JOHANNES KEPLER
UNIVERSITÄT LINZ

Eingereicht von
Lukas Frieske, BSc

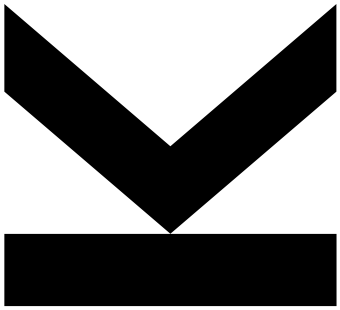
Angefertigt am
Institut für Wirtschaftsinformatik – Data & Knowledge Engineering

Beurteiler
o.Univ.-Prof. DI Dr. Michael Schrefl

Mitbetreuung
Ass.-Prof. Mag. Dr. Christoph Schütz

Oktober 2019

ENTWICKLUNG EINES DATA-CLEANING-PROZESSES FÜR DAS VOESTALPINE TREASURY



Masterarbeit

zur Erlangung des akademischen Grades

Master of Science

im Masterstudium

Wirtschaftsinformatik

JOHANNES KEPLER UNIVERSITÄT LINZ
Altenberger Straße 69
4040 Linz, Österreich
www.jku.at

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Masterarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die vorliegende Masterarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Ort, Datum

Unterschrift

Abstract

Data is one of the most valuable assets of a company. Nevertheless, the usage of standard software often leads to inconsistencies in a company's data due to insufficient consistency checks, negatively affecting data quality. In this master`s thesis, a data cleaning process was developed for the voestalpine Treasury to address the mentioned problem. The proposed data cleaning process examines the voestalpine Treasury`s data on a more detailed level than the built-in checking mechanisms of the standard software and therefore ensures appropriate data quality. A software system that enables the automated checking of the voestalpine Treasury`s data was implemented as a component of the data cleaning process. The software system utilizes business rules to identify inconsistencies in the data. The business rules were derived from the knowledge of the voestalpine Treasury`s domain experts and subsequently documented in natural language. Due to the structural weaknesses of natural language such as ambiguity, the business rules were formally represented using the Semantics of Business Vocabulary and Rules (abbr. SBVR). To enable the execution of the business rules by the software system, the SBVR business rules were transformed into corresponding SQL queries. The results of the automated checks of the software system are presented in a report document, which is accessible to the voestalpine Treasury`s employees and subsequently serves as guidance for the cleaning of the data. A first application of the data cleaning process proposed in this master`s thesis already led to the identification and elimination of a substantial number of inconsistencies, thus increasing data quality at voestalpine Treasury.

Kurzfassung

Daten sind von essenzieller Bedeutung für ein Unternehmen. Beim Einsatz einer Standardsoftware besteht jedoch die Gefahr, dass aufgrund von zu allgemeinen Überprüfungsmechanismen Inkonsistenzen im Datenbestand entstehen, welche die Datenqualität negativ beeinflussen. Um dies zu verhindern, wurde im Rahmen dieser Masterarbeit für das voestalpine Treasury ein Data-Cleaning-Prozess entwickelt, welcher den Datenbestand auf einer detaillierteren Ebene als die eingesetzte Standardsoftware überprüft und folglich die notwendige Datenqualität im voestalpine Treasury sicherstellt. Für die automatisierte Überprüfung der Datenbestände wurde als Teil des Data-Cleaning-Prozesses ein Softwaresystem implementiert, welches, unter Zuhilfenahme von Geschäftsregeln, Inkonsistenzen im Datenbestand identifiziert. Die Geschäftsregeln wurden im ersten Schritt nach den Vorgaben der Domänenexperten des voestalpine Treasury in natürlicher Sprache abgebildet. Aufgrund der strukturellen Schwachstellen der natürlichen Sprache, wie beispielsweise Mehrdeutigkeiten, wurden die Geschäftsregeln unter Verwendung von Semantics of Business Vocabulary and Rules (Abk. SBVR) formal abgebildet. Um die anschließende Verwendung durch das Softwaresystem zu ermöglichen, wurden die SBVR-Geschäftsregeln in die Form von SQL-Abfragen transformiert. Die Ergebnisse der automatisierten Überprüfung der Datenbestände werden den Mitarbeitern des voestalpine Treasury in Form eines Reports zur Verfügung gestellt, welcher die Grundlage für die Bereinigung des Datenbestands darstellt. Eine erste Anwendung des in dieser Masterarbeit beschriebenen Data-Cleaning-Prozesses im voestalpine Treasury führte bereits zur Identifikation und Bereinigung einer beträchtlichen Anzahl an Inkonsistenzen, wodurch die Datenqualität im voestalpine Treasury verbessert wurde.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Das voestalpine Treasury stellt die Inhouse-Bank des voestalpine Konzerns dar und ist unter anderem für die Sicherstellung der Liquidität sowie für die Steuerung von Zins-, Währungs- und Rohstoffpreisrisiken verantwortlich. Zur Ermittlung des Liquiditätsbedarfs, der Abschätzung der Zins-, Währungs- und Rohstoffpreisrisiken sowie zur Verwaltung der eingesetzten Finanzinstrumente, bedient sich das voestalpine Treasury einer Standardsoftware. Bei der Abbildung der Realität, unter Verwendung einer Standardsoftware, besteht jedoch die Gefahr, dass aufgrund von zu allgemeinen Überprüfungsmechanismen Inkonsistenzen im Datenbestand entstehen. Die für die richtige Durchführung der Geschäftsprozesse notwendige Datenqualität kann, durch die im Umfang der Standardsoftware enthaltenen Konsistenzprüfungen, nur unzureichend gewährleistet werden. Folglich können Inkonsistenzen, bei den auf den Datenbestand des voestalpine Treasury aufbauenden Auswertungen und Analysen, zu falschen Schlüssen führen. Dies kann aufgrund des Volumens der Transaktionen zu erheblichen Schäden für den gesamten voestalpine Konzern führen. Um die notwendige Datenqualität nachhaltig sicherstellen zu können, muss ein Data-Cleaning-Prozess entwickelt werden.

Das Problem inkonsistenter Datenbestände beim Einsatz von Standardsoftware wird durch folgendes Beispiel verdeutlicht. Im voestalpine Treasury werden alle Konzerngesellschaften, und die für die jeweiligen Konzerngesellschaften verantwortlichen Sachbearbeiter, in der Standardsoftware abgebildet. Von der Standardsoftware wird nur unzureichend geprüft, ob der verantwortliche Treasury-Sachbearbeiter ordnungsgemäß erfasst wurde. Wird eine Massenbenachrichtigung zur Meldung der Liquiditätsdaten versendet und der zuständige Sachbearbeiter scheint nicht, oder nur inkorrekt, im System auf, so führt dies zu einer fehlenden Meldung der Liquiditätsdaten. Dies kann bei Nichtentdeckung massive Auswirkungen auf die Liquiditätsplanung haben. Trotz des Einsatzes von detaillierten Handbüchern, welche die Geschäftsprozesse und die Abbildung der notwendigen Objekte in der Standardsoftware beschreiben, besteht das Risiko der fehlerhaften Eingabe von Daten durch die zuständigen Sachbearbeiter. Eine systemgestützte Überwachung der Datenbestände ist daher von allgemeinem Interesse.

Ziel dieser Masterarbeit ist es, einen Data-Cleaning-Prozess für das voestalpine Treasury zu entwickeln, welcher den Datenbestand und dessen Datenqualität auf einer detaillierteren Ebene als die eingesetzte Standardsoftware überprüft. Folglich soll eine Verbesserung der Datenqualität des unternehmensinternen Datenbestands, und somit die Steigerung der Zuverlässigkeit der darauf aufbauenden Analysen und Auswertungen erzielt werden.

Für die Durchführung dieser Masterarbeit wurden seitens des voestalpine Treasury Rahmenbedingungen vorgegeben, welche den Handlungsspielraum abgrenzen. Die derzeit im voestalpine Treasury eingesetzte technische Infrastruktur sowie die etablierten Geschäftsprozesse dürfen nicht abgeändert werden. Folglich sind Eingriffe an der zugrundeliegenden Datenbank, insbesondere in Bezug auf das eingesetzte Datenbankschema, nicht möglich. Des Weiteren können die eingesetzten (Standard-)Softwarelösungen und deren Programmlogik nicht abgeändert werden und stellen somit im Zuge dieser Masterarbeit eine weitere Vorgabe dar. Abschließend muss sichergestellt werden, dass der Einsatz des besagten Softwaresystems die verbleibende technische Infrastruktur, und im Weiteren die Geschäftsprozesse des voestalpine Treasury, in keiner Form negativ beeinflusst.

Um das bestehende Problem zukunftsorientiert zu beseitigen, wurden im Rahmen dieser Masterarbeit, unter Berücksichtigung der erwähnten Rahmenbedingungen, folgende Tätigkeiten durchgeführt.

Analyse des State-of-the-Art und konzeptionelle Entwicklung der Prozessarchitektur. In Abschnitt 2 wurde untersucht, wie vergleichbare Problemstellungen in der Literatur behandelt werden. Folglich wurden Methoden und Techniken identifiziert, welche zur Lösung der Problemstellung dieser Masterarbeit beitragen. Die Ergebnisse aus dieser Analyse wurden für die Umsetzung der weiteren Tätigkeiten im Rahmen dieser Masterarbeit verwendet. Des Weiteren wurde die Architektur des Data-Cleaning-Prozesses und dessen Komponenten festgelegt. Die Architektur des Data-Cleaning-Prozesses wird in Abschnitt 3 behandelt.

Festlegung und Abbildung der Konsistenzbedingungen. In Abschnitt 4 wird auf die Abbildung der Konsistenzbedingungen des voestalpine Treasury in Form von Geschäftsregeln eingegangen. Da die unternehmensinternen Konsistenzbedingungen in natürlicher Sprache, in Form von verschriftlichten Bedingungen, wie beispielsweise Handbüchern, oder in Form von Wissen der Domänenexperten vorlagen, mussten diese vorab formal abgebildet werden. Eben erwähntes Regelwerk wird anschließend im Zuge der Datenüberprüfungen vom Data-Cleaning-Prototyp verwendet. Im Rahmen dieser Masterarbeit wurde ein Teilbereich der Datenbestände des voestalpine Treasury, der Bereich „Master-Data-Management“ ausgewählt, um ein Beispielset an Geschäftsregeln zu erstellen, da eine ganzheitliche Untersuchung der umfangreichen Datenbestände den Rahmen überschreiten würde. Folglich fokussiert sich das erwähnte Regelwerk ausschließlich auf den ausgewählten Teilbereich.

Entwicklung eines Data-Cleaning-Prototyps und Erstellung des Data-Cleaning-Reports. Des Weiteren wurde ein Prototyp als Komponente des Data-Cleaning-Prozesses erstellt, welcher die automatisierte Überprüfung der Datenbestände des voestalpine Treasury abwickelt. Das Werkzeug ermöglicht es dem Anwender eine Prüfung der Datenbestände, auf der Grundlage der im vorherigen Absatz erwähnten Geschäftsregeln durchzuführen und umfassende Rückmeldung, in Form eines Data-Cleaning-Reports, über etwaige Problemfälle mit Handlungsbedarf, zu erhalten. Die Funktionalität des Data-Cleaning-Prototyps wird in Abschnitt 5 präsentiert. Der erwähnte Data-Cleaning-Report stellt das Endprodukt des Data-Cleaning-Prozesses dar und wird in Abschnitt 6 genauer betrachtet.

2 State-of-the-Art

In diesem Kapitel werden die Ergebnisse der State-Of-The-Art-Analyse aus der Literatur präsentiert. Das Kapitel wird in drei Unterkapitel gegliedert, welche sich auf drei Hauptgebiete der Masterarbeit konzentrieren. Im ersten Unterkapitel werden relevante Aspekte aus dem Bereich Data-Science behandelt, da diese Masterarbeit auf die Verbesserung der Datenqualität im voestalpine Treasury abzielt, und somit in diesem Bereich einzuordnen ist. Im zweiten Unterkapitel werden die Ergebnisse bezüglich der Dokumentation und der Abbildung von Geschäftsregeln diskutiert, da das gesammelte Wissen des voestalpine Treasury bezüglich der Konsistenz der unternehmensinternen Datenbestände mittels Geschäftsregeln umgesetzt wird. Im dritten Unterkapitel werden die unterschiedlichen Lösungswege bezüglich der Implementierung eines Data-Cleaning-Prototypen untersucht, da für das voestalpine Treasury ein prototypisches Softwaresystem entwickelt wurde, um die Prüfung der Datenbestände auf Basis der Geschäftsregeln durchzuführen.

2.1 Datenqualität und Data-Cleaning

Datenqualität wird in der Literatur oftmals als Grad der Eignung der Daten, für die Verarbeitung durch Anwendungen definiert [1]. Ein hoher Grad der Eignung wird folglich als hohe Datenqualität, ein niedriger Grad als schlechte Datenqualität bezeichnet. Die Merkmale, welche die Datenqualität beeinflussen, können somit von Unternehmen zu Unternehmen abweichen [1]. Welche Kriterien ausschlaggebend für die Datenqualität in einem Unternehmen sind, muss von facheinschlägig ausgebildeten Experten aus der jeweiligen Domäne definiert werden [1]. Fehler in den Datenbeständen eines Unternehmens wirken sich negativ auf die Datenqualität aus. Mangelnde Datenqualität kann unter Umständen signifikante wirtschaftliche Auswirkungen auf ein Unternehmen haben, folglich ist eine Sicherstellung hoher Datenqualität von allgemeinem Interesse [1]. Datenbestände, welche dazu tendieren eine Vielzahl an Fehler, wie beispielsweise Duplikate oder fehlende Werte zu beinhalten, werden als Daten schlechter Qualität bezeichnet. Um die Qualität der Datenbestände nachhaltig sicherstellen zu können, werden Methoden aus dem Bereich Data-Cleaning eingesetzt werden. Dabei zielt das Data-Cleaning auf das Identifizieren und Entfernen von Fehlern und Inkonsistenzen aus den Datenbeständen ab [2].

Rahm und Do [2] unterscheiden im Bereich Data-Cleaning auf Grund der unterschiedlichen Anforderungen zwischen Anwendungsfällen mit nur einer Datenquelle, wie beispielsweise eine einzelne Datenbank, und Anwendungsfällen mit mehreren Datenquellen. Im Falle mehrerer Datenquellen, wie oft bei einem Data Warehouse beobachtet werden kann, ist der Schritt des Data-Cleanings sogar noch wichtiger als im erwähnten Anwendungsfall mit nur einer Datenquelle [2]. Im Rahmen dieser Masterarbeit werden bei verteilten Geschäftsregeln (siehe Abschnitt 4.5) zwar externe Datenquellen kontaktiert, jedoch dienen diese lediglich zur Abfrage von Referenzwerten, und stellen somit kein Untersuchungsobjekt für die Data-Cleaning-Schritte dar. Folglich werden alle Data-Cleaning-Schritte ausschließlich an der Oracle Datenbank des voestalpine Treasury angewandt. Daher wird der Fokus im Bereich Data-Cleaning auf Probleme und Maßnahmen mit einer einzelnen Datenquelle gelegt. Abbildung 1 stellt die Klassifizierung der Probleme im Bereich Datenqualität nach Rahm und Do [2] grafisch dar.

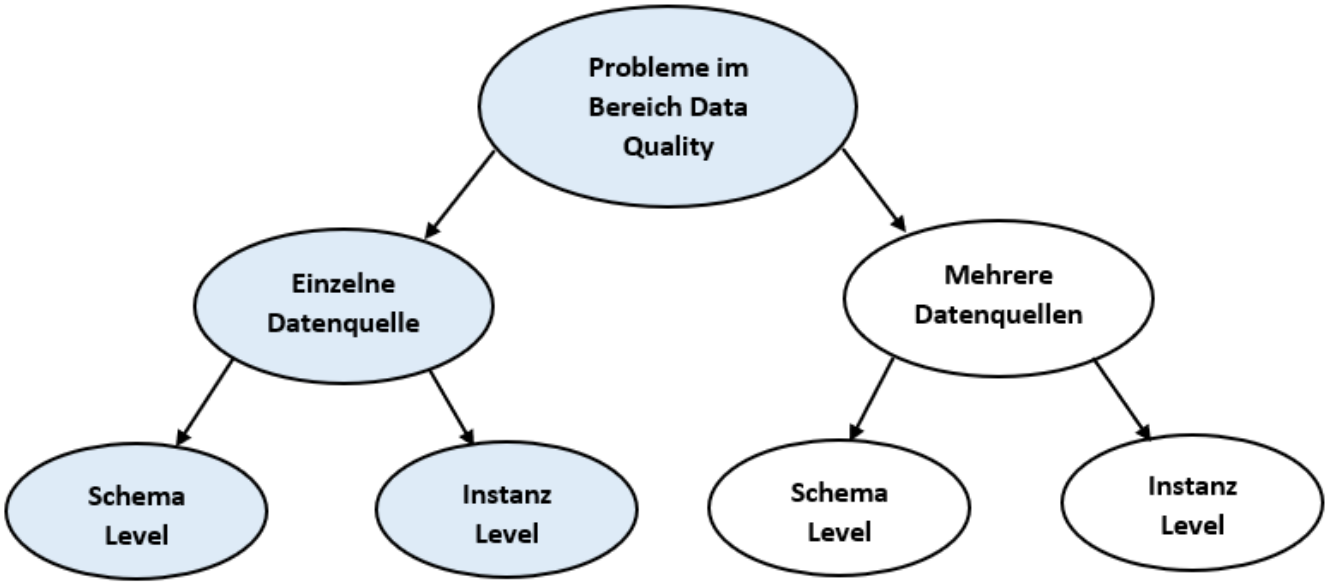


Abbildung 1: Klassifizierung der Probleme im Bereich Datenqualität nach Rahm und Do [2] S.3

Wie bereits im vorherigen Absatz erwähnt, werden im Rahmen dieser Masterarbeit nur lediglich die blau markierten Objekte aus Abbildung 1 behandelt. Ist nur eine einzelne Datenquelle ausschlaggebend für das Data-Cleaning, so sind die notwendigen Maßnahmen von der Beschaffenheit der Datenquelle abhängig. Rahm und Do [2] zufolge, steigt die Datenqualität in Datenquellen, welche ein Schema und Integritätsmechanismen verwenden, signifikant an. In Datenquellen, welche komplett ohne Schema bzw. Integritätsmechanismen arbeiten, wie beispielsweise herkömmliche Dateien, kann die Eingabe von Informationen nur schwer gesteuert werden, wodurch die Wahrscheinlichkeit der Eingabe von fehlerhaften Daten steigt. Werden Daten in einer Datenbank abgelegt, wie beispielsweise eine relationale Datenbank im Falle des voestalpine Treasury, so werden durch das Datenmodell bereits Restriktionen umgesetzt. Bei relationalen Datenbanken werden somit beispielsweise Referentielle Integrität oder einfache Attribut- bzw. Spaltenwerte, bereits durch das verwendete Datenmodell sichergestellt [2]. Da im Normalfall kein Endanwender eine direkte Verbindung zur Datenbank verwendet, und dort Daten einspeist, löscht, oder ändert, können ebenfalls durch die zuständige Applikation, welche die Abläufe Richtung Datenbank steuert, Integritätsbedingungen umgesetzt werden. Folglich können Probleme bezüglich Datenqualität auf Schema Level wie in Abbildung 1 dargestellt, durch mangelhafte Umsetzung von Integritätsbedingungen im Bereich des Datenmodells oder im Bereich der Applikation auftreten [2]. Laut Rahm und Do [2], können die Problempunkte im Bereich Datenqualität sowohl auf Schema- als auch Instanz-Level, in den Kategorien Attribut/Spalte, Datensatz, Datensatz-Typ und Quelle eingeordnet werden. Alle Problempunkte bezüglich Datenqualität auf Instanz-Level können nicht präventiv auf Schema-Level unterbunden werden, wie beispielsweise Schreibfehler bei der Eingabe. Tabelle 1 zeigt Beispiele für den besprochenen Sachverhalt auf Schema-Level. Ähnlich zu den Problempunkten auf Schema-Level, dargestellt in Tabelle 1, beschreiben Rahm und Do [2] verschiedene Probleme auf Instanz-Ebene wie in Tabelle 2 dargestellt.

Kategorie	Problem	Beispieldatensatz	Fehlergrund
Attribut	**Unerlaubter Wert**	bdate=30.13.63	Wert außerhalb des Wertebereichs
Datensatz	**Verletzung von Attribut-Abhängigkeiten**	Alter=21; bdate =12.02.63	Abhängigkeit „Alter= Aktuelles Datum – Geburtsdatum“ verletzt
Datensatz-Typ	**Eindeutigkeit (engl. Uniqueness) verletzt**	emp_1=(name=”John Walker”, SSN=”**123456**”) emp_2=(name=”Peter Way”, SSN=”**123456**”)	Eindeutigkeit der Sozialversicherungsnummer (SSN) verletzt
Quelle	**Referentielle Integrität verletzt**	emp=(name=”John Smith”, deptno=**127**)	Referenziertes Objekt (127) nicht definiert

Tabelle 1: Probleme bezüglich Datenqualität auf Schema-Level nach Rahm und Do [2]

Kategorie	Problem	Beispieldatensatz	Fehlergrund
Attribut	**Fehlender Wert**	tel=0000-000000	Platzhalterwerte oder Null-Werte
	Schreibfehler bei der Eingabe	Stadt=“Liinz“	(Recht-)Schreibfehler
	Unverständliche Werte, Abkürzungen	Erfahrung=“B“ Tätigkeit=“DB Prog“	
	(falsch) eingebettete Werte	Name=“J. Miller 12.02.63 New York”	Mehrere Attributwerte in einem Attribut (Bsp. Kommentarfeld)
	Attributwerte in falschen Attributen	Stadt=”Deutschland”	
Datensatz	**Verletzung von Attribut-Abhängigkeiten**	Stadt=”Linz” , plz=”1010”	Attribute Stadt und plz (Postleitzahl) müssen übereinstimmen
Datensatz-Typ	**Verletzung der Eingabeform**	$name_1$= “J. Walker”, $name_2$=”Way P.”	Üblich bei Freiformfeldern, Verletzung der Eingabekonvention
	Duplikate (Datensätze)	emp_1=(name=”John Walker”,...); emp_2=(name=”J. Walker”,...)	Mitarbeiter fälschlicherweise doppelt abgelegt, durch Eingabefehler
	Widersprüchliche Datensätze	emp_1=(name=”John Walker”, bdate=**12.02.63**); emp_2=(name=”John Walker”, bdate=**12.03.63**)	Dieselbe Entität aus der realen Welt ist durch verschiedene Datensätze abgebildet
Quelle	**Falsche Referenz**	emp=(name=”John Walker”, deptno=17)	Referenziertes Objekt ist angelegt, jedoch falsch

Tabelle 2: Probleme bezüglich Datenqualität auf Instanz-Level nach Rahm und Do [2]

Aufgrund des hohen Aufwands zur Bereinigung von Datenbeständen, weisen Rahm und Do [2] auf die Wichtigkeit von Mechanismen zur Prävention der Eingabe von fehlerhaften Daten hin. Folglich ist die Entwicklung eines den Anforderungen entsprechenden Datenmodells sowie adäquate Integritätsbedingungen von allgemeinem Interesse. Zusätzlich zu den Vorkehrungen auf Seiten der Datenablage, wie beispielsweise einer relationalen Datenbank, müssen von der zuständigen Applikation, welche den Zugriff auf die Datenbank steuert, weitere notwendige Integritätsmechanismen umgesetzt werden [2]. Neben Rahm und Do [2] weisen auch Swapna et al. [3] in ihrer Publikation unter den verschiedenen Typen von fehlerhaften Daten (engl. Dirty Data) auf nahezu dieselben Problempunkte bezüglich Datenqualität, wie in Tabelle 1 und Tabelle 2 dargestellt, hin. Des Weiteren erwähnen Prasad et al. [4] in ihrer Konferenzpublikation eine Untermenge der Problempunkte bezüglich Datenqualität als Fehlertypen (engl. Error Types). Im Rahmen der Analyse des State-of-the-Art wurden noch weitere Publikationen wie [5], [6] identifiziert, welche signifikante Übereinstimmungen mit den Ergebnissen von Rahm und Do [2] aufweisen, wobei [6] exakt dieselbe Einordnung vornahm.

2.2 Business-Rule-Engines

Die Abbildung und Umsetzung von Geschäftsregeln, um den Datenbestand eines Informationssystems möglichst fehlerfrei zu halten, haben sich in der Vergangenheit als eine vielversprechende Strategie etabliert [7]. Im Zuge der Umsetzung, werden Softwaresysteme, sogenannte Business-Rule-Engines, eingesetzt. Eine vereinfachte Darstellung nach Chisholm [8] der allgemeinen Architektur von Business-Rule-Engines ist in Abbildung 2 dargestellt.

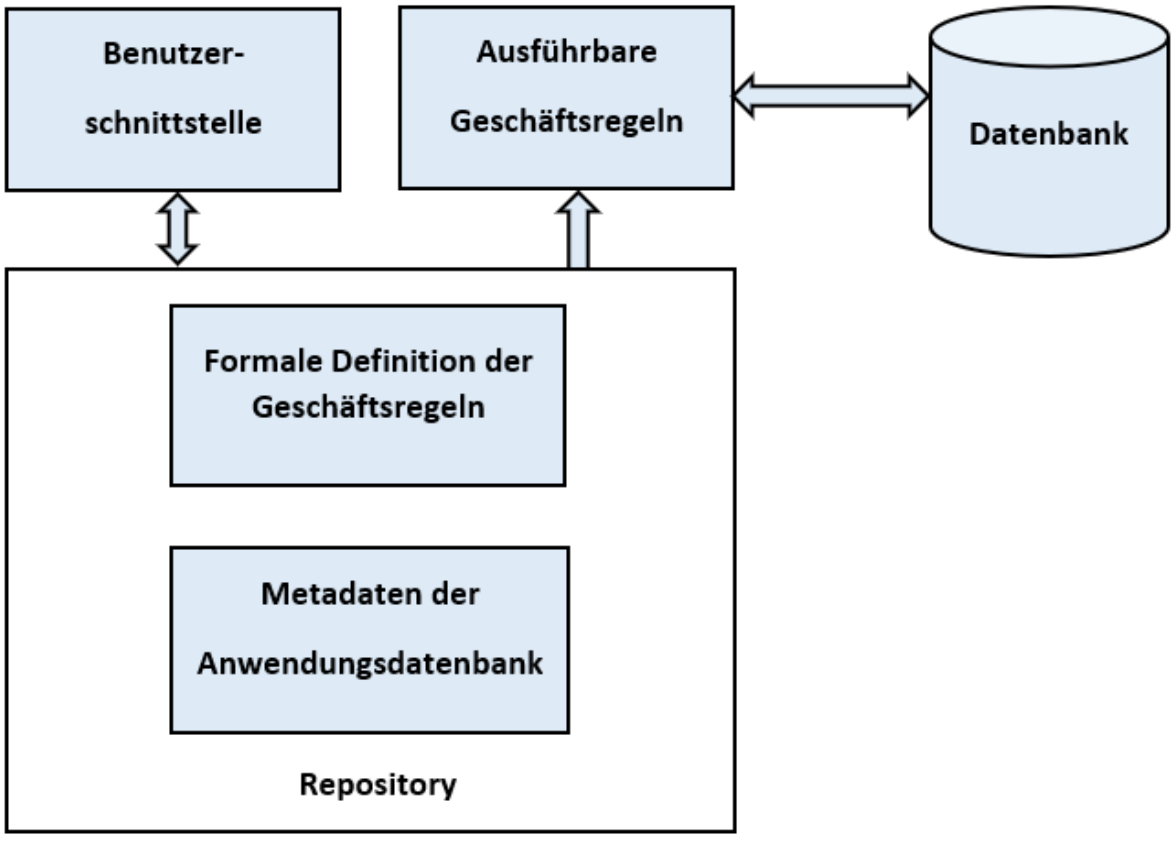


Abbildung 2: Komponenten einer Business-Rule-Engine nach Chisholm [8]

Das Repository bildet die Grundlage der Business-Rule-Engine und enthält sowohl die formale Definition der Geschäftsregeln als auch Metadaten der Anwendungsdatenbank. Unter Metadaten der Anwendungsdatenbank können Informationen zu jener Datenbank, deren Datenbestand durch die Business-Rule-Engine überprüft wird, verstanden werden [8]. Metadaten können Informationen bezüglich der Struktur der Anwendungsdatenbank, wie beispielsweise Informationen zu Tabellen und Spalten, enthalten [8]. Um den Datenbestand ordnungsgemäß prüfen zu können, muss die Business-Rule-Engine mit genügend Informationen ausgestattet sein, um durch die Daten der Anwendungsdatenbank navigieren zu können. Die formale Definition der Geschäftsregeln aus Abbildung 2 stellt jenes Regelwerk dar, welches verwendet wird, um die Datenbestände zu überprüfen. Die Geschäftsregeln bilden die notwendigen Restriktionen bezüglich Konsistenz der Daten ab, um den Datenhaushalt möglichst fehlerfrei zu halten. Über die Benutzerschnittstelle können die Geschäftsregeln entworfen und im Repository abgelegt werden. Um die Ausführung in der Anwendungsdatenbank zu ermöglichen, müssen die formalen Geschäftsregeln in eine ausführbare Form, wie beispielsweise SQL-Abfragen, übersetzt werden. Das Verwenden einer Business-Rule-Engine, welche die eben erwähnten Konzepte umsetzt, ermöglicht die Überprüfung der Anwendungsdatenbank ohne spezifische Programmlogik für jede einzelne Geschäftsregel zu implementieren [8].

Aktuell sind eine Vielzahl an Business-Rule-Softwaresystemen von verschiedenen Anbietern erhältlich, welche das Überprüfen von Datenbeständen, wie in den vorherigen Absätzen beschrieben, ermöglichen. Man kann jedoch erkennen, dass das Grundkonzept wie von Chisholm [8] beschrieben, bei allen Vertretern mit gewissen Abweichungen umgesetzt wird. In den nächsten beiden Abschnitten werden sowohl Lösungswege zur formalen Abbildung und Dokumentation von Geschäftsregeln, als auch Technologien zur anschließenden Umsetzung der Geschäftsregeln, welche im Rahmen der Literaturrecherche für den Einsatz im voestalpine Treasury in Erwägung gezogen wurden, diskutiert.

2.3 Dokumentation und Abbildung von Geschäftsregeln

Im diesem Unterkapitel werden Konzepte zur formalen Abbildung von Geschäftsregeln diskutiert. Grundsätzlich wurden im Rahmen der Literaturrecherche mehrere Ansätze zur formalen Abbildung von Geschäftsregeln identifiziert, jedoch wird im Zuge dieser Masterarbeit der Fokus auf die beiden am meisten in der Literatur vertretenen Konzepte gelegt. Im Zuge der Recherche konnten mehrere Publikationen, wie beispielsweise [9], [10], [11], [7] und [12] identifiziert werden, welche „Semantics of Business Vocabulary and Rules“, im Folgenden durch SBVR abgekürzt, als grundlegendes Konzept zur Definition und Modellierung von Geschäftsregeln verwenden. Folglich konnten die genannten Publikationen als verwandte Forschungsarbeiten (engl. related Work) eingestuft werden.

Sowohl Feuto-Njonko und El-Abed [9] als auch Ramzan et al. [10] beschreiben in ihren Publikationen die Transformation von in natürlicher Sprache verfassten Geschäftsregeln zu formal abgebildeten Geschäftsregeln, unter Zuhilfenahme von SBVR. Beide Publikationen weisen auf die Notwendigkeit einer formalen Abbildung von Geschäftsregeln, aufgrund der Schwachstellen der natürlichen Sprache, wie beispielsweise Mehrdeutigkeiten, hin. Natali und Liem [11] präsentieren in ihrer Publikation ein System, welches automatisierte Konsistenzüberprüfungen einer Datenbank ermöglicht. Natali und Liem [11] verwenden zur Abbildung der Geschäftsregeln ebenfalls SBVR und generieren daraus SQL-Abfragen, welche in der Datenbank ausgeführt werden. Folglich konnten in diesem Bereich Parallelen zu dieser Masterarbeit identifiziert werden. De Jesus und De Melo [7] bilden in ihrer Publikation ebenfalls Geschäftsregeln mittels SBVR ab. Die Autoren legen den Fokus auf die architektonischen Aspekte wie Geschäftsregeln im Rahmen von Informationssystemen implementiert werden können. Moschoyiannis et al. [12] behandeln in ihrer Publikation wie aus SBVR-Geschäftsregeln SQL-Abfragen korrekt

abgeleitet werden können. Die in diesem Absatz genannten Publikationen stellen für jeden Teilschritt des in Abbildung 17 dargestellten Prozesses zur Abbildung der Geschäftsregeln nützliche Informationen zur Verfügung, da im Rahmen dieser Masterarbeit eine Transformation der Geschäftsregeln von natürlicher Sprache zu SBVR und anschließend zu ausführbaren SQL-Abfragen notwendig ist.

