Masterarbeit

ERSTELLUNG EINER BEDARFSORIENTIERTEN DATENBANKLÖSUNG ZUR ASSET-VERWALTUNG IM LIFE CYCLE MANAGEMENT

ausgeführt am

Fachhochschul-Masterstudiengang
Automatisierungstechnik-Wirtschaft

von
Kerstin Ramian, BSc
2010322014

betreut und begutachtet von
Dr. Florian Hollomey

Gleisdorf, im November 2021

.....
Unterschrift

EHRENWÖRTLICHE ERKLÄRUNG

Ich erkläre ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die benutzten Quellen wörtlich zitiert sowie inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Unterschrift

DANKSAGUNG

An dieser Stelle gebührt mein besonderer Dank all jenen, die mich bei der Erstellung dieser Masterarbeit unterstützt und motiviert haben.

Primär bedanken möchte ich mich bei meinem Masterarbeitsbetreuer Dr. Florian Hollomey, der mir mit seinem herausragenden fachlichen Rat zur Seite gestanden ist und mir mit konstruktiver Kritik, hilfreichen Anregungen und seiner beruhigenden Art dabei geholfen hat, die vorliegende Arbeit zu erstellen und dabei die Ruhe zu bewahren.

Darüber hinaus bedanke ich mich bei meinem Vorgesetzten und Betreuer im Unternehmen Stefan Fuchs, MA, der mich mit zahlreichen Ideen maßgeblich dabei unterstützt hat, die Anforderungen des Unternehmens in ein Konzept überzuleiten.

Ausdrücklich von ganzem Herzen danken möchte ich meinem Lebensgefährten Manuel Jegg für seine Geduld und Unterstützung über die letzten Jahre hinweg und dass er mir in stressigen Phasen stets die kleinen Dinge des Alltags abgenommen hat. Er hat mir kontinuierlich die Kraft gegeben, die Zuversicht nicht zu verlieren und konsequent mein Ziel weiterzuverfolgen.

Abschließend möchte ich mich bei meiner Studien- und Arbeitskollegin Anika Jaindl, BSc bedanken, die mir über die gesamte Studienzeit als Schicksalsgenossin motivierend und unterstützend beiseite stand und mir in dieser Zeit eine sehr gute Freundin geworden ist.

KURZFASSUNG

Jeder Betreiber einer kritischen Infrastruktur muss gemäß der europäischen Richtlinie zur Netz- und Informationssicherheit (NIS) sicherstellen, dass alle Anlagen stets auf dem neuesten Stand der Technik sind. Die Dürr Austria GmbH erfüllt und befolgt die NIS-Richtlinie für ihren Auftraggeber, der für die kritische Infrastruktur von Tunneln in Österreich verantwortlich ist.

Die Verwaltung aller in einem Tunnel verbauten Assets ist umfangreich und eine lückenlose Dokumentation sämtlicher sicherheitsrelevanten Maßnahmen ist essenziell. Ziel dieser Masterarbeit war es, eine geeignete und individuelle Datenbanklösung zur Verwaltung dieser Assets zu entwickeln.

Basierend auf umfangreichen Recherchen zu Asset-Management, Datenmanagement und möglichen Datenbanklösungen, wurde die optimale Lösung für das Unternehmen ermittelt und ein User Interface Mock-up des Prototyps, unter Berücksichtigung der Vorgaben für die Visualisierung, erstellt. Dieser Prototyp veranschaulicht die grafische Benutzeroberfläche der zukünftigen Datenbank und demonstriert ihre Funktionalitäten. Die Ergebnisse zeigen, dass der Ansatz praktisch anwendbar und umsetzbar ist und die Implementierung der Datenbank im Unternehmen möglich ist.

Basierend auf allen Ergebnissen dieser Masterarbeit wird die Datenbank in naher Zukunft programmiert und im Unternehmen eingeführt.

ABSTRACT

The European Network and Information Security (NIS) directive determines that every operator of a critical infrastructure must ensure that their systems are always at the current state of the art. Dürr Austria GmbH fulfils and complies the NIS directive for its client who is responsible for the critical infrastructure of tunnels in Austria.

As the management of the tunnel’s built-in assets is widespread, the gapless documentation of all security-related actions is significant. The aim of this master thesis is to develop an appropriate and individual database solution to manage these assets.

Based on the extensive research on asset management, data management and possible database solutions the optimal solution for the company was determined and a user interface mock-up of the prototype considering the guidance for the visualization was established. This prototype illustrates the graphical user interface of the future database and demonstrates its functionalities. As the results demonstrate, the approach is practically applicable and feasible and the possibility of the implementation of the database solution has been proved.

Based on all results of this master thesis, the database will be programmed and implemented in the company.

INHALTSVERZEICHNIS

1 EINLEITUNG

Die 2001 von acht erfahrenen Fachleuten aus dem Bereich der Tunnel- und Verkehrssicherheit gegründete Dürr Austria GmbH zählt zu ihren Kernkompetenzen das Projekt- und Sitemanagement, Planungs- und Dokumentationsleistungen, die Prozessleittechnik mit allen damit verbundenen Aktivitäten wie Pflichtenhefterstellung, Systemprogrammierung, Systemvisualisierung, die Montage und Inbetriebnahme auf der Baustelle, sowie die Wartung und Instandhaltung des Gesamtsystems.[1]

Im Unternehmen Dürr Austria GmbH wird gemeinsam mit der Autobahnen- und Schnellstraßen-Finanzierungs-Aktiengesellschaft (ASFiNAG) als Auftraggeber ein neuartiges Projekt umgesetzt, welches unter Berücksichtigung des Netz- und Informationssystemsicherheitsgesetzes (NISG) sicherstellt, dass alle Anlagen der kritischen Infrastruktur stets am neusten Stand der Technik sind. Dürr Austria ist im Zuge des Life Cycle Managements in Österreich für 38 Tunnelanlagen verantwortlich, Tendenz steigend. Auf ca. 150 Tunnelkilometern sind mehr als 600 speicherprogrammierbare Steuerungen (SPS), rund 80 Server und 300 Touchpanels inklusive Visualisierungen verbaut.

1.1 Darstellung des Problems (Ist-Situation)

Alle elektronischen Komponenten, welche zur Steuerung einer Tunnelanlage essenziell sind, müssen gemäß NISG die Fähigkeit besitzen, Sicherheitsvorfällen vorzubeugen, diese zu erkennen, abzuwehren und zu beseitigen. Dies wird unter anderem durch regelmäßiges Patch-Management, Change-Management, Härtungsmaßnahmen, Logging, Datensicherung und Backups sowie Schwachstellenmanagement und Desaster Recovery erreicht. Im Life Cycle Management werden die bestehenden leittechnischen Anlagen des Kunden evaluiert, sukzessive erneuert und aktualisiert. Hierbei wird mit Hilfe von Microsoft Excel eine große Menge an Daten erhoben.

Bei diesen Komponenten handelt es sich in erster Linie um:

- Tunnelkopfrechner (TuKo)
- Lokale Steuereinheiten (LStE)
- Bedieneinheiten (Touchpanel, Touchmonitore)
- Bedienstationen (z.B.: (Not-) Bedienplätze)
- Intelligente Busklemmen
- Server
- Network Attached Storages (NAS)
- Infotafeln und Wechselverkehrszeichen

[1] Vgl. Dürr Austria GmbH (2021), Online-Quelle [15.11.2021].

Um die Datenmengen systematisch erfassen und regelmäßig auswerten sowie das Life Cycle Management administrieren zu können, wird eine Datenbank benötigt, die es dem Unternehmen ermöglicht je Komponente Informationen wie

- Configuration Item (CI) Name (z.B.: eindeutiger Hostname)
- Exakter Standort im Tunnel bzw. am Gelände (Straßenname, Kilometrierung, Richtungsfahrbahn, Ortstyp, Ortsbezeichnung, Geschoß, Raumname, Schranknummer)
- Systembezeichnung und Funktion
- Gerätehersteller
- Gerätebezeichnung
- Gerätetyp
- Gerätename
- IP-Adresse(n)
- Seriennummer
- Mögliche Redundanzen zu weiteren Komponenten
- Inbetriebnahmedatum
- Lieferantenname
- Zuständiger Instandhalter
- Komponentenbezeichnung
- Wartungstyp
- Softwarebezeichnung jeder einzelnen auf der Komponente befindlichen Software
- Verwendungszweck und Versionsnummer jeder einzelnen auf der Komponente befindlichen Software
- End-of-Life-Datum sowie Datum letzter Patch jeder einzelnen auf der Komponente befindlichen Software

zu erfassen und regelmäßig auf Handlungsbedarf und Optimierungspotential zu prüfen.

Als Quellen für sämtliche benötigten Informationen dienen sowohl bestehende Einkaufs-, Bestands- und Projektdaten sowie allfällige Instandhaltungs- und Lokalisationsdaten. Vor allem im Bereich der End-of-Life Ermittlung wird auf herstellerspezifische Informationen zurückgegriffen. Ein Überblick über die möglichen Quellen wird in Abbildung 1 grafisch dargestellt.

