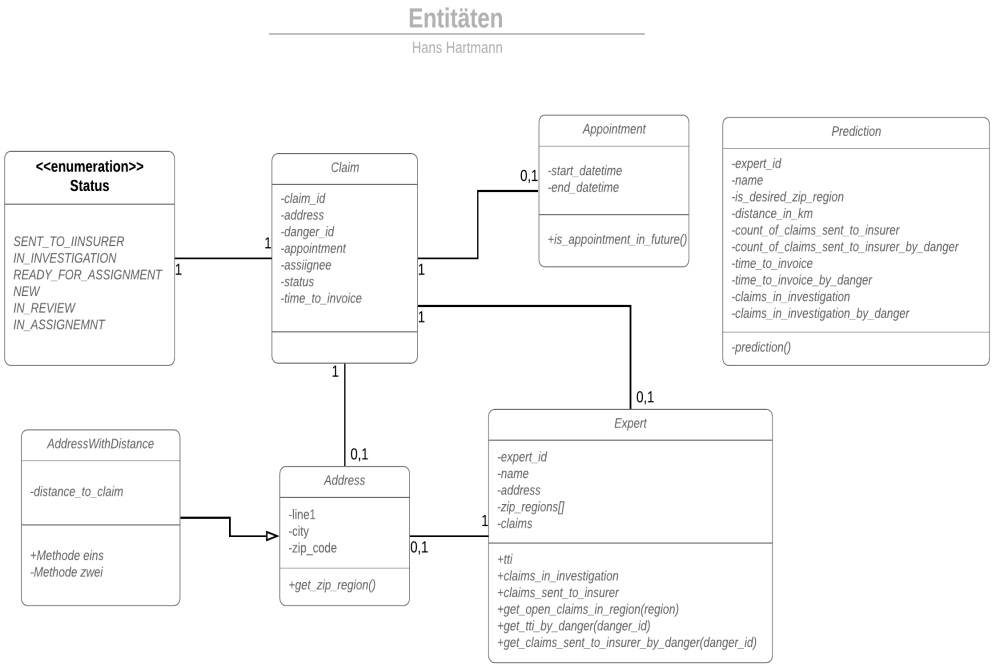
Zu den Entitäten gehören zum Domain-Layer auch Services. In der Domainschicht werde ich den folgenden Service implementieren:

- Vorhersage Service (PredictionService)

Dieser Service soll dafür genutzt werden, um die einzelnden Funktionen der Zuordnung abzubilden. In diesem Service sollen die Sachverständigen nach der Distanz, ihrer offenen Schadensfällen und ihrer Region gefunden und bewertet werden. Siehe Klassendiagramm Services: 3.6

Klassendiagramm der Entitäten:

Abbildung 3.4: Entitäten UML



Applikationsschicht

In der Applikationsschicht werden die Anwendungsfälle behandelt. Meine Software hat lediglich einen Anwendungsfall:

- Eine Bewertung zu allen möglichen Sachverständigen zu einem Schadensfall zu erhalten

Zudem werden ich in dieser Schicht abstrakte Repositories implementieren. Dadurch ist es möglich grundlegende Methoden für die spezifischen Repositories, welche in der Infrastrukturschicht implementiert werden, festzulegen. Siehe Klassendiagramm Repositories: 3.5

Infrastrukturschicht

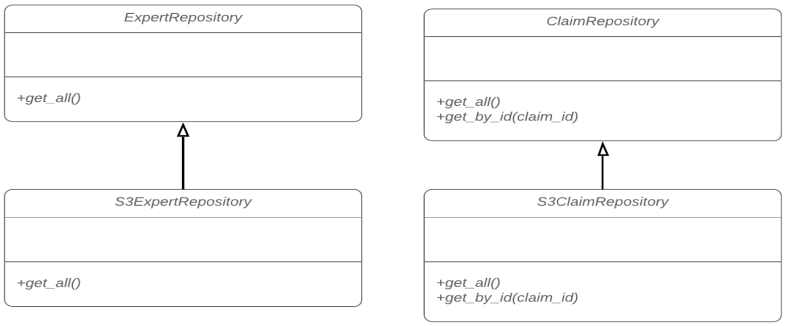
Die Infrastruktur ist die Schicht, in welcher das Programm gestartet wird. In dieser Schicht werden die verschiedenen spezifischen Repositories implementiert, um die gewünschten Daten zu erhalten. Diese spezifischen Repositories sind austauschbar, da die abstrakten Repositories durch die spezifischen Repositories erweitert werden und dadurch sichergestellt ist, dass die spezifischen Repositories die geforderten Funktionalitäten der Anwendung besitzen.

Folgende Repositories, mit den folgenden Funktionen, habe ich vor zu implementieren:

- Repository welches Schäden aus dem Amazon Storage ausliest
  - getAll() -> es sollen alle Schadensfälle, überarbeitet, aus der Datenbank geholt werden.
  - getById(ID) -> der Schadensfall zur gewünschten ID soll aus der Datenbank extrahiert werden.
- Repository welches Sachverständige aus dem Amazon Storage ausliest
  - getAll() -> es sollen alle Sachverständigen, überarbeitet, aus der Datenbank geholt werden.

Klassendiagramm der Repositories:

Abbildung 3.5: Repositories



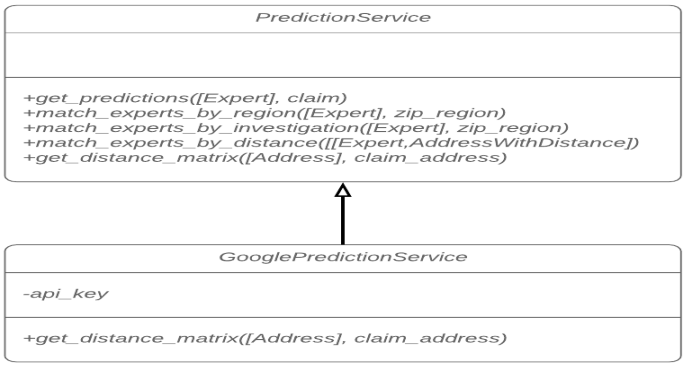
Zudem benötige ich in der Infrastruktur noch einen Service:

- Google Vorhersage Service (GooglePredictionService)

Dieser Service soll vom Vorhersage Service (PredictionService) erben und soll die Distanz zwischen einem Schadensfall und allen Sachverständigen via der Google Distance Matrix API ermitteln.

Klassendiagramm Services:

Abbildung 3.6: Services



Des Weiteren werden in der Infrastrukturschicht noch die Serializer und Deserializer benötigt. Diese sind für die Konvertierung zwischen Pythonobjekten und JSON zuständig. Zuerst müssen die JSON Daten in Pythonobjekte umgewandelt werden, dafür werden die Deserializer benutzt. Nachdem alle Daten zu den Sachverständigen ermittelt worden sind, werden die Pythonobjekte zu JSON umgewandelt, da die Daten als JSON an die Schnittstelle zurückgegeben werden sollen.

3.4.2 Schnittstelle

Wie in der Anforderungsanalyse erwähnt, sollen die gefundenen möglichen Sachverständigen, sortiert nach der besten Bewertung, als JSON an eine von Amazon Web Service betriebende Schnittstelle übergeben werden. Dort werden die übertragenden Daten von unseren Frontendteam ausgelesen und angezeigt.

Die Software benutzt die Google Distanz Matrix Schnittstelle, um die Distanz zwischen den Sachverständigen und der Adresse des Schadens zu ermitteln.

3.4.3 Benutzungsoberfläche

Die Software der Zuordnung von Schadensfällen zu Sachverständigen soll die Daten als JSON bereitstellen. Die Benutzeroberfläche wird vom Frontendteam implementiert. Diese soll alle relevanten Daten des Sachverständigen übersichtlich darstellen.

3.4.4 Tests

Die Software soll möglichst gut durch Tests auf Korrektheit geprüft werden. Es sollen Unit- und Integrationstest implementiert werden. Hierfür wird das Test Framework Pytest verwendet.

4 Vorbereitung der Daten

Die Daten, mit denen gearbeitet wird, müssen zuerst gewonnen werden und anschließend in eine konsistente Form gebracht werden.

4.1 Datengewinnung

Um eine datengetriebene Zuordnung zu realisieren, werden Daten benötigt. Die Daten, mit denen ich die Zuordnung realisiere, sind Primärdaten. Alle Schadensfälle, die claimsforce bisher bearbeitet hat, stehen mir zur Verfügung. Die Adressen der Sachverständigen, sowie die Postleitzahlengebiete, in welchen die Sachverständigen operieren möchten, wurden erhoben. Service Level Agreements ergeben sich aus der Zeitachse des Schadensfalles. Die Zeit von der Zuordnung des Termins bis zur Terminvereinbarung und die Zeit vom Termin, bis ein fertiger Bericht des Sachverständigen zu dem Schadensfall eingeht, kann man heranziehen, um zu überprüfen, wie schnell ein Sachverständiger Schadensfälle bearbeitet. Dies kann nach Schadenssegmenten gruppiert werden, wodurch eine Bewertung des Sachverständigen für die Art des Schadens durchgeführt werden kann.