Neben SBVR wurde ebenfalls in einigen Publikationen, wie [13], [14] und [15] die „Object Constraint Language“, im Weiteren durch OCL abgekürzt, identifiziert. Bajwa und Lee [13] stellen in ihrer Publikation einen Ansatz vor, wie Geschäftsregeln in Form von OCL-Constraints ausgedrückt werden können. Da Silva et al. [14] beschreiben in ihrer Publikation wie mittels OCL-Constraints Geschäftsregeln in relationalen Datenbanksystemen umgesetzt werden können. Demuth et al. [15] präsentieren ebenfalls eine Möglichkeit wie OCL zur Spezifikation von Geschäftsregeln für die anschließende Umsetzung in Form einer Datenbankanwendung verwendet werden kann. Folglich konnte auch Literatur identifiziert werden, welche den in Abbildung 17 abgebildeten Prozess, unter Verwendung von OCL, unterstützt.

Um die beiden erwähnten Konzepte in einem gemeinsamen Kontext betrachten zu können, und hinsichtlich des im Rahmen dieser Masterarbeit geplanten Einsatzes im voestalpine Treasury, die den Anforderungen am besten entsprechende Entscheidung treffen zu können, wurde ein Vergleich zwischen SBVR und OCL aufgestellt. Der Vergleich zwischen SBVR und OCL wurde nach der Analyse von Bajwa et al. [16] durchgeführt. Zusätzlich konnten Informationen von Cabot et al. [17] hinsichtlich der Transformation von OCL zu SBVR und den damit verbundenen Zusammenhängen der beiden Konzepte als Entscheidungshilfe verwendet werden. In den nachfolgenden Unterkapiteln werden die Grundlagen von SBVR und OCL erläutert, und wichtige Unterschiede der beiden Konzepte hervorgehoben.

2.3.1 Semantics of Business Vocabulary and Rules

Wie bereits im vorherigen Abschnitt beschrieben, stellt SBVR eine Möglichkeit dar, Geschäftsregeln formal abzubilden. SBVR wurde von der Object Management Group (abk. OMG) herausgegeben, und ist derzeit in der Version 1.4 verfügbar [18]. Die SBVR-Spezifikation ermöglicht die Erstellung eines Vokabulars und beliebig vielen Regeln sowie den Austausch letzterer zwischen Geschäftsleuten, Organisationen und Softwaresystemen [18]. Eben erwähntes SBVR-Vokabular und Regeln sollen die Semantik des Geschäftsvokabulars sowie der Geschäftsregeln einer Organisation abbilden [18]. Die Anwendungsdomäne von SBVR besteht in den Geschäftsleuten einer Organisation, unabhängig von den zugrundeliegenden Informationssystemen, welche die Geschäftsprozesse der Organisation unterstützen [18]. Der SBVR-Spezifikation zufolge [18], bietet SBVR einen Mehrwert in folgenden Punkten:

- Entfernung von Mehrdeutigkeiten bezüglich der Bedeutung der Geschäftsvokabeln und Geschäftsregeln.
- Dokumentieren der Bedeutung von Geschäftsvokabular und Geschäftsregeln in den verwendeten Wörtern der Geschäftsleute.
- Transformation von Vokabular und Regeln in der von Menschen ausgedrückten Form, in eine Form, welche von (Software-)Werkzeugen weiterverarbeitet werden kann.

- Untersuchung des erstellten SBVR-Content-Model, welches das Geschäftsvokabulars und die Geschäftsregeln beinhaltet, hinsichtlich Inkonsistenzen und Lücken.

- Anwendung des erstellten Geschäftsvokabulars sowie der Geschäftsregeln auf Geschäftssituationen der echten Welt.

Die exakte Definition des Geschäftsvokabulars und der Geschäftsregeln sowie aller Terme, welche zu deren Ausdruck verwendet werden, eliminiert das Risiko des Missverstehens letzterer, durch damit arbeitende Geschäftsleute [18]. Folglich kann sichergestellt werden, dass bei der Verwendung des SBVR-Vokabulars und den zugehörigen Geschäftsregeln kein Informationsverlust beim Austausch zwischen Menschen sowie zwischen (Software-)Werkzeugen auftritt (vgl. Punkt 1). Da Wörter aus dem Sprachschatz der Geschäftsleute verwendet werden, kann ebenfalls sichergestellt werden, dass Geschäftsleute, welche möglicherweise aus verschiedenen Bereichen oder Feldern kommen, beim Gebrauch des Vokabulars und der Regeln einen Ausdruck zu jeder Zeit mit genau einer Bedeutung assoziieren [18] (vgl. Punkt 2).
Nach der Erstellung des SBVR-Vokabulars und der Geschäftsregeln, können letztere in Form der SBVR-Content-Model-Datei zwischen Organisationen und (Software-)Werkzeugen ausgetauscht werden [18]. Zusätzlich besteht die Möglichkeit, das erstellte SBVR-Content-Model in eine XML-Datei zu überführen, und gegen die vorgesehene XSD-Datei zu validieren [18]. Wie bereits in den vorherigen Aufzählungspunkten erwähnt, können SBVR-Geschäftsregeln nur durch exakt definierte Terme ausgedrückt werden. Dies bedeutet im Weiteren, dass jeder in den Geschäftsregeln verwendete Term, durch einen Eintrag im SBVR-Vokabular eindeutig definiert sein muss. SBVR verwendet für die im Vokabular abgelegten Terme den Begriff Konzept (engl. Concept). Ein Konzept stellt ein Abbild eines Sachverhalts aus der Geschäftswelt dar, welcher anschließend im Vokabular festgehalten wird. SBVR wurde von der Object Management Group für die englische Sprache entworfen, weshalb der überwiegende Teil der zugänglichen Beispiele in Englisch gehalten ist. Laut Spezifikation [18] können das Vokabular und die Geschäftsregeln für andere Sprachen adaptiert werden, welches jedoch in erheblichen Einschränkungen bezüglich Austausches resultiert, da ein hoher Prozentsatz der SBVR-Community die englische Sprache verwendet. In Abbildung 3 ist ein Beispiel für ein Konzept aus einem SBVR-Vokabular dargestellt.

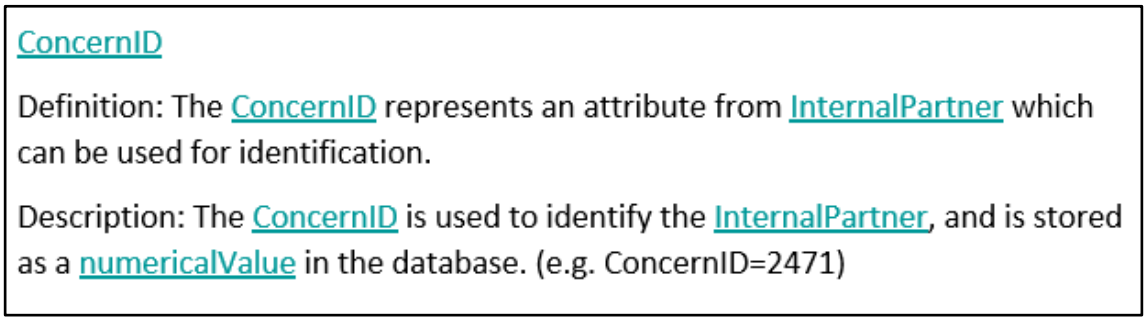


Abbildung 3: Beispiel SBVR Nomen-Konzept

In Abbildung 3 ist das Konzept `ConcernID` in Verbindung mit dessen Definition aus dem SBVR-Vokabular dargestellt. Ein wichtiges Merkmal von SBVR besteht in den unterschiedlichen Schriftarten und deren Farben. SBVR verwendet verschiedene Typen von Konzepten, welche eine unterschiedliche

Rolle ausüben, und folglich durch die visuellen Merkmale der Farben abgegrenzt werden. In Tabelle 3 sind die verschiedenen Typen von Konzepten aufgeführt.

Font Style	Originalbezeichnung	deutsche Bezeichnung	Beschreibung
Auto	Noun Concept	Nomen-Konzept	Nomen-Konzepte beschreiben allgemeine Konzepte, wie beispielsweise Auto, Haus oder Motorradfahrer
Österreich	Individual Concept	Namen-Konzept	Namen-Konzepte beschreiben spezifische Konzepte, wie beispielsweise Österreich, BillGates oder Ferrari
besitzt	Verb Concept	Verb-Konzept	Verb-Konzepte beschreiben Verben und/oder Präpositionen, wie beispielsweise besitzt, verkauft oder bearbeitet
Es ist notwendig, dass …	Keyword	Schlüsselwort	Schlüsselwörter werden verwendet um vollständige Sätze/Aussagen bilden zu können.

Tabelle 3: Verwendete Konzepte in SBVR [18]

Wird ein SBVR-Vokabular erstellt, so werden die einzelnen Konzepte, wie beispielsweise `internalPartner` und `numericalValue` (siehe Abbildung 3), beim Erzeugen eines neuen Konzepts, im obigen Beispiel `ConcernID`, hinterlegt. Folglich wird die Verbindung zwischen den einzelnen Konzepten im SBVR-Vokabular ersichtlich gemacht. Zusätzlich kann im Vokabular ebenfalls Freitext, sprich Text, welcher nicht farblich hinterlegt und somit kein anderes Konzept beschreibt, sondern weitere Informationen zu einem Konzept enthält, festgehalten werden. Werden Geschäftsregeln in SBVR definiert, so ist dies wie bereits erwähnt nicht möglich, da nur im Vokabular definierte Konzepte in den Regeln verwendet werden dürfen. In Abbildung 4 und Abbildung 5 wird ein Beispiel für eine Geschäftsregel, inklusive des verwendeten Vokabulars, dargestellt.

Abbildung 4: Beispiel für eine Geschäftsregel in SBVR

Die in Abbildung 4 dargestellte Geschäftsregel drückt aus, dass jede `ConcernID` ein `numericalValue` sein muss. Schlüsselwörter und Verb-Konzepte müssen im erstellten SBVR-Vokabular nicht erneut definiert werden, da bereits von der Object Management Group Definitionen für die meisten Schlüsselwort-Phrasen und Verb-Konzepte zur Verfügung gestellt werden. Jedoch müssen Definitionen für die Nomen-Konzepte und Namen-Konzepte im SBVR-Vokabular abgelegt werden. Für das in Abbildung 4 dargestellte Beispiel sieht das zugehörige Vokabular wie folgt aus.

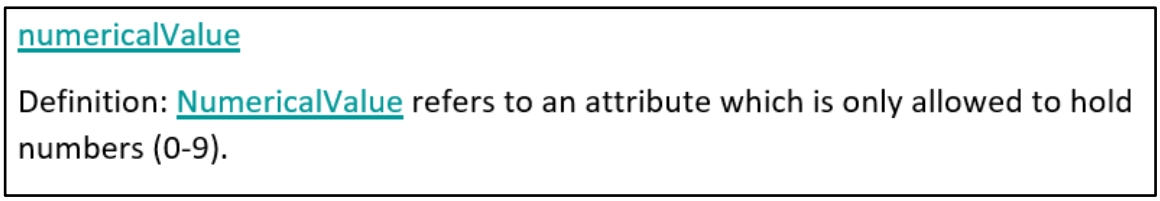


Abbildung 5: SBVR-Vokabel „numericalValue“

Wie in Abbildung 3 und Abbildung 5 dargestellt, müssen die Konzepte der SBVR-Geschäftsregel im SBVR-Vokabular auffindbar sein. Die Definition der Konzepte im SBVR-Vokabular kann unterschiedliche Formen annehmen. Grundsätzlich gilt jedoch, je mehr Informationen im SBVR-Vokabular festgehalten werden, desto besser können das Vokabular sowie die Geschäftsregeln eingesetzt werden [18]. Wird SBVR richtig eingesetzt, so stellt das SBVR-Content-Model, welches Vokabular und Geschäftsregeln beinhaltet, das Endergebnis dar. Folglich kann letzteres durch Geschäftsleute in der Organisation verwendet werden, oder mit der Generierung einer XML-Datei aus dem SBVR-Content-Model durch (Software-)Werkzeuge weiterverarbeitet werden.

2.3.2 Object Constraint Language

Die Object Constraint Language, im Folgenden als OCL abgekürzt, ist ein Bestandteil der Unified Modeling Language (Abk. UML), und ist somit neben SBVR ebenfalls eine Entwicklung der Object Management Group. OCL stellt eine Sprache zur Modellierung von Softwaremodellen dar, und wird in Verbindung mit UML verwendet [19]. Da OCL auf die definierten Typen, wie beispielsweise Klassen oder Schnittstellen aus dem UML Diagramm zurückgreift, kann OCL folglich nicht sinnvoll angewendet werden, ohne zumindest einige Grundlagen von UML ebenfalls umzusetzen [19]. Durch den Einsatz von OCL können zusätzliche Informationen, welche oft nicht in einem UML Diagramm dargestellt werden können, modelliert werden [19]. Wie der Name Object Constraint Language bereits andeutet, wurde OCL ursprünglich zur Modellierung von Constraints, welche als Beschränkungen oder Restriktionen verstanden werden können, entwickelt [19]. Der Funktionsumfang von OCL ist aktuell jedoch signifikant größer als nur die Modellierung von Restriktionen, beispielsweise können laut Warner und Kleppe [19] ebenfalls Abfragen, Referenzwerte und Geschäftsregeln mittels OCL umgesetzt werden. Da im Rahmen dieser Masterarbeit Geschäftsregeln von besonderem Interesse sind, wird ebenfalls der Fokus bezüglich OCL auf Geschäftsregeln gelegt. Wie bereits im Abschnitt 2.3 erwähnt, besteht die Notwendigkeit, die Geschäftsregeln formal abzubilden. Die Modellierung von Geschäftsregeln in OCL, basierend auf den Regeln in natürlicher Sprache, erfüllt diese Anforderung [14]. Wird ein OCL Constraint erstellt, so muss sich dieser immer auf ein Objekt einer Klasse aus dem UML-Klassendiagramm beziehen [14]. Dabei kann dieser entweder auf das Klassenobjekt selbst oder beispielsweise ein Attribut des Klassenobjekts abzielen [14]. Das erwähnte Objekt wird im Zuge der Erstellung von Geschäftsregeln als Regelanker bezeichnet [14]. In OCL unterscheidet man grundsätzlich zwischen zwei Arten von Geschäftsregeln. Es besteht die Möglichkeit derivative Regeln abzubilden, welche beispielsweise eine Kalkulation durchführen [14]. In Abbildung 6 wird ein Beispiel für eine derivative Regel dargestellt.

```
Context Motorrad::KostenVersicherung: Double
Post: result = Grundkosten + (PS * SchlüsselMotorsteuer)
```

Abbildung 6: derivative Regel in OCL

Mit Hilfe dieser OCL-Regel können die Versicherungskosten für ein Motorrad berechnet werden. `Context` gibt dabei das Kontextobjekt, bzw. den Regelanker an. Folglich bezieht sich diese OCL-Regel auf das Attribut `KostenVersicherung` des Klassenobjekts „Motorrad“. Die Post-Bedingung bewirkt die Kalkulation der Kosten, basierend auf den Versicherungsgrundkosten und den motorbezogenen Kosten. Als Endergebnis werden somit die gesamten Versicherungskosten im Attribut `KostenVersicherung` abgelegt.

Neben derivativen Regeln können mit Hilfe von OCL auch restriktive Regeln entworfen werden [14]. Dabei wird in OCL der Typ `invariant` (Abk. inv) verwendet. `Invariant` stellt einen booleschen Ausdruck dar, welcher für jede Instanz der Klasse „`Wahr`“ retournieren muss [14]. In Abbildung 7 wird ein Beispiel für eine restriktive Regel dargestellt.

```
Context Motorrad inv:
AnzahlRäder = 2
```

Abbildung 7: Beispiel restriktive Regel in OCL

Zusammenfassend kann festgehalten werden, dass OCL die Anforderungen zur Modellierung von Geschäftsregeln ebenfalls wie das im vorherigen Abschnitt behandelte SBVR erfüllt. Eine detailliertere Behandlung der Grundlagen von OCL ist im Rahmen dieser Masterarbeit nicht vorgesehen. Im folgenden Abschnitt werden jedoch die beiden vorgestellten Konzepte, SBVR und OCL, anhand der Publikation von Bajwa et al. [16] gegenübergestellt.

2.3.3 Analyse von SBVR im Vergleich zu OCL

Um die Entscheidung zwischen SBVR und OCL, hinsichtlich der formalen Abbildung der Geschäftsregeln und dem anschließenden Einsatz im voestalpine Treasury bestmöglich treffen zu können, werden in diesem Abschnitt beide Konzepte miteinander verglichen. Bajwa et al. [16] haben in ihrer Publikation verschiedene Kriterien identifiziert, an denen beide Konzepte verglichen werden können. Die erwähnten Kriterien werden in Abbildung 8 dargestellt.

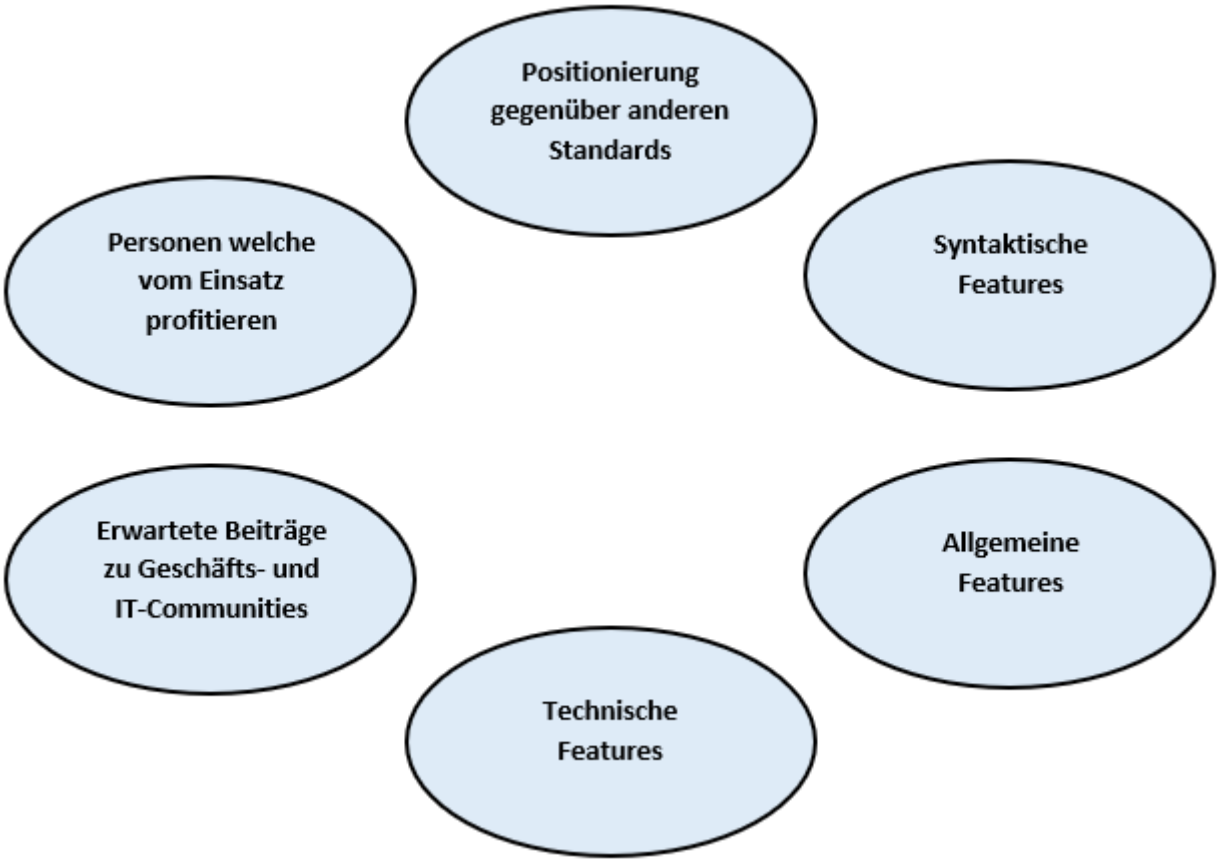


Abbildung 8: Kriterien zum Vergleich zwischen SBVR und OCL nach Bajwa et al. [16]

Im Bereich Positionierung gegenüber anderen Standards untersuchen Bajwa et al. [16] die Verbindungen zu anderen Standards im gemeinsamen Feld. Dabei können beispielsweise Wechselwirkungen oder Lücken im Vergleich zu anderen verwandten Standards identifiziert werden. Ebenfalls werden SBVR und OCL hinsichtlich ihrer syntaktischen Eigenschaften untersucht. Dabei werden die Basiselemente beider Konzepte zur Modellierung von Geschäftsregeln verglichen. Diese Untersuchung kann bei der Transformation von SBVR nach OCL, und in umgekehrter Richtung, sowie bei der Anwendung in verschiedenen Domänen von Vorteil sein [16]. Unter allgemeinen Features verstehen Bajwa et al. [16] die grundsätzlichen Charakteristiken, welche in einer SBVR-Geschäftsregel oder in einem OCL Constraint verwendet werden. Zusätzlich wird der Wirkungsgrad und die Auswirkungen von beiden Konzepten, im Bereich Software- und Geschäftsmodellierung untersucht [16]. Des Weiteren werden die technischen Eigenschaften beider Konzepte verglichen. Es wird untersucht, welche technischen Möglichkeiten sowie etwaige Probleme beim Einsatz von SBVR und OCL auftreten können. Bajwa et al. [16] erwähnten ebenfalls die Notwendigkeit, die erwarteten Beiträge beider Konzepte in Geschäfts- sowie IT-Communities zu untersuchen. Dabei können Eckdaten zu beispielsweise Werkzeug-Unterstützung, Community-Support oder Weiterentwicklung des jeweiligen Konzepts hervorgehoben werden, welche ebenfalls die Entscheidung zwischen SBVR und OCL beeinflussen können. Letztlich wird untersucht welche Personen vom Einsatz profitieren können, dabei können sowohl Anwender, welche direkt mit den jeweiligen Konzepten arbeiten, als auch Empfänger, welche lediglich vom Einsatz profitieren, identifiziert werden [16].

Positionierung gegenüber anderen Standards

Wie bereits erwähnt, sind SBVR und OCL beides Standards der Object Management Group, und wurden von letzterer in deren Model Driven Architecture (Abk. MDA), einem umfassenden Paket zu Softwaredesign und Softwareentwicklung, integriert [16]. Innerhalb der MDA werden SBVR und OCL jedoch in unterschiedliche Ebenen eingeordnet. In Abbildung 9 ist die Einordnung von SBVR und OCL innerhalb der MDA dargestellt.

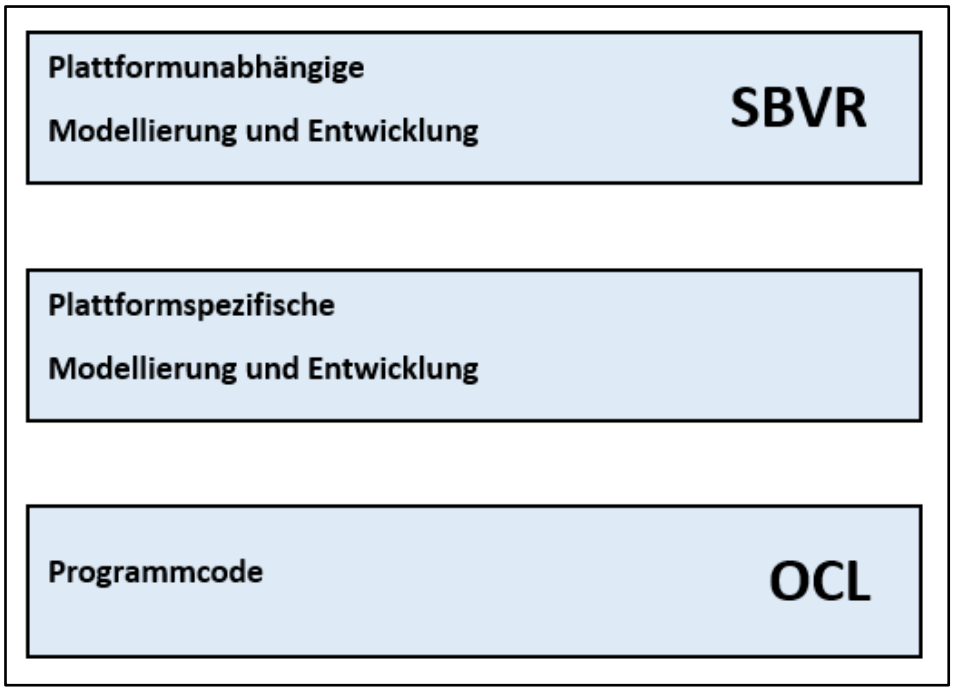


Abbildung 9: Ebenen innerhalb der MDA der Object Management Group [16]

SBVR wird zur Modellierung von Vokabular und Geschäftsregeln, zur Abbildung der Bedeutung der Geschäftsterminologie, auf der plattformunabhängigen Ebene eingesetzt. Folglich liegt die Hauptfunktion von SBVR in der Modellierung der Geschäftswelt, unabhängig von den darunterliegenden Informationssystemen (siehe 2.3.1) [16]. Im Vergleich dazu, dient OCL innerhalb der MDA zur Generierung präziser Modelle in Verbindung mit UML [16]. Wie bereits im Abschnitt 2.3.2 dargestellt, sind OCL-Konstrukte nicht in natürlicher Sprache verfasst, sondern in Form von Programmcode, weshalb OCL innerhalb der MDA auf der Ebene Programmcode eingeordnet wird.

Neben der Model Driven Architecture ist die Unified Modeling Language (Abk. UML) ebenfalls ein bedeutender Standard im Bereich der Modellierungssprachen. SBVR bietet sowohl die Möglichkeit Modelle in UML Modelle zu transformieren als auch umgekehrt [16]. Obwohl die Hauptfunktion von SBVR in der Modellierung der Geschäftswelt liegt, ist es durch die formale Repräsentation von SBVR ebenfalls möglich Softwareanforderungen zu modellieren [16]. Da OCL ohnehin Teil des UML Metamodells ist und als Zusatz zum UML Klassenmodell zur Modellierung von Constraints verwendet wird, kann hier eine sehr enge Verbindung mit UML festgestellt werden. Ebenfalls muss OCL wie in Punkt 2.3.2 beschrieben in Verbindung mit einem UML-Klassenmodell verwendet werden, um auf dessen Objekte referenzieren zu können. Dies bedeutet, dass OCL nicht als Einzellösung einsetzbar ist, und unterstreicht ein weiteres Mal die Verbindung zum UML Standard.

Business Process Management (Abk. BPM), und dessen zugehörige Komponenten wie beispielsweise Business Process Model and Notation (Abk. BPMN), sind ebenfalls namhafte Vertreter im Bereich der Geschäftsprozessmodellierung, welche die kontinuierliche Optimierung von Geschäftsprozessen als Ziel verfolgen [16]. Folglich haben Geschäftsregeln im Bereich BPM eine hohe Bedeutung, wodurch SBVR, und seine Fähigkeit Geschäftsregeln zu modellieren, zu einem wichtigen Werkzeug werden [16]. Es besteht die Möglichkeit ein SBVR-Vokabular in ein BPM-Modell zu importieren, um Geschäftsregeln besser modellieren zu können [16]. Daher kann hier ebenfalls eine Verbindung zwischen SBVR und einem verwandten Standard identifiziert werden. OCL hingegen hat keine direkte Verbindung zu BPM. OCL ist ebenfalls ein mächtiges Werkzeug zur Modellierung von Geschäftsregeln, jedoch wird OCL aufgrund seiner Struktur mit UML kombiniert, und nicht in Verbindung mit BPM eingesetzt [16].

Syntaktische Features

Bajwa et al. [16] vergleichen eine Reihe an syntaktischen Elementen von SBVR und OCL und heben deren Verbindung hervor. Im Zuge dieser Masterarbeit wird von einer detaillierten Beschreibung der syntaktischen Elemente beider Konzepte abgesehen, da dies den Rahmen überschreiten würde.

Allgemeine Features

Sowohl bei SBVR als auch OCL liegt das Hauptanwendungsgebiet in der konzeptuellen Modellierung, lediglich die Anwendungsdomäne ist unterschiedlich [16]. SBVR wird vorrangig zur Modellierung der Geschäftswelt eingesetzt, während OCL zur Softwaremodellierung verwendet wird [16]. Beide Konzepte sind deklarative Sprachen [16]. Während SBVR die Modellierung in natürlicher Sprache (Englisch) unterstützt, hat OCL seine eigene formale Struktur. Laut Bajwa et al. ist das Requirements-Engineering eine wichtige Phase im Prozess der Softwareentwicklung, wobei die Anforderungen häufig in natürlicher Sprache dokumentiert werden. SBVR kann im Requirements-Engineering wiederrum zur formalen Abbildung der Software- und Geschäftsanforderungen in natürlicher Sprache eingesetzt werden. Im Gegensatz dazu, wird OCL im späteren Verlauf, zur Modellierung präziser Modelle in Verbindung mit UML, eingesetzt [16]. Weiters haben Bajwa et al. [16] SBVR und OCL hinsichtlich der „Auswirkungen“ beider Konzepte verglichen. Da SBVR und OCL ausschließlich verwendet werden, um Geschäftsregeln auszudrücken, und von der Umsetzung letzterer vollständig entkoppelt sind, gelten sowohl SBVR als auch OCL als auswirkungsfreie Sprachen [16]. Letztlich wurde untersucht ob Geschäftsregeln, welche durch SBVR und OCL modelliert wurden, wohlgeformten Ausdrücken entsprechen. Im genaueren bedeutet dies, dass die Geschäftsregeln validiert und auf Konsistenz geprüft werden können [16]. Sowohl mit SBVR als auch durch OCL modellierte Geschäftsregeln erfüllen diese Anforderung [16].

Technische Features

Die semantischen Strukturen von SBVR basieren auf formaler Logik. Dabei vereint SBVR Aspekte der Prädikatenlogik, der Arithmetik und der Mengenlehre [16]. Ähnlich dazu, basiert OCL ebenfalls auf der Prädikatenlogik sowie der Mengenlehre [16]. Bajwa et al. [16] untersuchten ebenfalls die Erweiterbarkeit beider Konzepte. Ein SBVR-Vokabular kann durch Zusammenfügen mit einem anderen SBVR-Vokabular, welches andere Elemente enthält, beliebig erweitert werden [16]. Zusätzlich besteht die Möglichkeit, das SBVR-Vokabular in andere Sprachen als der englischen Sprache zu überführen [16]. OCL ist durch die Möglichkeit UML Objekte, wie beispielsweise Klassen oder Methoden einzubinden ebenfalls beliebig erweiterbar [16].

Erwartete Beiträge zu Geschäfts- und IT-Communities

Unter Beiträgen sind im Detail Publikationen zu den jeweiligen Konzepten oder Werkzeug-Support zu verstehen. Es konnten einige Publikationen (siehe 2.3) identifiziert werden, welche sich mit der Erstellung von Geschäftsregeln in SBVR beschäftigen. Darüber hinaus existieren weitere Publikationen im Bereich SBVR, wie beispielsweise [17] und [16], welche sich mit der Transformation von SBVR in Richtung anderer Technologien befassen. Im Bereich der SBVR-Werkzeuge wurden ebenfalls frei zugängliche Optionen, wie beispielsweise SBVR2OCL, SBeaVeR, SBVR Visual Editor und NL2SBVR identifiziert. Darüber hinaus existieren auch kostenpflichtige Produkte im Bereich SBVR, welche im Rahmen dieser Masterarbeit nicht angeführt werden. Bei den frei zugänglichen Optionen konnten Defizite im Bereich Weiterentwicklung, Wartung und Support identifiziert werden. Die Anzahl an einsetzbaren kostenlosen Werkzeugen im Bereich SBVR ist derzeit niedrig. Für OCL stellt sich eine ähnliche Situation dar. Es existieren einige Publikationen (siehe 2.3) welche sich mit OCL und dessen Anwendung beschäftigen. Bezüglich Werkzeug-Support konnte eine signifikant höhere Anzahl an Optionen, wie beispielsweise [20], [21] oder [22], verglichen mit den Ergebnissen im Bereich SBVR identifiziert werden. Eine detaillierte Evaluierung des Werkzeug-Angebots für SBVR und OCL wird in dieser Masterarbeit nicht durchgeführt. Aufgrund der Anzahl an Werkzeugen und Publikationen zu diesen Werkzeugen, könnte die Erstellung einer wissenschaftlichen Publikation, welche eine vergleichende Analyse in diesem Bereich durchführt, einen signifikanten Mehrwert darstellen.