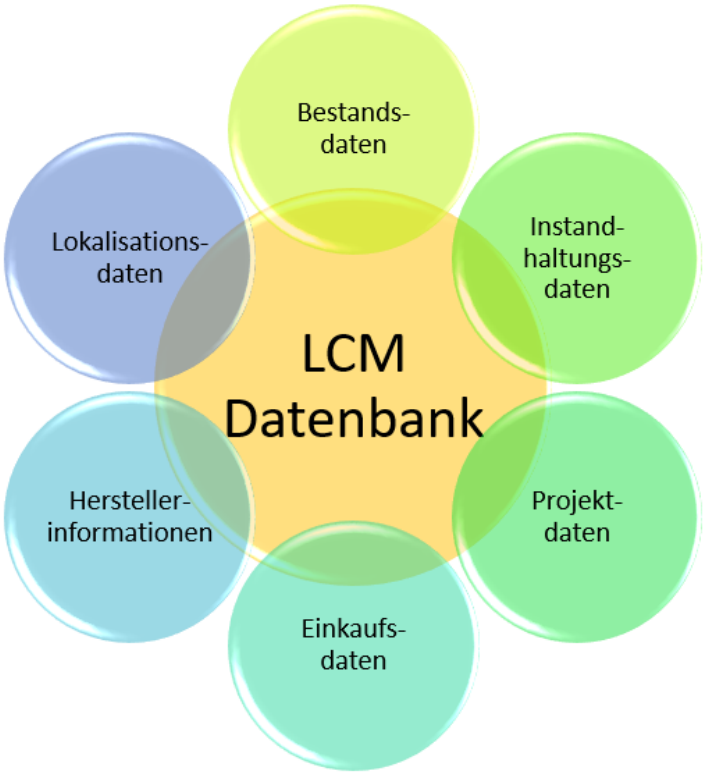

Abbildung 1: Quellen für die LCM Datenbank, Quelle: Eigene Darstellung.

1.2 Zielsetzung der Arbeit

Im Zuge dieser Arbeit sollen die theoretischen Grundlagen der Datenhaltung beleuchtet werden. Ebenso soll geprüft werden, welche Datenbanklösung den für den gewünschten Einsatz erforderlichen Anforderungen nachkommt und welche Prämissen das dazugehörige Frontend erfüllen soll. Ergänzend werden die Vor- und Nachteile gegenüber der bestehenden Microsoft Excel-Lösung betrachtet und ein konzeptioneller Prototyp der Datenbank als User Interface-Mockup modelliert.

In der späteren Datenbank sollen alle notwendigen Informationen erfasst und ausgewertet, sowie eine Anpassungs- und Erweiterungsmöglichkeit geboten werden. Die Probleme beim Ist-Zustand sollen dem Unternehmen zukünftig bestmöglich abgenommen werden.

2 ASSET-MANAGEMENT

2.1 Begriffsdefinition Asset und Asset-Management

Im Allgemeinen wird der Begriff Asset mit einem Vermögenswert oder -gegenstand gleichgesetzt. Ein Wirtschaftsgut oder eine Anlage, die den Aktivposten eines Unternehmens zugerechnet wird. Man kann sagen, es handelt sich dabei um eine Ressource unabhängig davon, ob es sich um Hard- oder Software handelt.

Beim Asset-Management geht es darum, Assets zu verwalten sowie die Eigentumsrechte und die Nutzung der Assets zu kennen. Mit Hilfe von Einkaufsaufzeichnungen erfolgt die Prüfung, ob es sich um ein Asset im engeren Sinne handelt. Hierbei wird geprüft, ob zum jeweiligen Asset Bestell- und Rechnungsinformationen vorliegen.[2]

2.2 Anwendung von Asset-Management

Unter Asset-Management werden systematische, koordinierte Praktiken und Aktivitäten verstanden, um Assets und deren Leistungen sowie zugehörige Risiken und Ausgaben während des Asset-Lebenszyklus optimal und nachhaltig zu verwalten.[3]

Gemäß ISO 55000:2014 unterliegt das Asset-Management folgenden Einflussfaktoren, welche sowohl beim Aufbau, bei der Implementierung, der Pflege als auch der kontinuierlichen Verbesserung des Asset-Managements berücksichtigt werden müssen:[4]

- Art und Zweck der Organisation
- Betriebskontext
- Finanzielle Beschränkungen und regulatorische Anforderungen
- Bedürfnisse und Erwartungen der Organisation und ihrer Stakeholder

Asset-Management legt den Fokus nicht direkt auf das Anlagegut selbst, sondern auf den Wert, den dieses für die Organisation darstellt und stellt somit sicher, dass dieses seinen geforderten Zweck erfüllt.[5]

[2] Vgl. Boerger (2021), S. 53.

[3] Vgl. Crespo/Macchi/Kumar (2020), S. 4.

[4] Vgl. ISO 55000 (2014), S. 1.

[5] Vgl. ISO 55000 (2014), S. 3.

Es dient der effektiven Kontrolle und Steuerung von Vermögenswerten, dem Management von Risiken und Chancen und ist das Werkzeug, um ein Gleichgewicht zwischen Kosten, Risiko und Leistung zu erreichen. Durch die Verwendung eines risikobasierten Ansatzes werden die organisatorischen Zielsetzungen in anlagenbezogene Entscheidungen sowie Pläne und Aktivitäten übersetzt.[6]

Gemäß ÖNORM EN16646:2015 wird das Anlagenmanagement wie folgt definiert:

„Anlagenmanagement ist als koordinierte Tätigkeiten einer Organisation zur Wertschöpfung mit Anlagen definiert. Insbesondere handelt es sich dabei um „das optimale Management des Lebenszyklus von Anlagen, um die angegebenen Geschäftsziele nachhaltig zu erreichen“.“[7]

Um ein effektives Asset-Management sicherzustellen, empfiehlt sich die Anwendung eines Anlagenmanagementsystems bzw. Datenbankmanagementsystems (DBMS).

Gemäß ÖNORM EN16646:2015 wird ein Managementsystem folgendermaßen charakterisiert:

„Ein Managementsystem für Anlagen (Anlagenmanagementsystem) ist eine Anzahl untereinander verbundener oder aufeinander wirkender Elemente einer Organisation, die Leitlinien und Ziele des Anlagenmanagements und die Prozesse zum Erreichen dieser Ziele festlegt. Ein Anlagenmanagementsystem ist nicht einfach nur ein Informationssystem – es enthält zugleich die Struktur, Rollen, Verantwortlichkeiten, Geschäftsprozesse, Pläne, Arbeitsvorgänge der Organisation.“[8]

Um ein systematisches Asset-Management erreichen zu können und die notwendigen Informationen im Asset-Management-System hinterlegen zu können, muss sich ein Unternehmen gezielt mit dem Asset Life Cycle, also dem Lebenszyklus eines Assets, auseinandersetzen.

Alle Assets durchlaufen einen bestimmten Lebenszyklus unabhängig davon, ob es sich um Hard- oder Software handelt. Assets, die sowohl physisch als auch einen rein virtuellen Wert für das Unternehmen darstellen, durchschreiten ihren eigenen Lebenszyklus. In den meisten Fällen besteht dieser aus:

- Kaufen
- Vorbereiten
- Einsetzen
- Erhalten
- Wiederverwenden
- Recyceln oder wegwerfen

[6] Vgl. ISO 55000 (2014), S. 1.

[7] ÖNORM EN16646 (2015), S. 7.

[8] ÖNORM EN16646 (2015), S. 9.

Je nach Kritikalität der Assets können diverse Schritte dem Zyklus hinzugefügt oder innerhalb des Zyklus kombiniert, sowie aus dem Zyklus entfernt werden.[9]

Abbildung 2 veranschaulicht exemplarisch einen solchen Asset Lebenszyklus.

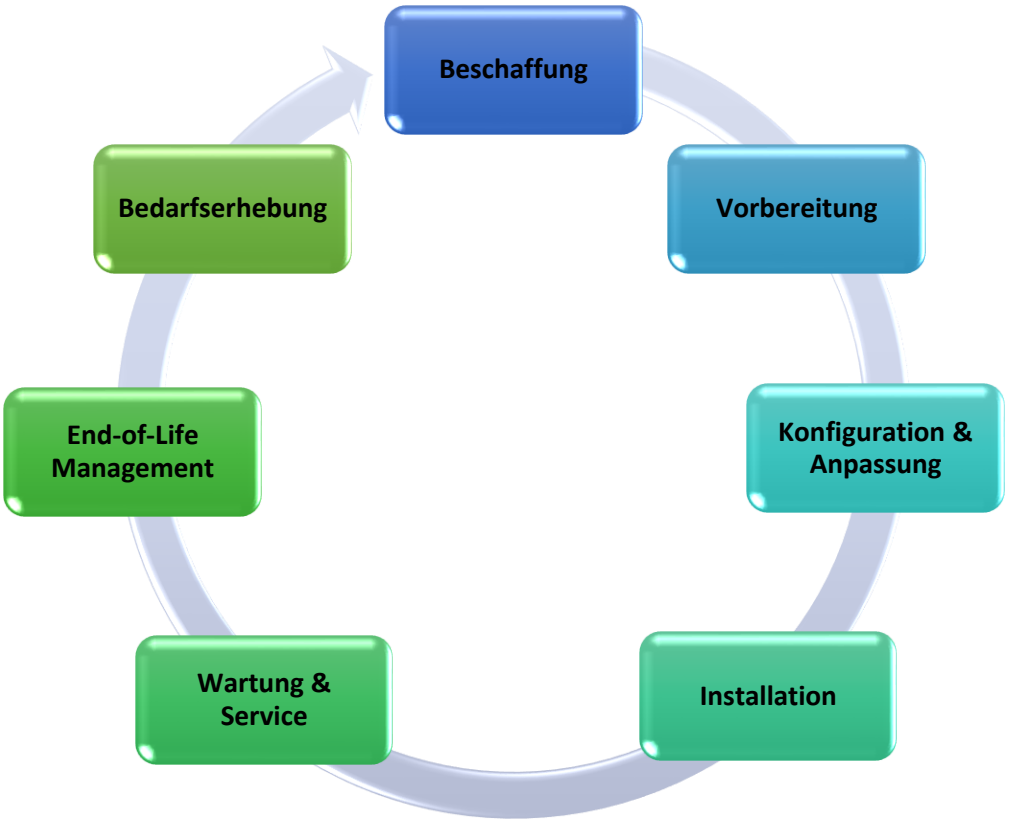

Abbildung 2: Stufen des Life Cycle, Quelle: Eigene Darstellung.