Es wäre zudem noch möglich anhand der vorhandenen Daten eine gewissen Qualität der Berichte in die Zuordnung mit einfließen zu lassen. Berichte der Sachverständigen können zurückgeschickt werden, mit der Bitte diese noch einmal zu überarbeiten. Dies ist ein Indiz dafür, dass der Sachverständige Hilfe bei der Erstellung von Berichten benötigt oder, dass er flüchtig und unvollständig arbeitet. Wenn der Sachverständige jedoch sehr erfahren ist und schon viele Schadensfälle bearbeitet hat, kann ausgeschlossen werden, dass dieser Hilfe benötigt. Es kann dann eher davon ausgegangen werden, dass der Sachverständige flüchtig und unvollständig gearbeitet hat, wodurch die Bewertung des Sachverständigen negativ ausfällt.

Zudem werden wie in 3.2 erwähnt, seit jüngster Zeit, Daten über die Qualität des Berichtes erhoben.

4.2 Datenintegration

Die Daten, welche ich benötige, um mein Projekt umzusetzen, werden bei claimsforce in Google Sheets, mehreren Dynamo DB Tabellen und im Firestore gespeichert. Personenbezogene und Daten, welche für die Bezahlung relevant sind, werden in den Dynamo DB Tabellen verschlüsselt.

Bei claimsforce wird ein Alt-System betrieben. Dadurch liegen die Daten für die Schäden, die claimsforce im Alt-System bearbeitet hat, im Firestore. Dies führt zu einer komplexeren Integration aller Schadensfälle, da nicht nur Schäden aus einem System integriert werden müssen, sondern aus zwei verschiedenen Datenbanksystemen.

Die Regionen mit den entsprechenden Postleitzahlen, in denen der Sachverständige arbeiten möchte, wurden in der Datengewinnung in einem Google Sheet abgelegt. Dies ist eine Übergangslösung und soll in der Zukunft auch in einer Dynamo DB abgelegt werden. Die Sachverständigen werden wie die Schäden im Alt-System im Firestore gespeichert und im aktuellen System in einer Dynamo DB.

Mit der Hilfe eines in NodeJS geschriebenen Tools werden die Daten aus den verschiedenen Quellen ausgelesen und kombiniert.

Aus den ausgelesenden Daten werden nun Schadensobjekte und Sachverständigenobjekte erstellt.

Die Schadensobjekte besitzen folgende Attribute, welche für meine Arbeit relevant sind

- Id des Schadens
- Versicherungsunternehmen, welches den Schaden aufgenommen hat
- Art des Schadens
- Status des Schadens
- Zeitachse des Schadens
- Adresse des Schadens
- Termin des Schadens
- die Schätzung des Schadens (in Euro)

- Sachverständige, welcher den Schaden bearbeitet

Sachverständige besitzen folgende releavante Attribute

- Id des Sachverständigen
- Name des Sachverständigen
- Wohnort des Sachverständigen
- Regionen des Sachverständigen, in welchen dieser Schäden bearbeiten möchte.
- Alle Schäden, die von dem jeweiligen Sachverständigen bearbeitet wurden

Die erstellten Objekte werden aus dem Alt- und Neusystem mit Hilfe einer Funktion kombiniert und in einem Objektspeicherservice von Amazon exportiert. Über eine gesicherte URL von Amazon können die Daten in mein Programm eingelesen werden.

4.3 Datenprofiling

Beim Datenprofiling wurden der Datensatz mit der Hilfe von verschiedenen Analysemethoden untersucht.

Das erste Problem, welches direkt ersichtlich war, war, dass ein Alt-System betrieben wird, in welchem andere Sachverständigen IDs verwendet werden als in dem neuen System. Dadurch hatten viele alte Schäden eine alte Sachverständigen ID als beauftragten Sachverständigen. Ein weiteres Problem war die Spalte 'Danger', welches das Segment oder die Art des Schadensfalls beschreibt. Hier war zu erkennen, dass es uneinheitliche Schreibweisen für die gleiche Art gab, so zum Beispiel bei den Leerzeichen, die unterschiedlich gesetzt werden. Für die Konsistenz musste gesorgt werden, da anhand des Segments des Schadensfall, der Sachverständige zum Teil bewertet wird.

Bei einer Null-Wert-Analyse der Spalte "Adresse" der Sachverständigen wurde festgestellt, dass 10 von 60 Sachverständige keine Adresse angegeben hatten. Das nahezu gleiche Ergebnis kam bei den Postleitzahlengebieten heraus. 8/60 hatten keine Postleitzahlengebiete angegeben, in welchen die jeweiligen Sachverständigen operieren möchten.

4.4 Datenbereingung

Das Datenprofiling hilft einem durch verschiedene Anaylse-Methoden, Fehler in dem Datensatz zu finden. Diese Fehler werden, nachdem man dieses gefunden hat, bereinigt.

Durch die Sachverständigen IDs im alten System ist es notwendig, alle alten IDs der Sachverständigen auf die neuen IDs zu übertragen. Hierfür wurde eine einfach Methode implementiert, welche statisch alle alten Sachverständigen IDs auf die neue überschreibt.

Des Weiteren mussten alle alten Sachverständigen IDs als Beauftragter eines Schadens, durch die neuen Sachverständigen IDs ersetzt werden. Dafür wurde eine weitere Methode implementiert. Die Methode holt sich alle Schadensfälle, iteriert durch diese und überschreibt jeden Beauftragten mit der neuen ID des Beauftragten.

Die Spalte 'Danger', also die Art des Schadens, wurde vereinheitlicht. Für dieselbe Bezeichnung gab es unterschiedliche Zeichenketten. Die Spalte war nicht konsistent. Daher wurden alle Werte, welche den gleichen Inhalt hatten, auf einen Einheitswert übertragen. Zum Beispiel war in der Spalte 'Danger' ein Wert Wasserschaden / Rohrbruch und ein identischer Wert, Wasserschaden/ Rohrbruch. Diese beiden Werte wurden zum Beipsiel auf Wasserschaden/Rohrbruch übertragen, um für die Konsistenz zu sorgen.

Die Adressen und Bereiche, in welchen die Sachverständigen operieren möchten, wurden nachgepflegt. Um die Adressen der Sachverständigen einzupflegen, musste Einsicht in die Verträge zwischen dem Sachverständigen und claimsforce genommen werden. Für die Ermittlung der Postleitzahlengebiete der Sachverständigen mussten Telefonate geführt und Emails geschrieben werden.

5 Implementierung

In diesem Abschnit werde ich die Implementierung meiner Software, die Zuordnung eines Schadenfalles zu einem Sachverständigen, beschreiben. Zunächst erläutere ich Allgemeines zur Software, um dann anschließend darzulegen, wie die User Stories sowie die nicht-funktionalen Anforderungen umgesetzt worden sind.

5.1 Allgemeines

Die Software, welche ich entwickelt habe, wurde in Python implementiert. Die gesamte Software wurde mittels objektorientierter Programmierung umgesetzt. Die objektorientierte Programmierung ermöglicht, strukturiert Programme zu entwickeln. Reale Objekte können direkt im Programm abgebildet werden. Jedes Objekt ist durch seine Identität, sein Verhalten und seinem Zustand identifizierbar. Der Zustand eines Objektes setzt sich aus den Eigenschaften und Verbindungen zu anderen Objekten zusammen. Das Verhalten wird durch die Methoden beschrieben und die Identität unterscheidet ein Objekt von anderen Objekten, selbst wenn die Objekte den gleichen Zustand und dasselbe Verhalten aufweisen. (vgl. (Buyya, Selvi und Chu, 2000) (p.19))

5.2 Realisierung der User Stories

In diesem Abschnitt werde ich die Implementierung der einzelnden User Stories erläutern und beschreiben, wie ich diese in Python umgesetzt habe.

5.2.1 Zuordnung via Region

Siehe User Story: 1

Der erste Schritt des automatisierten Prozesses ist, mögliche Sachverständige anhand Ihrer Region zu finden. Siehe: 3.1.4

Die Zuordnung via Region ist in Python sehr einfach umzusetzen. Nachdem alle Schadensobjekte und Sachverständigenobjekte erstellt sind, kann der Schaden via der Schadens_ID gefunden werden. Der gefundende Schaden hat ein Attribut 'Adresse', dies ist ein Adressenobjekt, welches 3 Attribute besitzt.

- Line1 (Straße des Schadens)
- Zip_Code (Postleitzahl des Schadens)
- City (Stadt des Schadens)

Es wurde eine Methode geschrieben, welche alle Sachverständigen und das Postleitzahlengebiet des Schadens entegegen nimmt und ein Set von Sachverständigen zurück gibt. Ein Set in Python ist eine ungeordnete Sammlung von Datentypen, durch die man iterieren kann, die änderbar (mutable) ist und keine dopppelten Elemente enthält.