Personen, welche vom Einsatz profitieren

Der letzte Aspekt welcher von Bajwa et al. [16] verglichen wurde, sind jene Personen, welche durch den Einsatz von SBVR oder OCL profitieren können. SBVR stellt einen Mehrwert für mehrere verschiedene Personengruppen dar. Zu diesen Personengruppen zählen unter anderen Geschäftsanalysten und -modellierer, Verantwortliche für Geschäftsvokabular und -regeln, Administratoren, Werkzeug-Entwickler oder Linguisten [16]. Durch OCL können ebenfalls mehrere Personen profitieren, wie beispielsweise alle Personen, welche mit UML Modelle arbeiten oder verantwortliche Personen im Requirements-Engineering [16].

Aus den im Abschnitt 2.3.3 erwähnten Aspekten kann eine signifikante Ähnlichkeit zwischen SBVR und OCL identifiziert werden. Beide Konzepte basieren auf mathematischen Grundlagen, welche die formale Modellierung von Geschäftsregeln ermöglichen [16]. Es wurden für beide Konzepte wissenschaftliche Publikationen identifiziert, welche die Materie behandeln. Ebenfalls wurde der Werkzeug-Support beider Konzepte untersucht, und die Notwendigkeit weiterführender Arbeiten in diesem Forschungsgebiet ausgesprochen. Trotz großer Ähnlichkeiten müssen jedoch zwei wichtige Aspekte, wie bereits in Abschnitt 2.3.1 , respektive 2.3.2 erwähnt, hervorgehoben werden. SBVR ermöglicht, im Gegensatz zu OCL, die Modellierung sämtlicher Elemente in natürlicher Sprache. Des Weiteren ist die Hauptfunktion, die konzeptuelle Modellierung, bei SBVR und OCL dieselbe, jedoch sind die Anwendungsdomänen unterschiedlich. SBVR-Vokabular und Geschäftsregeln werden zur Verwendung durch Geschäftsleute erstellt, und sind unabhängig von den darunterliegenden Informationssystemen. OCL wird im Gegensatz dazu, zur Modellierung von Softwaremodellen und den zugehörigen Restriktionen verwendet, und befindet sich daher im Bereich der Informationssysteme.

2.4 Implementierung und Umsetzung der Geschäftsregeln

Im vorherigen Abschnitt 2.3 wurden Möglichkeiten zur formalen Abbildung von Geschäftsregeln präsentiert. Jedoch decken die besprochenen Konzepte lediglich die Abbildung, nicht aber die Ausführung und Umsetzung der erstellten Geschäftsregeln ab. Folglich müssen für die Ausführung, im Genaueren, für die in der Einleitung beschriebene Überprüfung der Datenbestände, zusätzliche Komponenten eingesetzt werden. Die prominenteste Möglichkeit, welche in der Literatur erwähnt wurde und für diese Funktionalität eingesetzt werden kann, sind die in Abschnitt 2.2 beschriebenen Business-Rule-Engines. In den nachfolgenden Absätzen werden die Optionen im Bereich Business-Rule-Engines diskutiert. Im Rahmen dieser Masterarbeit werden die möglichen Optionen in zwei Kategorien eingeteilt. Die erste Kategorie besteht aus externen Business-Rule-Engine Produkten, welche derzeit am Markt kostenlos zugänglich sind. Kostenpflichtige Lösungen sind im Zuge dieser Masterarbeit nicht von Bedeutung. Die zweite Kategorie besteht aus möglichen Lösungen, welche selbst implementiert werden müssen. In dieser Kategorie wurden Publikationen aus diesem Bereich untersucht und mögliche Individuallösungen für das voestalpine Treasury hervorgehoben. In Abbildung 10 wird die Kategorisierung grafisch dargestellt.

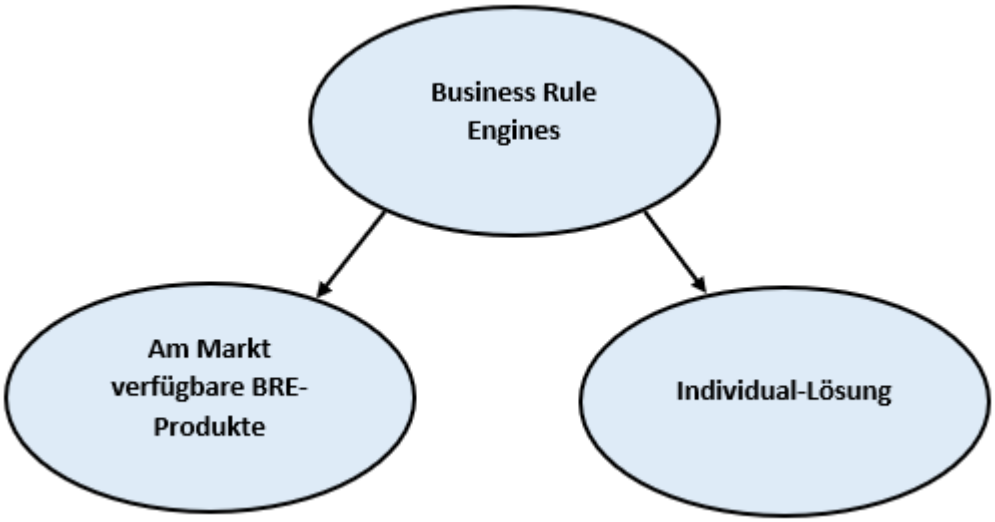


Abbildung 10: Kategorisierung von Business-Rule-Engines

2.4.1 Übersicht über am Markt verfügbare Business-Rule-Engines

In diesem Abschnitt werden verschiedene, am Markt verfügbare Business-Rule-Engine-Produkte, präsentiert. Aktuell ist eine Vielzahl von kostenlos, als auch kostenpflichtig zugänglichen Produkten erhältlich. Die nachfolgende Tabelle beinhaltet die verschiedenen kostenlos zugänglichen Optionen, welche im Rahmen der Recherche identifiziert werden konnten.

Business-Rule-Management-Systeme	Plattform
Drools [23]	Java
OpenRules [24]	Java
OpenL Tablets [25]	Java
Business-Rule-Engines	
Jess [26]	Java
JLisa [27]	Java
OpenLexicon [28]	Java
JRuleEngine [29]	Java
NxBRE [30]	.NET
SRE – Simple Rule Engine [31]	.NET
Drools.NET 3.0 [32]	.NET
Semantic-Web Business-Rules	
SweetRules [33]	Java

Tabelle 4: Verschiedene Optionen im Bereich Business-Rule-Engines

Wie in Tabelle 4 ersichtlich, konnten mehrere Produkte identifiziert werden, welche die benötigte Funktionalität einer Business-Rule-Engine abdecken. Der Einsatz der erwähnten Produkte erfordert im Gegensatz zu den in Abschnitt 2.4.2 erwähnten Optionen lediglich die Abbildung der eigenen Geschäftsregeln nach den spezifischen Vorgaben des Anbieters, jedoch keinen Implementierungsaufwand zur Erstellung der Business-Rule-Engine Funktionalität. Programmcode, welcher das Ablegen und Speichern der Geschäftsregeln oder das Durchlaufen der Geschäftsregeln übernimmt, wird beim Einsatz der in Tabelle 4 aufgeführten Produkte bereits zur Verfügung gestellt. Die in Tabelle 4 angeführten Produkte wurden in verschiedene Kategorien eingeteilt. Business-Rule-Management-Systeme (Abk. BRMS) stellen umfangreiche Pakete dar, welche zusätzlich zur Funktionalität der Business-Rule-Engine weitere Komponenten, wie beispielsweise Regel-Repositorien oder grafische Regel-Editoren beinhalten. Produkte mit der Kennzeichnung Business-Rule-Engine (Abk. BRE) ermöglichen es dem Benutzer Geschäftsregeln zu erstellen, diese abzulegen, und darauf basierend Überprüfungen durchzuführen, enthalten jedoch grundsätzlich keine zusätzlichen Komponenten. SweetRules [33] wurde getrennt von den anderen Produkten in Tabelle 4 angeführt, da SweetRules [33] vorrangig für das Erstellen von Geschäftsregeln im Bereich Semantic-Web entwickelt wurde. SweetRules [33] unterstützt sowohl die Rule Markup Language (Abk. RuleML) als auch die Semantic Web Rule Language (Abk. SWRL) oder OWL-Ontologien, und ist somit im Semantic-Web-Bereich einsetzbar. Eine detaillierte Untersuchung aller Produkte ist im Rahmen dieser Masterarbeit aufgrund des einzuhaltenden Umfangs nicht vorgesehen.

2.4.2 Individuallösungen im Bereich Business-Rule-Engines

Neben den im vorherigen Abschnitt besprochenen Optionen, welche lediglich das Erstellen eigener Geschäftsregeln voraussetzen, wurde im Bereich der Individuallösungen ebenfalls ein potenzieller Lösungsweg, für die Implementierung des Data-Cleaning-Prototyps, welches ein Ziel dieser Masterarbeit darstellt und die Funktionalitäten einer Business-Rule-Engine benötigt, identifiziert. Abdullah et al. [34] stellen in ihrer Publikation ein regelbasiertes System vor, welches mit Hilfe der Structured Query Language (Abk. SQL) implementiert wurde. Abdullah et al. [34] entwickelten in ihrem Anwendungsfall ein regelbasiertes System für ein IT-Unternehmen im Gesundheitsbereich. Die Architektur des implementierten regelbasierten Systems nach Abdullah et al. [34] wird in Abbildung 11 dargestellt.

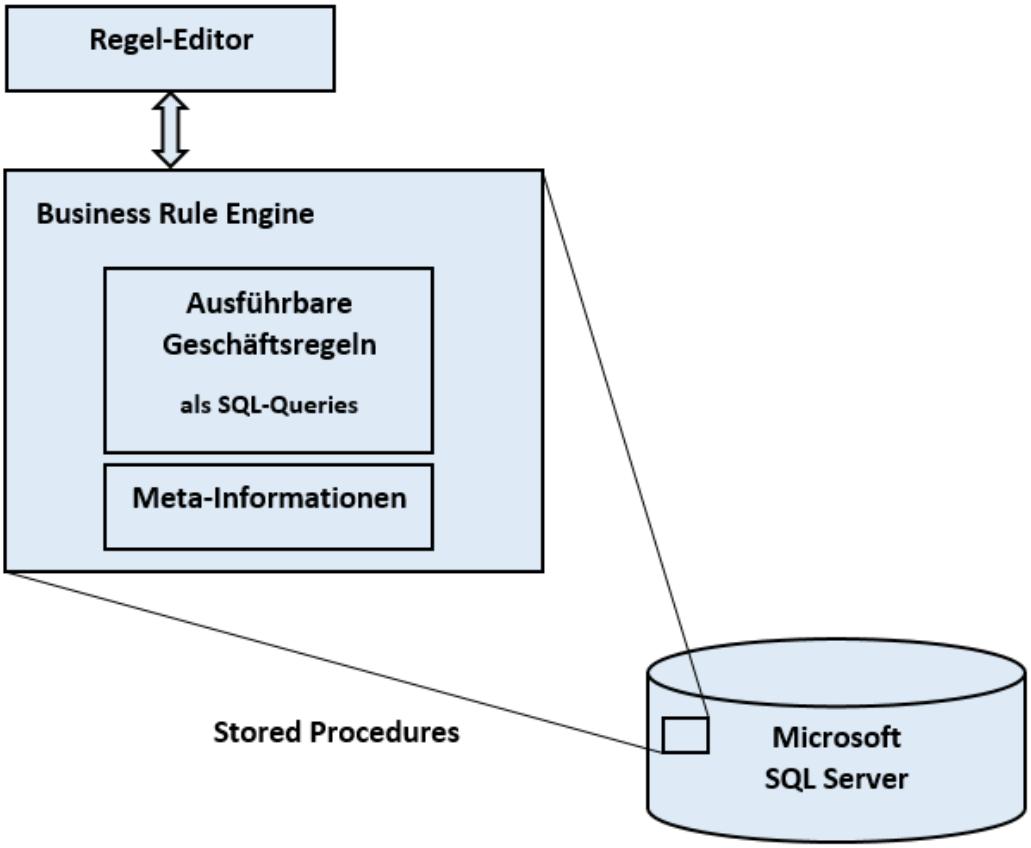


Abbildung 11: Architektur des regelbasierten Systems nach Abdullah et al. [34]

Abdullah et al. [34] verwenden in deren Anwendungsfall eine relationale Datenbank von Microsoft, welche zur Ablage der Unternehmensdaten benutzt wurde. Die Funktionalität der Business-Rule-Engine wurde in Form von mehreren Stored-Procedures in SQL, also direkt an der Datenbank, implementiert und abgelegt [34]. Im Vergleich dazu, verwenden bereits am Markt existierende Business-Rule-Engines wie in Tabelle 4 ersichtlich andere Plattformen wie beispielsweise Java oder .NET. Zusätzlich wurden mehrere Tabellen angelegt, um Meta-Informationen (vgl. 2.2) sowie die benötigten Geschäftsregeln speichern zu können [34]. Die ausführbaren Regeln, welche die Business-Rule-Engine zum Überprüfen der Daten verwendet, wurden in Form von SQL-Abfragen abgebildet [34]. Um die Erstellung der Geschäftsregeln zu unterstützen, haben Abdullah et al. [34] einen Editor entwickelt, welcher die Erstellung in englischer Sprache unterstützt und die Transformation in die ausführbaren SQL-Abfragen übernimmt. Abdullah et al. konnten folgenden Vorteil ihres regelbasierten Systems identifizieren. Da die Unternehmensdaten bereits in relationaler Form vorhanden sind, ermöglicht die Implementierung der Rule-Engine in Form von Stored-Procedures und das Verwenden von SQL-Abfragen um die Geschäftsregeln abzubilden, die Durchführung der Überprüfungen ohne eine vorherige Transformation der Unternehmensdaten in eine andere Form [34]. Folglich kann die Business-Rule-Engine direkt mit den Unternehmensdaten arbeiten, ohne diese abändern zu müssen.

In Abschnitt 2 wurden die relevanten Ergebnisse der Literaturrecherche für diese Masterarbeit präsentiert. Im nachfolgenden Abschnitt wird nun Bezug auf die Architektur des entwickelten Data-Cleaning-Prozesses im voestalpine Treasury, unter Verwendung der Ergebnisse der Literaturrecherche, genommen.

3 Architektur

In diesem Kapitel wird die Architektur des Data-Cleaning-Prozesses für das voestalpine Treasury präsentiert und die einzelnen Komponenten beschrieben. In Abbildung 12 wird der Data-Cleaning-Prozess grafisch dargestellt.

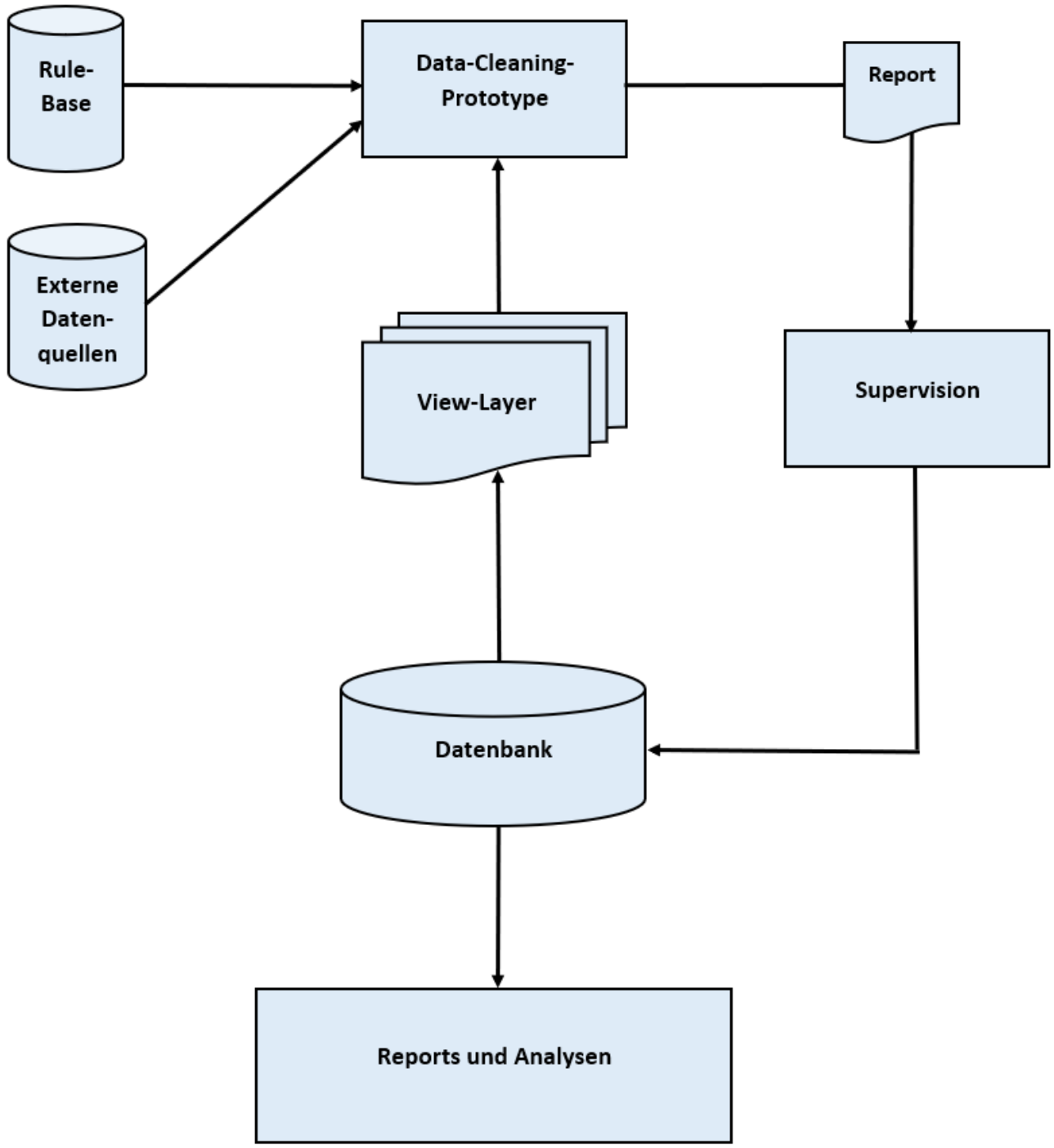


Abbildung 12: Architektur des Data-Cleaning-Prozesses im voestalpine Treasury

In den nächsten Absätzen werden die einzelnen Elemente des in Abbildung 12 dargestellten Data-Cleaning-Prozesses, erläutert. Zusätzlich wird der Prozessablauf erklärt und Rollen, sofern vorhanden, für die einzelnen Prozessschritte zugeteilt.

3.1 Datenbank

Im voestalpine Treasury wird eine relationale Datenbank von Oracle eingesetzt, welche die Grundlage für sämtliche Datenbestände des Treasury-Management-Systems (Abk. TMS) im Unternehmen darstellt. Eine Abänderung des Datenbankschemas ist aus Unternehmenssicht nicht möglich und somit nicht Gegenstand dieser Masterarbeit. Das Ziel besteht jedoch darin, die bestmögliche Qualität der Datenbestände zu erreichen und nachhaltig sicherzustellen. Dies ist besonders wichtig, da basierend auf den Datenbeständen des voestalpine Treasury eine Vielzahl an Auswertungen und Analysen durchgeführt werden, welche aufgrund des Volumens der damit verbundenen Finanzmittel erhebliche Auswirkungen auf den Erfolg des voestalpine Treasury haben können. Weitere Informationen zu den Auswertungen und Analysen werden im Abschnitt 3.8 behandelt.

3.2 View-Layer

Das in den Prozess eingebettete View-Layer-Element bezeichnet eine Abstraktionsebene über der eigentlichen Datenbankebene. Es sollen die relevanten Daten für die Tätigkeiten der Masterarbeit in eine Form gebracht werden, welche für die Data-Cleaning-Aufgaben des voestalpine Treasury erforderlich sind. Durch die Erstellung der benötigten Views werden vor allem Joins vorab durchgeführt, um diese aus den darauf aufbauenden Geschäftsregeln zu entfernen. Das nachfolgende Beispiel soll diesen Sachverhalt veranschaulichen.

Die Tabelle `Mitarbeiter` ist, wie in Abbildung 13 ersichtlich, mit fünf Spalten angelegt. Die Spalte `MaID` stellt die eindeutige Kennung für einen Mitarbeiter dar. Die Spalten `Vorname` und `Nachname` stellen den Vornamen, respektive den Nachnamen, eines Mitarbeiters dar. Die `Gehaltsgruppe` stellt einen Wert zwischen „A“ und „C“ dar, wobei „A“ die niedrigste, und „C“ die höchste Gehaltsgruppe ist. Die Spalte `SozVersID` stellt einen Fremdschlüssel dar, mit dem die Verbindung zu den Sozialversicherungsinformationen eines Mitarbeiters, aus der Tabelle `SVInformation`, hergestellt werden kann.

Tabelle Mitarbeiter

MaID	Vorname	Nachname	Gehaltsgruppe	SozVersID
1	Florian	Mayer	A	1221
2	Herbert	Gruber	B	1456
3	Hans	Huber	B	4511
4	Thomas	Steiner	C	8908

Abbildung 13:Tabelle Mitarbeiter

Die Tabelle `SVInformationen` enthält die Sozialversicherungsinformationen für einen Mitarbeiter. Die Spalte `SozVersID` stellt den Primärschlüssel in dieser Tabelle dar. Die Spalte `Abgabengruppe` stellt einen Wert zwischen „A“ und „C“ dar. Die Spalte `Rezeptgebühren` kann den Wert JA oder NEIN annehmen.

Tabelle SVInformation

SozVersID	Abgabengruppe	Rezeptgebühren
1221	A	JA
1456	B	NEIN
4511	A	NEIN
8908	C	NEIN

Abbildung 14: Tabelle SVInformation

Es wird eine View erstellt, welche über die `SozVersID` einen Join zwischen den Tabellen `Mitarbeiter` und `SVInformation` herstellt.

View Mitarbeiterinformationen

MaID	Vorname	Nachname	Gehaltsgruppe	SozVersID	Abgabengruppe	Rezeptgebühren
1	Florian	Mayer	A	1221	A	JA
2	Herbert	Gruber	B	1456	B	NEIN
3	Hans	Huber	B	4511	A	NEIN
4	Thomas	Steiner	C	8908	C	NEIN

Abbildung 15: View Mitarbeiterinformationen

Im Rahmen des Data-Cleaning-Prozesses soll der View-Layer die Durchführung der verschiedenen Integritäts- und Plausibilitätsprüfungen vereinfachen. Einerseits können die Daten durch Operationen wie Joins (siehe Beispiel) lesbarer gemacht werden, andererseits wird die Anwendung der Geschäftsregeln unterstützt. Das nachfolgende Beispiel soll die Anwendung einer Regel darstellen.

Es besteht die Regelung, dass Mitarbeiter aus der Gehaltsgruppe „C“ keine Rezeptgebührenbefreiung haben dürfen. In Abbildung 15 ist zu erkennen, dass Mitarbeiter Nummer 4 die Gehaltsgruppe „C“ besitzt, jedoch auch keine Rezeptgebühren zahlen muss. Aus dieser Erkenntnis geht hervor, dass hier eine potenzielle Verletzung der oben erwähnten Regel in den Datenbeständen vorliegt.
Ohne den Einsatz des View-Layers, würde die SQL-Abfrage, welche die Datenbestände basierend auf der definierten Regel überprüft, wie folgt aussehen:

```
SELECT m.*, sv.Abgabengruppe, sv.Rezeptgebühren
FROM Mitarbeiter m JOIN SVInformation sv ON m.SozVersID = sv.SozVersID
WHERE m.Gehaltsgruppe='C' AND sv.Rezeptgebühren='NEIN'.
```

Unter Verwendung des View-Layers, würde die SQL-Abfrage, welche die Datenbestände basierend auf der definierten Regel überprüft, wie folgt aussehen:

```
SELECT *
FROM Mitarbeiterinformationen
WHERE Gehaltsgruppe='C' AND Rezeptgebühren='NEIN'.
```

Mithilfe des View-Layers können somit die oftmals über mehrere Tabellen verteilten Daten vor der Durchführung der Konsistenzregeln in eine passende Struktur gebracht werden. Es ist ersichtlich, dass die Lesbarkeit sowie die Anwendbarkeit der aufgestellten SQL-Abfragen, welche basierend auf den Geschäftsregeln den Datenbestand überprüfen, unter Einsatz des View-Layers erheblich verbessert wurde.

3.3 Rule-Base

Unter dem Begriff Rule-Base soll im Rahmen dieser Masterarbeit das gesammelte Wissen des voestalpine Treasury, bezüglich der Konsistenz der Datenbestände, verstanden werden. Dieses eben erwähnte Wissen, soll die Rahmenbedingungen für die Durchführung des Data-Cleaning-Prozesses vorgeben. Im Detail bedeutet dies, dass alle überprüften Daten die vorgegebenen Rahmenbedingungen nicht verletzen dürfen. Sollte eine Verletzung vorliegen, so soll dies als Bestandteil des Data-Cleaning-Reports festgehalten werden. In diesem Punkt ist zu beachten, dass der Begriff Rule-Base nicht als Synonym für eine konkrete Implementierung einer Business-Rule-Engine, wie beispielsweise JBoss Drools (siehe Tabelle 4), verwendet wird. Jedoch ist die Modularität und Anpassbarkeit der Geschäftsregeln ein wichtiges Merkmal von Business-Rule-Technologien, und ist somit auch im Rahmen dieser Masterarbeit umzusetzen. Somit soll es für facheinschlägig ausgebildete Personen möglich sein, die Inhalte der Rule-Base nach den benötigten Anforderungen auszurichten.

3.4 Externe Datenquellen

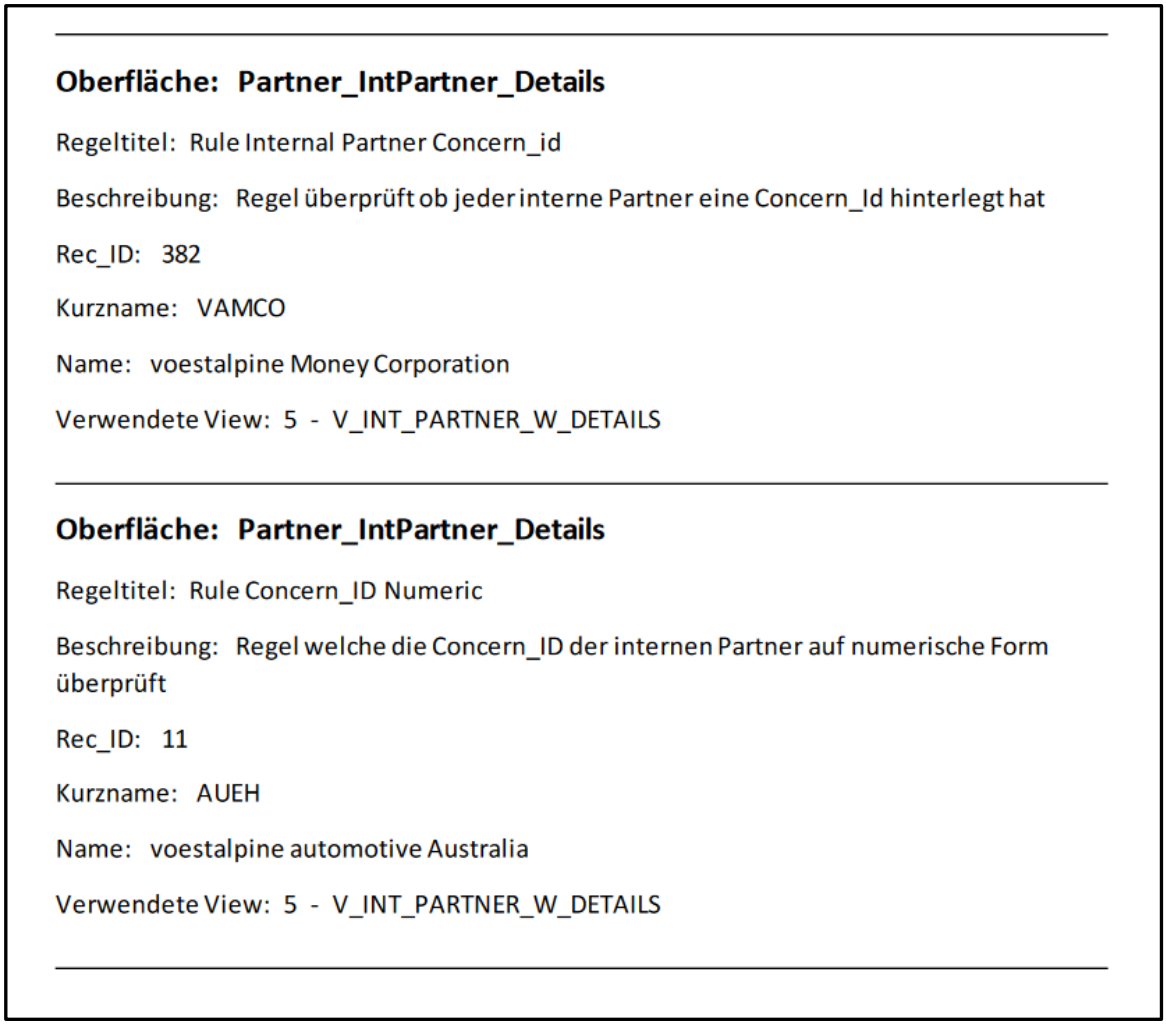
Zusätzlich zu den abteilungsinternen Datenbeständen werden externe Datenquellen, wie beispielsweise die Insighter-Datenbank, verwendet, um Informationen abzurufen, welche zur Steigerung der Datenqualität beitragen. Die Insighter-Datenbank stellt zusätzliche Informationen bezüglich Konzerngesellschaften und Mitarbeitern des voestalpine Konzerns zur Verfügung und stellt somit ein Beispiel für eine externe Datenquelle dar. Diese Datenbank wird von einer anderen konzerninternen Abteilung betrieben und verwaltet, ermöglicht es jedoch dem voestalpine Treasury per Lesezugriff aktuelle Informationen zu erhalten. Die erhaltenen Informationen können folglich für verschiedene Zwecke eingesetzt werden. Beispielsweise könnte eine Konzerngesellschaft aus dem Konzern ausgegliedert werden, und muss somit aus den zukünftigen Analysen und Reports exkludiert werden. Um in den Datenbeständen des voestalpine Treasury keine fehlerhaften Informationen zu behalten, wie den eben erwähnten Konzernstatus einer Gesellschaft, können die aktuellsten Informationen aus der Insighter-Datenbank abgerufen werden. Die Insighter-Datenbank stellt im Rahmen dieser Arbeit eine Referenzquelle dar, also eine Datenquelle, von der angenommen wird das die bezogenen Daten, sofern diese unterschiedlich zu den Daten des voestalpine Treasury sind, korrekt sind.

3.5 Data-Cleaning-Prototype

In diesem Schritt wird im Rahmen der Masterarbeit die eigentliche Implementierung des Prototyps verstanden, welcher das Herzstück des Data-Cleaning-Prozesses des voestalpine Treasury darstellt. Der Prototyp überprüft die mittels View-Layer bereitgestellten Datenbestände basierend auf den zugrundeliegenden Geschäftsregeln und generiert eine Rückmeldung in Form eines Data-Cleaning-Reports. Zusätzlich ist es möglich, mit dem Prototyp externe Datenquellen, wie die im obigen Abschnitt erwähnte Insighter-Datenbank, zu kontaktieren und relevante Informationen in den Cleaning-Schritt einfließen zu lassen.

3.6 Data-Cleaning-Report

Nach der Überprüfung der Datenstrukturen mittels Prototypen, wird ein Data-Cleaning-Report in Form eines Dokuments erstellt. Dieser gibt Aufschluss über den Zustand der überprüften Daten bezüglich der Konsistenzregelungen und wird im folgenden Prozessschritt von einem Sachbearbeiter des voestalpine Treasury analysiert. Gegebenenfalls werden die notwendigen Schritte, zur Behebung der Inkonsistenz, vom Sachbearbeiter eingeleitet. Ein Beispiel für einen Data-Cleaning-Report sieht wie folgt aus.



Oberfläche: Partner_IntPartner_Details

Regeltitel: Rule Internal Partner Concern_id

Beschreibung: Regel überprüft ob jeder interne Partner eine Concern_Id hinterlegt hat

Rec_ID: 382

Kurzname: VAMCO

Name: voestalpine Money Corporation

Verwendete View: 5 - V_INT_PARTNER_W_DETAILS

Oberfläche: Partner_IntPartner_Details

Regeltitel: Rule Concern_ID Numeric

Beschreibung: Regel welche die Concern_ID der internen Partner auf numerische Form überprüft

Rec_ID: 11

Kurzname: AUEH

Name: voestalpine automotive Australia

Verwendete View: 5 - V_INT_PARTNER_W_DETAILS

Abbildung 16: Ausschnitt Data-Cleaning-Report voestalpine Treasury

Abbildung 16 zeigt einen Ausschnitt aus einem Data-Cleaning-Report. Der Data-Cleaning-Report enthält beliebig viele Datenobjekte, welche gegen die Konsistenzregeln des voestalpine Treasury verstoßen, und somit einer Überprüfung durch einen Sachbearbeiter unterzogen werden müssen. Die horizontalen Linien stellen die Trennung zwischen den Objekten dar. Bei der Abarbeitung durch einen Sachbearbeiter kann dieser mit den Informationen, wie beispielsweise Oberfläche, Regeltitel, Regelbeschreibung und den angegebenen Attributen (vgl. Abbildung 5), das zu untersuchende Objekt im System eindeutig identifizieren und etwaige Änderungen vornehmen. Eine detaillierte Beschreibung des Data-Cleaning-Reports folgt im Abschnitt 6 .