Um eine möglichst hohe Datenqualität sicherstellen zu können, ist eine Optimierung und Standardisierung der Stammdaten unumgänglich. Hier wird unterschieden zwischen Formatstandards, welche eine einheitliche Codierung voraussetzen, Datenstandards, die eine Vereinheitlichung von Semantik und Struktur festlegen, Nachrichten-/Dokumentenstandards zur Homogenisierung von Verknüpfungen und Übertragungen, Prozessstandards, welche sämtliche organisatorischen Abläufe vereinheitlichen und Geschäftsstandards, die vor allem rechtliche Rahmenbedingungen im Fokus haben. Die Optimierung aller stammdatenrelevanten Standards bietet die Möglichkeit, Redundanzen und Obsoleszenzen aufzuzeigen und so die Datenqualität zu steigern.[10]

[9] Vgl. Boerger (2021), S. 54.

[10] Vgl. Gronwald (2020), S. 9.

2.3 Instandhaltung und Asset-Management

Insbesondere im Zuge von Wartungs- und Instandhaltungstätigkeiten (W&I-Tätigkeiten) dient ein Asset-Management als Unterstützung, um gezielte Entscheidungen zu treffen, welche Assets gewartet werden und welche Assets upgedatet oder ersetzt werden müssen. Hierbei muss im Zuge der Ausführung auch die Datenbank aktualisiert werden z.B.: welche Softwarestände wurden geändert, welche Teile wurden ersetzt etc.

Die ÖNORM EN16646:2015 beschreibt diesbezüglich die Trennung der Instandhaltungsprozesse in folgende Teilprozesse:[11]

- Instandhaltungsziele und -strategien;
- Planung der Instandhaltungstätigkeiten;
- Management der Ressourcen und Entwicklung;
- Durchführung der Instandhaltung;
- Nachkontrolle und kontinuierliche Verbesserung.

[11] ÖNORM EN16646 (2015), S. 12.

3 MÖGLICHKEITEN ZUR DATENHALTUNG

Der Duden beschreibt eine Datenbank als elektronisches System, in dem große Bestände an Daten zentral gespeichert sind.

3.1 Begriffsdefinition Daten und Datenbank

Eine Datenbank wird sowohl als eine Sammlung an Daten, also ein geordnetes Sammelwerk an Informationen definiert, als auch als Programm zur Verwaltung dieser Daten. Hierfür bedarf es unter anderem Funktionen wie erfassen, sortieren, filtern und anzeigen.[12]

Daten bzw. Informationen werden durch die folgenden Attribute beschrieben:

- Darstellung
- Verarbeitung
- Kombination
- Alter
- Original
- Vagheit
- Träger

Sie unterscheiden sich von materiellen Wirtschaftsgütern in diesen Eigenschaften, da Informationen durch Daten spezifiziert werden und mit Hilfe von Algorithmen und Datenstrukturen dargestellt und verarbeitet werden. Sie sind arbiträr kombinier-, kopier- und manipulierbar und können zumindest physikalisch nicht altern. Sämtliche Informationen sind ortsunabhängig und ihre Datenqualität ist in vielen Fällen vage und mehrdeutig. Darüber hinaus können Informationen vergleichsweise kosteneffizient erzeugt und vervielfältigt werden.[13]

Essenziell für eine hohe Datenqualität ist es, Daten stets nur einmal zentral zu speichern und das unabhängig von einer bestimmten Anwendung oder Programmiersprache. Dies bietet die Möglichkeit, Daten trotz Änderungen weiterhin verwenden zu können und Inkonsistenzen zu vermeiden. Darüber hinaus profitiert man von einem geringeren Speicherbedarf und schafft so zusätzliche Backupmöglichkeiten. Ein DBMS ermöglicht und kontrolliert den Zugriff auf sämtliche Daten sowohl lesend als auch schreibend und bietet die Möglichkeit, Benutzer*innen weitere Rechte und Einschränkungen zuzuweisen wie dem Hinzufügen oder Löschen von Daten.[14]

[12] Vgl. Kersken (2019), S. 1135 f.

[13] Vgl. Meier/Kaufmann (2016), S. 1.

[14] Vgl. Hartmut/Jochen/Gerd (2016), S. 338.

Unternehmen können zahlreiche Motive für die Einführung einer Datenbank haben. Es müssen beispielsweise große Informationsmengen gespeichert und verwaltet werden oder es werden gezielte Auswertungen oder Analysen aus einer Fülle an Informationen benötigt. Oftmals besteht der Wunsch, Redundanzen oder Fehler in den gespeicherten Daten zu vermeiden.[15]

3.2 Definition Datenbankmanagementsystem

Ein Datenbankmanagementsystem beschreibt eine Software, die im Allgemeinen die Funktionen bietet Daten hinzuzufügen, zu löschen, zu ändern und zu suchen. Es bietet die Möglichkeit Daten dauerhaft zu speichern und in unterschiedlichsten Formen zur Verfügung zu stellen. Es kann mehreren Benutzern paralleler Zugriff auf die Daten gewährt werden und eine Benutzerverwaltung ermöglicht jedem Benutzer individuelle Zugriffsrechte zuzuweisen. Auf diese Weise wird unterschieden, welcher Benutzer Daten lesen, schreiben oder löschen darf. Verzichtet die Datenbanksoftware beispielsweise auf eine Benutzerverwaltung bzw. -einschränkung, bedeutet dies einen geringeren administrativen Aufwand und ermöglicht einen einfacheren und schnelleren Umgang mit diesem Datenbanksystem. Das DBMS ermöglicht logische Verknüpfungen einzelner Inhalte unter Berücksichtigung von spezifischen Integritätsregeln und ihr Einsatz gewährleistet die Konsistenz von Daten sowohl logisch als auch physikalisch (z.B. bei Systemausfällen).[16]

3.3 Architektur von Datenbanksystemen

Der generelle Aufbau eines Datenbanksystems besteht aus drei Elementen: der Datenbank, welche die Basis jedes Datenbanksystems darstellt, dem Datenbankmanagementsystem und der Anwendersoftware als Schnittstelle zum Benutzenden. Diese*r befüllt die Datenbank mit großen Datenmengen, welche wiederum strukturiert vom Datenbanksystem verarbeitet und gespeichert werden. Aktuelle Systeme werden in drei Schichten bzw. Ebenen gegliedert, welche die Möglichkeit bieten sämtliche Daten separat zu bearbeiten und dank des DBMS anwendungsunabhängig zu speichern und mittels Benutzerverwaltung individuell zur Verfügung zu stellen. Die Trennung erfolgt konzeptuell, logisch und physisch.[17]

Eine grafische Darstellung eines solchen Datenbanksystems ist in Abbildung 3 ersichtlich.

[15] Vgl. Theis (2021), S. 23.

[16] Vgl. Piepmeyer (2011), S. 4 ff.

[17] Vgl. Schneider (2004), S. 2 ff.

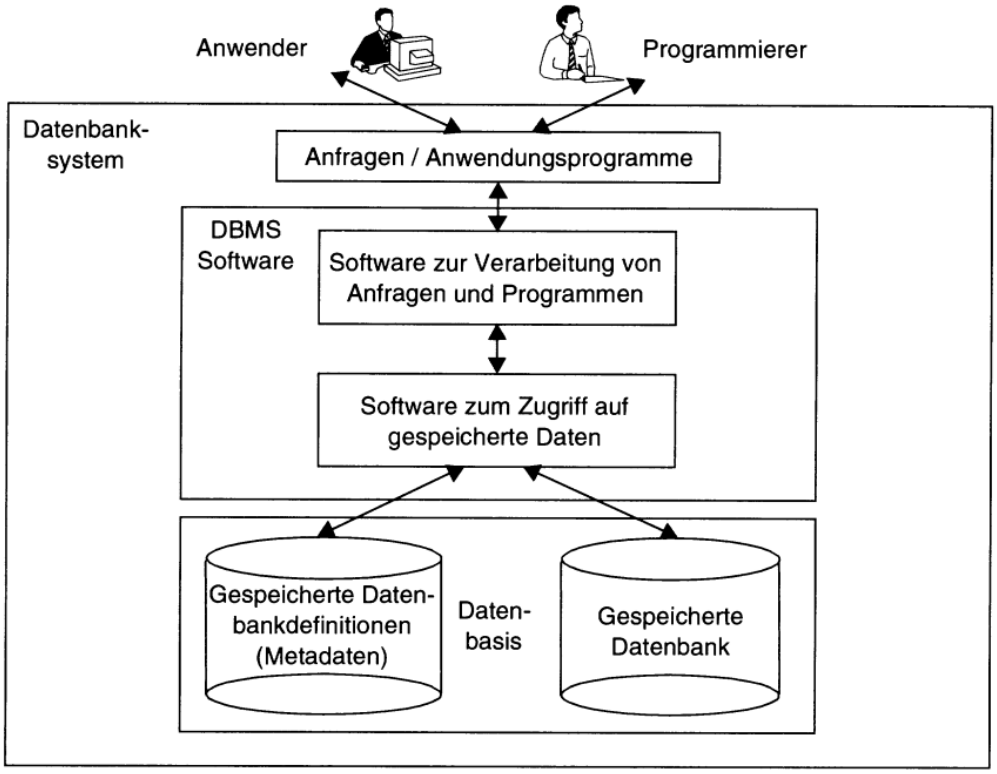

Abbildung 3: Vereinfachte Sicht eines Datenbanksystems, Quelle: Schneider (2004), S. 4.