Diese Methode iteriert nun durch alle Sachverständigen und überprüft, ob das Postleitzahlengebiet des Schadens in der Liste der Postleitzahlengebiete des Sachverständigen vorhanden ist. Falls der Sachverständige in dem Postleitzahlengebiet operiert, wird er mit allen anderen Sachverständigen, welche in dem Postleitzahlengebiet arbeiten, als Set zurückgegeben.

Python-Code:

```
@staticmethod
@StaticTimer
def match_experts_by_zip_regions(experts: Set[Expert], zip_region: int)
  -> Set[Expert]:
    return {expert for expert in experts
            if zip_region in expert.zip_regions
           }
```

5.2.2 Zuordnung via Schäden in Bearbeitung

Siehe User Story: 2

Der zweite Schritt des automatisierten Prozesses besteht darin, mögliche Sachverständigen anhand ihrer Schäden in Bearbeitung zu erhalten. Siehe: 3.1.4

Die Zuordnung via Schäden in Bearbeitung ist komplexer. Hier benötigt man alle Sachverständigenobjekte mit deren zugehörigen Schadensfällen und wieder das Postleitzahlengebiet des Schadenfalls.

Eine Funktion wurde für diese Anforderung implementiert. Diese nimmt alle Sachverständigen und das Postleitzahlengebiet des Schadens als Parameter entgegen und gibt ein Set von Sachverständigen zurück.

Sachverständigenobjekte haben alle Schadensfälle, welche sie bei Claimsforce bearbeitet haben, als Attribut. In der Klasse des Sachverständigen wurde eine Methode implementiert, welche alle Schadensfälle, die im Status IN_INVESTIGATION sind, zurückgibt. Siehe: (5.2.4)

In der Klasse Appointment (Termin) gibt es eine Methode, welche überprüft, ob der Termin in der Zukunft liegt oder schon stattgefunden hat. Falls kein Termin gefunden wird, wird davon ausgegangen, dass noch kein Termin vereinbart worden ist und somit der Termin noch nicht stattgefunden hat.

Python-Code:

```
@property
def is_in_future(self) -> bool:
    if self.start_datetime is None:
        return True
    return datetime.now() < self.start_datetime
```

Des Weiteren wurde eine Methode in der Klasse des Sachverständigen implementiert, welche alle offenen Schadensfälle in der jeweiligen Region mit einem Termin in der Zukunft zurückgibt. Siehe: (5.2.4) In dieser Methode werden alle offenen Schadensfälle des Sachverständigen durchiteriert und überprüft, ob der Termin des offenden Schadensfall in der Zukunft liegt und ob die Region des Schadens, welcher der Sachverständige in Bearbeitung hat, dieselbe Region ist wie die Region des zuzuweisenden Schadensfalls.

Alle Sachverständigen, welche einen Schadensfall in der Schadensregion haben und bei denen der Termin in der Zukunft liegt, werden als Set zurückgegeben.

Python-Code:

```
@staticmethod
@StaticTimer
def match_experts_by_investigation(experts: Set[Expert], zip_region: int)
  -> Set[Expert]:
  return {expert for expert in experts
          if len(expert.get_open_claims_in_region(zip_region))
         }
```

5.2.3 Zuordnung via Distanz

Siehe User Story: 3

Die letzte Zuordnung von möglichen Sachverständigen im automatisierten Prozess ist die Zuordnung via Distanz. Siehe: (3.1.4)

Für dieses Zurordnung wird eine Schnittstelle benötigt, bei der man Anfragen über Distanzen zu mehreren Adressen stellen kann und diese in einer einzigen Anfrage erhält. Wenn für jeden Sachverständigen und zu der Schadensadresse eine einzelne Anfrage gestellt werden müsste und nachfolgend jedes Mal auf die Antwort der Schnittstelle gewartet werden müsste, wäre das Programm sehr langsam.

Ich habe mich für die Google Distance Matrix API entschieden. Mit der Googel Distanz Matrix ist es möglich die Distanzen von mehrere Adressen in nur einer Anfrage zu erhalten.

Für diese Zuordnung habe ich einen Service implementiert, welcher in der Infrastrukturschicht liegt. Dieser heißt GoogleVorhersageService. Dies habe ich aus folgendem Grund gemacht: Es könnte in der Zukunft eine passendere Schnittstelle von einem anderen Anbieter gefunden werden. Falls dies der Fall ist, können die Schnittstellen sehr leicht ausgetauscht werden. Den Rest der Implementierung müsste man nicht bearbeiten.

Der GoogleVorhersageService hat eine Funktion, welche die Adressen der Sachverständigen als Set und die Adresse des Schadens als Parameter entgegennimmt. Diese Funktion gibt ein Set von AdressenMitDistanzenobjekten zurück.

AdressenMitDistanzenobjekten bestitzen folgende Attribute:

- Line1 (Straße des Schadens)
- Zip_Code (Postleitzahl des Schadens)
- City (Stadt des Schadens)
- Distance_to_claim (Distanz von der Adresse zum Schaden)

Die Funktion extrahiert alle Adressen der Sachverständigen. Beim Extrahieren werden zwischen den Attributen Trennzeichen gesetzt und die einzelnen Attribute als Zeichenkette in einer Liste gespeichert. Das Trennzeichen zwischen den Attributen ist ein Komma. Nachdem alle Zeichenketten aus den Attributen der Adressen in einer Liste gespeichert sind, kann man die einzelnden Zeichenketten der Liste mit einem weiteren Trennzeichen, dem Querstrich, verknüpfen lassen. Die Attribute der Schadensadresse werden ebenso extrahiert und auch mit einem Komma voneinander separiert.

Gegeben sind zwei Adressen von Sachverständigen:

1. Adresse: Line1: Bauernrosenweg 17 zip_code: 22177 city: Hamburg
2. Adresse: Line1: Westfeld 12 zip_code: 21635 city: Jork

Zudem eine Schadensaddresse:

1. Schadensadresse: Line1: Am Sandtorkai 23/24 zip_code: 20457 city: Hamburg

Zuerst werden die Attribute der Adressen mit einem Komma ausgelesen.

List[0] = Bauernrosenweg 17, 22177, Hamburg

List[1] = Westfeld 12, 21635, Jork

Nun wird die Liste mit einem weiteren Trennzeichen verknüpft, wodurch man die folgende Zeichenkette erhält:

origin = Bauernrosenweg 17, 22177, Hamburg | Westfeld 12, 21635, Jork

destination = Am Sandtorkai 23/24, 20457, Hamburg

Nachdem die passenden Zeichenketten für die Google Distance Matrix API erstellt worden sind, kann die Anfrage gestartet werden. (Siehe Zeile: 17 in get_distance_matrix)

Um eine Anfrage zu starten, benötigt man einen Google Schnittstellenschlüssel. Die Anfragen an die Google Distance Matrix API kosten ab einer gewissen Anzahl an Anfragen Geld. Für 5$ erhält man 1000 Antworten der API.

Die Antwort der Schnittstelle ist ein JSON, welches folgendermaßen strukturiert ist:

- 'destination_adresses':Am Sandtorkai 23/24, 20457, Hamburg
- 'origin_adresses':'Bauernrosenweg 17, 22177, Hamburg', 'Westfeld 12, 21635, Jork'
- 'rows':
  - 'elements':
  - 'distance':
    - 'text':'23 minutes'
    - 'value':8600,
- 'status':'OK'
  - 'elements':
  - 'distance':
    - 'text':'45 minutes',
    - 'value':33800
- 'status':'OK'

In der Reihenfolge, wie die Adressen in die Schnittstelle gegeben werden, kommen die Antworten auch von Google zurück. Daher ist es möglich, die vorher erstellte Liste der Adressen zur Erstellung von AdressenMitDistanzobjekten zu benutzen. (Siehe Zeile: 7). Nach Prüfung des Status der Antwort, iterieren wir durch die 'rows' der Antwort von der Google Distance Matrix API und erstellen AdressenMitDistanzobjekten (Siehe Zeile: 33). Da in Python eine Liste eine geordnete Menge von Werten ist, ist sichergestellt, dass die richtige Adresse der Sachverständigen zur jeweiligen richtigen Distanz der Antwort gehört. Daher wird der Eintrag aus der Adressenliste vom Index X genutzt, um die Adresse zu erhalten und derselbe Index X genutzt, um die dazugehörige Distanz aus der Antwort von der Google Distance Matrix API auszulesen. Somit erhalten wir AdressenMitDistanzobjekt.

```
def get_distance_matrix(self, experts_addresses: Set[Address], claim_address:
  -> Set[AddressWithDistance]:
      destination = claim_address.line1 + "," +
                    claim_address.zip_code + "," +
                    claim_address.city

      addresses = [expert_address.line1 + "," +
                   expert_address.zip_code + "," +
                   expert_address.city
                   for expert_address in experts_addresses
                   if expert_address is not None]
      separator = '|'
      source = separator.join(addresses)

      url = 'https://maps.googleapis.com/maps/api/distancematrix/json?'

      response = requests.get(url + 'origins=' + source +
                              '&destinations=' + destination +
                              '&key=' + self._api_key)
      response = response.json()
      tmp_addresses = set()
      for i in range(0, len(response['rows'])):
          line1 = addresses[i].split(',')[0]
          zip_code = addresses[i].split(',')[1]
          city = addresses[i].split(',')[2]

          if response['rows'][i]['elements'][0]['status'] == 'OK':
              distance_to_claim =
               response['rows'][i]['elements'][0]['distance']['value'] / 1000
          else:
              distance_to_claim = None

          address_with_distance =
             AddressWithDistance(line1=line1, zip_code=zip_code, city=city,
                                 distance_to_claim=distance_to_claim)
```

```
36
37              tmp_addresses.add(address_with_distance)
38
39          return tmp_addresses
```