3.7 Supervision

Die Informationen des Data-Cleaning-Reports geben Aufschluss über mögliche Inkonsistenzen in den Datenbeständen des voestalpine Treasury und dienen den Sachbearbeitern als Unterstützung bei deren Behebung. Im Weiteren bedeutet dies, dass der Prototyp im Rahmen dieser Masterarbeit zu keiner Zeit Änderungen automatisiert durchführen darf, da dies der Autorisierung der Sachbearbeiter bedarf und nur von letzteren manuell durchgeführt werden darf.

3.8 Reports und Analysen

Das voestalpine Treasury bedient sich einer Vielzahl an Reports und Analysen, welche ein wichtiges Werkzeug zur Einschätzung verschiedener Geschäftsentscheidungen darstellen. Reports und Analysen werden im voestalpine Treasury in verschiedenen Bereichen eingesetzt. Folgende Punkte stellen Beispiele für die Einsatzbereiche dar:

- Liquiditätsstatus
- Finanzplan
- Bankenlimits
- Nettopositionslimits
- FX-Berichte (engl. Foreign Exchange)
- Commodityberichte

Wie bereits in der Einleitung erwähnt, kann eine Verfälschung der Reports und Analysen, welche auf einen fehlerhaften Datenbestand zurückzuführen ist, aufgrund des finanziellen Volumens im voestalpine Treasury erhebliche Schäden verursachen. Daher ist es von allgemeinem Interesse, die Datenbestände mittels Data-Cleaning-Prozess fehlerfrei zu halten, um etwaige Schäden zu unterbinden.

Im aktuellen Kapitel wurde der Data-Cleaning-Prozess des voestalpine Treasury ganzheitlich betrachtet und die einzelnen Elemente präsentiert. In den nachfolgenden Kapiteln wird eine detaillierte Analyse der Hauptbereiche, an denen im Rahmen dieser Masterarbeit gearbeitet wurde, durchgeführt. Der erste der erwähnten Hauptbereiche ist die Abbildung der Geschäftsregeln, der zweite Hauptbereich behandelt die Umsetzung und Implementierung des Data-Cleaning-Prototyps und der dritte Hauptbereich befasst sich mit dem Endergebnis des Data-Cleaning-Prozesses, dem Data-Cleaning-Report.

4 Abbildung der Geschäftsregeln im voestalpine Treasury

Wie bereits in Abbildung 2 dargestellt, benötigt eine Business-Rule-Engine Geschäftsregeln, um die Prüfung von Datenbeständen durchführen zu können. Geschäftsregeln können, wie in Abschnitt 2.3 dargestellt, in verschiedenen Formen abgebildet werden. Beispielsweise können Geschäftsregeln lediglich in natürlicher Sprache, schriftlich auf Papier, abgebildet sein. Geschäftsregeln in natürlicher Sprache sind oftmals der Ausgangspunkt für weitere Modellierarbeit, da diese in einem Unternehmen bereits schriftlich oder in Form von Wissen in den Köpfen der Domänenexperten, vorliegen. Aufgrund der Struktur der natürlichen Sprache, welche Problempunkte, wie beispielsweise Mehrdeutigkeiten [9], im Bereich der Weiterverarbeitung durch Maschinen aufweist, ist eine formale Abbildung der Geschäftsregeln von allgemeinem Interesse. Wurden die Geschäftsregeln in natürlicher Sprache, unter Verwendung von beispielsweise SBVR oder OCL (siehe Abschnitt 2.3) formal abgebildet, so muss anschließend untersucht werden, wie eine Transformation in eine für die Business-Rule-Engine ausführbare Form umgesetzt werden kann. Wird eine am Markt erhältliche Business-Rule-Engine, wie beispielsweise Drools verwendet, so müssen die formal abgebildeten Regeln korrekt in die von Drools verwendete Abbildungsform überführt werden. Ein anderes Business-Rule-Engine-Produkt verwendet unter Umständen eine andere Form zur Abbildung der Geschäftsregeln. Grundsätzlich gilt somit, dass bei der Verwendung von am Markt zugänglichen Business-Rule-Engines, die Geschäftsregeln an das von den Entwicklern vorausgesetzte Format angepasst werden müssen. Bei der Implementierung einer eigenen Business-Rule-Engine, kann die Form, in der die ausführbaren Geschäftsregeln abgebildet werden, dementsprechend selbst gewählt werden. Im nächsten Abschnitt werden die damit verbundenen Designentscheidungen, welche im voestalpine Treasury getroffen wurden, vorgestellt.

4.1 Designentscheidungen

Da die Abbildung der Geschäftsregeln und die Umsetzung des Data-Cleaning-Prototyps, welcher die Business-Rule-Engine im Rahmen dieser Masterarbeit darstellt, in enger Verbindung zueinander stehen und demnach gegenseitige Auswirkungen mit sich bringen, müssen an diesem Punkt bereits Informationen aus Kapitel 5 vorweggenommen werden. Diese Informationen unterstützen das Verständnis bezüglich der gewählten Methoden zur Abbildung der ausführbaren Geschäftsregeln. In Abbildung 17 werden die getroffenen Designentscheidungen und der Prozess zur Abbildung der Geschäftsregeln im voestalpine Treasury durch die grün markierten Elemente dargestellt.

Im voestalpine Treasury waren bisher noch keine Geschäftsregeln dokumentiert, das gesammelte Wissen bezüglich Datenqualität und Datenkonsistenz war lediglich in den Köpfen der verschiedenen Domänenexperten vorhanden. Im voestalpine Treasury stellen die in den verschiedenen Bereichen tätigen Sachbearbeiter die Domänenexperten im Rahmen dieser Masterarbeit dar. Da das Wissen nur verteilt vorlag, konnte eine Überprüfung der aus dem Wissen abgeleiteten Geschäftsregeln nie ganzheitlich durchgeführt werden. An diesem Punkt wurde im Rahmen dieser Masterarbeit angesetzt und mit der Dokumentation der Geschäftsregeln in natürlicher Sprache, in schriftlicher Form, begonnen.

Wie bereits im Abschnitt 2.3 erwähnt, konnten für den nächsten Prozessschritt, der formalen Abbildung der Geschäftsregeln, zwei Optionen identifiziert werden. SBVR und OCL wurden im Rahmen der Anwendung im voestalpine Treasury in Erwägung gezogen, deren Charakteristiken abgewogen, und die den Anforderungen am besten entsprechende Option ausgewählt. Letztendlich wurde SBVR aus den nachfolgenden Gründen ausgewählt, um im Rahmen dieser Masterarbeit die formale Abbildung der Geschäftsregeln zu unterstützen.

Der primäre Fokus von SBVR und OCL liegt auf der Modellierung, jedoch unterstützen sie unterschiedliche Anwendungsdomänen. SBVR wurde zur Verwendung durch Geschäftsleute entwickelt und stellt eine Option zur Geschäftsmodellierung dar, welche unabhängig von den zugrundeliegenden IT-Systemen ist. OCL wurde hingegen zur Verwendung durch IT-Personal, zur Modellierung von Softwaresystemen entwickelt. Im Rahmen dieser Masterarbeit wurde SBVR ausgewählt, da die Verwendung des Vokabulars und der Geschäftsregeln durch die Sachbearbeiter des voestalpine Treasury möglich ist. SBVR stellt somit wie in Abbildung 17 dargestellt, ein Bindeglied zwischen der natürlichen Sprache und den ausführbaren Geschäftsregeln dar und dient als Arbeitsbasis für die stetige Weiterentwicklung des gesamten Data-Cleaning-Prozesses, da SBVR die Mitarbeit der Sachbearbeiter, welche keine IT-Experten sind, ermöglicht. OCL kann in dem eben genannten Punkten nicht dieselbe Funktionalität bereitstellen und ist deshalb für den Einsatz im voestalpine Treasury nicht ausgewählt worden.

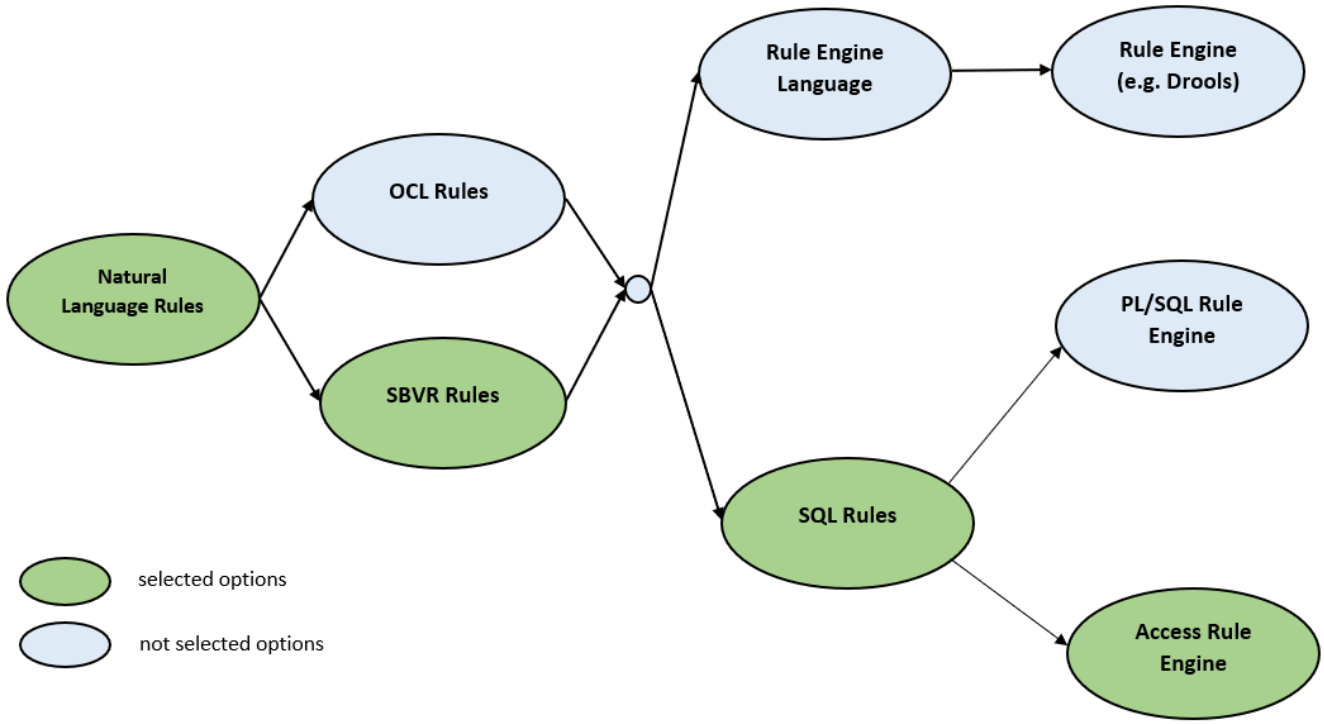


Abbildung 17: Umsetzungsoptionen und Auswahl im Rahmen dieser Masterarbeit

Der nächste Schritt im Prozess der Abbildung der Geschäftsregeln aus Abbildung 17, das Überführen der SBVR-Geschäftsregeln in eine ausführbare Form, hängt, wie erwähnt, unmittelbar mit der Implementierung des Data-Cleaning-Prototyps zusammen. Wird eine externe Business-Rule-Engine-Lösung verwendet, so müssen die Geschäftsregeln in das vorgegebene Format überführt werden. Im Rahmen dieser Masterarbeit wurde nach Abwägung der Optionen mit der Leitung des voestalpine Treasury entschieden, den Data-Cleaning-Prototyp als Individuallösung zu implementieren. Ein Grund wieso eine Individuallösung angestrebt wurde, ist das vorhanden sein der notwendigen Kompetenzen im voestalpine Treasury im Bereich der Softwareentwicklung, um zukunftsorientiert die Wartung des Data-Cleaning-Prototyps zu übernehmen. Des Weiteren wurde vom Einsatz eines externen Produkts, aufgrund der besonderen Anforderungen, wie beispielsweise das in Abbildung 12 erwähnte kontaktieren von weiteren externen Datenquellen, abgesehen. Folglich stellt die Freiheit bei der Implementierung von Individuallösungen, im Falle des voestalpine Treasury einen erheblichen Vorteil dar.

Das im Abschnitt 2.4.2 vorgestellte regelbasierte System von Abdullah et al. [34], war für die Implementierung des Data-Cleaning-Prototyps für das voestalpine Treasury, wegweisend. Abdullah et al. [34] verwendeten SQL-Abfragen zur Abbildung der Geschäftsregeln im Rahmen der Überprüfung der Datenbestände. Des Weiteren wurde das regelbasierte System in Form von Stored-Procedures implementiert. Diese Architektur wurde ebenfalls für den Data-Cleaning-Prototyp angedacht, jedoch wurde die Implementierung in Form von Stored-Procedures, in Rücksichtnahme auf die im voestalpine Treasury vorhandenen Kompetenzen abgeändert. Eine Vorgabe des voestalpine Treasury besteht in der Weiterbetreuung des Data-Cleaning-Prototyps durch einen facheinschlägig ausgebildeten Mitarbeiter, über die Zeit der Beschäftigung des Autors hinaus. Daher wurde im Bereich der Implementierung, sofern keine Qualitätseinbußen auftraten, vorrangig auf die eben erwähnte Person Rücksicht genommen. Aufgrund dieser Gegebenheiten wurde entschieden, die ausführbaren Regeln nach Abdullah et al. [34] auf Basis von SQL-Abfragen zu entwickeln. Der Data-Cleaning-Prototyp und dessen Business-Rule-Engine-Funktionalität, wurden in Form einer Access-Datenbank, unter Zuhilfenahme von Visual Basic for Applications (Abk. VBA) erstellt.

In den folgenden drei Unterkapiteln wird der beschriebene Prozess zur Abbildung der Geschäftsregeln, unter Berücksichtigung der in diesem Kapitel vorgestellten Designentscheidungen, an einem durchgängigen Beispiel erläutert.

4.2 Geschäftsregeln in natürlicher Sprache

Wie bereits im Abschnitt 4.1 erwähnt, stellen die Geschäftsregeln in natürlicher Sprache den Ausgangspunkt für den Prozess zur Abbildung der Geschäftsregeln im voestalpine Treasury dar. Dabei wurde im Rahmen dieser Masterarbeit, wie in der Einleitung beschrieben, der Fokus auf einen Teilbereich der Daten gelegt, da ansonsten der Rahmen überschritten werden würde. Die Geschäftsregeln wurden vom Autor im Interview mit den Sachbearbeitern des voestalpine Treasury entgegengenommen und in natürlicher Sprache niedergeschrieben. Das Endergebnis dieses Schrittes war ein Dokument, mit den gesammelten Geschäftsregeln in natürlicher Sprache. Die nachfolgende Abbildung zeigt ein Beispiel für die niedergeschriebenen Geschäftsregeln in natürlicher Sprache. Alle Geschäftsregeln wurden in englischer Sprache dokumentiert, da die Konzernsprache der voestalpine Englisch ist.

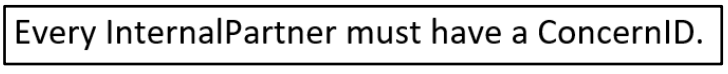


Abbildung 18: Beispielregel in natürlicher Sprache

Die in Abbildung 18 dargestellte Geschäftsregel drückt aus, dass jeder interne Partner im System der voestalpine Treasury eine `ConcernID` benötigt. Man kann erkennen, dass eine Weiterverarbeitung aufgrund der Struktur der natürlichen Sprache, ohne zusätzliche Informationen sich als schwer gestaltet. Beispielsweise können unterschiedliche Sachbearbeiter unter dem Begriff `InternalPartner` verschiedene Dinge verstehen. Folglich kann bei der Weiterverarbeitung oder Weitergabe der Geschäftsregeln Informationsverlust auftreten. Ebenfalls können Mehrdeutigkeiten in der natürlichen Sprache zu Problemen in der Weiterverarbeitung führen. Die nachfolgende Abbildung stellt diesen Sachverhalt dar.

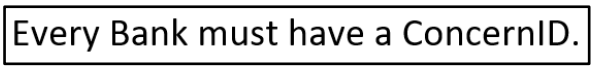


Abbildung 19: Beispielregel Mehrdeutigkeiten in natürlicher Sprache

Die in Abbildung 19 dargestellte Geschäftsregel weist eine erhebliche Schwäche auf. Die Bedeutung des Begriffes „Bank“, kann ohne zusätzliche Informationen, nicht eindeutig festgelegt werden. Da der Begriff „bank“ (zu Deutsch Bank), in der englischen Sprache sowohl ein Finanzinstitut beschreibt als auch Flussufer bedeutet, jedoch die dieselbe Schreibweise verwendet, kann die Bedeutung dieses Begriffes in einer Geschäftsregel nicht eindeutig definiert werden. Diese Tatsache kann sowohl bei einer maschinellen als auch bei einer Weiterverarbeitung durch Benutzer erhebliche Probleme auslösen. Werden die Geschäftsregeln beispielsweise im Rahmen einer Besprechung durch mehrere Sachbearbeiter verwendet, so kann der Fall auftreten, dass mehrere Sachbearbeiter den Begriff „bank“ als Finanzinstitut interpretieren, und andere Sachbearbeiter die Bedeutung desselben Begriffs als Ufer verstehen. Oftmals kann dieses Problem zwar durch domänenspezifischen Kontext gelöst werden, jedoch kann in Form von natürlicher Sprache keine Sicherheit gewährleistet werden.

4.3 Geschäftsregeln unter Verwendung von SBVR

Um die im letzten Unterkapitel angesprochenen Problempunkte der natürlichen Sprache zu unterbinden, wurde, wie in Abschnitt 4.1 beschrieben, im voestalpine Treasury entschieden, die Geschäftsregeln in natürlicher Sprache, unter Zuhilfenahme von SBVR (siehe 2.3.1), in ein SBVR-Vokabular und dazugehörige Geschäftsregeln zu überführen. In Abbildung 20 ist die Geschäftsregel aus Abbildung 18, nach der Überführung in das SBVR-Format dargestellt.

Abbildung 20: Beispiel Geschäftsregel in SBVR

Die von SBVR verwendeten Elemente und die Bedeutung der Farbtöne im SBVR-Vokabular und den Geschäftsregeln, wurden ausführlich in Abschnitt 2.3.1 beschrieben. In Abbildung 20 ist erkennbar, dass zwei SBVR-Vokabeln, auch Konzepte genannt, verwendet werden. Der Geschäftsregel muss demnach ebenfalls die verwendeten Vokabeln beigelegt werden.

Wird nun die in Abbildung 20 dargestellte Geschäftsregel verwendet, so gibt die Vokabel `InternalPartner`, aus dem zugehörigen SBVR-Vokabular, Aufschluss über die Bedeutung des verwendeten Konzepts. Wie bereits in Abschnitt 2.3.1 erwähnt, kann das SBVR-Vokabular unterschiedliche Formen annehmen. In Abbildung 21 ist grundlegendes Beispiel dargestellt. Um nun die verwendeten Vokabeln zu komplettieren, ist in Abbildung 22 der Vokabeleintrag für `ConcernID` dargestellt.

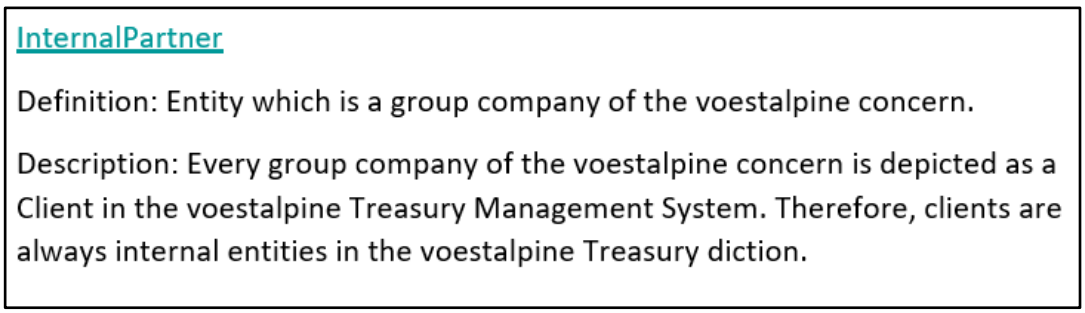


Abbildung 21: SBVR-Vokabel InternalPartner

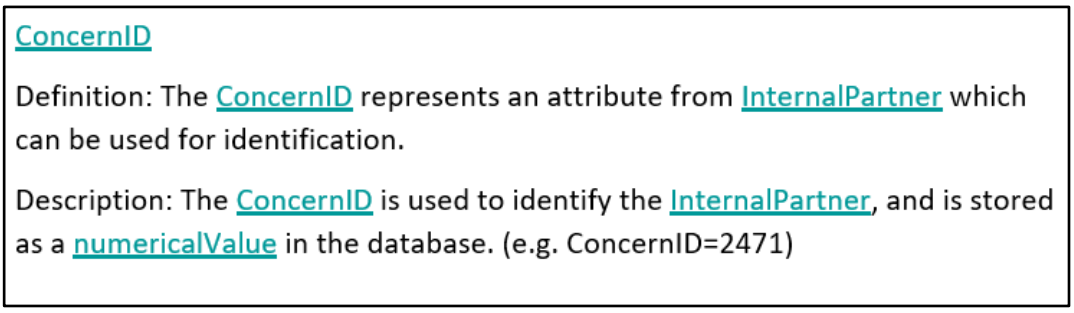


Abbildung 22: SBVR-Vokabel ConcernID

Da alle verwendeten Vokabeln aus den Geschäftsregeln im SBVR-Vokabular aufscheinen müssen, können bei der Verwendung der Geschäftsregeln keine Missverständnisse bezüglich Bedeutung der verwendeten Konzepte aufkommen. Die verbleibenden Bestandteile der SBVR-Geschäftsregel wie Schlüsselwörter (oranger Farbton) oder Verb-Konzepte (blauer Farbton) müssen, wie in Abschnitt 2.3.1 beschrieben, nicht erneut im eigenen Vokabular definiert werden, da hier Definitionen der Object Management Group für die gängigsten Phrasen verwendet werden können. Ist eine Geschäftsregel und das verwendete Vokabular nach dem SBVR-Format korrekt angelegt, so können diese sowohl zur Weiterverarbeitung durch Menschen als auch (Software-)Werkzeugen verwendet werden. Im Falle des voestalpine Treasury werden die Geschäftsregeln und das Vokabular im Rahmen von Workshops zur Weiterentwicklung des Data-Cleaning-Prozesses verwendet. Ebenfalls kann das SBVR-Content-Model, welches das Vokabular und die Geschäftsregeln beinhaltet, in das XML-Format überführt werden und so zwischen Organisationen und (Software-)Werkzeugen ausgetauscht werden.

Um nun im Rahmen einer Business-Rule-Engine automatisiert Überprüfungen der Datenbestände durchführen zu können, müssen die Geschäftsregeln jedoch in eine ausführbare Form transformiert werden. SBVR deckt diese Funktionalität nicht ab, da der Fokus auf Geschäftsleute, unabhängig von den zugrundeliegenden IT-Systemen, gelegt wird. Folglich musste im Rahmen dieser Masterarbeit eine Lösung gefunden werden, um die Geschäftsregeln aus dem SBVR-Format in eine ausführbare Form zu überführen.

4.4 Ausführbare Geschäftsregeln in Form von SQL-Abfragen

Im Folgenden wird auf den Prozess der Abbildung der Geschäftsregeln aus Abbildung 17 eingegangen. Wie bereits erwähnt, müssen die SBVR-Geschäftsregeln für den Einsatz, in Verbindung mit dem Data-Cleaning-Prototyp, in eine ausführbare Form gebracht werden. Diese ausführbare Form wird im Rahmen dieser Masterarbeit durch die Darstellung der Geschäftsregeln als SQL-Abfragen erreicht. Nach der Abbildung der Geschäftsregeln werden diese im Speicher der Business-Rule-Engine abgelegt. Eine detaillierte Beschreibung der Abläufe der Business-Rule-Engine wird im Kapitel 5 erläutert. In den nachfolgenden Absätzen wird auf die für das voestalpine Treasury umgesetzte Struktur der SQL-Abfragen, welche die Geschäftsregeln abbilden, eingegangen. Ein wichtiger Unterschied zwischen den Geschäftsregeln in natürlicher Sprache sowie den SBVR-Geschäftsregeln und den SQL-Abfragen, welche die erwähnten Regeln repräsentieren, besteht in der allgemeinen Struktur. Da im Rahmen von SQL-Abfragen keine Regeln wie in natürlicher Sprache (siehe Abbildung 18) oder SBVR-Geschäftsregeln (siehe Abbildung 20) erstellt werden können, muss eine andere Strategie verfolgt werden. Die nachfolgende Abbildung 23 stellt die Vorgänge im Rahmen der Transformation dar. Die ersten beiden Schritte von der Geschäftsregel in natürlicher Sprache zur SBVR-Geschäftsregel wurden bereits im oberen Teil dieses Kapitels beschrieben. Im dritten Schritt werden die SBVR-Geschäftsregeln durch SQL-Abfragen repräsentiert. Im Rahmen dieses Schrittes werden die SQL-Abfragen so angelegt, um jene Datensätze zu identifizieren, welche gegen die SBVR-Geschäftsregeln verstoßen. In Abbildung 23 wird das bereits in Abbildung 18 und Abbildung 20 gezeigte Beispiel weitergeführt.

Die SBVR-Geschäftsregel drückt die Notwendigkeit des Vorhandenseins einer `ConcernID` für jeden `InternalPartner` aus. Folglich sind jene Datensätze in unserer Datenbank von Interesse, welche diese Vorgabe nicht erfüllen, da kein Eintrag in der Spalte `ConcernID` existiert und die Spalte somit den Wert `null` beinhaltet. Die im unteren Teil der Abbildung 23 abgebildete SQL-Abfrage erfüllt exakt diese Überprüfung, und liefert alle Datensätze, welche der Geschäftsregel zufolge eine `ConcernID` hinterlegt haben sollten, jedoch in Wirklichkeit den Wert `null`, also keinen Eintrag, hinterlegt haben. Die Datensätze im Ergebnis der SQL-Abfrage stellen somit potenzielle Fehler in den Datenbeständen dar, da sie gegen die Geschäftsregel verstoßen haben, und werden für die Untersuchung durch einen Sachbearbeiter des voestalpine Treasury in den Data-Cleaning-Report geschrieben. Resultierend aus diesem Beispiel kann festgehalten werden, dass im Rahmen dieser Masterarbeit die SQL-Abfragen indirekt die Einhaltung der Geschäftsregeln sicherstellen, da Datensätze, welche gegen eine Geschäftsregel verstoßen, identifiziert werden.

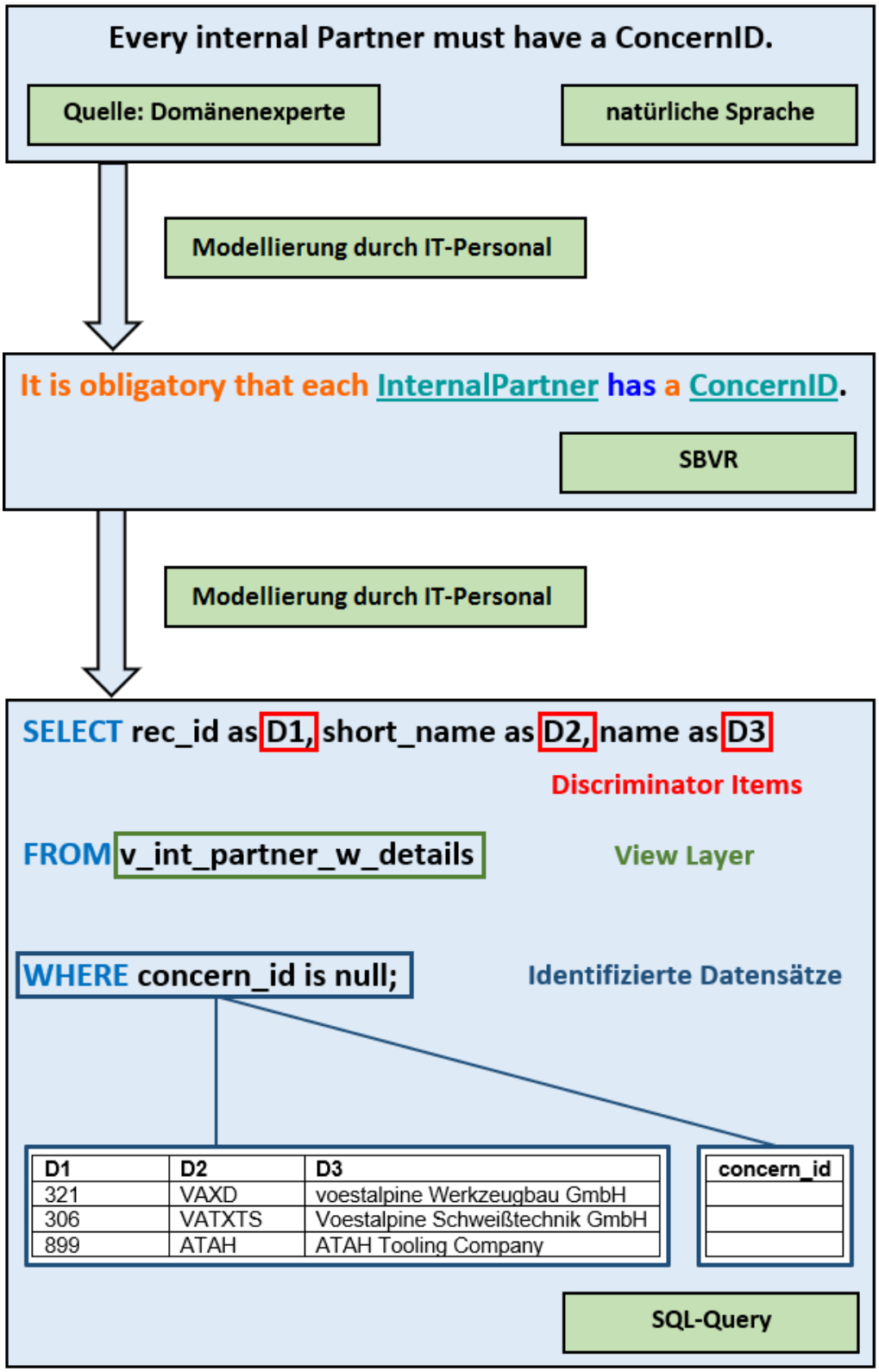


Abbildung 23: Detailprozess zur Abbildung der Geschäftsregeln im voestalpine Treasury

Um die für die Anwendung im voestalpine Treasury notwendige Funktionalität bereitstellen zu können, wurden gewisse Designentscheidungen bezüglich der Struktur des SQL-Abfragen umgesetzt, welche eingehalten werden müssen. Diese Designentscheidungen wurden in Abbildung 23 im unteren Bereich durch farbige Rechtecke und dazugehörige Begriffe hervorgehoben. Im Select-Teil der SQL-Abfrage befindet sich die erste Konvention, die Discriminator Items, welche im Rahmen dieser Masterarbeit eingeführt wurde. Diese Konvention bezüglich der Struktur des Select-Teils der SQL-Abfragen sieht vor, dass jene Spalten aus der Datenbank selektiert werden, die das Objekt, worum es in der Geschäftsregel geht, bestmöglich beschreiben. In dem in Abbildung 23 behandelten Beispiel ist das Objekt, welches überprüft wird, ein interner Partner. Folglich werden Spalten ausgewählt, welche diesen bestmöglich identifizieren. Das Auswählen der passenden Spalten ist ausschlaggebend für die Weiterverarbeitung im Data-Cleaning-Report sowie für den Schritt der Supervision (siehe Abbildung 12) durch einen Sachbearbeiter des voestalpine Treasury. Erhält ein Sachbearbeiter den Data-Cleaning-Report, und führt die manuelle Überprüfung und etwaige Bereinigung der im Report identifizierten Datensätze durch, so müssen die Objekte, wie beispielsweise ein interner Partner, durch die im Report vermerkten Attribute bestmöglich identifiziert werden können. Das nachfolgende Beispiel soll den beschriebenen Sachverhalt erläutern.