Charakteristiken von Datenbanksystemen sind wie folgt:

- Datenunabhängigkeit
- Effizienter Datenzugriff
- Gemeinsame Datenbasis
- Nebenläufiger Datenzugriff
- Redundanz
- Konsistenz der Daten
- Integrität der Daten
- Datensicherheit
- Bereitstellung von Backup- und Recovery-Verfahren
- Stellen von Anfragen
- Bereitstellung verschiedenartiger Benutzerschnittstellen
- Flexibilität
- Schnellere Entwicklung von neuen Anwendungen

So wird gewährleistet, dass eine Datenbank unabhängig von einem bestimmten Anwendungsprogramm betrieben werden kann, der verfügbare Speicher effizient genutzt wird und unterschiedliche Benutzer*innen auf für sie bestimmte Daten Zugriff haben. Es muss sichergestellt werden, dass sämtliche Daten vollständig und zentral, möglichst ohne unnötige Redundanzen, gespeichert sind und jederzeit wiederhergestellt

werden können. Eine einfache und vollständige Datenmanipulationsmöglichkeit (z.B.: Bearbeitung, Auswertung, Optimierung, Berichterstellung, etc.) sollte gegeben sein.[18]

3.4 Transaktionen und Konsistenz

Es wird zwischen „ACID“ und „BASE“ Systemen unterschieden.

ACID ist ein Akronym und steht für „Atomicity“ (Atomität), „Consistency“ (Konsistenz), „Isolation“ und „Durability“ (Dauerhaftigkeit). Diese Eigenschaften beschreiben, dass alle Transaktionen als kleinste unteilbare Einheit betrachtet werden. Des Weiteren muss sichergestellt werden, dass sich Daten ausnahmslos in konsistentem Zustand befinden. Ein Überführen von Daten findet nur statt, wenn vor und nach der Übertragung die Konsistenz aller Daten gewährleistet ist. Sobald Inkonsistenzen im Zuge der Transaktion erkannt werden, wird diese unterbrochen und rückgängig gemacht. Durch die Isolation werden Transaktionen isoliert betrachtet und diese haben keinen Einfluss auf andere Transaktionen. Die Dauerhaftigkeit beschreibt die Persistenz der Daten und stellt sicher, dass sogar nach einem Neustart oder Systemfehler die Daten in der Datenbank gespeichert bleiben.[19]

BASE gilt als Gegenstück zu ACID-Systemen und beschreibt die flexibleren Eigenschaften von NoSQL-Systemen, welche in Kapitel 3.5.7 detailliert beschrieben werden. BASE steht für „Basically Available“ (grundsätzlich erreichbar), „Soft-State“ (weicher Status) und „Eventual Consistency“ (schlussendlich konsistent). Diese Eigenschaften bedeuten, dass BASE-Systeme hochverfügbar sind, dies jedoch auf Kosten der Konsistenz der Daten. Soft-State steht für diesen Umstand, wenn auf Anfragen reagiert wird, jedoch noch nicht sämtliche Änderungen an das System übertragen wurden. Die Datenbank ist in diesem Übergangszeitraum „eventual consistent“ und ist erst zu einem späteren Zeitpunkt vollständig.[20]

Abbildung 4 zeigt einen Überblick über die Gegensätzlichkeiten der Eigenschaften von ACID und BASE.

[18] Vgl. Schneider (2004), S 5 ff.

[19] Vgl. Kurowski (2012), S. 11.

[20] Vgl. Technische Hochschule Mittelhessen (3) (2015), Online-Quelle [02.09.2021].

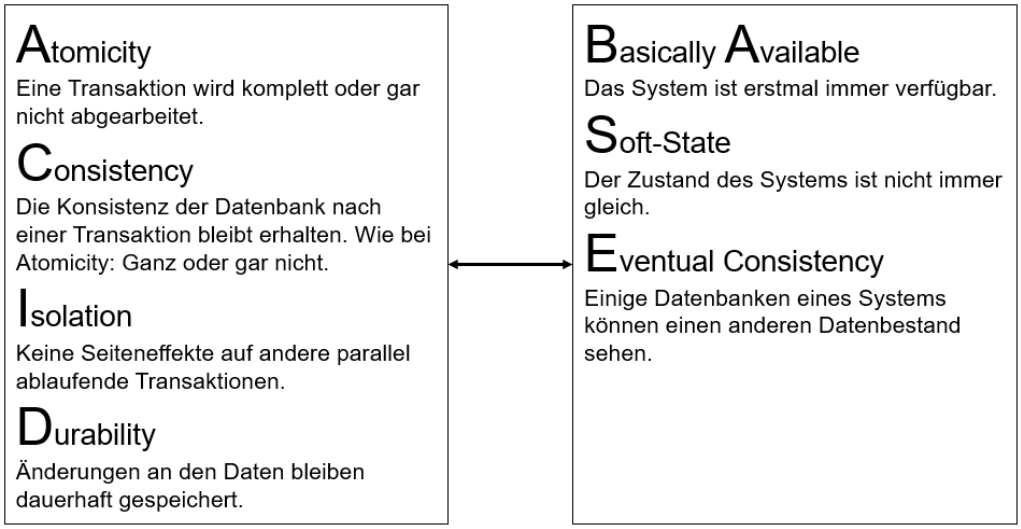

Abbildung 4: Gegenüberstellung ACID und BASE, Quelle: Kurowski (2012), S. 15 (leicht modifiziert).

3.5 Datenbankmodelle

Es werden im Allgemeinen folgende Datenbankmodelle unterschieden:

- Einzeltabellen
- Hierarchische Datenbanken
- Netzwerkdatenbanken
- Relationale Datenbanken
- Objektorientierte Datenbanken
- Objektrelationale Datenbanken
- NoSQL-Datenbanken

Jeder dieser Datenbanktypen bietet unterschiedliche Eigenschaften und Möglichkeiten, welche nachfolgend beschrieben werden.

3.5.1 Einzeltabellen

Bei Einzeltabellen handelt es sich um die simpelste Art von Datenbanken. Sie bestehen aus einer einzelnen Tabelle mit Informationen. Ihr Anwendungsgebiet ist die Speicherung einfacher Daten z.B. zur Adressverwaltung. Der grundsätzliche Aufbau einer Einzeltabelle besteht aus Spalten, in denen die Daten in Kategorien aufgeteilt werden und Zellen, die jeweils einzelne Informationen beinhalten. Diese werden auch als Datenfelder bezeichnet. Ein Datensatz ist eine Kombination einzelner Zellen zu einer Zeile und enthält alle Informationen zu einer Entität. Einzeltabellen bieten die Möglichkeit einer Sortierung nach Kategorie, nach unterschiedlichen Parametern wie alphabetisch auf- und absteigend, nach Zellinhalt etc. Eine Such- und Filteroption sowie die Möglichkeiten verschiedener Formatierungen sind in Einzeltabellen ebenfalls gegeben. Als Nachteile einer Einzeltabelle können sowohl eine mangelnde Änderungsmöglichkeit mehrerer Inhalte gleichzeitig als auch eine fehlende Inhaltsüberprüfung genannt werden. Ändert sich ein Datensatz z.B. für mehrere Mitarbeiter*innen oder Abteilungen gleichzeitig, muss

dieser überall einzeln angepasst werden, was insbesondere bei umfangreichen Einzeltabellendatenbanken einen enormen Zeitaufwand bedeuten kann. Ist der Inhalt eines Datenfeldes nicht konsistent, z.B. durch einen Tippfehler, wird damit die Logik einer Tabelle beeinträchtigt und sämtliche Such- und Filtermöglichkeiten werden beeinflusst.[21]

Abbildung 5 illustriert eine minimalisierte Version einer Einzeltabelle.

Nachname	Vorname	Geburtsdatum	Wochenstunden	Dienstjahre	Beschäftigungsgruppe
Groß	Elisabeth	23.09.1963	38,5	2	E
Lackner	Josef	08.03.1959	38,5	15	F
Kainz	Manuela	19.04.1972	25,0	7	D
Sommer	Herbert	30.11.1965	38,5	4	D

Abbildung 5: Beispiel für eine Einzeltabelle, Quelle: Eigene Darstellung.

3.5.2 Hierarchische Datenbanken

Hierarchische Datenbanken weisen eine Baumstruktur auf (1:1 und 1:n Beziehungen), welche ein einfaches und schnelles Bearbeiten der Daten durch ihre Linearität ermöglicht. Operationen wie Suchen, Hinzufügen und Bearbeiten sowie Löschen können simpel durchgeführt werden, Querbeziehungen sind allerdings durch die starre Hierarchie schwer umsetzbar (z.B. gleicher Artikel von diversen Herstellern). Übersichtsmäßige Auswertemöglichkeiten gestalten sich schwierig und aufwändig.[22]

Ein vereinfachtes Beispiel für ein hierarchisches Datenbankmodell ist in Abbildung 6 dargestellt.