Nachdem das Set der AdressenMitDistanzenobjekten erstellt und zurückgegeben wurde, wird ein Dictionary erstellt. Ein Dictionary in Python ist eine Sammlung, die ungeordnet, änderbar und indizierbar ist. In Python haben Dictionaries Schlüssel und Werte. Das Dictionary, welches erstellt wird, hat als Schlüssel den Sachverständigen und als Wert das Objekt AdressenMitDistanz. Um das Dictionary zu erstellen wird durch alle Sachverständigen iteriert und eine Überprüfung auf die Gleichheit der Adresse des Sachverständigen und aller Adressen des Sets AdressenMitDistanzenobjekten durchgeführt.(Siehe Zeile: 16) Falls die Adressen identisch sind, wird ein Dictionary erstellt, welches als Schlüssel den aktuellen Sachverständigen hat und als Wert das gefundene AdressenMitDistanzobjekt.(Siehe Zeile: 19)

```
1  def get_distance(self, experts: Set[Expert], claim_address: Address)
2    -> Dict[Expert, AddressWithDistance]:
3
4          addresses = {expert.address for expert in experts}
5
6          addresses_with_distances =
7              self.get_distance_matrix(experts_addresses=addresses,
8                                       claim_address=claim_address)
9
10         experts_with_distances = dict()
11         for expert in experts:
12             if expert.address is not None:
13                 distance_to_claim_set =
14                     {address_with_distance
15                     for address_with_distance in addresses_with_distances
16                     if address_with_distance.__eq__(expert.address)
17                     }
18                 if len(distance_to_claim_set) > 0:
19                     experts_with_distances[expert] =
20                         distance_to_claim_set.pop()
```

```
21                  else:
22                      experts_with_distances[expert] = None
23              else:
24                  experts_with_distances[expert] = None
25
26          return experts_with_distances
```

Nun sind alle Werte vorhanden, die benötigt werden, um Sachverständige auf Grundlage der Distanz zur Schadensadresse zu finden.

Hierfür wurde nun eine Methode implementiert, die das Dictionary von Sachverständigen zu den AdressenMitDistanzobjekten als Parameter entgegennimmt und ein Set von möglichen Sachverständigen zurückgibt. In der Methode wird durch das Dictionary durchiteriert und überprüft, ob die Distanz des Sachverständigen weniger als 100km zum Schadensfall beträgt. Falls dies der Fall ist, wird der Sachverständige zum Set der möglichen Sachverständigen für diesen Schadensfall hinzugefügt.

```
@staticmethod
 @StaticTimer
 def match_experts_by_distance(
   experts_with_distances: Dict[Expert, AddressWithDistance])
  -> Set[Expert]:

     return {expert for expert in experts_with_distances
               if experts_with_distances[expert] is not None
               and experts_with_distances[expert].distance_to_claim < 100
             }
```

5.2.4 Bewertung der möglichen Experten

Siehe User Story: 4

Nachdem alle Sachverständigen ermittelt worden sind, welche für den Schadensfall in Frage kommen, sollen diese bewertet werden. Siehe: 3.1.4

Zuerst werden die drei Sets der gefundenden Sachverständigen miteinander vereinigt. Falls ein Sachverständiger in zwei oder allen der drei Sets enthalten ist, ist durch das Set

sichergestellt, dass dieser nur einmal in der Liste der vereinigten möglichen Sachverständigen vorhanden ist. Ein Set enthält keine doppelten Elemente.

Nun wird durch alle möglichen Sachverständigen durchiteriert und dabei werden die Daten, welche für die Bewertung vonnöten sind, berechnet und erfasst.

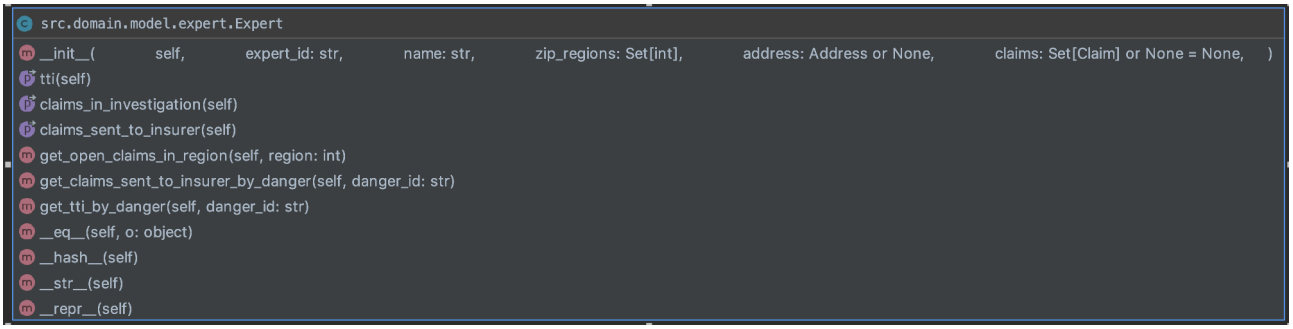
Für die Bewertung wurde eine Klasse Prediction erstellt. Alle möglichen Sachverständigen werden in Predictionobjekte konvertiert.

Diese Klasse hat folgen Attribute:

1. Sachverständigen_ID
2. Name des Sachverständigen
3. Ist der Fall in der Region, in welcher der Sachverständige operieren möchte?
4. TTI des Sachverständigen
5. TTI des Sachverstädingen in der gleichen Schadensart, wie der zuzuweisende Schadensfall
6. Distanz zum Schadens
7. Anzahl der Schadensfälle, die der Sachverständige bearbeitet hat
8. Anzahl der Schadensfälle, die der Sachverständige bearbeitet hat und in der gleichen Schadensart, wie der zuzuweisende Schadensfall liegt
9. Anzahl der Schadensfalle, die der Sachverständige in Bearbeitung hat
10. Alle Schadensfälle, die der Sachverständige in der Region des zuzuweisenden Schadensfall hat und bei denen der Termin noch nicht stattgefunden hat

Um die möglichen Sachverständigen in Predictionobjekte umzuwandeln, wurden mehrere Methoden in die Klasse des Sachverständigen implementiert. Da Sachverständige alle Ihrere Schadensfälle, welche Sie bearbeitet haben, als Attribut haben, kann man diese Daten leicht berechnen und erfassen. Diese Methoden möchte ich kurz vorstellen. Sachverständigenklasse mit Methoden:

Abbildung 5.1: Sachverständigen Klasse



Das erste wichtige Merkmal bei der Bewertung ist die Angabe, ob der Sachverständige in dem Postleitzahlengebiet des Schadensfall operieren möchte. Es wird geprüft, ob das Postleitzahlengebiet des Schadenfalls mit einem der gewünschten Postleitzahlengebiete des Sachverständigen übereinstimmt.

Ein weiterer wichtige Parameter für die Bewertung eines Sachverständigen ist die Distanz zwischen dem Sachverständigen und dem Schadensfall. Durch das Dictionary von Sachverständigen zu AdressenMitDistanzobjekten ist es einfach, die Distanz zu erhalten. Man benötigt lediglich den dazugehörigen Sachverständigen, um das AdressenMitDistanzobjekt auszulesen und die Distanz des Sachverständigen zum Schadensfall zu erhalten.

Wie die weiteren wichtigen Prameter für die Bewertung (Siehe Attribute der Predictionklasse: 1) ermittelt werden, möchte ich erläutern.