Abbildung 24: Ausschnitt aus dem Data-Cleaning-Report

In Abbildung 24 ist ein Ausschnitt, welcher einen potenziell fehlerhaften Datensatz repräsentiert, aus dem Data-Cleaning-Report dargestellt. Der Sachbearbeiter bearbeitet nun diesen Datensatz und muss überprüfen ob dieser Datensatz rechtmäßig im Data-Cleaning-Report inkludiert wurde, und somit ein Fehler vorliegt. Um diese Überprüfung durchzuführen steigt der Sachbearbeiter im Treasury-Management-System ein und navigiert zur Ansicht der Partner, da aus dem Report ersichtlich ist, dass die Partneroberfläche betroffen ist (siehe Abbildung 24). Um nun den richtigen Partner, aus den rund fünfhundert Partnern zur Überprüfung heranziehen zu können, nutzt er die Informationen aus dem Data-Cleaning-Report. An diesem Punkt zeigt sich die Nützlichkeit der Discriminator Items. Durch die Steuerung der im Ergebnis der SQL-Abfrage beinhalteten Spalten, und der Hinterlegung der dazugehörigen Discriminator Tags im Bereich der Metadaten der Geschäftsregel, kann ein Sachbearbeiter den zu überprüfenden Datensatz leicht identifizieren. Die Discriminator Tags und deren Funktionsweise werden im Kapitel 5 zusammen mit dem Data-Cleaning-Prototyp behandelt.

Der Sachbearbeiter nimmt somit die Informationen aus dem Data-Cleaning-Report wie in diesem Beispiel die `Rec_ID` (Eindeutige Identifikation im Treasury-Management-System), den `Kurznamen` (Abkürzung für den Konzerngesellschaftsnamen, oft verwendet in Suchfunktionen) oder den `Namen`,

und kann dadurch sofort zum Untersuchungsobjekt navigieren und den potenziellen Fehler beheben. Würden diese Informationen nicht bereitgestellt werden, so kann der Sachbearbeiter nicht nachvollziehen um welchen Datensatz es sich handelt, und die Information des Vorhandenseins eines Fehlers aus dem Data-Cleaning-Report wäre von keinem Wert. Ebenfalls können nicht einfach alle Spalten im Select-Teil der SQL-Abfrage inkludiert werden. Es würden zwar auch die relevanten Spalten aus Abbildung 24 in der Ergebnismenge sein, jedoch können die verwendeten Views eine hohe Anzahl an Spalten beinhalten, wodurch eine Weiterverarbeitung im Data-Cleaning-Report aufgrund der Unübersichtlichkeit unmöglich wird. Aufgrund des eben besprochenen Sachverhalts wurde das Konzept der Discriminator Items umgesetzt, und in die Struktur der ausführbaren Geschäftsregeln, und die SQL-Abfragen, die sie repräsentieren, eingebettet. Um die Discriminator Items in der Programmlogik des Data-Cleaning-Prototyps generisch behandeln zu können, und somit den Namen der einzelnen Spalten, wie beispielsweise `Rec_ID`, nicht in der Programmlogik festlegen zu müssen, wurden Aliasnamen für alle im Select-Teil der SQL-Abfragen beinhalteten Items vergeben. Dabei steht der Aliasname `D1` (siehe Abbildung 24) lediglich für das erste Discriminator Item, der Aliasname `D2` für das zweite, und `D3` für das dritte. Die Anzahl der verwendeten Discriminator Items wurde durch die Untersuchung der Datenbankobjekte, wie beispielsweise Partner oder Kontaktpersonen, durch den Autor vorläufig auf drei gesetzt. Dabei wurde vom Autor untersucht wie viele Discriminator Items für die verschiedenen Datenbankobjekte notwendig sind, um diese hinsichtlich der Weiterverarbeitung durch die Sachbearbeiter, eindeutig zu identifizieren. Zusätzlich wurde anschließend ein Versuchstest mit den Sachbearbeitern durchgeführt wo letzteren ein Report mit allen unterschiedlichen Datenbankobjekten, und drei dazugehörigen Discriminator Items, vorgelegt wurde. Dieser Versuchstest konnte von allen Sachbearbeitern mit den Informationen der drei Discriminator Items erfolgreich durchgeführt werden. Sollten sich die Anforderungen im voestalpine Treasury in Zukunft ändern, so wurden bei der Implementierung des Data-Cleaning-Prototyps bereits die notwendigen Vorkehrungen getroffen. Die Anzahl der Discriminator Items kann nach Belieben abgeändert werden, ohne die Programmlogik des Data-Cleaning-Prototyps zu verändern. Eine genauere Beschreibung wird ebenfalls in Kapitel 5 vorgenommen.

Neben den Discriminator Items ist in Abbildung 23 ebenfalls der Begriff View-Layer dargestellt. Wie bereits im Kapitel 3 Architektur vorgestellt, wird im Rahmen dieser Masterarbeit nicht direkt auf die die Daten enthaltenden Datenbanktabellen zugegriffen, sondern auf die für die Überprüfung optimierten Views, welche, wie in Abschnitt 3.2 beschrieben, bereits vorab Joins zwischen mehreren Datenbanktabellen umsetzen, um diese Joins aus dem Verantwortungsbereich der SQL-Abfragen zu entfernen. Alle verfügbaren Views wurden den Anforderungen entsprechend vom Autor erstellt und mit dem Präfix „v_“ versehen (vgl. Abbildung 23, Abbildung 24). Der Begriff „identifizierte Datensätze“ aus Abbildung 23 beschreibt jene Ergebnismenge an Datensätzen aus der Datenbank, welche gegen überprüfte Geschäftsregel verstoßen haben, somit potenziell fehlerhaft sind, und daher in den Data-Cleaning-Report aufgenommen werden. Nach der Überprüfung aller Regeln werden die identifizierten Datensätze formatiert und im Data-Cleaning-Report abgebildet.

4.5 Verwendete Regelarten im voestalpine Treasury

Im nachfolgenden Unterkapitel werden die derzeit im voestalpine Treasury verwendeten Regelarten präsentiert. Die Regelarten, und die einzelnen Geschäftsregeln, welche jeweils einer Regelart angehören, wurden vom Autor nach den Instruktionen der Leitung des voestalpine Treasury angelegt. Um die verschiedenen Regelarten anhand von Beispielen erläutern zu können, werden Ausschnitte aus den Data-Cleaning-Reports verwendet. Detaillierte Informationen zum Data-Cleaning-Report folgen in Kapitel 6 . Im Rahmen dieser Masterarbeit wurden zu Demonstrationszwecken rund 50 Beispielregeln vom Autor, in Kooperation mit dem voestalpine Treasury, erarbeitet. Diese 50 Beispielregeln konnten in vier verschiedene Regelarten eingeteilt werden. Die grundlegendste der vier Regelarten setzt an dem Punkt an, wo die Überprüfungen der Standardsoftware nicht mehr ausreichen. Die Standardsoftware übernimmt beispielsweise datenbanktechnische Überprüfungen wie das Einhalten eines Primärschlüssels oder der referentiellen Integrität. Seitens der Standardsoftware kann jedoch nicht erschöpfend überprüft werden, ob ein spezifisches Attribut in der Datenbank bei allen Datensätzen mit einem Wert versehen wurde, da dies von Kunde zu Kunde unterschiedlich sein kann, und dadurch Implementierungsaufwand für den Softwarelieferanten entstehen würde. Nach Abschluss dieser Masterarbeit müssen seitens des voestalpine Treasury Ressourcen investiert werden, und die vorhandene Basis an Geschäftsregeln ausgebaut werden, um das volle Potenzial des in dieser Masterarbeit behandelten Data-Cleaning-Prozess zu erreichen. Um in Zukunft weitere Arten von Geschäftsregeln im voestalpine Treasury hinzufügen zu können, wurden bei der Implementierung die notwendigen Vorkehrungen getroffen. Grundsätzlich kann der Data-Cleaning-Prototyp jede Geschäftsregel verarbeiten, unter der Voraussetzung, dass diese durch eine korrekte SQL-Abfrage ausgedrückt werden kann. Bei verteilten Geschäftsregeln ist die externe Datenquelle ein zusätzlicher Faktor. Letztere muss die benötigten Daten in der richtigen Form zur Verfügung stellen, um neue verteilte Geschäftsregeln hinzufügen zu können. Eine genauere Beschreibung der verteilten Geschäftsregeln folgt in Abschnitt 5.5 . In Abbildung 25 werden die derzeit im voestalpine Treasury eingesetzten Regelarten dargestellt.

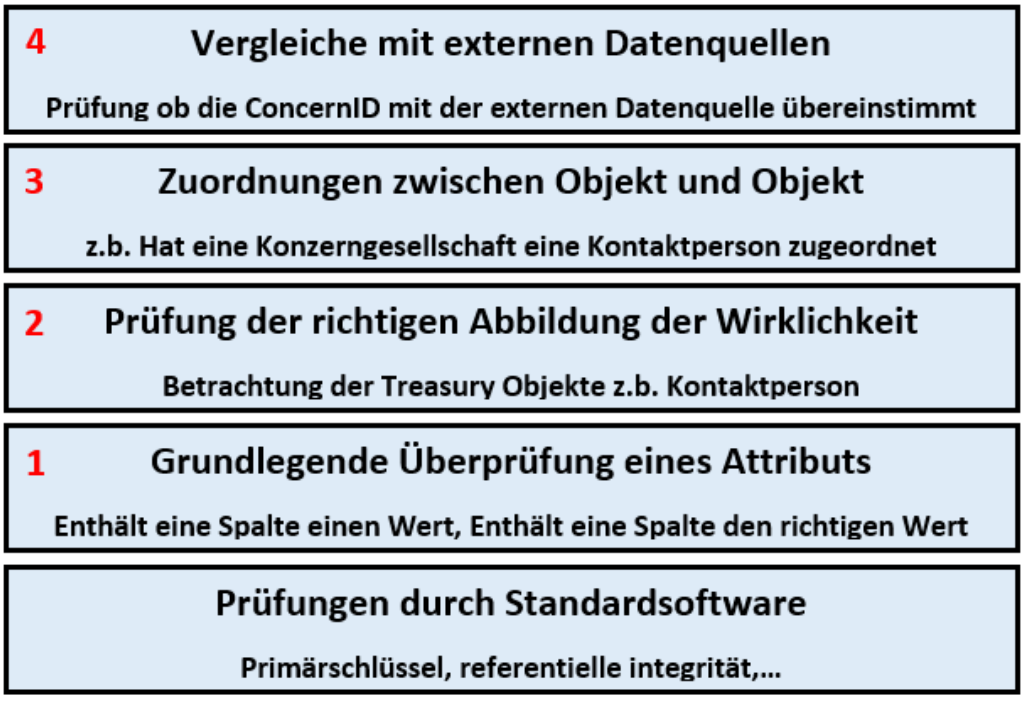


Abbildung 25: Verschiedene Regelarten im voestalpine Treasury

Ein Beispiel für die erste Regelart aus Abbildung 25, einer grundlegenden Überprüfung eines Attributs, wurde bereits als durchgängiges Beispiel in Kapitel 4 präsentiert. Die erwähnte Geschäftsregel überprüft, ob jeder `InternalPartner` eine `ConcernID` hinterlegt hat. Ein weiteres Beispiel für diese Art von Geschäftsregel ist die Überprüfung ob die `ConcernID` eines Partners ein vierstelliger numerischer Wert ist. Geschäftsregeln in dieser Form, welche ein Attribut eines voestalpine Treasury-Objekts, wie beispielsweise eines internen Partners oder einer Kontaktperson überprüfen, werden im Rahmen dieser Masterarbeit in die Kategorie 1 eingereiht. Ein hoher Prozentsatz der Geschäftsregeln im Beispielset führt Überprüfungen der Kategorie 1 durch, da eine Vielzahl an Verbesserungsmöglichkeiten aus diesem Bereich, in den Datenbeständen identifiziert werden konnte. In Abschnitt 2.1 wurden die verschiedenen Probleme bezüglich Datenqualität vorgestellt. Geschäftsregeln der Kategorie 1 tragen dazu bei, die in Tabelle 2 angeführten Fehlerquellen auf Attribut-Ebene zu identifizieren, und eine Behebung durch die Sachbearbeiter des voestalpine Treasury einzuleiten.

Geschäftsregeln der Kategorie 2 betrachten bereits das Treasury-Objekt als Ganzes, und fokussieren sich nicht auf ein einzelnes Attribut. Das nachfolgende Beispiel stellt eine Geschäftsregel der Kategorie 2 dar, welche den Ausschluss von Duplikaten der Kontaktpersonen sicherstellt. Im voestalpine Konzern ist die Mailadresse eines Mitarbeiters eindeutig. Sollten zwei Mitarbeiter den selben Vornamen als auch Nachnamen haben, so erhält einer der beiden Mitarbeiter nicht die standardmäßige vorname.nachname@voestalpine.com Mailadresse, sondern wird mit einer leicht abgeänderten Mailadresse ausgestattet. Daher wurde entschieden, die Duplikat-Erkennung bei den Kontaktpersonen über die Mailadresse in Verbindung mit dem Vornamen und Nachnamen der Kontaktperson durchzuführen.

The EmailAdress of an active ContactPerson must be unique.

Natürliche Sprache

It is obligatory that each ContactPersonEmail of an active ContactPerson is unique.

SBVR

```
SELECT c1.rec_id as D1, c1.name as D2, c1.email as D3
FROM v_contact_person c1
WHERE c1.contact_person_status = 'Active' and EXISTS (SELECT c2.rec_id FROM
v_contact_person c2 WHERE c1.email = c2.email
AND c1.rec_id <> c2.rec_id AND c2.contact_person_status = 'Active')
ORDER BY c1.email asc;
```

SQL-Query

Abbildung 26: Beispiel für eine Geschäftsregel der Kategorie 2

Mit der in Abbildung 26 dargestellten Geschäftsregel werden Duplikate im Bereich der Kontaktpersonen identifiziert. Im ersten Schritt wird eine Mehrfachverwendung der Mailadresse unter den Kontaktpersonen durch die Durchführung der SQL-Abfrage hervorgehoben. In Verbindung mit den doppelt verwendeten Mailadressen werden dem Sachbearbeiter im Data-Cleaning-Report die `Rec_ID`, der `Vorname` und der `Nachname` der Kontaktperson angegeben. Wurden die potenziellen Duplikate im Bereich der Kontaktpersonen im Data-Cleaning-Report abgebildet, so erhält der Sachbearbeiter die in Abbildung 27 dargestellten Informationen.

Abbildung 27: Ausschnitt Data-Cleaning-Report für die Geschäftsregel Kategorie 2

Der Sachbearbeiter kann nun, basieren auf den in Abbildung 27 dargestellten Informationen, über die Beibehaltung beider Datensätze oder die Entfernung eines Datensatzes entscheiden. Wie in Abbildung 27 ersichtlich, liegt definitiv ein Fehler im Datenbestand vor, da entweder die Mailadresse, entgegen den konzernweiten Richtlinien, für zwei unterschiedliche Personen der echten Welt doppelt verwendet wurde oder dieselbe Person aus der echten Welt mit zwei unterschiedlichen `Rec_ID` doppelt angelegt wurde. Geschäftsregeln der Kategorie 2 tragen folglich zur Identifikation und Behebung von Fehlern auf den in Tabelle 2 angeführten Ebenen des Datensatzes sowie des Datensatz-Typs bei.

Geschäftsregeln der Kategorie 3 zielen auf die Überprüfung von Zuordnungen und Beziehungen von Treasury-Objekten, wie beispielsweise Partner und Kontaktpersonen, ab. Im nachfolgenden Beispiel wird eine Geschäftsregel der Kategorie 3 behandelt, welche die Zuordnung zwischen einer Konzerngesellschaft (interner Partner) und den Kontaktpersonen überprüft. Seitens des voestalpine Treasury ist vorgegeben, dass jede Konzerngesellschaft mindestens eine Kontaktperson mit der Rolle „Account Manager“ hinterlegt haben muss. Die nachfolgende Geschäftsregel überprüft diese Vorgabe und liefert jene Konzerngesellschaften zurück, welche diese Bedingung nicht erfüllen.

Every internalPartner must have at least one ContactPerson with the role AccountManager.

Natürliche Sprache

It is obligatory that each internalPartner has at least one ContactPerson whose functionalRole is AccountManager.

SBVR

```
SELECT erg.D1, erg.D2, erg.D3
FROM (SELECT c1.rec_id as D1, c1.short_name as D2, c1.name as D3,
ROW_NUMBER () OVER (PARTITION BY c1.rec_id, c1.short_name,
c1.name ORDER BY c1.rec_id, c1.short_name, c1.name) AS
rownumber
FROM v_partner_contfunct_contpers c1
WHERE not exists (SELECT c2.rec_id FROM
v_partner_contfunct_contpers c2 WHERE c1.rec_id = c2.rec_id AND
c2.partner_function = 'AccountManager')) erg
WHERE erg.rownumber = 1;
```

SQL-Query

Abbildung 28: Geschäftsregel der Kategorie 3

Mit der Überprüfung der in Abbildung 28 dargestellten Geschäftsregel erhält der Sachbearbeiter alle Konzerngesellschaften im Data-Cleaning-Report zurück, welche keine Kontaktperson mit der Rolle „Account Manager“ hinterlegt haben. Der Sachbearbeiter bekommt die Informationen wie in Abbildung 29 dargestellt zur Verfügung gestellt.

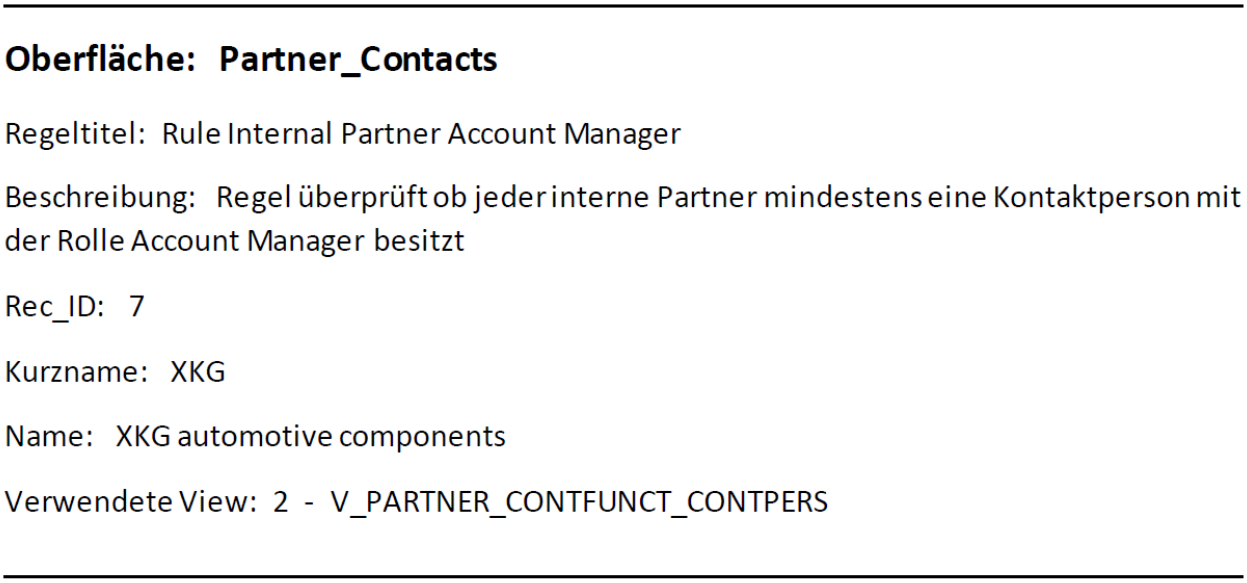


Abbildung 29: Ausschnitt Data-Cleaning-Report für die Geschäftsregel der Kategorie 3

Führt der Sachbearbeiter anschließend die manuelle Supervision durch, so navigiert er im Treasury-Management-System zum internen Partner mit dem `Namen` „XKG automotive components“ oder verwendet die `Rec_ID` oder den `Kurznamen` zur Identifikation des internen Partners. Anschließend kann der Sachbearbeiter dem internen Partner eine Kontaktperson zuweisen, um den Fehler zu beheben. Geschäftsregeln der Kategorie 3 tragen folglich zur Identifikation und Behebung von Fehlern auf der in Tabelle 2 angeführten Ebene des Datensatzes bei, da dieser Sachverhalt auf die Verletzung einer Attribut-Abhängigkeit, zwischen dem internen Partner und der Kontaktperson, heruntergebrochen werden kann.

Die vierte Kategorie der Geschäftsregeln stellen im Rahmen dieser Masterarbeit die verteilten Geschäftsregeln dar, welche Abgleiche mit externen Datenquellen ermöglichen. Verteilte Geschäftsregeln ermöglichen Überprüfungen der Datenbestände des voestalpine Treasury, unter Zuhilfenahme von Daten externer Datenquellen. Im Gegensatz zu den nicht-verteilten Geschäftsregeln, werden die verteilten Geschäftsregeln nicht gänzlich in Form einer SQL-Abfrage abgebildet. Es werden zwar jeweils für die Datenbank des voestalpine Treasury und für die externe Datenquelle SQL-Abfragen zur Beschaffung der Daten generiert, jedoch wurde der Abgleich der Daten mittels Programmlogik im Access-Modul umgesetzt. Eine genaue Beschreibung der Funktionalität von verteilten Geschäftsregeln folgt in Abschnitt 5.5 . Es wurden externe Datenquellen identifiziert, welche vollständig korrekte Daten, für Teilbereiche der Datenbestände des voestalpine Treasury zur Verfügung stellen. Folglich werden diese Datenquellen im Rahmen dieser Masterarbeit Referenzquellen genannt. Ein durchgängiges Beispiel, welches die Überprüfung einer verteilten Geschäftsregel und die damit verbundene Funktionalität abbildet, wird in Abschnitt 5.5 ausführlich behandelt. Wie bereits im oberen Teil dieses Unterkapitels erwähnt, werden, nach Abschluss dieser Masterarbeit und der endgültigen Inbetriebnahme des Data-Cleaning-Prozess im voestalpine Treasury, weitere Geschäftsregeln und Regelarten hinzugefügt. Für diesen Schritt werden Ressourcen in Form von Arbeitszeit der Mitarbeiter des voestalpine Treasury benötigt, um unter der Führung des Autors die bestmögliche Basis an Geschäftsregeln, für die Überprüfung der Datenbestände aufzubauen.

Im aktuellen Kapitel wurde der Prozess der Abbildung der Geschäftsregeln für das voestalpine Treasury detailliert beschrieben und anhand eines Leitbeispiels dargestellt. Es wurden bereits notwendige Information bezüglich der Implementierung des Data-Cleaning-Prototyps, welche zum Verständnis der Abbildung der Geschäftsregeln beigetragen haben, vorweggenommen. Im folgenden Kapitel werden die Implementierung und Umsetzung des Data-Cleaning-Prototyps als Business-Rule-Engine für das voestalpine Treasury präsentiert.

5 Data-Cleaning-Prototyp

Der Data-Cleaning-Prototyp ist, wie in Abbildung 12 ersichtlich, ein Kernelement des Data-Cleaning-Prozesses, welcher im voestalpine Treasury implementiert wurde. Der Data-Cleaning-Prototyp erlaubt es, mittels den in der Rule-Base abgelegten Geschäftsregeln im Form von SQL-Abfragen (siehe Kapitel 4), den Datenbestand des voestalpine Treasury auf potenzielle Fehler zu überprüfen und den Sachbearbeitern eine Rückmeldung in Form des Data-Cleaning-Reports zur Verfügung zu stellen. Der Data-Cleaning-Prototyp verfügt somit über die in Abschnitt 2.2 vorgestellte Funktionalität einer Business-Rule-Engine. In Abbildung 30 ist der vereinfachte Ablauf bei der Durchführung der Überprüfungen des Data-Cleaning-Prototyps in Form eines UML-Aktivitätsdiagramms dargestellt.

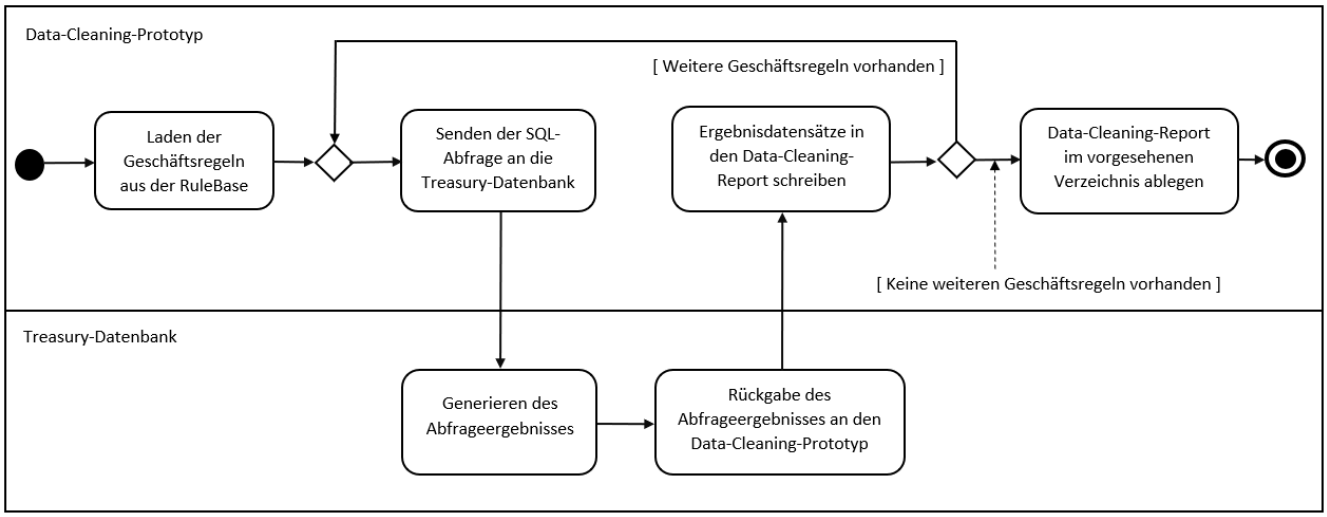


Abbildung 30: UML-Aktivitätsdiagramm - vereinfachter Ablauf der Überprüfung durch den Data-Cleaning-Prototyp

Als erster Schritt werden die in der Rule-Base abgelegten Geschäftsregeln geladen, um anschließend die Überprüfung vornehmen zu können. Anschließend werden alle Regeln nach demselben Schema abgearbeitet. Die SQL-Abfrage, welche die Geschäftsregel repräsentiert, wird an die Treasury-Datenbank, den zentralen Datenspeicher des voestalpine Treasury, übergeben. Anschließend wird seitens der Datenbank das Abfrageergebnis vorbereitet und wieder an den Data-Cleaning-Prototyp zurückgegeben. Im darauffolgenden Schritt werden die in der Ergebnismenge inkludierten Datensätze in den Data-Cleaning-Report aufgenommen. Nach Abarbeitung aller Geschäftsregeln wird der Report im Dateisystem, im vordefinierten Ordner, abgelegt. Um diese Funktionalität bereitzustellen, verwendet der Data-Cleaning-Prototyp mehrere Komponenten. Eine Übersicht über den Komponentenaufbau des Data-Cleaning-Prototyps und den relevanten Datenbankkomponenten ist in Abbildung 31 dargestellt.

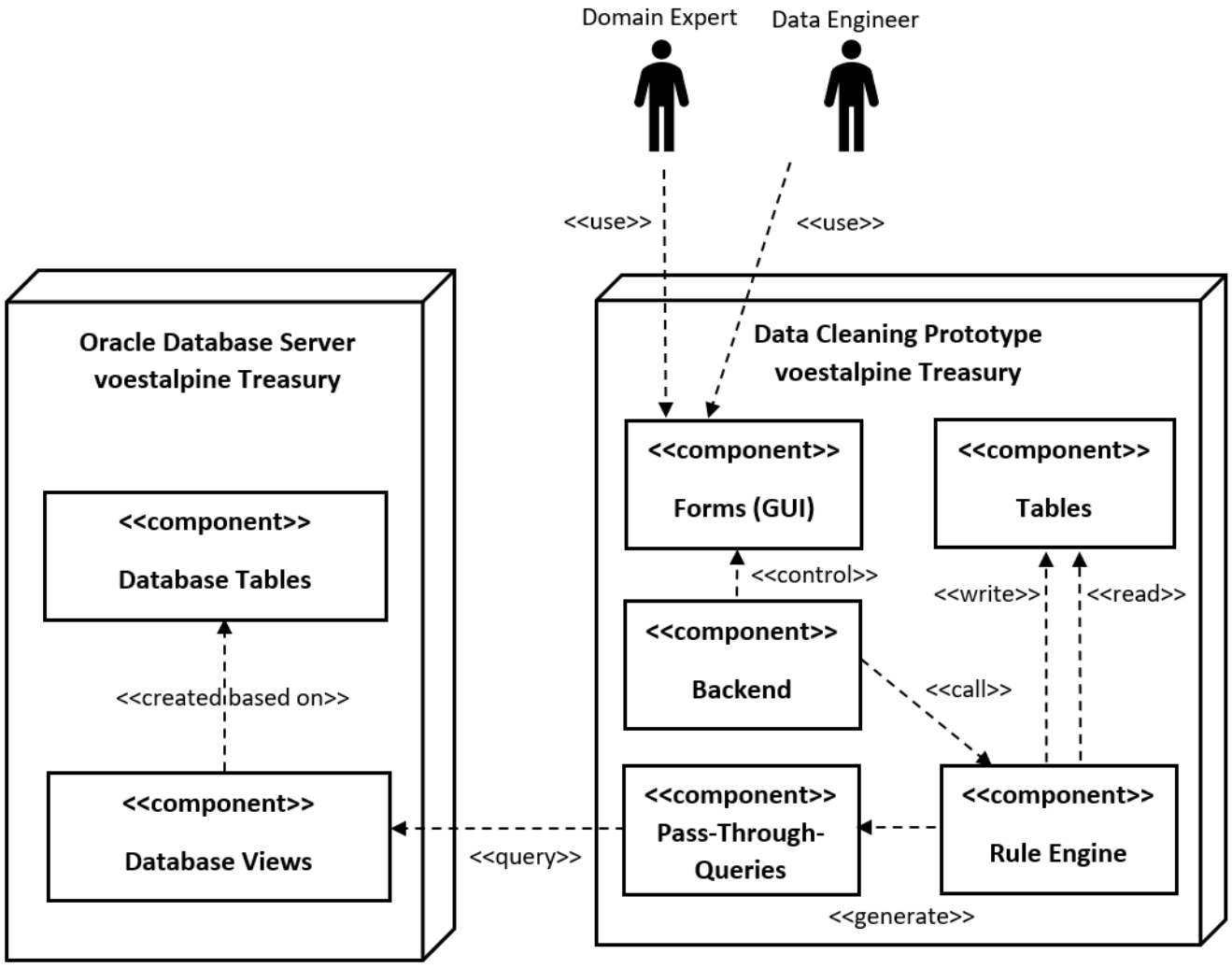
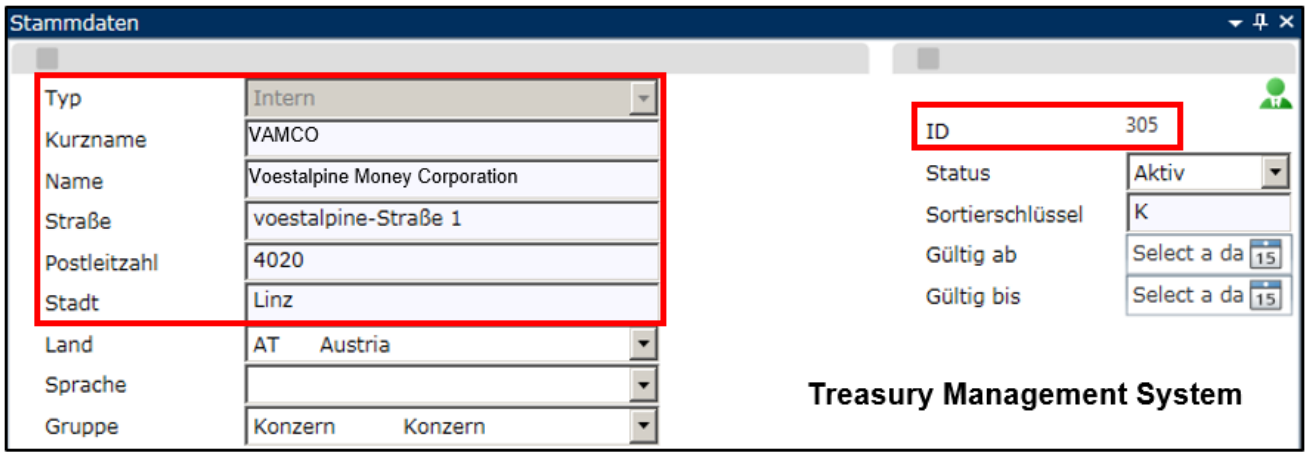


Abbildung 31: Komponentendiagramm Oracle-Datenbank und Data-Cleaning-Prototyp

5.1 Views und Tabellen in der Oracle-Datenbank des voestalpine Treasury

Um die in Abbildung 30 dargestellte Funktionalität zu ermöglichen wurden verschiedene Vorkehrungen getroffen werden, welche die Basis für den Einsatz des Data-Cleaning-Prototyps bilden. Seitens der Treasury-Datenbank wurden die zu verwendenden Views angelegt werden, welche bei der Erstellung der Geschäftsregeln im Data-Cleaning-Prototyp hinterlegt werden können. Werden somit neue Geschäftsregeln hinterlegt, muss entweder durch den Autor oder durch facheinschlägig ausgebildete Personen des voestalpine Treasury untersucht werden, ob eine neue View in der Treasury-Datenbank angelegt werden muss. Die erstellten Views beziehen sich, wie in Abbildung 31 dargestellt, auf die Datenbanktabellen, welche die Produktivdaten des Treasury-Management-Systems und somit den zentralen Datenbestand des voestalpine Treasury, beinhalten. Die Definitionen der Views werden direkt als Datenbankobjekt in der Treasury-Datenbank angelegt, und werden bei der Durchführung der Überprüfung vom Data-Cleaning-Prototyp verwendet. Um den in Abbildung 12 dargestellten Schritt der Supervision durch die Sachbearbeiter bestmöglich zu unterstützen, wurde die Designentscheidung bezüglich der erstellten Views getroffen, dass jede View die Inhalte einer Oberfläche im Treasury-Management-System abbildet. Da die Sachbearbeiter ihr tägliches Geschäft im Treasury-Management-System abwickeln, sind diese bestens mit den verschiedenen Oberflächen vertraut, und können dadurch Änderungen am Datenbestand rasch vornehmen. In Abbildung 32 wird ein Beispiel für den erwähnten Sachverhalt präsentiert. Der obere Teil der Abbildung stellt einen Partner-Eintrag aus dem

Bereich „Master Data Management“ dar. Hier werden grundlegende Stammdaten wie beispielsweise `Typ`, `Kurzname` oder `Name`, in Abbildung 32 rot markiert, zum Partner abgelegt. Im unteren Teil der Abbildung 32 ist das Gegenstück in Form einer Datenbankview abgebildet. Diese stellt die Daten aus der Oberfläche, bzw. der Datenbanktabellen im Hintergrund, für die Weiterverarbeitung durch den Data-Cleaning-Prototyp zur Verfügung.