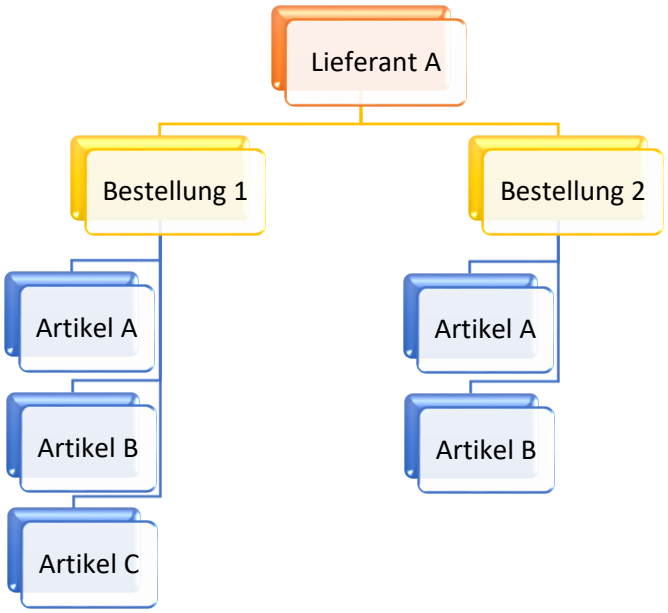

Abbildung 6: Beispiel für ein hierarchisches Modell, Quelle: Eigene Darstellung.

[21] Vgl. Kersken (2019), S. 1140 ff.

[22] Vgl. Hartmut/Jochen/Gerd (2016), S. 339.

3.5.3 Netzwerkdatenbanken

Die ersten Netzwerkdatenbanken verwendeten Graphen als grundlegende Datenstruktur. Verglichen mit hierarchischen Systemen können reellere Abbildungen umgesetzt werden. Beziehungen sind leichter abbildbar, aber mit einem höheren Aufwand verbunden als bei hierarchischen Systemen. Übersichtsmäßige Auswertemöglichkeiten gestalten sich schwierig und aufwändig.[23]

Abbildung 7 zeigt eine vereinfachte Darstellung einer Netzwerkdatenbank.

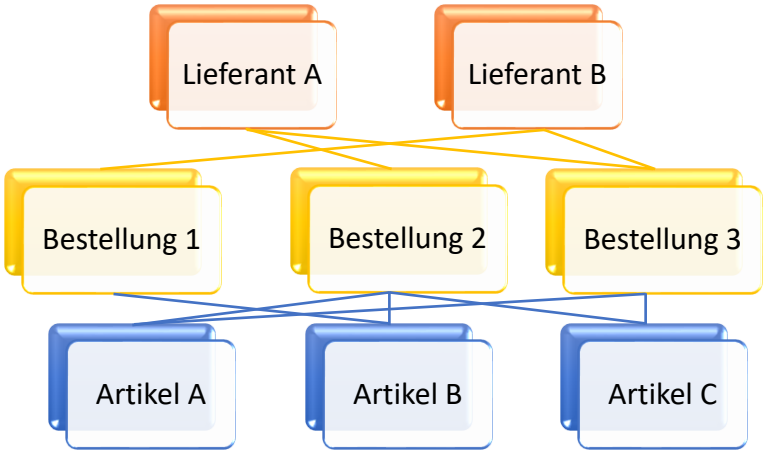

Abbildung 7: Vereinfachte Darstellung eines Netzwerkdatenbankmodells, Quelle: Eigene Darstellung.

3.5.4 Relationale Datenbanken

Die Basis relationaler Datenbanken sind Einzeltabellen, welche beliebig miteinander verknüpft werden. Diese Vorgehensweise sichert die Konsistenz der Daten, da sämtliche Informationen nur einmal gespeichert werden. Aufgrund dieser Tatsache gilt die relationale Datenbank als wichtigstes Datenbankmodell in der Praxis. Grundlage solcher Datenbanken ist die relationale Algebra und sie werden beschrieben durch die Begrifflichkeiten: Attribut, Tupel, Relation und Beziehung (engl. „relationship“). Als Attribut wird die Tabellenspalte (A_1 bis A_n) und als Tupel ein einzelner Datensatz definiert. Eine Mehrheit an Tupeln gilt als Relation r(R) und ist die Bezeichnung für eine Tabelle. Somit gilt $r(R) = r(A_1, \ldots, A_n)$.[24]

Ein Beispiel für eine Relation wird in Abbildung 8 dargestellt.

Attribut
Tupel
Primärschlüssel
Relation

NR	NAME	KAUFNR	BETRAG
101	Vogel AG	20210101	13.580,23
102	Meister GmbH	20210223	5.101,18
103	Holz OG	20210608	1.395,07
104	Grün KG	20210415	19.346,26

Abbildung 8: Beispiel einer Relation, Quelle: Eigene Darstellung.

[23] Vgl. Hartmut/Jochen/Gerd (2016), S. 339.

[24] Vgl. Kersken (2019), S. 1143 ff.

Zwischen unterschiedlichen Tabellen werden mittels Verknüpfungen Beziehungen zwischen dem Primärschlüssel eines Datensatzes und einer anderen Tabelle hergestellt. Es wird zwischen drei Beziehungsarten unterschieden:

- 1:1-Beziehung
- 1:n-Beziehung (Eins-zu-Viele-Beziehung)
- m:n-Beziehung (Viele-zu-Viele-Beziehung)

Während bei der 1:1-Beziehung genau ein Datensatz mit einem weiteren einer anderen Tabelle in Beziehung gesetzt wird, wird bei der 1:n-Beziehung ein Tabellendatensatz mit mehreren Datensätzen einer anderen Relation verknüpft. Diese Form der Beziehung ermöglicht es Diskrepanzen zu vermeiden, weshalb in der Praxis überwiegend die 1:n-Beziehung zur Anwendung kommt. m:n-Beziehungen kombinieren eine beliebige Anzahl an Tupeln mit anderen Tupeln. Diese Relation wird indirekt umgesetzt, indem mehrere Tabellen mittels 1:n-Beziehung miteinander verbunden werden. Relationale Datenbanken werden in der standardisierten Abfragesprache Structured Query Language (SQL) umgesetzt.[25]

Abbildung 9 stellt die Beziehungen zwischen den Tabellen einer einfachen relationalen Datenbank dar. Die Tupel „NR“, „KAUFNR“ und „ARTNR“ dienen als Primärschlüssel der jeweiligen Tabelle. Zwischen den Tabellen „Adressen“ und „Artikel“ besteht eine indirekte m:n-Beziehung.

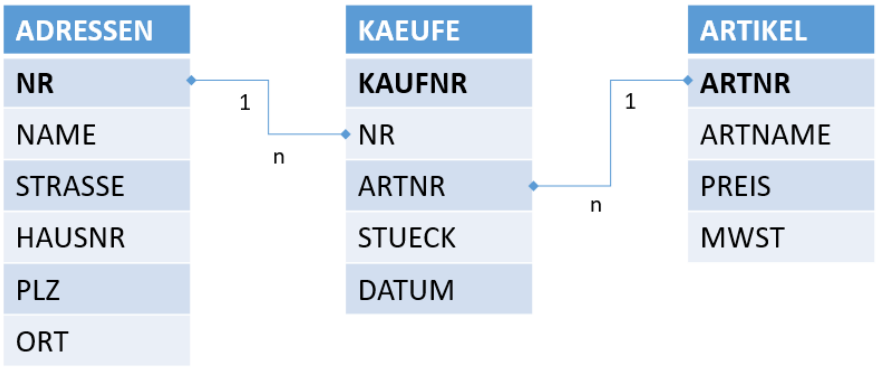

Abbildung 9: Beziehungen zwischen den Tabellen einer einfachen relationalen Datenbank, Quelle: Kersken (2019), S. 1146 (leicht modifiziert).

Mithilfe von sechs sogenannten „Normalformen“ wird aufeinander aufbauend Schritt für Schritt dafür Sorge getragen, dass in einer relationalen Datenbank Redundanzen und Inkonsistenzen bestmöglich verhindert werden:

- Erste Normalform (1NF)
- Zweite Normalform (2NF)
- Dritte Normalform (3NF)
- Boyce-Codd-Normalform (BCNF)
- Vierte Normalform (4NF)
- Fünfte Normalform (5NF)

[25] Vgl. Kersken (2019), S. 1144 ff.