Methode claims_sent_to_insurer

Um die Anzahl der Schadensfälle herauszufinden, die der Sachverständige bearbeitet hat, wurde die Methode claims_sent_to_insurer implementiert. Diese Methode gibt alle Schadensfälle zurück, die dem Sachverständigen zugeordnet sind und die den Status: 'SENT_TO_INSURER' haben. Nachdem man die Schadensfälle hat, kann mit der Abfrage der Größe des Sets, die Anzahl der Schadensfälle bestimmen werden, die der Sachverständige bearbeitet hat.

```
@property
def claims_sent_to_insurer(self) -> Set[Claim]:
```

```
  return {claim for claim in self.claims
          if claim.status == Status.SENT_TO_INSURER.value
         }
```

Methode get_claims_sent_to_insurer_by_danger

Die Anzahl der Schadenfälle, welche der Sachverständige in der gleichen Schadensart bearbeitet hat wie der zuzuweisende Schadensfall, wird mittels einer Methode ermittelt. Diese Methode gibt alle Schadensfälle des Sachverständigen zurück, welche den Status: 'SENT_TO_INSURER' haben und die, die die gleiche Schadensart wie der zuzuordnende Schaden haben.

```
def get_claims_sent_to_insurer_by_danger(self, danger_id: str) -> Set[Claim]:
  return {claim for claim in self.claims_sent_to_insurer
          if claim.danger_id == danger_id
         }
```

Methode claims_in_investigation

Um die Anzahl der Schadensfälle zu erhalten, welche der Sachverständige in Bearbeitung hat, wurde eine weitere Methode implementiert. Diese ist nahezu identisch mit der Methode: claims_sent_to_insurer, jedoch wird hier nicht auf den Status: 'SENT_TO_-INSURER' geprüft, sonder auf den Status: 'IN_INVESTIGATION'.

```
@property
def claims_in_investigation(self) -> Set[Claim]:
  return {claim for claim in self.claims
          if claim.status == Status.IN_INVESTIGATION.value
         }
```

Methode get_open_claims_in_region

Eine weitere Methode wurde implementiert, welche alle Schadensfälle des Sachverständigen, die in der Region des zuzuweisenden Schadensfall liegen und bei welchem der Termin in der Zukunft liegt, ermittelt und zurückgibt.

```
def get_open_claims_in_region(self, region: int) -> Set[Claim]:
  return {
      claim for claim in self.claims_in_investigation
      if claim.is_appointment_in_future
      and claim.address.zip_region == region
      }
```

Methode tti

Die 'time to invoice' des Sachverständigen wird ermittelt, indem die Summe des Attributes time_to_invoice aller Schadensfälle des Sachverständigen aufsummiert werden und dann durch die gesamte Anzahl der Schadensfälle des Sachverständigen dividiert wird.

```
@property
def tti(self) -> float:
  return sum([
    claim.time_to_invoice for claim in self.claims
    ]) / max(len(self.claims), 1)
```

Methode tti_by_danger

Die 'time to invoice' des Sachverständigen in der gleichen Schadensart wie die des zuzuweisenden Schadensfalls wird durch eine Methode ermittelt. Diese Methode erstellt ein Set des Attributes time_to_invoice, nimmt jedoch nur die Schadensfälle, welche in derselben Schadensart liegen wie der zuzuweisende Schadensfall. Nachdem das Set erstellt ist, kann die Summe über das Set gebildet werden und durch die Anzahl der Schadensfälle dividiert werden. So erhält man den Durchschnitt der Leistung des Sachverständigen in einer spezifischen Schadensart.

```
def get_tti_by_danger(self, danger_id: str) -> float:
  expert_claims = {claim.time_to_invoice for claim in self.claims
                   if claim.danger_id == danger_id
                  }
  return sum(expert_claims) / max(len(expert_claims), 1)
```

Methode prediction

Die Daten für die Bewertung des Sachverständigen sind nun vollständig. Zuerst wird überprüft, ob der mögliche Sachverständige für den Schadensfall in der Schadensregion operieren möchte und ob dieser offene Schadensfälle in der Region des zuzuweisenden Schadensfalls, mit einem Termin in der Zukunft, hat. Falls dies der Fall ist, wird eine Variable namens prediction auf 20 gesetzt. Falls der Sachverständige in der Region operiert, jedoch keinen aktuellen Schadensfall in der Region hat, wird die Variable prediction auf 5 gesetzt. Falls beide Möglichkeiten nicht zutreffen, ist die Variable prediction standardmäßig auf 1.

Nun wird überprüft, ob der Sachverständige Schadensfälle in der gewissen Schadensart bearbeitet hat. Falls dies der Fall ist wird der 'time to invoice' Wert der Schadesart genommen. Ansonsten wird der durchschnittliche gesamte 'time to invoice' Wert des Sachverständigen für die Berechnung der Bewertung benutzt.

Die Bewertung wird folgendermaßen berechnet:

Im Zähler der Berechnung sind Werte, die, je größer sie sind, eine bessere Bewertung bedeuten. Im Nenner der Berechnung sind Werte, welche je größer sie sind, eine schlechtere Bewertung nach sich ziehen. Daher sind folgende Werte im Zähler der Berechnung:

- prediction
- Anzahl der Schadensfälle, die der Sachverständige bearbeitet hat

Im Nenner sind folgende Werte:

- 'time to invoice'
- Anzahl der Schadensfälle, welcher der Sachverständige aktuell bearbeitet
- Distanz zwischen dem Wohnort des Sachverständigen und dem Schaden

Die Formel der Berechnung ist folgendermaßen:

```
prediction = (prediction + self.count_of_claims_sent_to_insurer_by_danger)
             / (self.time_to_invoice_by_danger +
             len(self.claims_in_investigation) + distance) * 100
```

Ergebnisse als JSON bereitstellen

Siehe User Story: 5

Die Predicitionobjekte können nun in JSON umgewandelt werden und an das Frontendsystem übergeben werden. Dies ist der letzte Prozessschritt der Software. Siehe 3.1.4

Hierfür wurde für alle relevanten Klassen ein Serializer implementiert, welche das jeweilige Objekt als Zeichenkette, in der Form von JSON zurückgeben. JSON ist ein kompaktes Datenformat, welches sich gut für den Austausch von Daten zwischen Anwendugen eignet.(T. Bray, 2017)

```
@staticmethod
def get_json_from_predictions(predictions: List[Prediction]) -> [str, Any]:
    return [
        {
            "id": prediction.expert_id,
            "name": prediction.name,
            "prediction": prediction.prediction,
            "countOfSentToInsurerClaims":
                prediction.count_of_claims_sent_to_insurer,
            "countOfSentToInsurerClaimsByDanger":
                prediction.count_of_claims_sent_to_insurer_by_danger,
            "desiredZipRegion": prediction.is_desired_zip_region,
            "avgTimeToInvoice": prediction.time_to_invoice,
            "avgTimeToInvoiceByDanger": prediction.time_to_invoice_by_danger,
            "claimsInInvestigation": list(
                ClaimSerializer.claim_serializer(claim)
                 for claim in prediction.claims_in_investigation
                 ),
            "claimsInZipRegion": list(
                ClaimSerializer.claim_serializer(claim)
                 for claim in prediction.claims_in_investigation_per_region
                 ),
            "distanceInKm": prediction.distance_in_km
        } for prediction in predictions
    ]
```

5.3 Nicht-Funktionale Anforderungen

5.3.1 Funktionalität

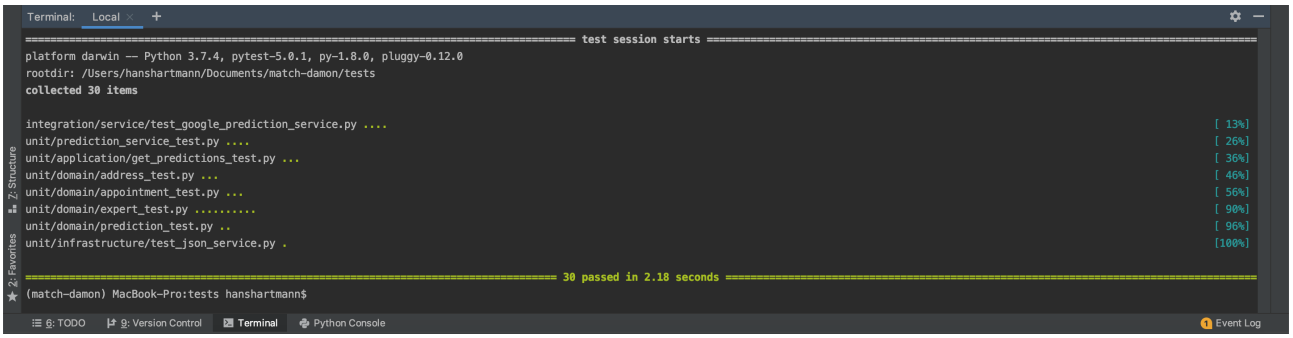
Die Software hat alle Funktionen, die in der Anforderungsanalyse erfasst worden sind, implementiert. Alle User Stories wurden umgesetzt und sind ausführbar.

5.3.2 Zuverlässigkeit

Die Software wird im Amazon Web Service eingesetzt. Falls ein Laufzeitfehler auftritt, werden alle relevanten Mitarbeiter per E-mail informiert. Dadurch werden Fehler sofort sichtbar und die Beseitigung des Fehlers kann direkt geschehen.

Um die Zuverlässigkeit zu steigern, wurden, wie im Entwurf erwähnt, Unit sowie Integrations Tests implementiert. Diese stellen die Funktionalitäten der Software sicher.