Ausschnitt View

Rec_ID	Partner_Type	Short_name	Name	Street	City	Postal_Code	...
305	Intern	VAMCO	Voestalpine Money Corporation	Voestalpine-Straße 1	Linz	4020	
			...				

Abbildung 32: Daten aus dem Treasury-Management-System und der zugehörigen View

5.2 Tabellen des Data-Cleaning-Prototyps zur Speicherung der Metadaten

Wurden die Views in der Treasury-Datenbank erstellt, so können diese auf der grafischen Benutzeroberfläche des Data-Cleaning-Prototyps ausgewählt werden. Die grafische Benutzeroberfläche wurde mittels der in Microsoft Access verfügbaren Formulare umgesetzt und stellt ebenfalls eine wichtige Komponente des Data-Cleaning-Prototyps dar. Wie bereits in Abbildung 2, im Abschnitt 2.2 erwähnt, benötigt eine Business-Rule-Engine eine Schnittstelle für die Eingabe von Informationen durch einen Benutzer. Diese Schnittstelle muss nicht in Form einer grafischen Benutzeroberfläche implementiert werden, jedoch erleichtert diese die Steuerung der Business-Rule-Engine signifikant. Im Rahmen dieser Masterarbeit wurde vom Autor entschieden, eine grafische Benutzeroberfläche für das Anstoßen der Überprüfung sowie das Anlegen und Löschen von Geschäftsregeln, Views, und Oberflächen zu erstellen. Um die Daten, welche über die grafische Benutzeroberfläche eingegeben werden, ordnungsgemäß ablegen zu können wurden im Access-Modul mehrere Datenbanktabellen angelegt. In Abbildung 33 wird das Datenbankschema im Access-Modul dargestellt.

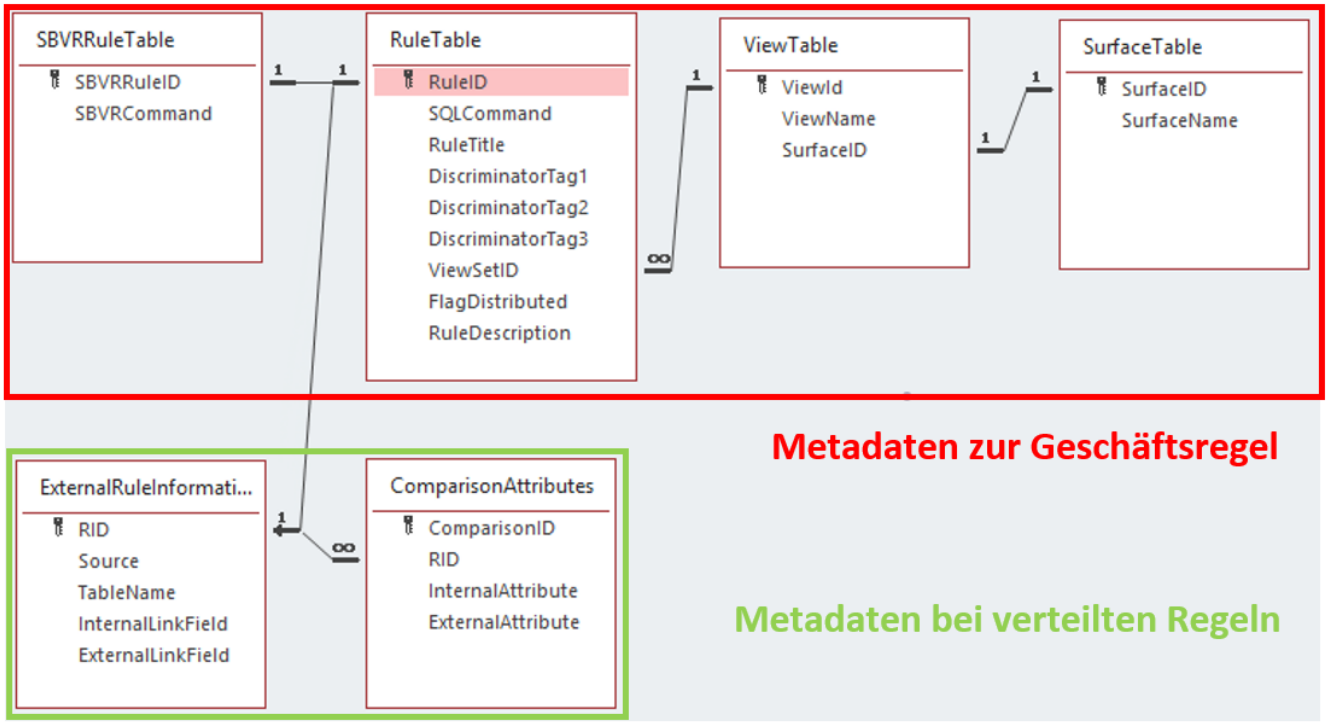


Abbildung 33: Datenbanktabellen im Access-Modul

In der Tabelle `RuleTable` werden alle grundlegenden Informationen zur Geschäftsregel abgelegt. Dabei wird jede Regel durch eine eindeutige `RuleID` identifiziert. Die Spalte `RuleID` ist technisch durch einen Integer-Wert realisiert, folglich kann eine `RuleID` durch einen Zahlenwert wie beispielsweise „321“ repräsentiert werden. Die Spalte `SQLCommand` beinhaltet die in Kapitel 4 beschriebene SQL-Abfrage der Geschäftsregel und bestimmt somit die Abfrageanweisung, welche bei der Überprüfung an die Treasury-Datenbank gesendet wird. `RuleTitle` stellt den Titel der Geschäftsregel dar. Dabei ist es wichtig einen sprechenden Namen, wie beispielsweise „Fehlender Partner“, im Titel der Regel anzugeben, damit bei der Bearbeitung durch die Sachbearbeiter keine Missverständnisse auftreten. Die Spalte `RuleDescription` stellt ebenfalls zusätzliche Informationen für die Sachbearbeiter zur Verfügung. In der `RuleDescription` soll ein kurzer Text zur Beschreibung der Regel, wie beispielsweise „Regel überprüft ob bei jedem Partner eine `ConcernID` hinterlegt wurde“, verfasst werden. Die Spalten `DiscriminatorTag 1-3` stellen die notwendigen Komplementärinformationen zu den in Kapitel 4 vorgestellten Discriminator Items dar. Werden wie in Kapitel 4 beschrieben beispielsweise die Datenbankspalten `RecID`, `short_name` und `name` als Discriminator Items festgelegt, so muss sichergestellt werden, dass ein Sachbearbeiter im Data-Cleaning-Report nicht den technischen Namen der Datenbankspalten vorfindet. Es muss der technische Spaltenname, wie beispielsweise `short_name`, in brauchbare Informationen für den Sachbearbeiter umgewandelt werden, da in diesem Punkt nicht vorausgesetzt werden kann, dass ein Sachbearbeiter über Kenntnisse des Datenbankschemas der Treasury-Datenbank verfügt. Folglich werden bei der Erstellung der Geschäftsregel im Data-Cleaning-Prototyp zusätzliche Meta-Informationen in Form der Discriminator Tags hinzugefügt, um die Weiterverarbeitung im Data-Cleaning-Report zu erleichtern. Sollte sich die Anzahl der Discriminator Items ändern, so muss lediglich weitere Spalten in der `RuleTable` angelegt werden. Dieser Vorgang ist im Rahmen dieser Masterarbeit derzeit nicht automatisiert worden, wurde aber als mögliche Erweiterung beim Verlassen des Prototyp-Stadiums des Data-Cleaning-Softwaresystems vorgemerkt.

Über die RuleID kann ebenfalls die Verbindung zur SBVRRuleTable hergestellt werden. Diese beinhaltet die SBVR-Geschäftsregel für jeden im Data-Cleaning-Prototyp angelegte Geschäftsregel. Dies ist vor allem nützlich, um nachvollziehen zu können, welche SBVR-Geschäftsregel hinter der im Data-Cleaning-Prototyp angelegten Regel steckt. Folglich kann auch in diesem Punkt die Verbindung über die RuleID hergestellt werden. Zusätzlich wurde vom Autor für eine weitere mögliche Erweiterung das Einbinden der SBVR-Geschäftsregel in den Data-Cleaning-Report vorgesehen. Es besteht die Möglichkeit, die SBVR-Geschäftsregeln mit dem dazugehörigen Mockup, wie beispielsweise Farbtöne und Unterstreichungen (siehe Abschnitt 2.3.1), im Access-Modul abzulegen und diese Informationen im Data-Cleaning-Report den Sachbearbeitern zur Verfügung zu stellen.

Die Spalte ViewSetID ermöglicht die Verbindung zur Tabelle ViewTable sowie SurfaceTable. Die Informationen in der ViewTable des Access-Moduls stellen lediglich zusätzliche Meta-Informationen für die Geschäftsregel dar und dürfen nicht mit den View-Datenbankobjekten in der Treasury-Datenbank verwechselt werden. Im Rahmen dieser Masterarbeit wurden bezüglich der Beziehung zwischen Geschäftsregel, verwendeter View und zugehöriger Oberfläche (engl. Surface) notwendige Designentscheidungen getroffen. Wie in Abbildung 33 dargestellt, können mehrere Geschäftsregeln derselben View zugeordnet werden. Jedoch repräsentiert jede View genau eine Oberfläche aus dem Treasury-Management-System. Die Spalte ViewSetID stellt somit den Fremdschlüssel in der Tabelle RuleTable dar, um die für die Geschäftsregel verwendete View zu referenzieren. In der Tabelle ViewTable wurde eine eindeutige Identifikation der View durch die Spalte ViewID umgesetzt. Zusätzlich kann der Name der View hinterlegt werden. Letztlich kann durch den Fremdschlüssel SurfaceID zur zugehörigen Oberfläche navigiert werden. In der Tabelle SurfaceTable können die Oberflächen mit dem dazugehörigen Namen und der eindeutigen Identifikation angelegt werden.

Die Spalte FlagDistributed gibt an, ob die Geschäftsregeln Informationen aus einer externen Datenquelle bezieht und somit unter die Bezeichnung „verteilte Geschäftsregel“ fällt. Der Begriff verteilte Geschäftsregel wurde im Rahmen dieser Masterarbeit für Regeln festgelegt, welche zusätzliche Informationen, neben den Daten der Treasury-Datenbank verwenden, um komplexere Überprüfungen zu ermöglichen. Beispielsweise kann ein Abgleich von Werten der eigenen Datenbank, mit Werten einer externen Datenquelle, wie der in Abschnitt 3.4 vorgestellten Insighter-Datenbank, vorgenommen werden, um die Richtigkeit der eigenen Datensätze zu verifizieren. Handelt es sich um eine verteilte Regel, so werden in der grafischen Benutzeroberfläche zusätzliche Informationen eingegeben, welche in den in Abbildung 33 grün markierten Tabellen abgelegt werden, und spezielle Metadaten für diese Art von Regeln darstellen.

In der Tabelle ExternalRuleInformation werden die Einträge ebenfalls über die RID identifiziert, welche mit dem Wert der RuleID aus der Tabelle RuleTable übereinstimmt. Die Spalte Source beinhaltet den Namen der zu verwendenden externen Datenquelle, wie beispielsweise „Insighter“. Die Spalte TableName beinhaltet den Namen der Tabelle, welche im Rahmen der Überprüfung in der externen Datenquelle abgefragt wird. Die Spalten InternalLinkField und ExternalLinkField beinhalten die notwendigen Informationen, um eine Verbindung zwischen den Daten der Treasury-Datenbank und den Daten der externen Datenquelle herzustellen. Die beiden Spalten beinhalten folglich die notwendigen Spaltennamen der Treasury-Datenbank, respektive der externen Datenquelle, um einen Verbund zwischen den beiden Datenquellen zu simulieren und somit die Daten beider Datenquellen miteinander vergleichen zu können. Wurde die Verbindung zwischen den internen und den externen Daten erfolgreich hergestellt, so können mehrere Werte verglichen werden. In der Tabelle ComparisonAttributes existieren die Spalten InternalAttribute sowie ExternalAttribute. In den erwähnten Spalten werden die zu vergleichenden Attribute aus der

Treasury-Datenbank, sowie der externen Datenquelle, abgelegt. Auch in der Tabelle `ComparisonAttributes` werden die Metadaten über die `RuleID` eindeutig identifiziert. Abschließend kann festgehalten werden, dass wenn die Spalte `FlagDistributed` in der `RuleTable` gesetzt wird, zusätzliche Meta-Informationen zur Geschäftsregel in den Tabellen `ExternalRuleInformation` und `ComparisonAttributes` abgelegt werden, um die Kommunikation und den Vergleich der Daten mit externen Datenquellen zu ermöglichen. Eine genauere Beschreibung des Prozesses bei der Durchführung von verteilten Geschäftsregeln sowie die Informationen zur Implementierung folgen in einem späteren Abschnitt.

5.3 Grafische Benutzeroberfläche

Die besprochenen Datenbanktabellen werden für die Speicherung der notwendigen Metadaten verwendet. Diese Metadaten werden bei der Erstellung der Geschäftsregeln im Data-Cleaning-Prototyp vom Benutzer eingegeben. In diesem Schritt wird unter der Bezeichnung „Benutzer“ kein Sachbearbeiter des voestalpine Treasury verstanden, sondern entweder der Autor oder eine facheinschlägig ausgebildete Person, welche die Erstellung der Geschäftsregeln übernimmt. Um die verwendete View sowie die zugehörige Oberfläche bei der Erstellung der Geschäftsregeln hinterlegen zu können, müssen diese beiden Objekte vorab im Data-Cleaning-Prototyp angelegt werden. Dazu navigiert der Benutzer wie in Abbildung 34 dargestellt, im Data-Cleaning-Prototyp auf den Reiter Surfaces. Dort werden in einer Tabelle alle derzeit angelegten Oberflächen angezeigt. Zusätzlich ist das Anlegen neuer Oberflächen unter „Add a Surface“ möglich. Dazu muss lediglich ein Name vom Benutzer eingegeben werden, die Vergabe eindeutiger Identifikationen wird vom System im Hintergrund erledigt.

Klickt der Benutzer auf die Schaltfläche „Add Surface“, so wird die Oberfläche im System erstellt und die Ansicht erneuert. Anschließend ist die neu erstellte Oberfläche ebenfalls in der Tabelle ersichtlich. Der nächste Schritt ist das Anlegen einer View. Wie im oberen Teil erwähnt, stellt eine View immer das Gegenstück zu einer Oberfläche dar. Daher muss bei der Erstellung einer View die zugehörige Oberfläche angegeben werden. Mit der Eingabe eines Namens für die View und dem Betätigen der Schaltfläche „Register View“, erstellt der Benutzer die View im System. In Abbildung 35 ist die Ansicht im Data-Cleaning-Prototyp zur Erstellung der Views abgebildet. Im oberen Teil wurde bereits erwähnt, dass die Views als Datenbankobjekte in der Treasury-Datenbank angelegt werden. Anschließend werden die Views durch den gerade beschriebenen Vorgang im Data-Cleaning-Prototyp registriert. Durch diese Registration wird der Eintrag im Data-Cleaning-Prototyp erstellt. Um zu vermeiden, dass Views in der Datenbank angelegt, jedoch nicht registriert wurden, oder Views im Data-Cleaning-Prototyp registriert werden, welche in der Datenbank noch nicht angelegt wurden, wurde ein Kontrollsystem, ersichtlich um unteren Teil der Abbildung 35, angelegt.

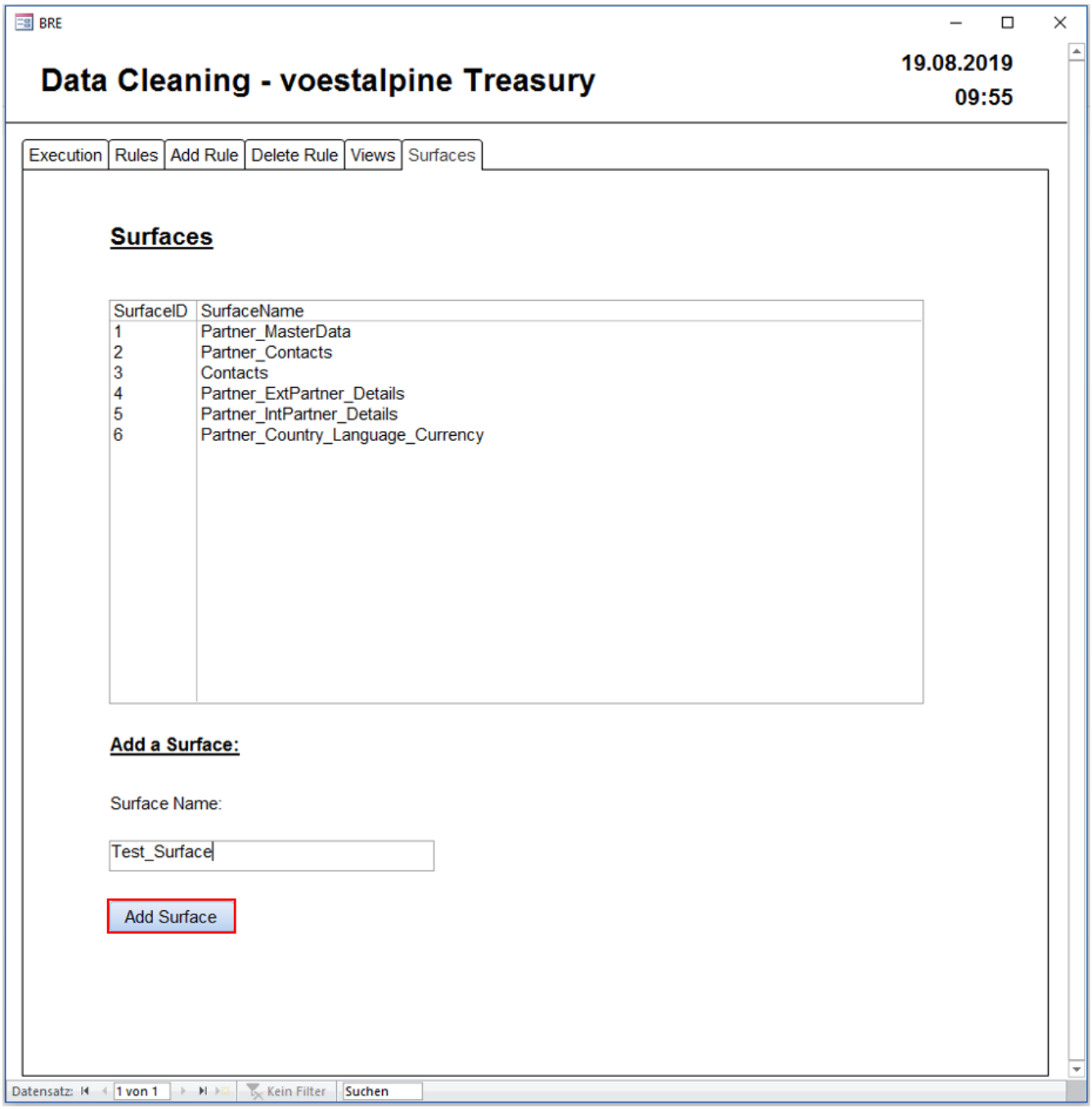


Abbildung 34: Data-Cleaning-Prototyp Ansicht Oberfläche anlegen

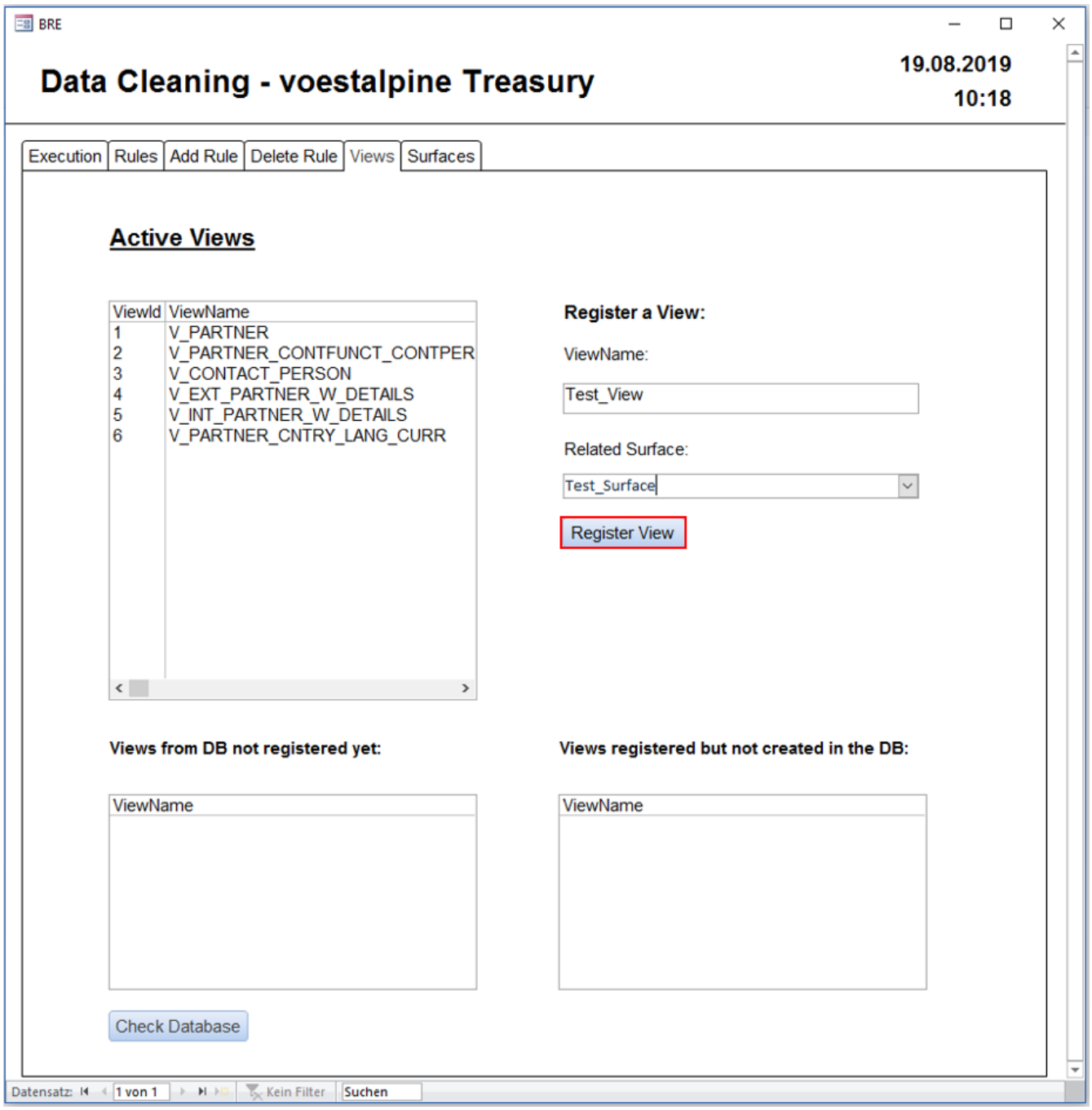


Abbildung 35: Data-Cleaning-Prototyp Ansicht View anlegen

Wird nun wie in Abbildung 35 gezeigt, eine View mit dem Namen `Test_View` angelegt, welche in der Datenbank nicht erstellt wurde, so erscheint nach Betätigung der „Check Database“ Schaltfläche die in Abbildung 36 dargestellte Ansicht. Im unteren Teil der Ansicht wird die `Test_View` in der rechten Box hinzugefügt. Die rechte Box beinhaltet Views, welche im Data-Cleaning-Prototyp erstellt wurden, wo jedoch kein dazugehöriges Datenbankobjekt in der Treasury-Datenbank existiert. In der linken Box werden alle Views, welche bereits als Datenbankobjekte in der Treasury-Datenbank erstellt wurden, jedoch nicht im Data-Cleaning-Prototyp registriert wurden, angezeigt.

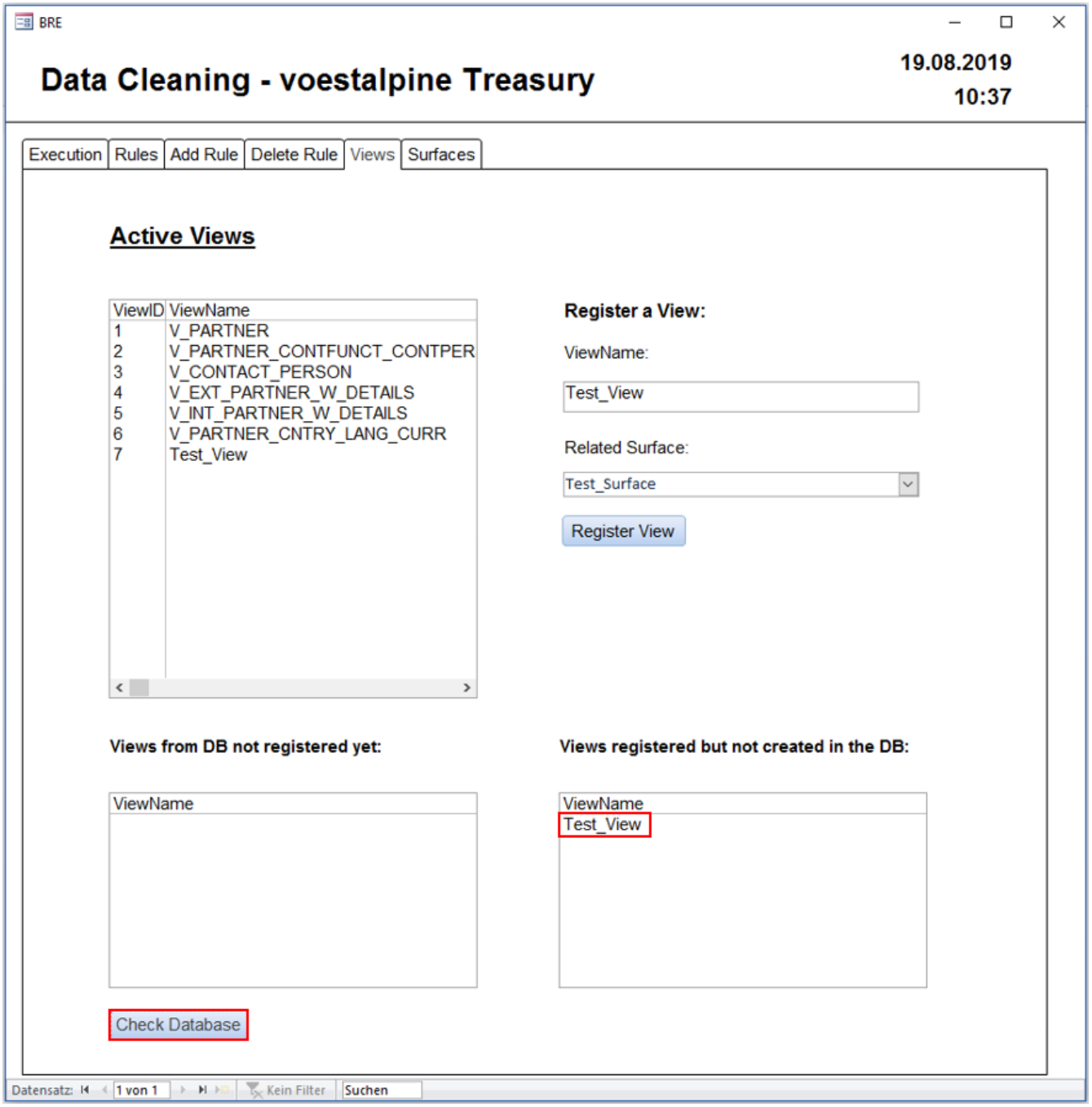


Abbildung 36: Data-Cleaning-Prototyp Abgleich Views

Wurden die Oberflächen sowie die Views erstellt, so ist der nächste Schritt das Erstellen der Geschäftsregel. Um diese Aufgabe zu erledigen navigiert der Benutzer zum Reiter „Add Rule“. Geschäftsregeln können, wie in Abbildung 38 dargestellt, im Data-Cleaning-Prototyp auf zwei verschiedene Arten angelegt werden. Entweder die Informationen werden direkt in der Oberfläche eingegeben, oder es wird eine Textdatei importiert, welche die Geschäftsregeln in strukturierter Form enthält. Beim Import einer Textdatei müssen die einzelnen Felder der Geschäftsregeln aus Abbildung 33, wie beispielsweise der `SQLCommand`, mit „|“ getrennt angegeben werden. Es ergibt sich die in Abbildung 37 dargestellte Struktur, welche beim Import eingehalten werden muss.

SQLCommand | RuleTitle | DiscriminatorTag1 | DiscriminatorTag2 | DiscriminatorTag3 | ViewSetID | ...

FlagDistributed | RuleDescription | SBVRCommand | Source | TableName | InternalLinkField | ...