Die erste Normalform legt fest, dass sämtliche Tupel atomar und somit nicht weiter zerlegbar sind und Wiederholungen gleicher Informationen vermieden werden müssen (z.B. eine Person mit mehreren Telefonnummern). Zusätzlich wird in der zweiten Normalform festgelegt, dass sämtliche Informationen in einer Tabelle nur vom Primärschlüssel abhängig sein dürfen. Die dritte Normalform gibt vor, dass keine Information, die keinen Primärschlüssel darstellt, von einer anderen Nichtschlüsselinformation funktionell abhängig sein darf. Die BCNF gilt als strengere Version der dritten Normalform und stellt sicher, dass, falls der Primärschlüssel aus mehreren Feldern besteht, eine Aufteilung in Einzeltabellen durchgeführt wird, in denen diese Felder Primärschlüssel darstellen. Die vierte Normalform kommt bei mehrwertigen Abhängigkeiten zum Einsatz. Hier können aufgrund mehrerer Spalten in einer Tabelle keine exakten 1:n-Beziehungen mehr hergestellt werden z.B. wenn eine Person mehrere Programme in unterschiedlichen Programmiersprachen einsetzt. Gelöst wird dieses Problem beispielsweise durch Unterteilung der Ergebnistabelle in zwei Einzeltabellen. Bei der fünften Normalform dürfen nur triviale Join-Abhängigkeiten vorhanden sein. Diese kommen in Tabellen vor, welche in mehreren Einzeltabellen mit demselben Primärschlüssel aufgespaltet werden können. Die Trivialität wird dadurch beschrieben, dass eine Verknüpfung dieser Tabellen keine Redundanzen erzeugt. Die Normalisierung sollte bei der Modellierung der Datenbank berücksichtigt werden.[26]

Abbildung 10 zeigt einen grafischen Überblick über die Normalformen und ihre Charakterisierung.

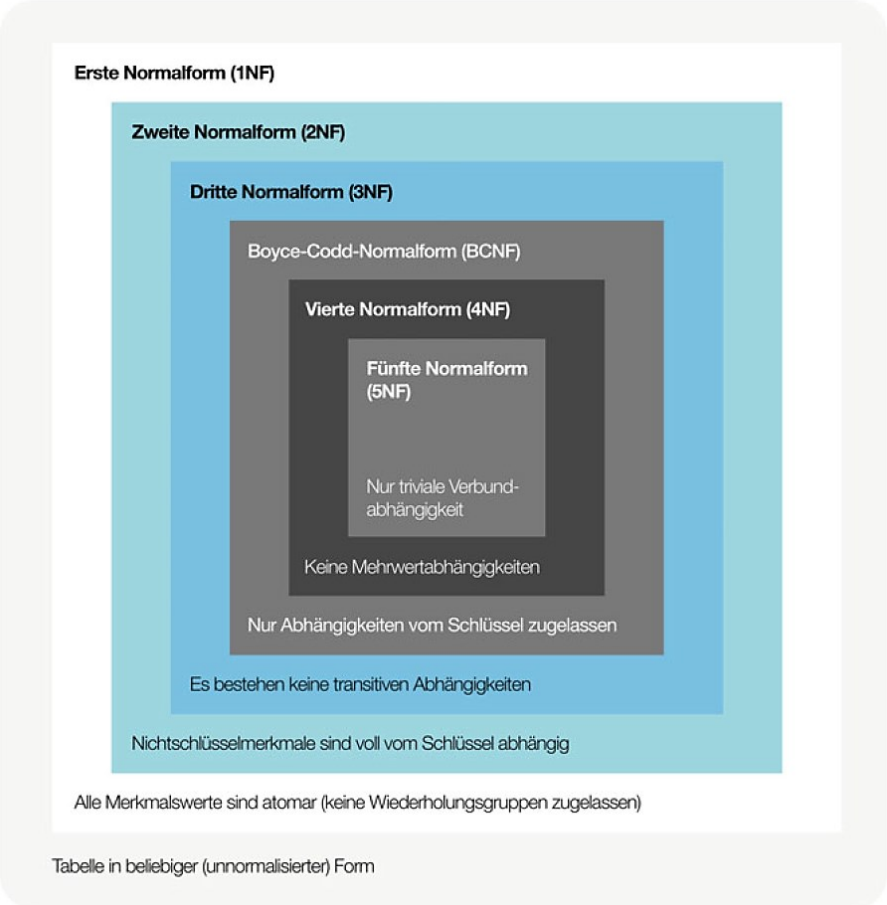

Abbildung 10: Übersicht über die Normalformen und ihre Charakterisierung, Quelle: Meier/Kaufmann (2016), S. 38.

[26] Vgl. Kersken (2019), S. 1149 ff.

3.5.5 Objektorientierte Datenbanken

Mit Hilfe von objektorientierten Datenbanken können Strukturen umgesetzt werden, die bei relationalen Datenbanken mit den Regeln zur Normalisierung nicht umsetzbar sind. Ähnlich der objektorientierten Programmierung kommen bei diesen Datenbanken Klassen und Objekte zum Einsatz. Umgesetzt wird dies in vielen Fällen in der Object Definition Language (ODL), oftmals in Kombination mit der Object Query Language (OQL). Diese ermöglichen es, objektorientierte Datenstrukturen zu erstellen und mit Werten zu hinterlegen, sind jedoch nicht im selben Ausmaß wie SQL standardisiert. Häufig sind die objektorientierten Programmiersprachen C++, C# oder Java in Verwendung. Eine Klasse (engl. „class“) nimmt in objektorientierten Datenbanken die Stelle einer Tabelle in relationalen Datenbanken ein. Sowohl im relationalen Umfeld als auch in der Objektorientierung wird die Spalte einer Tabelle als Attribut (engl. „attribute“) deklariert. Die einzelnen Tupel bzw. Datensätze werden z.B. mit Hilfe der Datentypen „string“, „short“ oder „long“ definiert. Ein Tupel stellt gleichzeitig eine Instanz einer Klasse dar. Dies ermöglicht es, Beziehungen direkt zu referenzieren. Komplexe Kombinationen z.B. Datums- oder Entfernungsangaben werden mittels Struktur (engl. „struct“) umgesetzt und somit als Gruppe dargestellt. Werden Daten als Liste benötigt, wird dies z.B. mittels „array“ realisiert.[27]

Ein vereinfachtes Beispiel für eine Tabelle einer objektorientierten Datenbank ist in Abbildung 11 ersichtlich.

```
class Kauf {
   attribute long KaufNr;
   relationship Adressen Kunde;
   relationship Artikel Ware;
   attribute short Stueck;
   attribute struct Datum {
      short Tag;
      short Monat;
      short Jahr;
   } KaufDatum;
}
class Ort {
   attribute string Name;
   struct Entfernung {
      relationship Ort Zielort;
      attribute short Kilometer;
   };
   array (struct Entfernung) Entfernungen;
}
```

Abbildung 11: Codebeispiel einer Tabelle in einer objektorientierten Datenbank, Quelle: Kersken (2019), S. 1158 (leicht modifiziert).

3.5.6 Objektrelationale Datenbanken

Objektrelationale Datenbanken gelten als Mischform von objektorientierten und relationalen Datenbanken. Dies wird ermöglicht durch Erweiterungen der SQL-Sprache mit objektorientierten Eigenschaften wie Objektidentifikation, Datentypen für Listen, Mengen und Felder sowie abstrakte und parametrierbare Datentypen und benutzerdefinierbare Methoden. Sie bieten die Möglichkeit Klassen zu definieren und

[27] Vgl. Kersken (2019), S. 1155 ff.

Objekte zu strukturieren und zu identifizieren. Methoden und Vererbung werden unterstützt und die Datenkapselung ermöglicht es, dass interne Definitionen von Objekten nach außen unsichtbar bleiben. Ein großer Teil der in der Praxis verwendeten Datenbanksysteme ist relational aufgebaut. Um die Notwendigkeit einer Migration in objektorientierte oder objektrelationale Datenbanken zu vermeiden, kann mittels Zuordnung (engl. „mapping“) auf relationale Daten zugegriffen werden. Ein Beispiel für ein objektrelationales Mapping ist in Abbildung 12 ersichtlich. Die Informationen in den relationalen Tabellen „Autor“, „Buch“ und „Verfasst“ werden in einem relationalen Datenbankmanagementsystem (RDBMS) gespeichert. Zwischen „Autor“ und „Buch“ wird mittels objektorientierter Programmierung (OOP) eine komplexe objektorientierte Verbindung hergestellt. Eine objektrelationale Mapping Software (ORM-Software) stellt automatisch eine Verbindung zwischen dem RDBMS und der OOP her. Auf diesem Weg können relationale Datenbanken und Objektorientierung miteinander verknüpft werden.[28]

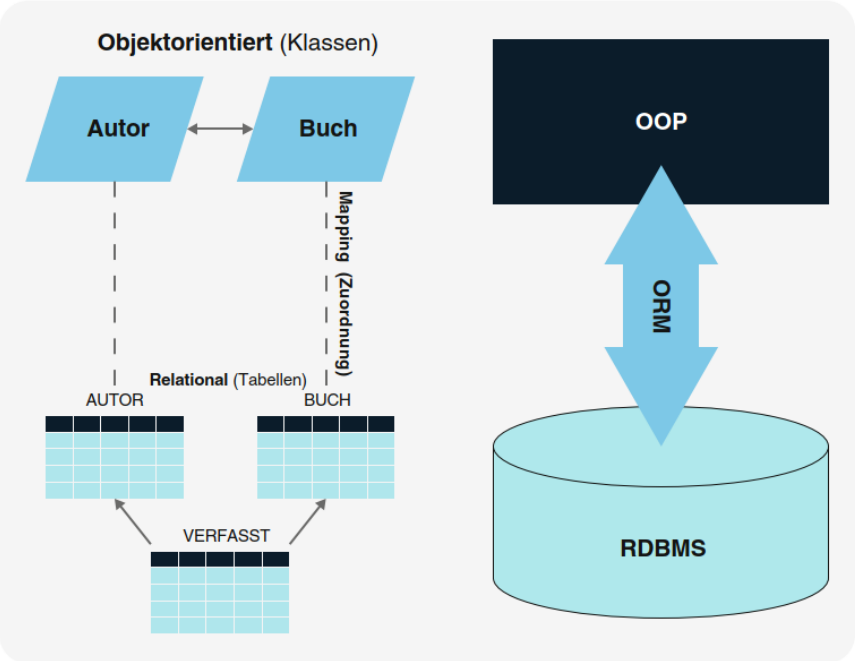

Abbildung 12: Objektrelationales Mapping, Quelle: Meier/Kaufmann (2016) S. 207.