Abbildung 5.2: Tests



Um zu Überprüfen, wie die Abdeckung der Tests ist wurde ein Python Plugin benutzt. Dies heißt pytest-cov. Dieses Plugin misst, wieviele Anweisungen des Sourcecodes durch Tests abgedeckt sind.

Pfad	Statements	Miss	Prozent der Abdeckung
match-damon/src/application/use_case/get_predictions.py	18	0	100%
match-damon/src/domain/model/address.py	16	2	88%
match-damon/src/domain/model/address_with_distance.py	11	0	100%
match-damon/src/domain/model/appointment.py	14	2	86%
match-damon/src/domain/model/claim.py	38	3	92%
match-damon/src/domain/model/expert.py	42	3	93%
match-damon/src/domain/model/prediction.py	39	5	87%
match-damon/src/domain/service/prediction_service.py	62	8	87%
match-damon/src/infrastructure/service/google_prediction_service.py	30	0	100%
match-damon/src/application/repository/claim_repository.py	10	2	80%
match-damon/src/application/repository/expert_repository.py	9	1	89%

Tabelle 5.1: Anweisungsabdeckung

Wie aus der Tabelle zu entnehmen, wurde eine Anweisungsabdeckung der relevanten Klassen von über 80% gemessen. Bei einigen Klassen liegt diese bei 100%. Insgesamt wurden mehr als 90% der Anweisungen im Sourcecode durch Tests ausgeführt.

5.3.3 Benutzbarkeit

Wie in der Anforderungsanalyse erwähnt, muss der Mitarbeiter lediglich zwei Knöpfe betätigen, um einen Schaden einem Sachverständigen zuzuordnen. Dadurch, dass der Mitarbeiter keinen Text eingeben muss, ist die Software ziemlich benutzerfreundlich und es können fehlerhafte Texteingaben ausgeschlossen werden. Jedoch kann es passieren, dass ein Mitarbeiter aus Versehen einem "falschen" Sachverständigen einen Schaden zuweist. Dazu kann es kommen, wenn der Mitarbeiter in die falschen Zeile rutscht und dort den Knopf "Schaden zuweisen" betätigt. Dieser Fehler lässt sich jedoch leicht bereinigen; der Mitarbeiter muss nur den zugewiesenden Schadensfall von dem Sachverständigen abziehen.

Siehe Screenshot Benutzungsoberfläche: 6.1

5.3.4 Effizienz

Die Software darf nicht länger als 10 Sekunden für eine Anfrage brauchen. In Python ist es möglich mit einem Decorator, jede Funktion auf Ihre Performance zu überprüfen. Decorator erlauben einem neue Funktionalitäten zu einer bestehenden Methode hinzuzufügen, ohne die Struktur der Methode zu ändern. Hierfür wird eine Klasse implementiert, welche die auszuführende Methode als Parameter im Konstuktor entgegennimmt. Danach implementiert man die Python Klassen-Methode __call__, welche aufgerufen wird, wenn eine Instanz der Klasse aufgerufen wird. In dieser __call__ Methode wird die Zeit des Startpunktes erfasst, danach wird die auszuführende Methode ausgeführt und nach der Ausführung der Methode wird ein zweites mal die Zeit erfasst. Nun ist es möglich, durch einen Vergleich der Start- und Endzeit die Ausführungszeit der Methode zu erfassen und diese sich anzeigen zu lassen.

```
class StaticTimer:

    def __init__(self, function):
        self.function = function

    def __call__(self, *args, **kwargs):

        environment = os.environ.get('ENVIRONMENT')
        t1 = time.time()
        result = self.function(*args, **kwargs)
        t2 = time.time() - t1
        if environment != 'Production':
            print('{} ran in: {} sec'.format(self.function.__name__, t2))

        return result
```

Ein Decorator kann in Python mit Hilfe des “@” Symbols und des Namens der Klasse über eine Methode geschrieben werden, wodurch die Decorator Klasse ausgeführt wird, wenn die Methode aufgerufen wird. Siehe: 5.2.3

Im Durchschnitt lag die Messung des Programmes bei 4,5 Sekunden. Durch die zeitliche Messung der einzelnden Methoden konnten einige Performance Engpässe erkannt und beseitigt werden.

Je mehr Schadensfälle und Sachverständige hinzukommen, desto anspruchsvoller wird es werden, die Performance des Programmes beizubehalten. Mit steigender Anzahl an Daten, steigt auch die Zeit für die Erstellung und Bewertung der möglichen Sachverständigen für den Schadensfall. Jeden Tag kommen neue Schadensfälle hinzu, daher sollte man die Performance des Programmes im Auge behalten und gelegentlich neue Messung durchführen und diese auswerten.

5.3.5 Änderbarkeit

Durch die Clean Code Architektur ist das Programm sehr leicht änderbar. Neue Anwendugsfälle können problemlos implementiert werden. Die Schnittstellen oder Datenbanken, welche genutzt werden, können einfach ausgetauscht werden. Es muss lediglich eine Klasse bearbeitet werden. Siehe: 3.4.1 Die Erweiterbarkeit ist in diesem Fall auch sehr wichtig, da immer weitere Geschäftsregeln und spätere individuelle Kriterien einzelner Versicherungsunternehmen hinzukommen werden.

5.3.6 Übertragbarkeit

Wie in der Anforderungsanalyse erwähnt, sollen die ermittelten und berechneten Daten an eine Schnittstelle übergeben werden. Auf diese Schnittstelle kann jedes System zugreifen. Daher ist es möglich, die ermittelten Daten zu den Sachverständigen mit jedem System zu erfassen und mit diesen Daten einen geeigneten Sachverständigen zu einem Schadensfall zu finden.

Falls das Programm nicht mehr über den Amazon Web Service eingesetzt werden sollte, muss sichergestellt sein, dass auf den auszuführenden System die richtige Python Version mit allen Abhängigkeiten installiert ist. Die Abhängigkeiten und die Python Version sind im Projekt in der Datei “Pipfile” gespeichert.

5.3.7 Tracking

Die Auswahl des Mitarbeiters, welcher den Schadensfall einem Sachverständigen zuweist, kann nachverfolgt werden. Die Liste von möglichen Sachverständigen ist nach der Bewertung sortiert. Dadurch ist es möglich zu überprüfen mit welchem Prozentsatz vorgeschlagene Sachverständige ausgewählt worden sind. Falls es dort Abweichung gibt und zum

Beispiel sieht, dass häufig ein Sachverständiger ausgewählt wurde, welcher nur auf den hinteren Plätzen der Bewertung liegt, kann die Formel für die Bewertung der Sachverständigen überarbeitet werden. Zudem kann eine Analyse, warum ein Sachverständiger, welcher trotz schlechterer Bewertung ausgewählt worden ist, erfolgen. Das Ergebnis des Trackings sieht folgendermaßen aus:

Abbildung 5.3: Tracking

Event Label	Unique Events ↓	Event Value
	260 % of Total: 28.82% (902)	242 % of Total: 72.67% (333)
1. 0062479a-59b6-46ca-854c-b0013a2c2f16	2 (0.77%)	1 (0.41%)
2. 01d02f04-7886-4240-a449-7d335a67e69c	2 (0.77%)	1 (0.41%)
3. 03e2ee2b-8710-4eb0-8dcb-086f24f0ba50	2 (0.77%)	3 (1.24%)
4. 054c5f1d-c3ed-4bfd-b91e-43df90a66c8e	2 (0.77%)	3 (1.24%)
5. 067676ae-3acd-4f53-9db6-992bec67e5df	2 (0.77%)	2 (0.83%)
6. 06cbe4aa-9904-40ac-a5ac-76fe160c9e12	2 (0.77%)	2 (0.83%)
7. 0c90f1f0-143e-4e66-b1a8-489334c5949c	2 (0.77%)	2 (0.83%)
8. 0d4fc09c-0d82-4c09-9447-0be4548e18eb	2 (0.77%)	1 (0.41%)
9. 0df09dfd-a0c5-4444-a127-ced80620e124	2 (0.77%)	3 (1.24%)
10. 0fc9116f-cc34-45a3-aee3-aa66a74cf821	2 (0.77%)	3 (1.24%)

Das Event-Label zeigt die Schadens ID für den zugeordneten Schadensfall. Die Unique Events sind die Anzahl der Events, die stattgefunden haben. Ein Event ist immer eine Aktion. Diese Aktion könnte zum Beispiel sein, dass ein Schadensfall einem Sachverständigen zugeordnet worden ist. Zudem sind die Annahme oder Ablehnung des Schadenfalls durch den Sachverständigen Events. Im Idealfall sollten immer zwei Unique Events eintreten, die Zuordnung des Schadenfalls und die Annahme des Schadensfalls durch den Sachverständigen. Das Event Value ist die Auswahl des vorgeschlagenen Sachverständigen. Hier wird ersichtlich, welcher Sachverständige aus der sortierten Liste ausgewählt wurde.