ExternalLinkField | InternalAttribute1, ExternalAttribute1;[InternalAttributeN, ExternalAttributeN;]

Abbildung 37: Struktur der Geschäftsregeln beim Fileimport

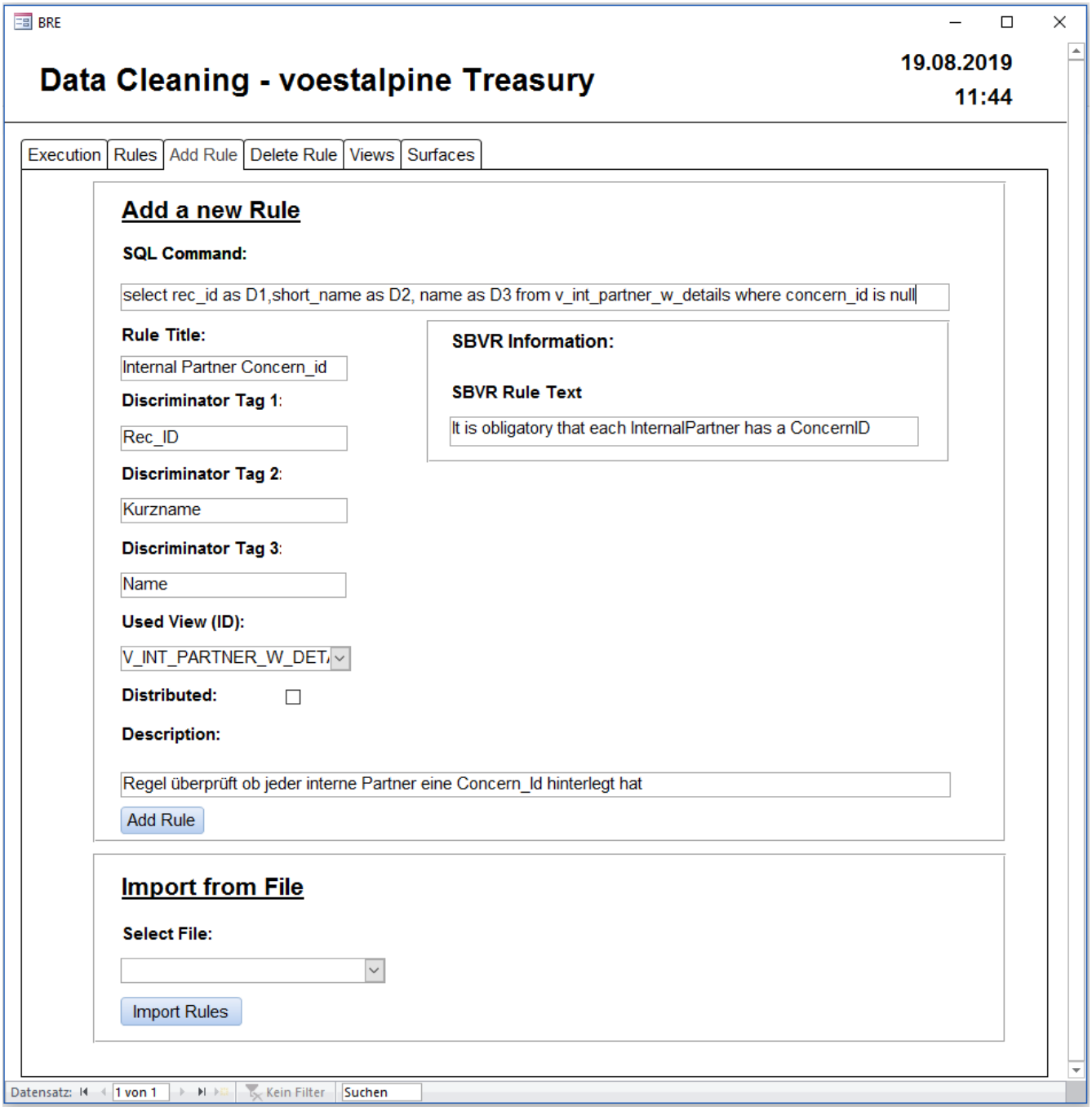


Abbildung 38: Data-Cleaning-Prototyp Ansicht Geschäftsregel anlegen

Handelt es sich bei der Geschäftsregel um eine verteilte Regel, so müssen die rot markierten zusätzlichen Informationen eingegeben werden. Alle Bestandteile des Fileimports sind mit den Datenbanktabellen in Abbildung 33 abgestimmt. Sollten in einer Geschäftsregel mehrere Attribute der

internen Datenbank, mit Attributen der externen Datenquelle verglichen werden, so können diese optional, wie in Abbildung 37 dargestellt, im Fileimport angegeben werden.

Entscheidet sich ein Benutzer, die Informationen für eine Geschäftsregel direkt an der Oberfläche einzugeben, muss dies im oberen Teil der in Abbildung 38 dargestellten Ansicht durchgeführt werden. Der Sachverhalt wird an dem bereits in Kapitel 4 Gezeigten Beispiel weitergeführt. Der `SQLCommand` stellt die mehrmals erwähnte Anweisung an die Datenbank in Form einer SQL-Abfrage dar. Diese SQL-Abfrage repräsentiert eine zuvor in SBVR erstellte Geschäftsregel. Der Text der erstellten SBVR-Geschäftsregel wird vom Benutzer im Feld `SBVR Rule Text` hinterlegt, und ermöglicht somit die Verbindung zwischen der Geschäftsregel im SBVR-Format und der Geschäftsregel in Form der SQL-Abfrage. Zusätzlich müssen der Regeltitel im Feld `RuleTitle`, die Discriminator Tags in deren zugeordneten Feldern und die Regelbeschreibung im Feld `Description` hinterlegt werden. Des Weiteren muss die verwendete View über das Drop-Down Feld `UsedView` ausgewählt werden. Die Checkbox `Distributed` gibt an, ob es sich um eine verteilte Geschäftsregel handelt. Ist diese mit einem Haken ausgefüllt, so wird eine verteilte Geschäftsregel angelegt. Im Falle einer nicht-verteilten Geschäftsregel ist der Anlageprozess mit dem abschließenden Klick auf die Schaltfläche `Add Rule` beendet und die Regel wird bei der Nächsten Ausführung der Überprüfungen berücksichtigt.

Handelt es sich um eine verteilte Regel, so werden mit dem Bestätigen der Checkbox weitere Eingabefelder, dargestellt in Abbildung 39, sichtbar. Im Feld `SourceName` kann der Benutzer eine der vordefinierten Datenquellen auswählen und somit festlegen, dass die Geschäftsregel aus dieser externen Datenquelle Informationen bezieht. Des Weiteren muss der Tabellenname angegeben werden, aus dem die Daten bezogen werden. Hierbei wird der Benutzer im Drop-Down Feld unterstützt, indem alle möglichen Tabellen der externen Datenquelle vorab abgefragt und eingefügt werden. Die `LinkFields` stellen die erwähnten Spalten dar, über die eine Verbindung mit den Daten der externen Datenquelle hergestellt werden kann. Hier wird der Benutzer ebenfalls mit vorgefertigten Eingabemöglichkeiten unterstützt. Abschließend muss der Benutzer die zu vergleichenden Spalten angeben und die Spaltenpaare per Klick auf die Schaltfläche `Add` der Liste hinzufügen. Mit einem abschließenden Klick auf die Schaltfläche `Add Rule` wird die Geschäftsregel abgelegt und bei der nächsten Durchführung der Überprüfungen berücksichtigt. Unter dem Reiter „Delete Rule“ findet der Benutzer die Möglichkeit eine oder mehrere erstelle Geschäftsregeln zu löschen. Unter dem Reiter „Rules“ können alle aktuell erstellten Geschäftsregeln mit allen zugehörigen Informationen eingesehen werden.

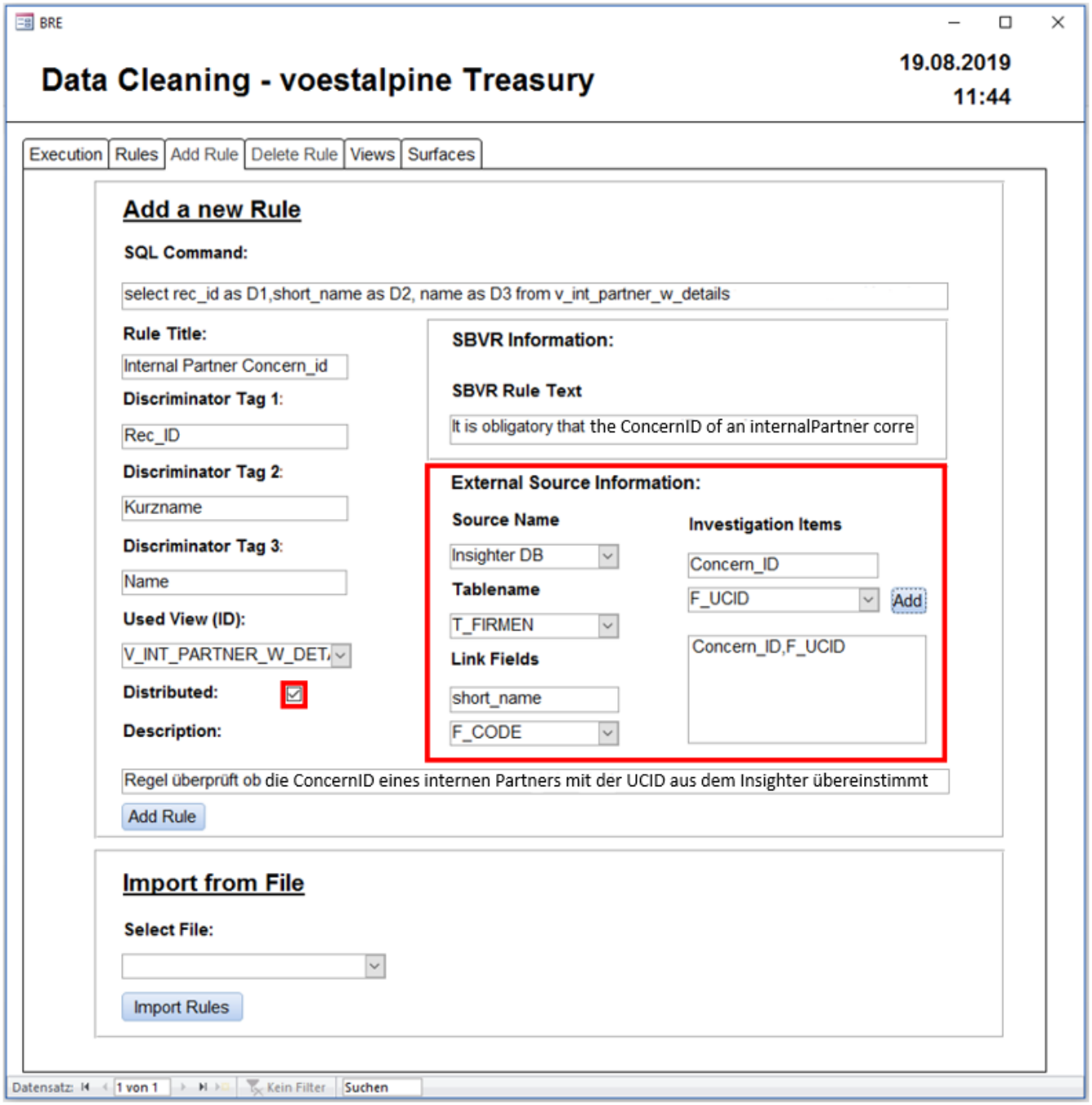


Abbildung 39: Data-Cleaning-Prototyp Ansicht verteilte Geschäftsregel erstellen

Um nach der Erstellung der Geschäftsregeln eine Überprüfung der Datenbestände anzustoßen, muss der Benutzer auf den Reiter „Execution“ wechseln. In der in Abbildung 40 dargestellten Ansicht erhält der Benutzer Informationen über die aktuellen Einstellungen des Data-Cleaning-Prototyps. Der `Scan Type` gibt Aufschluss darüber, ob die Datenbestände vollständig überprüft werden, oder ob nur gewisse Bereiche für eine Überprüfung vorgesehen sind. Im Rahmen dieser Masterarbeit wurde seitens des voestalpine Treasury lediglich eine vollständige Überprüfung vorgesehen. Eine Eingrenzung auf Teilbereiche wurden jedoch vom Autor als mögliche Erweiterung vorgemerkt. Es wird ebenfalls die Anzahl der aktiven Regeln in der in Abbildung 40 gezeigten Ansicht dargestellt. Es wurden mehrere Regeln ausgewählt, um im Rahmen dieser Masterarbeit als Beispielregeln die Funktionalität des Data-Cleaning-Prototyps zu demonstrieren. Im voestalpine wird nach Beendigung dieser Masterarbeit, der Ausbau der aktuellen Regelbasis kontinuierlich durchgeführt, um ein bestmögliches Ergebnis zu erzielen.

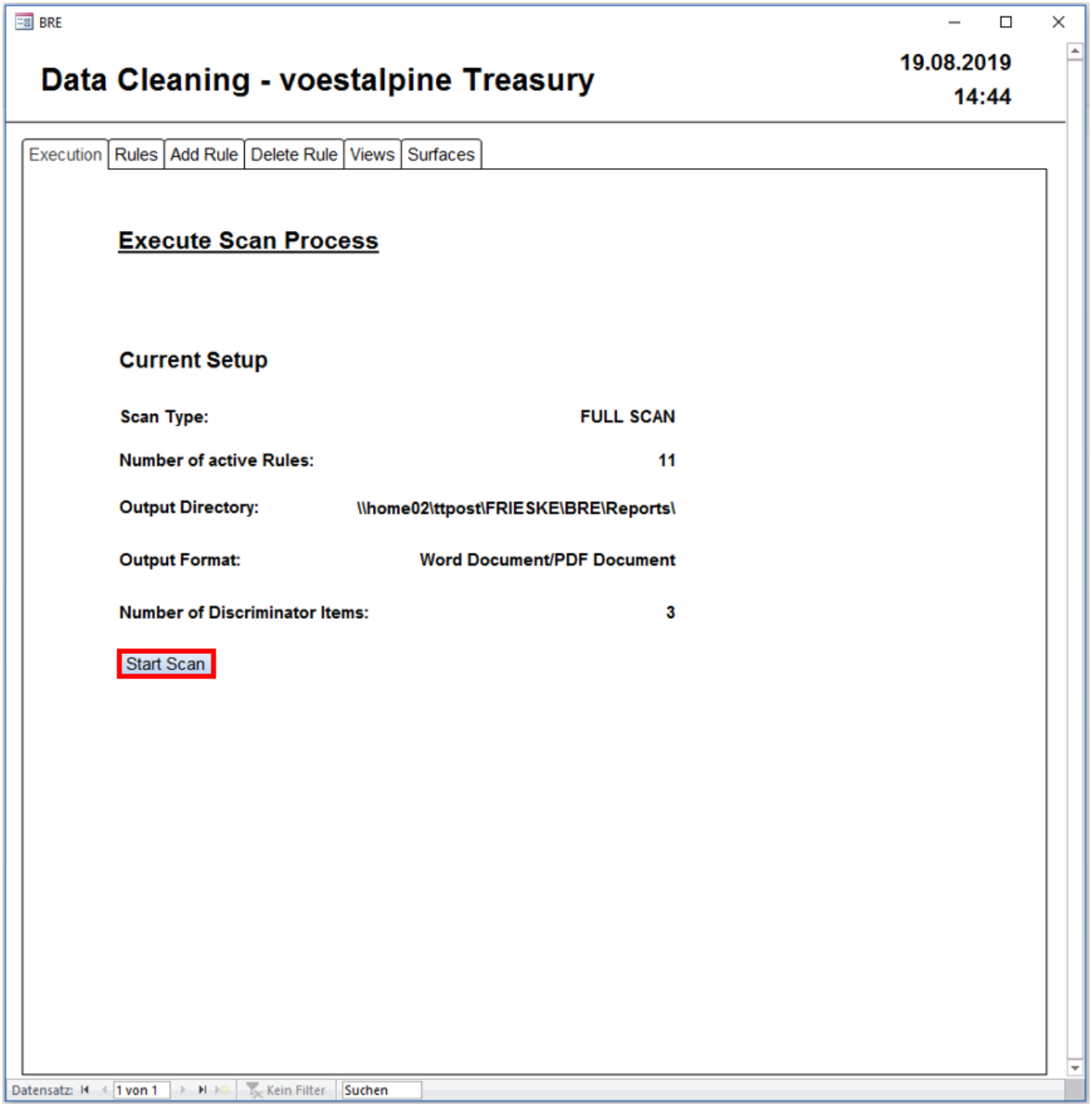


Abbildung 40: Data-Cleaning-Prototyp Ansicht Überprüfung starten

Der Dateipfad und das Ausgabeformat, in dem die Data-Cleaning-Reports nach der Erstellung durch den Prototyp abgelegt werden, ist ebenfalls in der Benutzeroberfläche ersichtlich. Den Anforderungen des voestalpine Treasury zufolge, wird derzeit jeder Data-Cleaning-Report sowohl als Word-Dokument als auch als PDF-Dokument abgespeichert. Zusätzlich wird die Anzahl der Discriminator Items in der Benutzeroberfläche angezeigt. Um den Überprüfungsprozess zu starten, muss der Benutzer die Schaltfläche `Start Scan`, in der in Abbildung 40 dargestellten Ansicht, betätigen. Wurde der Überprüfungsprozess erfolgreich durchgeführt, so erscheint eine Nachrichtenbox mit dem Inhalt „Scan erfolgreich durchgeführt!“ als Rückmeldung für den Benutzer.

5.4 Rule-Engine, Backend und Pass-Through-Abfragen

Die Funktionalität des Data-Cleaning-Prototyps wurde mittels Visual Basic für Applikationen (Abk. VBA), in Verbindung mit den in Abbildung 31 dargestellten Komponenten, umgesetzt. VBA ist eine von Microsoft für die Office-Anwendungen, wie beispielsweise Word oder Excel, konzipierte Sprache, welche standardmäßig mit der Verwendung letzterer verfügbar ist. Mittels VBA können zwar Klassen und Objekte abgebildet werden, jedoch wurde der Fokus auf die prozedurale Programmierung und den reibungslosen Einsatz in Verbindung mit den Office-Anwendungen, gelegt. Der Code-Editor lässt sich in Microsoft Access über den Reiter `Entwurf` und die Schaltfläche `Code anzeigen` aufrufen. Im Rahmen dieser Masterarbeit wurden zur Implementierung des Backends des Data-Cleaning-Prototyps Eventprozeduren, in Verbindung mit den Steuerelementen der grafischen Benutzeroberfläche, verwendet. Eventprozeduren (vgl. engl. „Event Handler“) ermöglichen das Reagieren auf gewisse Aktionen des Benutzers, wie beispielsweise einen Klick auf eine Schaltfläche. Mit dieser Methode konnte die Verarbeitung der eingegebenen Daten sowie das Anstoßen der Überprüfungen gesteuert werden. Zusätzlich wurde ein Code-Modul erstellt, welches die Funktionalität der Rule-Engine umsetzt. In Abbildung 41 ist die Beziehung zwischen der grafischen Benutzeroberfläche, den Eventprozeduren des Backends und dem Codemodul der Rule-Engine dargestellt.

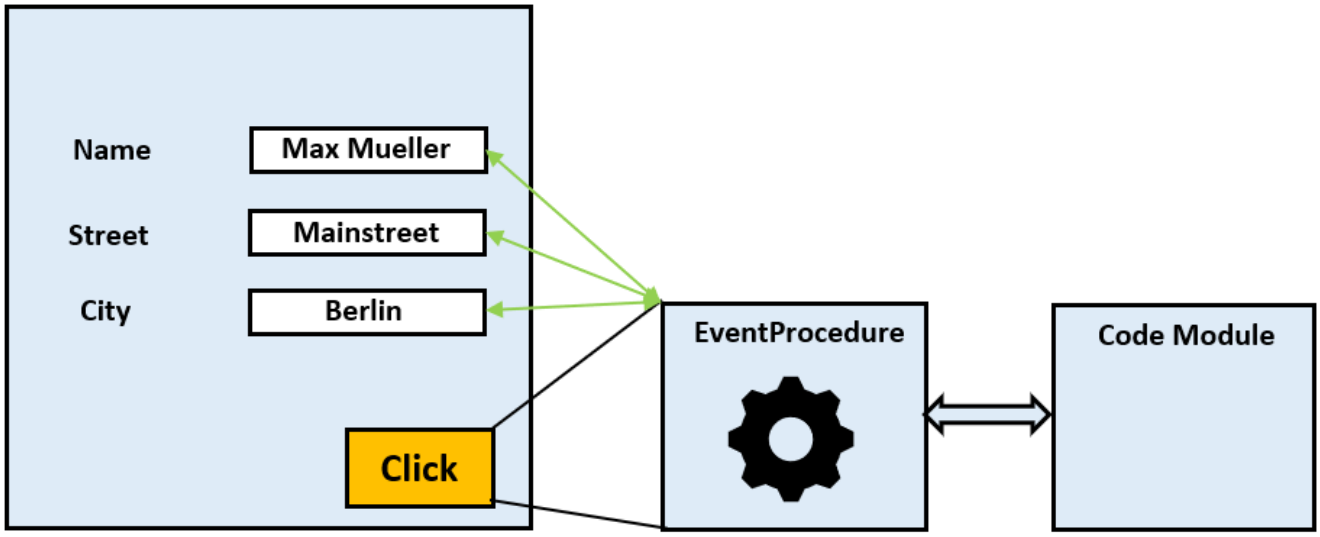


Abbildung 41: Beziehung grafische Oberfläche, Eventprozeduren des Backends und Code-Modul der Rule-Engine

Neben dem Quellcode, welcher die Funktionalität im Data-Cleaning-Prototyp ermöglicht, muss ebenfalls die Verbindung zur Datenbank des voestalpine Treasury sowie externen Datenquellen wie der Insighter-Datenbank sichergestellt werden. Im Rahmen dieser Masterarbeit wurden Pass-Through-Abfragen verwendet, um die Abfrage zur Datenbank zu übermitteln und das Ergebnis zurück zum Access-Modul zu geleiten. In diesem Szenario stellen die Pass-Through-Abfragen Transportkapseln dar, welche die Abfrageanweisung aufnehmen und an der Datenbank abliefern und anschließend das Abfrageergebnis aufnehmen und an das Access-Modul rückliefern. Die Pass-Through-Abfragen erhalten die Verbindungsinformationen durch das hinterlegen einer ODBC-Datenquelle (engl. Open Database Connectivity), wobei diese vorab am Computer eingerichtet werden muss. Um eine ODBC-Datenquelle einrichten zu können, kann über die Suchfunktion „ODBC-Datenquellen“ eingegeben werden, und die Applikation gestartet werden. Um die Verbindung mit einer Datenbank herstellen zu können, muss der zugehörige Treiber, wie beispielsweise der „Oracle-Treiber“, vorhanden sein. Nach der Eingabe eines Verbindungsnamens, des TNS-Service-Namens sowie einem and der Datenbank registrierten

Benutzernamen mit Passwort, ist die ODBC-Datenquelle vollständig eingerichtet. Folglich kann diese anschließend beim Erstellen einer Pass-Through-Abfrage im Access-Modul für die Verbindung ausgewählt werden. Wurde eine Pass-Through-Abfrage richtig angelegt, so kann diese im Quellcode verwendet werden. Die Verbindung kann bei Bedarf durch Anweisungen im Quellcode aufgerufen werden, um Daten von der Datenbank zu beziehen.

5.5 Prozess zur Überprüfung von verteilten Geschäftsregeln

In den vorherigen Unterkapiteln wurden die Komponenten des Data-Cleaning-Prototyps detailliert beschrieben. In diesem Kapitel wird auf den Abarbeitungsprozess bei verteilten Geschäftsregeln und auf dessen Unterschiede, im Vergleich zu den nicht-verteilten Geschäftsregeln, eingegangen. Am Beginn dieses Kapitels wurde zur Einleitung bereits der Ablauf der Überprüfungen durch den Data-Cleaning-Prototyp erwähnt. Abbildung 30 stellt die dabei notwendigen Schritte grafisch dar. Handelt es sich um verteilte Geschäftsregeln, so müssen zusätzliche Schritte, für das Kontaktieren der externen Datenquelle, durchgeführt werden. Diese zusätzlichen Anforderungen betreffen, wie in Abschnitt 5.3 erwähnt, bereits das Erstellen der Geschäftsregel, da weitere Informationen hinterlegt werden müssen. Um die Abweichungen zu nicht-verteilten Geschäftsregeln hervorheben zu können, wird ein Beispiel durchgeführt. Im voestalpine Treasury hat jeder interne Partner (Konzerngesellschaften) eine eindeutige `ConcernID` hinterlegt. Diese `ConcernID` ist eine vierstellige numerische Kennung. Es wurde nun beobachtet, dass bei einigen internen Partnern keine, oder eine falsche, `ConcernID` hinterlegt ist. Es ist jedoch bekannt, dass die Rechtsabteilung der voestalpine AG ebenfalls die `ConcernID`, unter dem Namen `UCID`, in ihrem System pflegt. Die Datenbestände der Rechtsabteilung stellen in diesem Szenario eine Referenzquelle, folglich eine Quelle bei der angenommen wird das ihre Informationen vollständig korrekt sind, dar. Aus diesem Grund wurde eine Regel angelegt, welche die `ConcernID` der internen Partner aus der Datenbank des voestalpine Treasury, mit den Informationen aus der Insighter-Datenbank vergleicht. Da bei verteilten Geschäftsregeln, im Gegensatz zu nicht-verteilten Geschäftsregeln, der Abgleich der Informationen nicht direkt in der SQL-Abfrage, welche an die Datenbank des voestalpine Treasury gesendet wird, durchgeführt werden kann, sondern mittels Programmlogik umgesetzt wurde, ändert sich die Struktur der SQL-Abfrage. Bei der Eingabe der SQL-Abfrage im Feld `SQL Command`, wie in Abbildung 38 dargestellt, muss keine Where-Bedingung angegeben werden. Es werden lediglich die zu überprüfenden Daten abgefragt, um diese dem Data-Cleaning-Prototyp zur Verfügung zu stellen. Der SQL-Command der verteilten Regel aus diesem Beispiel, ist Abbildung 42 dargestellt.

```
SELECT rec_id as D1, short_name as D2, name as D3 FROM v_int_partner_w_details;
```

Abbildung 42: SQL-Command einer verteilten Geschäftsregel

Die in Abbildung 42 gezeigte SQL-Abfrage stellt sicher, dass der Data-Cleaning-Prototyp die zu überprüfenden Datensätze erhält. In diesem Beispiel werden die internen Partner, welche in der View `v_int_partner_w_details` abgelegt sind, aus der Datenbank des voestalpine Treasury abgefragt und dem Data-Cleaning-Prototyp übergeben. Es ist wichtig zu erkennen, dass eine verteilte Geschäftsregel die Discriminator Items in derselben Art wie eine nicht-verteilte Geschäftsregel verwendet. Wie in Abbildung 39 dargestellt, ergeben sich ebenfalls keine Unterschiede beim Regeltitel (engl. `RuleTitle`), den Discriminator Tags, der verwendeten View, der Regelbeschreibung oder dem SBVR-Befehl. Lediglich im Bereich der roten Markierung in Abbildung 39 und in der SQL-Abfrage treten Unterschiede zu einer nicht-verteilten Geschäftsregel auf. Wurde die verteilte Geschäftsregel korrekt angelegt, so kann der Data-Cleaning-Prototyp die Überprüfung durchführen.

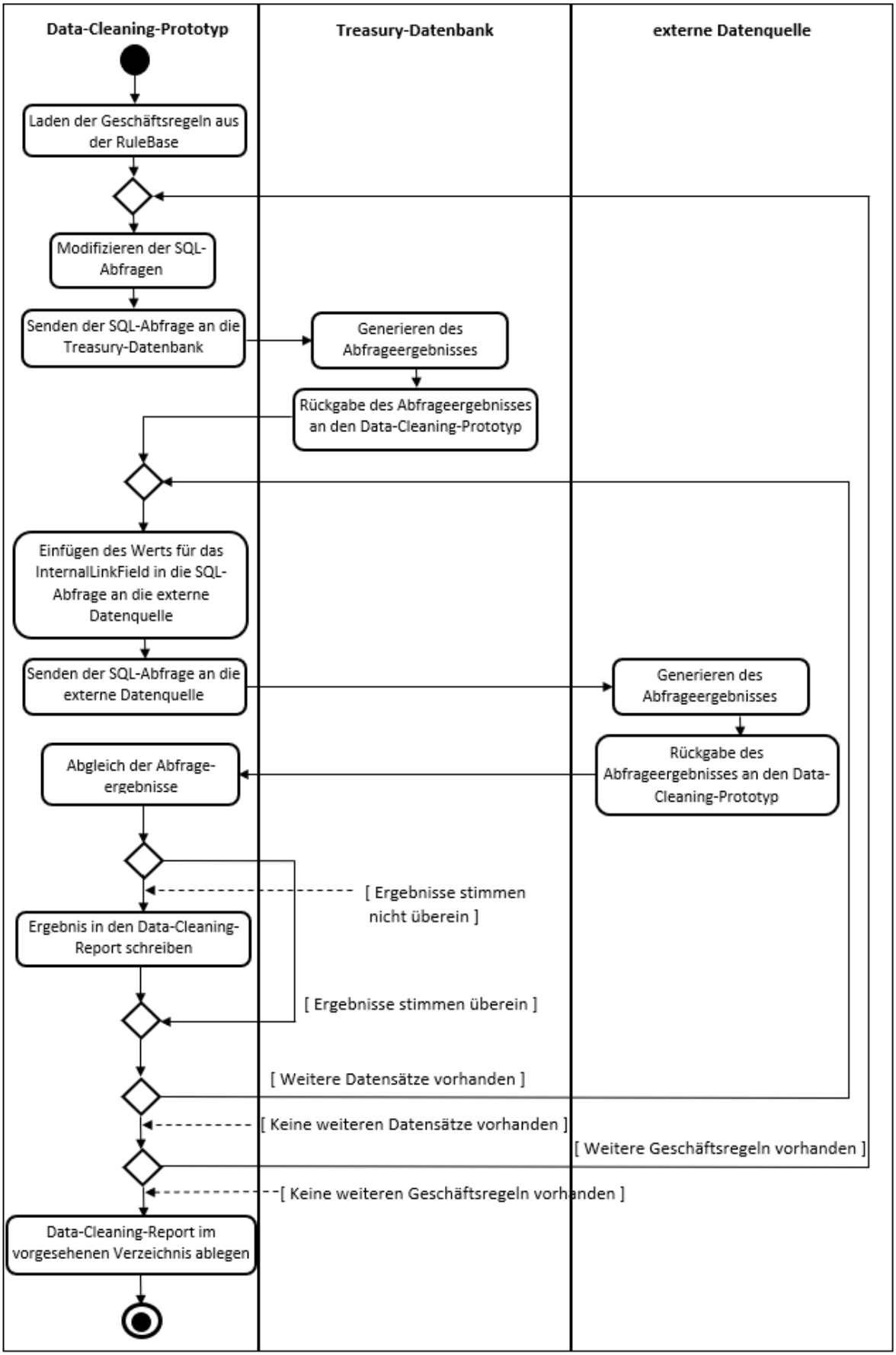


Abbildung 43: UML-Aktivitätsdiagramm – Ablauf bei verteilten Geschäftsregeln

Aus technischer Sicht mussten für die Überprüfung zusätzliche Funktionen implementiert werden, um verteilte Geschäftsregeln umsetzen zu können. Folglich hat sich der in Abbildung 30 dargestellte Ablauf verändert. In Abbildung 43 ist der Ablauf zur Überprüfung von verteilten Geschäftsregeln dargestellt.

Der Beginn des Ablaufs zur Überprüfung von verteilten Geschäftsregeln ist ähnlich zu dem in Abbildung 30 abgebildeten Prozess bei nicht-verteilten Geschäftsregeln, jedoch wird, im Gegenteil zu Abbildung 30, die SQL-Abfrage nicht direkt an die Datenbank weitergegeben. Vorab muss sowohl die SQL-Abfrage für die Treasury-Datenbank als auch die SQL-Abfrage für die externe Datenquelle präpariert werden, um die notwendigen Daten zurückzuliefern. Dieser Vorgang wird automatisch im Hintergrund, mit den bei der Erstellung der Geschäftsregel hinterlegten Informationen, durchgeführt. In Abbildung 44 ist der Ablauf der Modifikation der SQL-Abfragen in Form eines UML-Aktivitätsdiagramms dargestellt.