3.5.7 NoSQL-Datenbanken

NoSQL stand ursprünglich für „no“ SQL also „kein“ SQL. Mittlerweile wird diese Datenbankform als „not only“ SQL, also „nicht nur“ SQL interpretiert. NoSQL-Datenbanken unterscheiden sich von relationalen Systemen insbesondere durch ihre größere Skalierbarkeit und der Möglichkeit, sehr große Datenmengen verarbeiten und mit Hilfe spezialisierter Abfrageoperationen auswerten zu können. Ein großer Teil der bestehenden NoSQL-Datenbanken ist als Open-Source verfügbar.[29]

[28] Vgl. Meier/Kaufmann (2016), S. 206 ff.

[29] Vgl. Kleppmann (2019), S. 31 f.

Abbildung 13 stellt die wesentlichen Eigenschaften von SQL-Datenbanken und NoSQL-Datenbanken gegenüber.

	SQL-Datenbank	NoSQL-Datenbank
Art	Eine Datenbank für alles	Verschiedene Datenbankmodelle, wie z. B. Dokumenten-, Graph-, Key-Value- und Spaltendatenbank
Datenspeicherung	Individuelle Daten (z. B. „Buchtitel“) werden in Zeilen in einer Tabelle gespeichert und bestimmten Attributen zugeordnet (z. B. „Autor“, „Erscheinungsjahr“ usw.). Die Datensätze werden in separaten Tabellen gespeichert und bei komplexen Suchanfragen vom System zusammengeführt.	NoSQL-Datenbanken nutzen keine Tabellen, sondern je nach Typ ganze Dokumente, Key-Values, Graphen oder Spalten.
Schemata	Datentyp und -struktur werden im Vorhinein festgelegt. Um neue Informationen zu speichern, muss die gesamte Datenbank angepasst werden (und zu diesem Zweck offline gehen).	Flexibel. Neue Datensätze können sofort hinzugefügt werden. Strukturierte, halbstrukturierte und unstrukturierte Daten können zusammen abgespeichert werden, eine vorherige Konvertierung ist nicht notwendig.
Skalierung	Vertikale Skalierung. Ein einzelner Server muss die Leistung des kompletten Datenbanksystems tragen, was einen Leistungsabfall bei großen Datenmengen zur Folge hat.	Horizontale Skalierung. Jeder Administrator kann neue Commodity- und Cloud-Server hinzufügen, die NoSQL-Datenbank sendet die Daten automatisch an alle Server.
Entwicklungsmodell	Open Source (z. B. MySQL) oder Bezahl-Software (Oracle Database)	Open Source oder Bezahl-Software
ACID-Eigenschaften: Atomicity, Consistency, Isolation, Durability	Bei SQL-Datenbanken sind alle ACID-Eigenschaften gegeben.	Damit NoSQL-Datenbanken flexibel und horizontal skalierbar bleiben, werden ACID-Transaktionen meistens nicht unterstützt. Stattdessen wird das BASE-Modell verwendet (Basically Available, Soft State, Eventually Consistant). Demnach gilt: Verfügbarkeit vor Konsistenz.
Leistung	Nutzen bei erhöhtem Datenaufkommen Indizes. Um die Leistung von SQL-Systemen zu erhöhen, müssen Abfragen, Indizes und Struktur optimiert werden.	Durch das Nutzen von Cloud-Servern und Hardware-Clustern haben NoSQL-Datenbanken eine deutlich höhere Leistungsstärke.
APIs	Abfragen zur Speicherung und zum Abrufen von Daten werden mit SQL (Structured Query Language) übermittelt.	Daten werden über objektbasierte APIs gespeichert und abgefragt.

Abbildung 13: Gegenüberstellung SQL und No-SQL Datenbanken, Quelle: Ionos SE (2019), Online-Quelle [15.09.2021].

Die flexibleren Methoden von NoSQL-Datenbanken verglichen mit relationalen Systemen ermöglichen die Be- und Verarbeitung unstrukturierter Daten. Anstelle von Tabellen werden flexible Techniken zur Organisation der Daten angewendet. NoSQL-Datenbanken verwenden z.B. Wertepaare, Spalten, Graphen oder Dokumente, um mit Hilfe flexibler Strukturen, horizontaler Skalierung und gleichmäßiger Verteilung von Kapazitäten große Datenmengen zu verarbeiten.[30]

[30] Vgl. Ionos SE (2019), Online-Quelle [15.09.2021].

NoSQL-Systeme lassen sich in vier Kerntechnologien unterteilen:

- Schlüssel-Wert-Datenbanken (engl. „key-value-database“)
- Spaltenorientierte Datenbanken (engl. „column-oriented-database“)
- Graphdatenbanken (engl. „graph-oriented-database“)
- Dokumentenorientierte Datenbanken (engl. „document-oriented-database“)

Schlüssel-Wert-Datenbanken (engl. „key-value-database“)

Schlüssel-Wert-Datenbanken gelten als die älteste und simpelste Form von NoSQL-Datenbanken. Datensätze werden mittels Verwendung sogenannter Schlüssel-Wert-Paare (engl. „key-value-pair“) verwaltet. Jeder Schlüssel enthält einen Wert (engl. „value“) aus einer zufälligen oder strukturierten Verkettung von Zeichen und wird als „String“, „Liste“, „Set“ oder „Hash“ deklariert. Zum Beispiel erhält jeder Schlüssel (engl. „key“) einen Wert und einen Hash. Somit entsteht ein Schlüssel-Werte-Paar. Ein Hash kann mehrere Werte enthalten. Wesentlicher Vorteil von Schlüssel-Wert-Systemen ist ihre Effizienz durch eine schnelle Lese- und Schreibgeschwindigkeit und durch ihre Skalierbarkeit aufgrund ihres einfachen Schemas. Durch das Schlüssel-Werte-System können Änderungen einfach mittels Versionierung der Daten durchgeführt werden. Komplexe Abfragen können in Key-Value-Datenbanken aufgrund fehlender Suchalgorithmen jedoch nur sehr aufwändig durchgeführt werden.[31]

Spaltenorientierte Datenbanken (engl. „column-oriented-database“)

In der Regel findet die Speicherung von Daten zeilenweise statt. In den meisten Fällen wird allerdings nur auf wenige Datensätze dieser Zeile zugegriffen. In spaltenorientierten Datenbanken werden alle Werte einer Zeile stattdessen in einer Spalte abgelegt. Beim Zugriff wird auf diese Weise nur auf jene Dateien zugegriffen, die für die gewünschte Abfrage benötigt werden, was die Anzahl der Abfragen reduziert und somit die Bereitstellung der Daten beschleunigt. Abbildung 14 veranschaulicht die Speicherung in Spalten verglichen mit einem relationalen Modell. Dieses System ist auf gleiche Weise mit nicht relationalen Daten umsetzbar. Wenn sich Datensätze in der spaltenorientierten Speicherung wiederholen, ermöglicht diese Speicherungsform zusätzlich die Komprimierung der enthaltenen Daten, was eine Erhöhung der Effizienz der Datenbank zur Folge hat.[32]

[31] Vgl. Begerow Beratungsgesellschaft mbH & Co. KG (1) (2021), Online-Quelle [16.09.2021].

[32] Vgl. Kleppmann (2019), S. 103 ff.

Tabelle fact_sales

date_key	product_sk	store_sk	promotion_sk	customer_sk	quantity	net_price	discount_price
140102	69	4	NULL	NULL	1	13.99	13.99
140102	69	5	19	NULL	3	14.99	9.99
140102	69	5	NULL	191	1	14.99	14.99
140102	74	3	23	202	5	0.99	0.89
140103	31	2	NULL	NULL	1	2.49	2.49
140103	31	3	NULL	NULL	3	14.99	9.99
140103	31	3	21	123	1	49.99	39.99
140103	31	8	NULL	233	1	0.99	0.99

Spaltenorientiertes Speicherlayout:

date_key file contents:	140102, 140102, 140102, 140102, 140103, 140103, 140103, 140103
product_sk file contents:	69, 69, 69, 74, 31, 31, 31, 31
store_sk file contents:	4, 5, 5, 3, 2, 3, 3, 8
promotion_sk file contents:	NULL, 19, NULL, 23, NULL, NULL, 21, NULL
customer_sk file contents:	NULL, NULL, 191, 202, NULL, NULL, 123, 233
quantity file contents:	1, 3, 1, 5, 1, 3, 1, 1
net_price file contents:	13.99, 14.99, 14.99, 0.99, 2.49, 14.99, 49.99, 0.99
discount_price file contents:	13.99, 9.99, 14.99, 0.89, 2.49, 9.99, 39.99, 0.99

Abbildung 14: Relationale Daten spaltenweise statt zeilenweise speichern, Quelle: Kleppmann (2019), S. 104.