6 Bewertung und Ausblick

In diesem Abschnitt erfolgt die Bewertung der Arbeit und ein kurzer Ausblick, wie weiter zu verfahren sein wird.

6.1 Bewertung

Die datengetriebene Zurordnung von Sachverstädingen zu Schadensfällen ist erfolgreich implementiert worden. Die Mitarbeiter, welche die Software benutzen, sind sehr zufrieden und begrüßen die Zeitersparnisse. Zudem erleben die Mitarbeiter es als Gewinn, dass sie sich nicht mehr durch diverse Dokumente durcharbeiten müssen, um lediglich ein paar Daten zu den Sachverständigen zu erhalten. Da claimsforce ein wachsendes StartUp ist, wird es bei steigender Zahl an Sachverständigen und Schadensfällen immer schwieriger die Daten für einen einzelnden Sachverständigen zu ermitteln. Die Automatisierung des Prozesses durch die Implementierung in Python war sehr vonnöten und ist erfolgreich umgesetzt worden. Die einfache Handhabung via einem Klick eine Liste aller Sachverständigen zu erhalten, die für den Schadensfall in Frage kommen, wird von den Mitarbeitern als große Erleichterung bei der Ausführung ihrer Arbeit gesehen. Auch die Möglichkeit alle relevanten Daten der möglichen Sachverständigen mit einem Blick sehen zu können ist ein zeitsparender und arbeitserleichternder Gewinn. Dieser Gewinn wird durch die Automatisierung des Zuordnungsprozesses in Python ermöglicht. Die gewonnene Zeit kann von den Mitarbeitern nun für produktivere Arbeiten genutzt werden.

Die Benutzungsoberfläche für die Mitarbeiter sieht folgendermaßen aus:

Abbildung 6.1: Benutzungsoberfläche

An der Formel für die Bewertung eines Sachverständigen muss noch weiter gearbeitet werden. Die Analyse des Trackings hat ergeben, dass bei circa 35 Prozent der Zuordnungen der erste Sachbearbeiter ausgewählt wurde, der zweite zu circa 30 Prozent und alle Sachbearbeiter, welche schlechter bewertet worden sind, circa 35 Prozent der Fälle ausmachen.

Aufgrund des Trackings, ist es möglich, die Schadensfälle mit ihren Zuordnungen zu analysieren. Dies ermöglicht einem die Hintergründe zu erfahren, warum ein Sachverständiger ausgewählt wurde, welcher nicht als bester bewertet worden ist.

Dass in nur 35% der Fälle ein Sachverständiger vom ersten Platz ausgewählt wurde, kann damit zusammenhängen, dass die Daten, die ich zur Verfügung hatte nicht alle vergangene Daten des Sachverständigen beinhaltet haben. Ich hatte nur Zugriff auf die Schadensfälle, welche bei claimsforce bearbeitet worden sind. Dies sind im Moment circa 2000 Schadensfälle. Es wäre jedoch wünschenswert gewesen, wenn alle vergangene Schadensfälle und alle aktuellen Schadensfälle des Sachverständigen von allen Versicherungsunternehmen zur Verfügung gestanden hätten. Dadurch wäre zum einen eine genauere Bewertung der Sachverständigen möglich, da dann die gesamte Expertise des Sachverständigen vorgelegen und es mehr Informationen über die Qualität ihrer Arbeit gegeben

hätte. Zum anderen würde der Zugriff auf alle Schadensfälle, die der Sachverständige bei allen Versicherungsunternehmen in Bearbeitung hat, eine Erleichterung für die Planung der Fahrrouten bedeuten. Eine Implementierung von der Erstellung von Fahrtrouten wurde noch nicht umgesetzt. Dies wäre jedoch wünschenswert, da dies die Produktivität der Sachverständigen erheblich steigern würde. Die Sachverständigen könnten dadurch Fahrtwege kombinieren und würden deutlich an Zeit sparen. Hierfür werden jedoch, wie bereits erwähnt, mehr Daten benötigt.

Zudem wäre es eine Bereicherung, wenn die Daten über die Qualität der Berichte, die die Sachverständigen erstellt haben, vorhanden wären. Dies ermöglicht eine umfassendere Bewertung der Sachverständigen. Die 'time to invoice' gibt einen Überblick, wie schnell ein Sachverständiger einen Schadensfall bearbeitet hat, jedoch gibt die Qualität des Berichtes einen Einblick, wie gut der Sachverständige gearbeitet hat.

Die Formel für die Bewertung könnte mit den gewünschten Daten überarbeitet werden. Wie in 5.2.4 erwähnt, sind Werte, je größer sie sind und eine bessere Bewertung nach sich ziehen im Zähler und Werte, je größer sie sind und eine schlechtere Bewertung bedeuten, im Nenner. Durch den Zugriff auf alle Schadensfälle, die ein Sachverständiger bearbeitet hat, würde wie oben erwähnt die Expertise genauer bewertet werden können, wodurch die Formel der Bewertung noch genauer werden würde.

Die Daten über die Qualität der Berichte der Sachverständigen würden im Nenner mit der Anzahl der Schadensfälle, die der Sachverständige bearbeitet hat und der Variable prediction addiert werden. Je besser ein Bericht bewertet wurde, desto besser ist der Sachverständige geeignet. Die Qualität der Berichte der Sachverständigen lassen sich nach den jeweiligen Schadensarten gruppieren. Dadurch lässt sich sehr gut erschließen, wie gut sich der Sachverständige in der Schadensart des zuzuordnenden Schaden auskennt.

Nach einer Änderung der Formel sollte überprüft werden, ob sich die Positionen der zugeordneten Sachverständigen verbessern. Dies ist durch das Tracking sehr leicht umzusetzen. Die aktuelle Statistik, für die Auswahl der Sachvertändigen, die an erstere Stelle stehen, sollte sich verbessern.

6.2 Ausblick

Im Moment gibt es noch keine Daten zu der Verfügbarkeit der Sachverständigen. Claimsforce speichert aktuell nicht, ob Sachverständige krank oder im Urlaub sind. Dies wird

in naher Zukunft implementiert. Wenn diese Daten vorhanden sind, ist die Prüfung, ob der Sachverständige überhaupt zur Verfügung steht, sehr wichtig. Die Zuweisung eines Schadensfalls bei Krankheit oder Urlaub kann dann nicht erfolgen. Hierfür sind einige folgende zusätzliche Fragen bedeutsam:

- Soll ein Sachverständiger einen Tag vor seinem Urlaub noch einen Schadensfall zugewiesen bekommen?
- Soll ein Sachverständiger am letzten Tag seines Urlaubes Schadensfälle zugewiesesn bekommen?
- oder macht es Sinn einem Sachverständigen nur Schadensfälle zuzuweisen, wenn er im Moment zur Verfügung steht.

Wenn ein Sachverständiger einen Tag vor seinem Urlaub einen Schadensfall zugewiesen bekommt, ist es nahezu undenkbar, dass dieser es schafft, an einem Tag den Termin mit dem Versicherten zu vereinbaren, diesen wahrzunehmen und einen Bericht zu verfassen, welcher noch am selben Tag abgeschickt wird. Daher würde ich diesen Fall ausschließen und einem Sachverständigen, welcher am nächsten Tag in Urlaub fährt nicht berücksichtigen.

Am letzten Tag eines Urlaubes können nur Schadensfälle zugewiesen werden, welche nicht von aktueller Relevanz sind. Wasserschäden und Rohrbrüche würde ich ausklammern, da bei diesen Schadensfällen eine sofortige Handlung vonnöten ist. Bei Schadensfällen, welche keine sofortige Handlung benötigen wäre dies denkbar. Die einfachste Variante wäre, einem Sachverständigen nur Schadensfälle zuzuweisen, solange dieser zur Verfügung steht und den Schadensfall noch vor seinem Urlaub bearbeiten und fertigstellen kann.

Die Daten über die Qualität der Berichte der Sachverständigen können bald genutzt werden. Diese Daten sollen hilfreich sein für eine aussagekräftigere Bewertung.

Des Weiteren haben verschiedene Versicherungsunternehmen verschiedene Anforderungen an die Zuordnung von Sachverständigen zu Schadensfällen. Die unterschiedlichen Anforderungen wurden noch nicht implementiert. Ein Versicherungsunternehmen, welches mit claimsforce zusammenarbeitet, hat eigene Sachverständige und möchte zudem das Netzwerk von Sachverständigen, welche extern für claimsforce arbeiten, in Anspruch nehmen. Für die Allokation von Sachverständigen zu Schadensfällen möchte dieses Versicherungsunternehmen die geschätze Schadenssumme als wichtigen Faktor mit aufnehmen. Schäden, welche eine größere Schadenssumme als 10 000 Euro aufweisen, sollen nur an

die eigenen Sachverständigen des Versicherungsunternehmen vermittelt werden. Schäden, die kleiner als 10 000 Euro sind, sollen externen claimsforce Sachverständige zugeordnet werden. Falls die eigenen Sachverständigen des Versicherungsunternehmen ausgelastet sind, dürfen auch die externen claimsforce Sachverständigen einen Schadensfall über 10 000 Euro für das Versicherungsunternehmen bearbeiten.