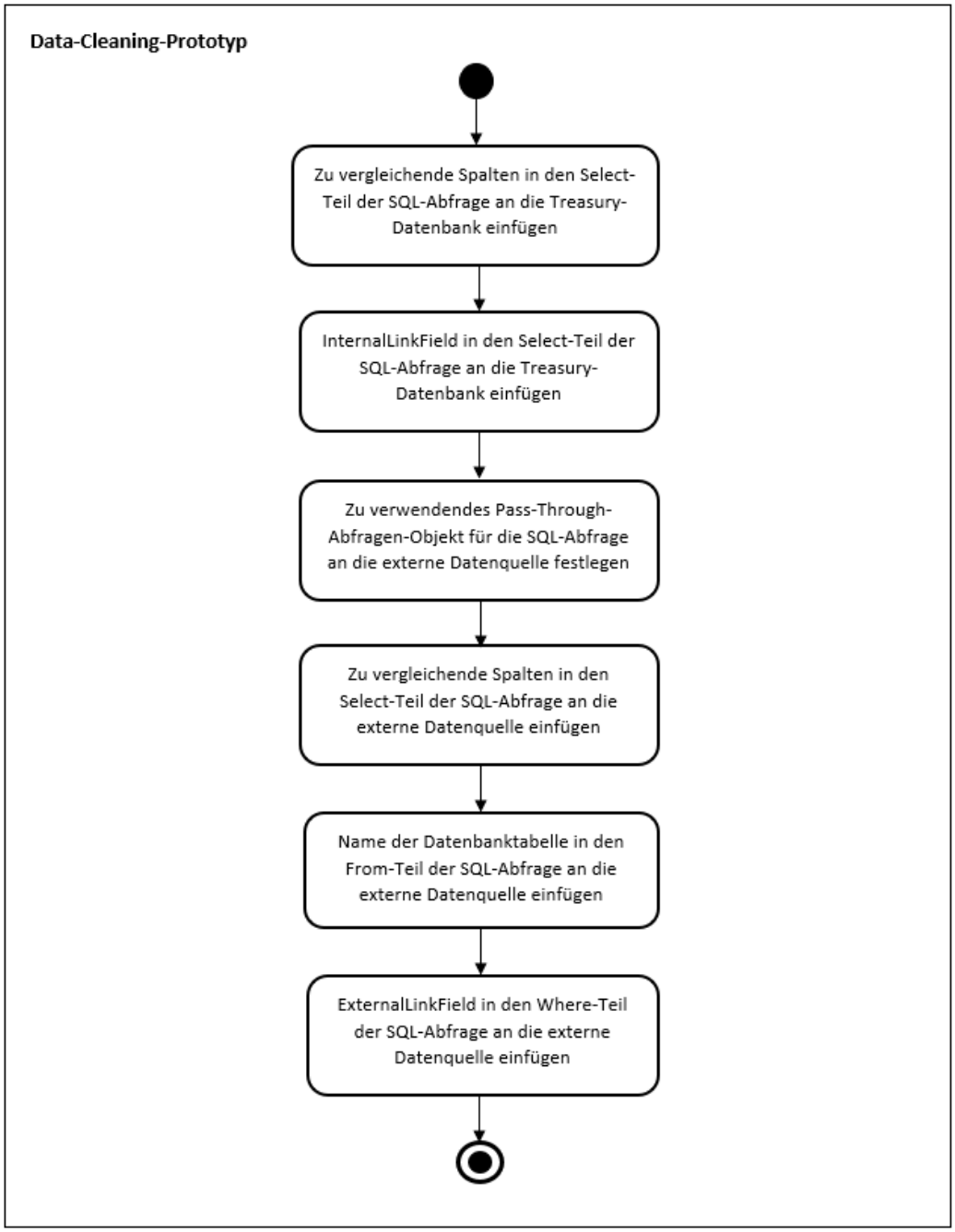


Abbildung 44: UML-Aktivitätsdiagramm - Modifikation der SQL-Abfragen

Die SQL-Abfrage, welche an die Datenbank des voestalpine Treasury gesendet wird (vgl. Abbildung 42) wird abgeändert, um im Abfrageergebnis die zu vergleichenden Spalten zu enthalten. Im angegebenen Beispiel soll die `ConcernID` aus der Datenbank des voestalpine Treasury, mit der `F_UCID` aus der Insighter-Datenbank verglichen werden (siehe Abbildung 39). Folglich muss die SQL-Abfrage, welche an die Datenbank des voestalpine Treasury gesendet wird, die Spalte `ConcernID` als Ergebnis rückliefern, um anschließend einen Abgleich im Data-Cleaning-Prototyp zu ermöglichen. Daher wird die Zeichenkette der SQL-Abfrage vom Data-Cleaning-Prototyp abgeändert und die Spalte `ConcernID` in den Select-Teil eingefügt. Nach demselben System muss das `InternalLinkField`, in unserem Beispiel der `short_name`, im Abfrageergebnis beinhaltet sein, um eine Verbindung mit den Daten der externen Datenquelle herstellen zu können. Da nicht angenommen werden kann, dass beide Datenquellen dasselbe oder ein ähnliches Schema besitzen, muss über Attribute, welche den gleichen Inhalt verkörpern, eine Verbindung hergestellt werden. Diese Attribute wurden vom Autor in den beiden Datenquellen identifiziert und bei der Erstellung der Geschäftsregeln als `InternalLinkField` sowie `ExternalLinkField`, wie in Abbildung 39 dargestellt, hinterlegt. Diese Verbindung übernimmt die Funktionalität eines Joins indem das `InternalLinkField` mit dem `ExternalLinkField` im Where-Teil der SQL-Abfrage der externen Datenquelle abgeglichen wird (vgl. Abbildung 45). In unserem Beispiel wurde identifiziert, dass sowohl die Datenbank des voestalpine Treasury als auch die Insighter-Datenbank, den Kurznamen (engl. `short_name`), gemäß den konzernweiten Vorgaben verwendet, und somit eine Verbindung hergestellt werden kann.

Nachdem die SQL-Abfrage, die an die Datenbank des voestalpine Treasury gesendet wird, mit den zu vergleichenden Spalten und der Spalte, welche die Verbindung ermöglicht, ausgestattet wurde, muss ebenfalls die SQL-Abfrage für die externe Datenquelle behandelt werden. Die SQL-Abfrage der externen Datenquelle wird ebenfalls automatisch im Hintergrund durch den Data-Cleaning-Prototyp vorbereitet. Dabei werden die bei der Erstellung der Geschäftsregel abgespeicherten Informationen verwendet. Als erster Schritt wird die zu verwendende Datenquelle, und damit das zugehörige Pass-Through-Abfragen-Objekt, welches die Verbindungsinformationen enthält, festgelegt.

```
SELECT F_UCID FROM T_FIRMEN WHERE F_CODE = '" & short_name & "';
```

Abbildung 45: Vorlage für die generierte SQL-Abfrage der externen Datenquelle

Anschließend wird die SQL-Abfrage vom Data-Cleaning-Prototyp zusammengesetzt, welche im Select-Teil die zu vergleichenden Attribute, in unserem Beispiel die `F_UCID` der Insighter-Datenbank, enthält. Im From-Teil wird der Name der Datenbanktabelle, aus der die Daten bezogen werden, in unserem Beispiel `T_FIRMEN`, eingefügt. Im Where-Teil muss das `ExternalLinkField`, welches als Gegenstück zum `InternalLinkField` die Verbindung ermöglicht, in unserem Beispiel `F_CODE`, hinzugefügt werden. Abbildung 45 stellt eine Vorlage für die zusammengesetzte SQL-Abfrage dar.

Beim Abgleich der Abfrageergebnisse kann somit über die in Abbildung 45 dargestellte SQL-Abfrage auf den zugehörigen Wert aus der externen Datenquelle zugegriffen werden. Zwischen den einzelnen Hochkommata in Abbildung 45 wird automatisch der Kurzname, über den die Verbindung hergestellt wird, eingefügt. Folglich stellt die Zeichenfolge zwischen den einzelnen Hochkommata den in VBA verwendeten Platzhalter dar. Werden mehrere Attribute in einer Geschäftsregel zum Vergleichen zusammengefasst, so würden diese Attribute sowohl im Select-Teil der Abfrage an die Datenbank des voestalpine Treasury als auch im Select-Teil der Abfrage für die externe Datenquelle aufscheinen. Beim Abgleich der Daten wird der interne Wert mit dem externen Wert verglichen. Entspricht somit der Eintrag

für die `ConcernID` aus der Datenbank des voestalpine Treasury nicht dem Wert für die `F_UCID` aus der externen Datenquelle, liegt ein potenzieller Fehler vor und der Datensatz wird, wie bei nicht-verteilten Geschäftsregeln, in den Data-Cleaning-Report geschrieben.

Im Rahmen der verteilten Geschäftsregeln werden nur Vergleiche von internen und externen Daten durchgeführt. Komplexere Operationen, wie der Abgleich von Beziehungen zwischen unterschiedlichen Datenbankobjekten, wurden derzeit nicht als Anforderung vom voestalpine Treasury vorgebracht. Ein Beispiel für eine komplexere Operation sieht wie folgt aus. Im voestalpine Treasury werden Konzerngesellschaften und deren Kontaktpersonen, inklusive der Rolle der Kontaktperson, abgespeichert. Dieses Konstrukt mittels einer externen Datenquelle zu überprüfen, gestaltet sich, aufgrund der verschiedenen Datenbankschemata, als schwierig. Folglich ist derzeit nicht vorgesehen Objekte wie Konzerngesellschaften oder Kontaktpersonen und deren Beziehungen untereinander mit externen Datenquellen abzugleichen. Diese Thematik wurde vom Autor ebenfalls als mögliche Erweiterung vorgesehen, sofern weitere externe Datenquellen mit den notwendigen zu vergleichenden Informationen bereitgestellt werden.

Im aktuellen Kapitel wurden die unterschiedlichen Komponenten des Data-Cleaning-Prototyps, sowie die Abläufe beim Überprüfen von nicht-verteilten und verteilten Geschäftsregeln behandelt. Wie in Abbildung 30 und Abbildung 43 dargestellt, ist der Data-Cleaning-Report das Endprodukt der Verarbeitung durch den Data-Cleaning-Prototyp. Im nachfolgenden Kapitel wird der Data-Cleaning-Report analysiert und relevante Aspekte hervorgehoben.

6 Data-Cleaning-Report

Der Data-Cleaning-Report stellt das Endprodukt dar, welches nach der Verarbeitung der Geschäftsregeln und den damit verbundenen Überprüfungen, vom Data-Cleaning-Prototyp generiert wird. Der Data-Cleaning-Report wird von den Sachbearbeitern des voestalpine Treasury als Entscheidungsgrundlage bezüglich der potenziellen Fehler im Datenbestand verwendet. Wird ein Datensatz im Data-Cleaning-Report aufgeführt, so hat dieser gegen mindestens eine Geschäftsregel verstoßen und muss folglich einer Überprüfung durch einen Sachbearbeiter unterzogen werden. Der Data-Cleaning-Report wird nach Abschluss der Überprüfungen durch den Data-Cleaning-Prototyp in einem vorab definierten Dateipfad abgelegt. Die Sachbearbeiter des voestalpine Treasury wurden mit den notwendigen Zugriffsrechten versehen und können somit jederzeit auf die generierten Dokumente zugreifen. Aus Gründen der Übersichtlichkeit erzeugt der Data-Cleaning-Prototyp eine Verzeichnisstruktur, welche die generierten Dokumente nach dem Datum der Erzeugung sortiert. In Abbildung 46 ist diese Dateistruktur grafisch dargestellt.

Wird ein neuer Data-Cleaning-Report erstellt, so wird überprüft, ob im Basisverzeichnis bereits ein Ordner mit dem Jahr existiert. Ist dies nicht der Fall, so wird automatisch ein Ordner mit dem aktuellen Jahr angelegt. Dasselbe wird für den Ordner auf Monatsebene durchgeführt. Die Monate werden mit den dazugehörigen Zahlen repräsentiert. Schließlich werden die Dokumente, unter Verwendung des aktuellen Datums im Format „DDMMYY“, im Monatsordner abgelegt. Das Basisverzeichnis kann jederzeit über die Konfiguration des Data-Cleaning-Prototyps geändert werden.

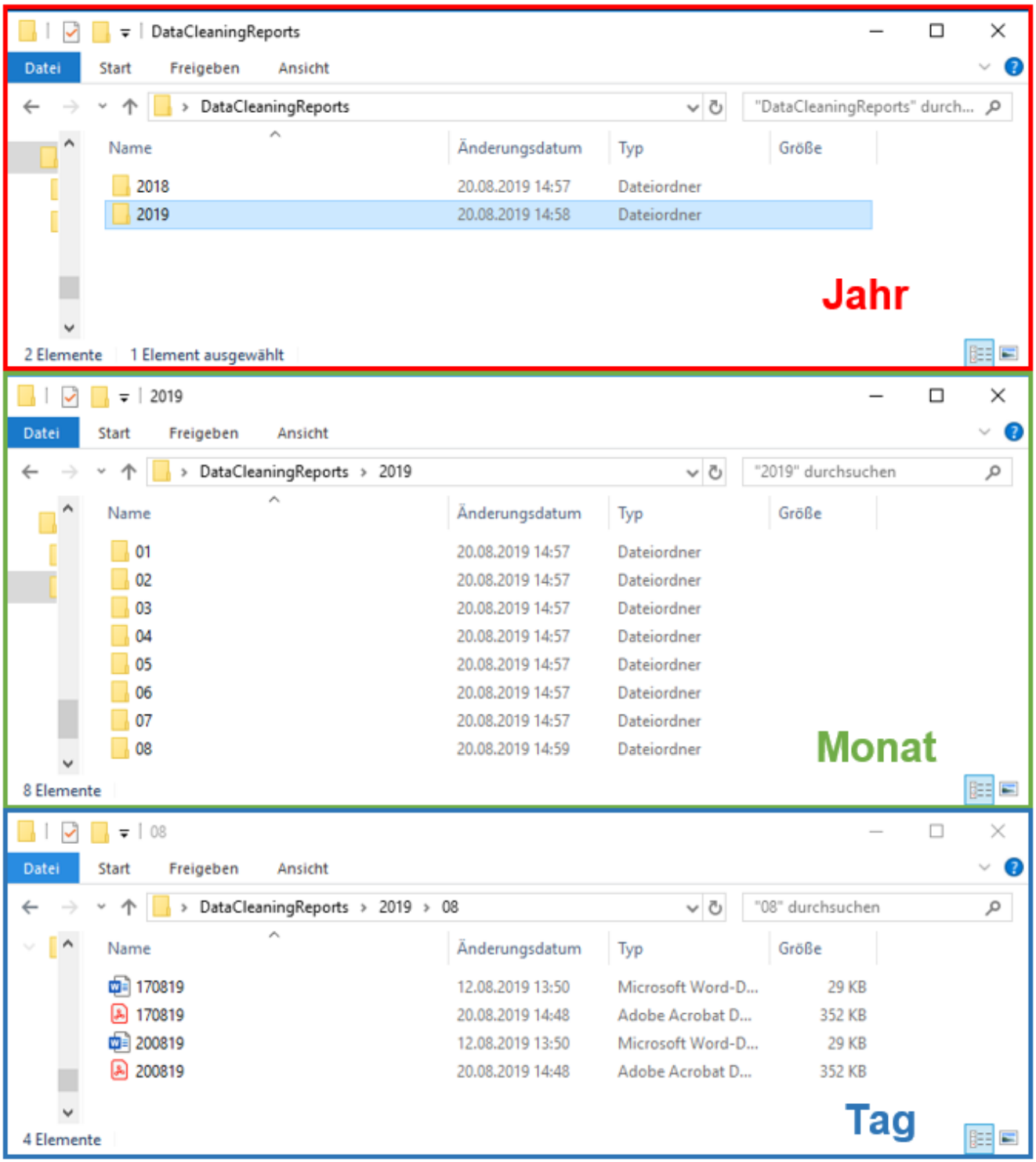


Abbildung 46: Dateistruktur Data-Cleaning-Reports

Innerhalb des Data-Cleaning-Reports werden die potenziell fehlerhaften Datensätze untereinander gereiht abgebildet. Zwischen den einzelnen Datensätzen befindet sich eine horizontale Trennlinie. In Abbildung 47 ist Datensatz aus einem Data-Cleaning-Report abgebildet.

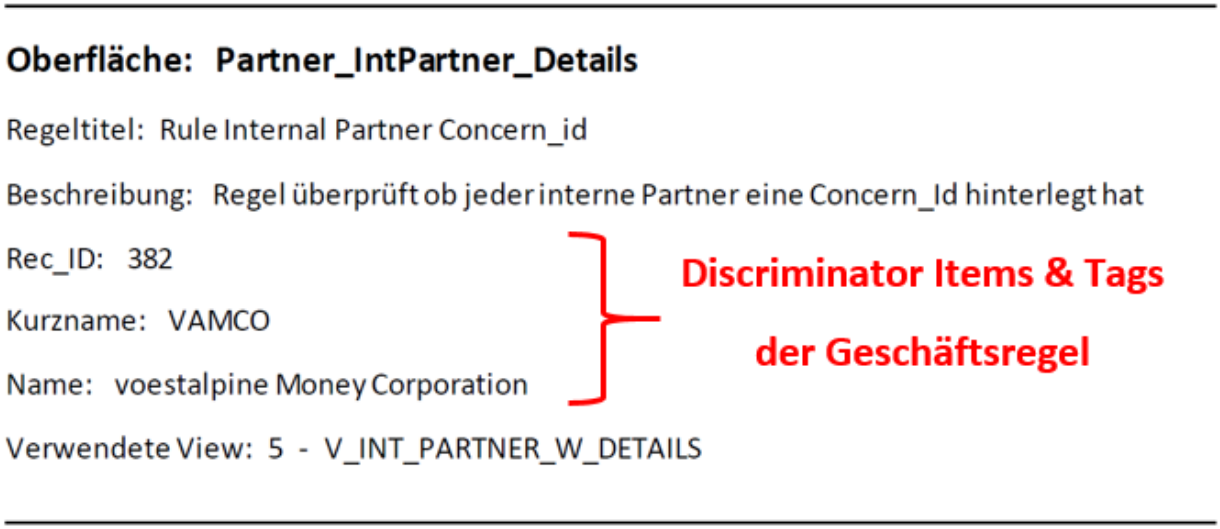


Abbildung 47: Ausschnitt Data-Cleaning-Report

Das Layout der einzelnen Datensätze ist an die Anforderungen der Sachbearbeiter des voestalpine Treasury angepasst. Da die Sachbearbeiter in das Treasury-Management-System einsteigen, um den fehlerhaften Datensatz zu begutachten, sind die Informationen bezüglich der Oberfläche am oberen Ende jedes Datensatzes angegeben. Folglich erhält der Sachbearbeiter unverzüglich die Information wohin er navigieren muss, um den fehlerhaften Datensatz einsehen zu können. Der Regeltitel und die Regelbeschreibung geben Aufschluss über die Art des Fehlers im Datensatz. In dem in Abbildung 47 gezeigten Beispiel, wurde beim dargestellten Datensatz keine `Concern_ID` hinterlegt. Dieser verstößt somit gegen die Geschäftsregel und muss durch einen Sachbearbeiter überprüft und abgeändert werden.

Im mittleren Teil jedes Datensatzes befinden sich die Discriminator Items mit den dazugehörigen Discriminator Tags. In dem in Abbildung 47 dargestellten Beispiel sind die `Rec_ID`, der `Kurzname` und der `Name` als Discriminator Tags hinterlegt und geben dem Sachbearbeiter Auskunft über den fehlerhaften Datensatz. Die Werte neben den Discriminator Tags werden durch die Discriminator Items hinterlegt. Überprüft ein Sachbearbeiter den in Abbildung 47 gezeigten Datensatz, so kann dieser über die erwähnten Informationen umgehend das zugehörige Objekt im Treasury-Management-System identifizieren. Folglich ist dieser Abschnitt eines Datensatzes von Regel zu Regel unterschiedlich, und kann über die Hinterlegung der Discriminator Items und Discriminator Tags bei der Erstellung der Geschäftsregel gesteuert werden. Abschließend wird die verwendete View angezeigt. Diese Information dient vorrangig zur Aufklärung von Unklarheiten seitens der Sachbearbeiter mit dem Autor. Dem Autor wird durch die hinterlegte Information ermöglicht, schneller Rückschlüsse über die Herkunft potenzieller Unklarheiten zu treffen.

Werden potenziell fehlerhafte Datensätze bei der Verarbeitung von verteilten Geschäftsregeln identifiziert, so werden diese grundsätzlich in derselben Struktur im Data-Cleaning-Report abgebildet. Da jedoch bei verteilten Geschäftsregeln ebenfalls die Daten der externen Datenquelle zur Verfügung stehen und die Daten aus den verwendeten Datenquellen im Rahmen dieser Masterarbeit als vollständig korrekt gelten, werden diese als Soll-Wert vorgeschlagen.

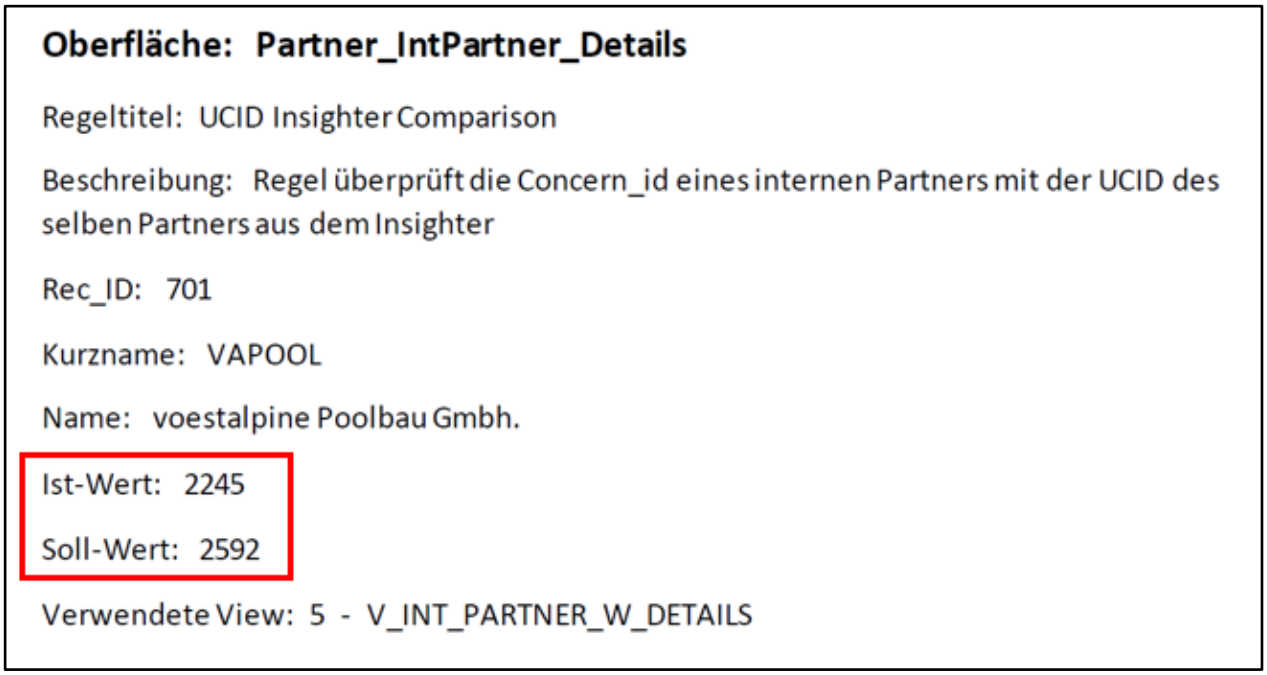


Abbildung 48: Ausschnitt Data-Cleaning-Report bei verteilten Geschäftsregeln

Wie in Abbildung 48 dargestellt, kann der Sachbearbeiter bei verteilten Geschäftsregeln zusätzlich den Wert aus der externen Datenquelle zur Entscheidungsfindung verwenden. Werden mehrere Attribute verglichen, so werden mehrere Ist- und Soll-Werte im Data-Cleaning-Report angeführt. Neben den Datensätzen wird in der Fußzeile des Data-Cleaning-Reports das Datum und die Uhrzeit der Erzeugung hinterlegt.

Der Data-Cleaning-Report wird technisch durch die Verwendung von VBA-Code (Abk. für Visual Basic für Applikationen) erzeugt. Der erwähnte VBA-Code ist fester Bestandteil des Data-Cleaning-Prototyps und sorgt für die Generierung des Reports sowie die Ablage des Reports in der Verzeichnisstruktur. VBA stellt in diesem Punkt eine reibungslose Interaktion zwischen den Microsoft-Office Anwendungen zur Verfügung und ermöglicht so das Bearbeiten von Word-Dokumenten oder Excel-Arbeitsmappen, unter Verwendung von VBA-Code innerhalb der Access-Datenbank. Es wird bei jeder Generierung eines Data-Cleaning-Reports ein Vorlagedokument geöffnet, mit Datensätzen befüllt, und anschließend abgespeichert und im PDF-Format exportiert.

7 Fazit und Ausblick

Die Einführung des in dieser Masterarbeit beschriebenen Data-Cleaning-Prozesses wurde sowohl von der Leitung als auch den Mitarbeitern des voestalpine Treasury mit positiven Rückmeldungen aufgenommen. Das aktuell verwendete Beispielset an Geschäftsregeln konnte bereits Ergebnisse in Form eines Data-Cleaning-Reports mit 138 Seiten erzielen. Bei der Überprüfung wurden 1121 Datensätze auf deren Richtigkeit hinsichtlich der Geschäftsregeln untersucht. Davon konnten 374 Datensätze als fehlerhaft identifiziert werden. Die fehlerhaften Datensätze wurden von den Sachbearbeitern des voestalpine Treasury begutachtet und die notwendigen Änderungen wurden vorgenommen. Folglich wurde durch die Identifizierung und Behebung der fehlerhaften Datensätze nicht nur die Datenqualität im voestalpine Treasury verbessert, es wurde zusätzlich ein Bewusstsein

bezüglich Datenqualität geschaffen. Durch die Erregung der Aufmerksamkeit der Mitarbeiter, bezüglich dem Nutzen der Sicherstellung einer hohen Datenqualität der internen Datenbestände, konnte durch die Mithilfe der Mitarbeiter die Regelbasis ausgebaut werden. Die Beteiligung der Mitarbeiter des voestalpine Treasury, in der Rolle der Domänenexperten, ist auch nach Abschluss dieser Masterarbeit von hoher Bedeutung. Workshops zum Ausbau der Regelbasis, unter Zusammenarbeit der Domänenexperten sowie dem Autor, wurden seitens der Leitung des voestalpine Treasury bereits zugesichert. Zusätzlich wurden die notwendigen Ressourcen zur Wartung und Erweiterung des Data-Cleaning-Prototyps für die Zukunft bereitgestellt.

Des Weiteren wurden Möglichkeiten zur Erweiterung des Data-Cleaning-Prozesses sowie des Data-Cleaning-Prototyps identifiziert. Die Auswahl eines Editor-Werkzeugs für das Erstellen der SBVR-Geschäftsregeln könnte eine Verbesserung des in Abbildung 17 abgebildeten Prozess zur Abbildung der Geschäftsregeln erzielen. Der Benutzer wird vom Editor-Werkzeug sowohl bei der Erstellung des SBVR-Vokabulars als auch bei der Erstellung der SBVR-Geschäftsregeln unterstützt. Dem Benutzer werden beispielsweise Arbeiten wie das Highlighting, welches eine wichtige Bedeutung bei der Verwendung von SBVR besitzt, abgenommen. Zusätzlich kann durch den Einsatz eines Editor-Werkzeugs die Überprüfung der syntaktischen Korrektheit des SBVR-Vokabulars sowie der SBVR-Geschäftsregeln durchgeführt werden. Derzeit sind jedoch nur wenige hochwertige Optionen im Bereich der Editor-Werkzeuge für SBVR am Markt verfügbar, daher ist es wichtig ein geeignetes Produkt auszuwählen. Eine ressourcenintensive Alternative stellt die Implementierung eines eigenen SBVR-Editors dar.

Da SBVR unterschiedliches Highlighting für seine verschiedenen Elemente verwendet, werden wichtige Informationen visuell übermittelt. Derzeit werden die SBVR-Geschäftsregeln, wie in Abbildung 23 dargestellt, in SQL-Abfragen transformiert und anschließend durch Eingabe in der grafischen Benutzeroberfläche (siehe Abbildung 38) im Speicher des Data-Cleaning-Prototyps abgelegt. Die Informationen der SBVR-Geschäftsregeln werden aktuell lediglich als Klartext abgelegt, um einen Rückschluss zwischen der Geschäftsregel im SQL-Format und derselben Geschäftsregel im SBVR-Format zu ermöglichen. Eine mögliche Erweiterung in diesem Bereich ist die Speicherung der SBVR-Geschäftsregeln im Data-Cleaning-Prototyp unter Erhaltung der im Highlighting vermittelten Informationen. Dadurch können die SBVR-Geschäftsregeln, inklusive der im Highlighting enthaltenen Informationen, ebenfalls im Data-Cleaning-Report inkludiert werden, wodurch letzterer verbessert werden kann.

8 Referenzen

[1] F. Naumann, „Datenqualität," *Informatik Spektrum,* pp. 27-31, Februar 2007.

[2] E. Rahm und H. Hai Do, „Data Cleaning: Problems and Current Approaches," *Bulletin of the Technical Committee on Data Engineering,* December 2000.

[3] S. Swapna, P. Niranjan, P. Srinivas und P. Swapna, „Data Cleaning for Data Quality," in *2016 International Conference on Computing for Sustainable Global Developement*, 2016.

[4] K. H. Prasad, T. Faruquie, S. Joshi, S. Chaturvedi, V. Subramaniam und M. Mohania, „Data Cleansing Techniques for large Enterprise Datasets," in *Annual SRII Global Conference*, 2011.

[5] K. Ali und M. A. Warraich, „A framework to implement Data Cleaning in Enterprise Data Warehouse for robust Data Quality," in *2010 International Conference on Information and Emerging Technologies*, 2010.

[6] J. Gerken, „Data Cleaning in Data Mining and Warehousing," 2015. [Online]. Available: https://www.researchgate.net/publication/265078718_Data_Cleaning_in_Data_Mining_and_Warehousing_I. [Zugriff am 26 09 2019].

[7] J. De Jesus und A. De Melo, „An architectural pattern to implement business rules in information systems," in *IEEE 17th Conference on business informatics*, 2015.

[8] M. Chisholm, How to Build a Business Rule Engine, San Francisco, CA: Elsevier, 2004.

[9] P. B. Feuto Njonko und W. El Abed, „From Natural Language Business Requirements to Executable Models via SBVR," in *International Conference on Systems and Informatics*, 2012.

[10] S. Ramzan, I. S. Bajwa und I. Asif Naeem, „A model Transformation from NL to SBVR," in *Ninth International Conference on Digital Information Management (ICDIM 2014)*, 2014.

[11] V. Natali und I. Liem, „Automated Consistency Checking Using SBVR," in *International Conference on Data and Software Engineering*, 2015.

[12] S. Moschoyiannis, A. Marinos und P. Krause, „Generating SQL Queries from SBVR Rules," in *International Symposium, RuleML*, Washington, DC, USA, 2010.

[13] I. S. Bajwa und M. G. Lee, „Transformation Rules for Translating Business Rules to OCL Constraints," in *7th European Conference, ECMFA*, Birmingham, UK, 2011.

[14] G. Z. Da Silva und J. M. De Souza, „Enforcement of Business Rules in Relational Databases using Constraints," in *XVIII Simpósio Brasileiro de Bancos de Dados*, Brasil, 2003.

[15] B. Demuth, H. Hussmann und S. Loecher, „OCL as a Specification Language for business Rules in Database Applications," in *4th International Conference on the Unified Modeling Language*, Toronto, Canada, 2001.

[16] I. S. Bajwa, B. Bordbar und M. Lee, „SBVR vs OCL: A Comparative Analysis if Standards," in *14th IEEE International Multitopic Conference*, Karachi, Pakistan, 2011.

[17] J. Cabot und R. R. Pau, „From UML/OCL to SBVR specifications: A challenging transformation," *Information Systems,* pp. 417-440, 2010.

[18] Object Management Group, „Spezifikation SBVR Version 1.4," 05 2017. [Online]. Available: https://www.omg.org/spec/SBVR/About-SBVR/. [Zugriff am 18 03 2019].

[19] J. Warmer und A. Kleppe, The Object Constraint Language Second Edition, Addison-Wesley, 2003.

[20] M. Gogolla, F. Büttner und M. Richters, „USE: A UML-based Specification Environment for Validating UML and OCL.," *Science of Computer Programming,* pp. 27-34, 2007.

[21] M. Roldan und F. Duran, „Dynamic Validation of OCL Constraints with mOdCL," in *ECEASST*, 2011.

[22] A. Queralt, A. Artale, D. Clavanese und E. Teniente, „OCL-Lite: Finite Reasoning on UM- L/OCL Conceptual Schemas," *Data and Knowledge Engineering,* pp. 1-22, 2012.

[23] „Drools Business Rules," Red Hat, [Online]. Available: https://www.drools.org/. [Zugriff am 21 08 2019].

[24] „OpenRules Business Rules," OpenRules Inc., [Online]. Available: https://openrules.com/. [Zugriff am 21 08 2019].

[25] „OpenL Tablets - Easy busiess Rules," OpenL Tablets, [Online]. Available: openl-tablets.org/. [Zugriff am 21 08 2019].

[26] „Jess business Rules," Sandia National Laboratories, [Online]. Available: https://www.jessrules.com/jess/. [Zugriff am 21 08 2019].

[27] M. Beedle, „JLisa Business Rules," [Online]. Available: http://jlisa.sourceforge.net/. [Zugriff am 21 08 2019].

[28] „Openlexicon Business Rule Engine," [Online]. Available: http://freshmeat.sourceforge.net/projects/openlexicon. [Zugriff am 21 08 2019].

[29] „JRule Business Rule Engine," [Online]. Available: jruleengine.sourceforge.net/. [Zugriff am 21 08 2019].

[30] „NxBRE, Business Rule Engine on Github," [Online]. Available: https://github.com/ddossot/NxBRE. [Zugriff am 21 08 2019].

[31] „SRE - Simple Rule Engine on Github," [Online]. Available: https://github.com/codeyu/SimpleRuleEngine. [Zugriff am 21 08 2019].

[32] „Drools.NET 3.0," [Online]. Available: http://www.csharpopensource.com/droolsdotnet/. [Zugriff am 21 08 2019].

[33] „Sweet Rules Business Rules," [Online]. Available: sweetrules.semwebcentral.org/. [Zugriff am 21 08 2019].

[34] U. Abdullah, M. J. Sawar und A. Ahmed, „Design of a rule based system using Structured Query Language," in *Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, Chengdu, China, 2009.

[35] Object Management Group, „SBVR EU-Rent Example," 26 08 2016. [Online]. Available: https://www.omg.org/spec/SBVR/About-SBVR/. [Zugriff am 19 03 2019].