Um die Belastung der Sachverständigen herauszufinden, könnte ein Algorithmus implementiert werden. Dieser würde die 'time to invoice' der Sachverständigen analysieren und bewerten, wie die 'time to invoice' sich verhält, wenn dieser X Schäden in Bearbeitung hat und einen weiteren Schaden zugewiesen bekommen würde. Dies wäre ein weiterer relevanter Parameter für die Bewertung des Sachverständigen, da man abschätzen könnte, wie sich die 'time to invoice' mit einem Schadensfall mehr, entwickelt.

Je mehr Versicherungsunternehmen mit claimsforce zusammenarbeiten und die Tools von claimsforce benutzen, desto mehr Anforderungen und Bedingungen für die Allokation von Sachverständigen für einen Schadensfall werden hinzukommen.

7 Zusammenfassung

Der manuelle Prozess der datengetriebenen Zuordnung von Sachverständigen zu einem Schadensfall ist sehr zeitinsiv und erfordert eine erhebliche Expertise des Mitarbeiters, welcher die Schadensfälle an die Sachverständigen zuordnet. Des Weiteren muss der Mitarbeiter sich durch unzählige Dokumente arbeiten, um die relevanten Daten der Sachverständigen zu erhalten. Der automatisierte Prozess, welcher in dieser Arbeit in Python implementiert wurde, lässt die Zuordnung von einem Sachverständigen zu einem Schadensfall einerseits schneller durchführen und andererseits müssen die Mitarbeiter sich nicht mehr durch unzählige Dokumente durcharbeiten und keine Expertise über die Sachverständigen haben. Zudem werden mehrere Faktoren der Zuordnung beim automatisierten Prozess berücksichtigt. Der manuelle Prozess beeinhaltete lediglich die Zuordnung via der Regionen, in welchen die Sachverständigen operieren möchten. Falls daraufhin kein Sachverständiger gefunden worden ist, wird die Distanz zwischen dem Schadensfall und dem Wohnort des Sachverständigen oder der Expertise des Mitarbeiters über die Sachverständigen genutzt, um den Schadensfall einem Sachverständigen zuzuordnen. Durch die gestiegende Zahl der Zuordnungsverfahren infolge des automatisierten Prozesses, erhält der zuzuordnende Mitarbeiter eine größere Auswahl an einsetzbaren Sachverständigen und neue Erfahrungswerte über einzelne Sachverständige. Die neuen Erfahrungswerte ergeben sich aus den angezeigten relevanten Informationen zu jedem vorgeschlagenden Sachverständigen. Die neuen Zuordnungsverfahren wären ohne eine Automatisierung nicht möglich umzusetzen, daher war die Automatisierung des Prozesses vonnöten. Die gefundenen Sachverständigen werden nach ihrern bisherigen und aktuellen Schadensfällen bewertet, dadurch hat der Mitarbeiter eine noch bessere Übersicht, welcher Sachverständiger für den Schadensfall gut geeignet ist und kann infolgedessen eine bessere und schnellere Auswahl treffen.

Literaturverzeichnis

[Apel u. a. 2010] Apel, D ; Behme, W ; Eberlein, R: *Datenqualität erfolgreich steuern Praxislösungen für Business-Intelligence-Projekte*. Hanser Verlag, 2010. – ISBN 9783446425019

[Buyya u. a. 2000] Buyya, R D. ; Selvi, T D. ; Chu, Mr. X.: *Object Oriented Programming with Java: Essentials and Applications*. Addison Wesley Verlag, 2000. – ISBN 9783827312822

[Danesh und Kock 2005] Danesh, A ; Kock, N: An experiment study of process representation approaches and their impact on perceived modeling quality and redesign success Business Process Management. (2005)

[Frank und Prasse 1997] Frank, U ; Prasse, M: Zur Standardisierung objektorientierter Modellierungssprachen: Eine kritische Betrachtung des State of the Art am Beispiel der Unified Modeling Language. (1997). – URL https://www.researchgate.net/publication/242568639_Zur_Standardisierung_objektorientierter_Modellierungssprachen_Eine_kritische_Betrachtung_des_State_of_the_Art_am_Beispiel_der_Unified_Modeling_Language

[Gaitanides 1983] Gaitanides, M: *Prozessorganisation. Entwicklung, Ansätze und Programme prozessorientierter Organisationsgestaltung*. Verlag Vahlen, 1983. – ISBN 978-3-800-60991-8

[Hesse und Mayr 2008] Hesse, W ; Mayr, H: Modellierung in der Softwaretechnik - eine Bestandsaufnahme. (2008). – URL https://link.springer.com/article/10.1007/s00287-008-0276-7

[IBM] IBM: Schreiben eines Berichts zur Datenbereinigung. . – URL https://www.ibm.com/support/knowledgecenter/de/SS3RA7_15.0.0/com.ibm.spss.crispdm.help/writing_cleaning_report.htm

[Ki Hyun Tae 2019] Ki Hyun Tae, Young Hun Oh Hyunsu Kim Steven Euijong W.: Data Cleaning for Accurate, Fair and Robust Models : A Big Data - AI Integration Approach. (2019). – URL https://arxiv.org/pdf/1904.10761.pdf

[Martin 2000] Martin, Robert C.: *Pattern-orientierte Software-Architektur*. Addison Wesley Verlag, 2000. – ISBN 9783827312822

[Martin 2018] Martin, Robert C.: *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Pearson Education, Inc., 2018. – ISBN 978-0-13-449416-6

[Maydanchik 2007] Maydanchik, A: *Data Quality Assessment*. Technics Publications, 2007. – URL https://play.google.com/books/reader?id=g43XBgAAQBAJ&hl=de&printsec=frontcover&source=gbs_atb_hover&pg=GBS.PA17. – ISBN 978-0-9771400-2-2

[Meerkamm 2011] Meerkamm, s: *Ein Rahmenwerk für das Prozessdesign zur Identifikation, Klassifikation und Umsetzung von Anforderungen*, Universität Bayreuth, Dissertation, 2011. – URL https://epub.uni-bayreuth.de/253/1/Dissertation_Meerkamm.pdf

[Mittelstraß 1995] Mittelstrass, J: *Mittelstraß, J. Enzyklopädie Philosophie und Wissenschaftstheorie*. J.B. Metzler, 1995. – ISBN 978-3-476-02103-8

[Naumann und Leser 2006] Naumann, F ; Leser, U: *Informationsintegration: Architekturen und Methoden zur Integration verteilter und heterogener Datenquellen*. dpunkt.verlag GmbH, 2006. – URL https://www.dpunkt.de/openbooks/informationsintegration.pdf

[Rossak 2013] Rossak, I: *Datenintegration*. Hanser Verlag, 2013. – ISBN 978-3-446-43221-5

[Schwarzer und Krcmar 1995] Schwarzer, B ; Krcmar, H: *Grundlagen der Prozessorientierung - eine vergleichende Untersuchung in der Elektronik- und Pharmaindustrie*. Deutscher Universitätsverlag, 1995. – URL https://www.springer.com/de/book/9783824461363. – ISBN 978-3-322-95452-7

[T. Bray 2017] T. Bray, Ed: The Javascript Object Notation. (2017). – URL https://tools.ietf.org/pdf/rfc8259.pdf

[Vom Brocke 2003] Vom Brocke, J: *Referenzmodellierung Gestaltung und Verteilung von Konstruktionsprozesse*. Logos Verlag Berlin, 2003. – URL https://www.researchgate.net/profile/Jan_vom_Brocke/

publication/295549962_Referenzmodellierung_Gestaltung_und_Verteilung_von_Konstruktionsprozessen_2_Auflage/links/589b0c02aca2721f0db428db/Referenzmodellierung-Gestaltung-und-Verteilung-von-Konstruktionsprozessen-2-Auflage.pdf. – ISBN 978-3-8325-0179-2

[Wandt 2015] Wandt, Dr H.: Datenqualität. (2015). – URL http://www.digitalwiki.de/datenqualitaet/

Glossar

Business Process Modeling Notation Prozessmodellierungssprache.

Data Warehouse Eine für Analysezwecken zentrale Datenbank, die Daten aus mehreren Datenquellen zusammenführt.

Ereignisgesteuerte Prozesskette Prozessmodellierungssprache.

Unified Modelling Language Modellierungssprache.

time to invoice Die Dauer der Bearbeitung eines Schadens.

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „– bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] – ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen."

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: _____

Vorname: _____

dass ich die vorliegende Bachelorarbeit – bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Datenbasierte Zuordnung von Schadensfällen zu Sachverständigen in der Versicherungsbranche

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

_____	_____	_____
Ort	Datum	Unterschrift im Original