Graphdatenbanken (engl. „graph-oriented-database“)

In Graphdatenbanken werden Objekte möglichst realitätsnah und explizit als Graphenmodell gespeichert und zueinander in Beziehung gesetzt. Graphen können zwar ebenfalls in relationalen Systemen gespeichert werden, jedoch ist eine Wiederherstellung des ursprünglichen Graphen sehr komplex. Graphen bestehen aus Knoten (Objekte) und Kanten (Beziehungen), welche mittels Algorithmen vernetzt werden.[33]

Ein wesentliches Merkmal von Graphdatenbanken ist die Flexibilität von Anwendungen. Die Kante eines jeden Knoten kann zu einem anderen Knoten führen. Die Kanten und Knoten sind jedoch nicht geordnet und es wird auf die Knoten mittels Index oder eindeutigem Identifikator (engl. „identifier“, ID) zugegriffen.[34]

[33] Vgl. Begerow Beratungsgesellschaft mbH & Co. KG (2) (2021), Online-Quelle [16.09.2021].

[34] Vgl. Kleppmann (2019), S. 63.

Abbildung 15 demonstriert einen Graphen eines Beziehungsmodells mit Knoten und Kanten.

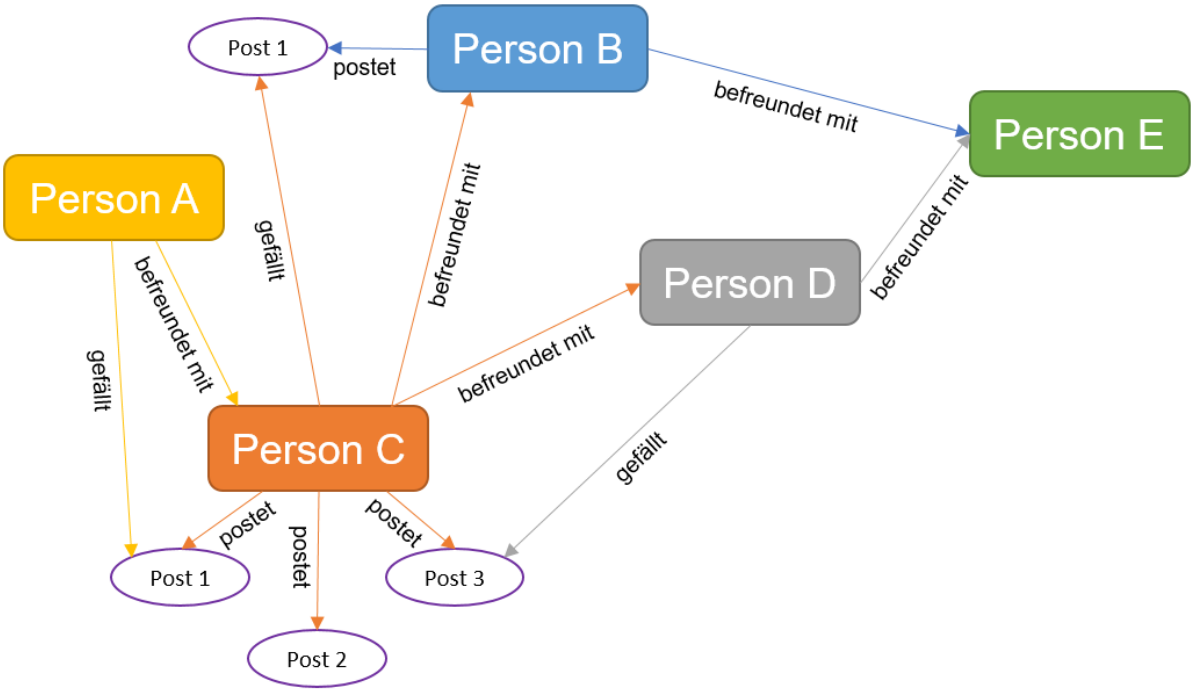

Abbildung 15: Graph eines Beziehungsmodells bestehend aus Knoten und Kanten, Quelle: Eigene Darstellung.

Dokumentenorientierte Datenbanken (engl. „document-oriented-database“)

Dokumentenorientierte Datenbanken sind durch die fehlende Vorgabe eines bestimmten Schemas äußerst flexibel und es wird je Schlüsel ein Dokument gespeichert. Sie sind vom Prinzip ähnlich wie Schlüssel-Wert-Datenbanken aufgebaut und verwenden als Datenformate sogenannte JSON („JavaScript Object Notation“) Objekte oder XML („Extensible Markup Language“) Dokumente. Durch die transparente Struktur können nicht nur die Dokumente selbst gesucht und geändert werden, sondern es können auch Änderungen in den Dokumenten selbst durchgeführt werden.[35]

Im Gegensatz zu Graphdatenbanken, in denen in einigen Fällen sogar alles mit allem in Beziehung gesetzt sein kann, ist in dokumentenorientierten Datenbanken eine Verknüpfung zwischen Dokumenten eher selten.[36]

Beispiele für NoSQL-Datenbanken sind:

- Cassandra (spaltenorientierte Datenbank)
- CouchDB (dokumentenorientierte Datenbank)
- Neo4j (cloudbasierte Graphdatenbank) und
- Redis (Key-Value-Datenbank)

[35] Vgl. Technische Hochschule Mittelhessen (1) (2015), Online-Quelle [16.09.2021].

[36] Vgl. Kleppmann (2019), S. 66.

3.6 Datenbanksprachen

Diverse Datenbanksprachen ermöglichen es, auf vielfältige Benutzeranforderungen einzugehen. Während relationale Abfragesprachen die Möglichkeit bieten, sowohl die Datenbank zu erstellen als auch Benutzerrechte zu vergeben oder die Inhalte von Tabellen zu ändern oder auszuwerten, bieten z.B. graphbasierte Sprachen vordefinierte Optionen und Routinen um Graphprobleme, wie z.B. das Ermitteln von kurzen Pfaden, zu lösen. Wie in Abbildung 16 dargestellt, finden Datenbanksprachen insbesondere in der Datenarchitektur, Anwendungsprogrammierung und Datenanalyse Verwendung.[37]

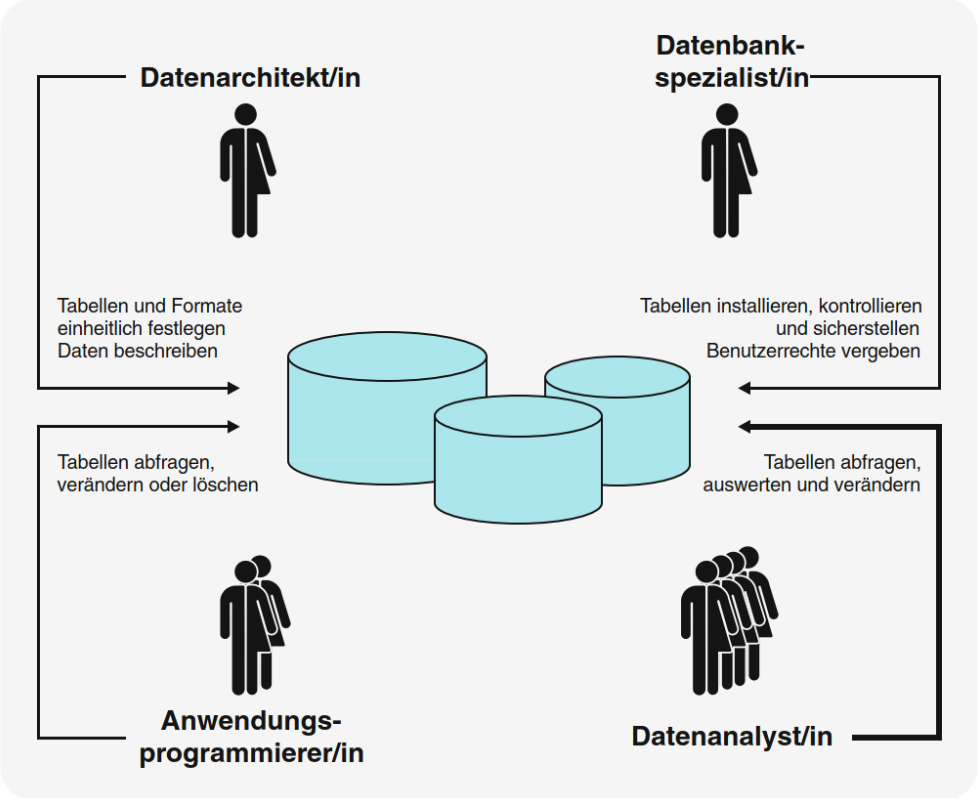

Abbildung 16: Verwendung einer Datenbanksprache (Beispiel SQL), Quelle: Meier/Kaufmann (2016) S. 92.

3.6.1 Relationenalgebra

Als theoretische Basis für relationale Datenbanken gilt die sogenannte Relationenalgebra. Algebraische Operatoren wirken auf Relationen beispielsweise zur mengenorientierten Vereinigung $\cup$, Durchschnittsberechnung $\cap$, Subtraktion $\setminus$ oder zur Berechnung des kartesischen Produkts $\times$. Die einheitlichen algebraischen Eigenschaften erlauben es, mehrere Operatoren zu kombinieren. Relationenorientierte Operatoren wie Projektion π, Selektion σ, Verbund $\bowtie$ und Division $\div$ ermöglichen die Reduktion der Relationen auf Teilmengen, das Verbinden von zwei Relationen, die Selektion einzelner Tupel oder die Generierung einer Teiltabelle durch Division zweier Relationen.[38]

[37] Vgl. Meier/Kaufmann (2016), S. 91 f.

[38] Vgl. Meier/Kaufmann (2016), S 93 f.

Abbildung 17 verdeutlicht die Unterschiede der mengenorientierten und relationenorientierten Operatoren.

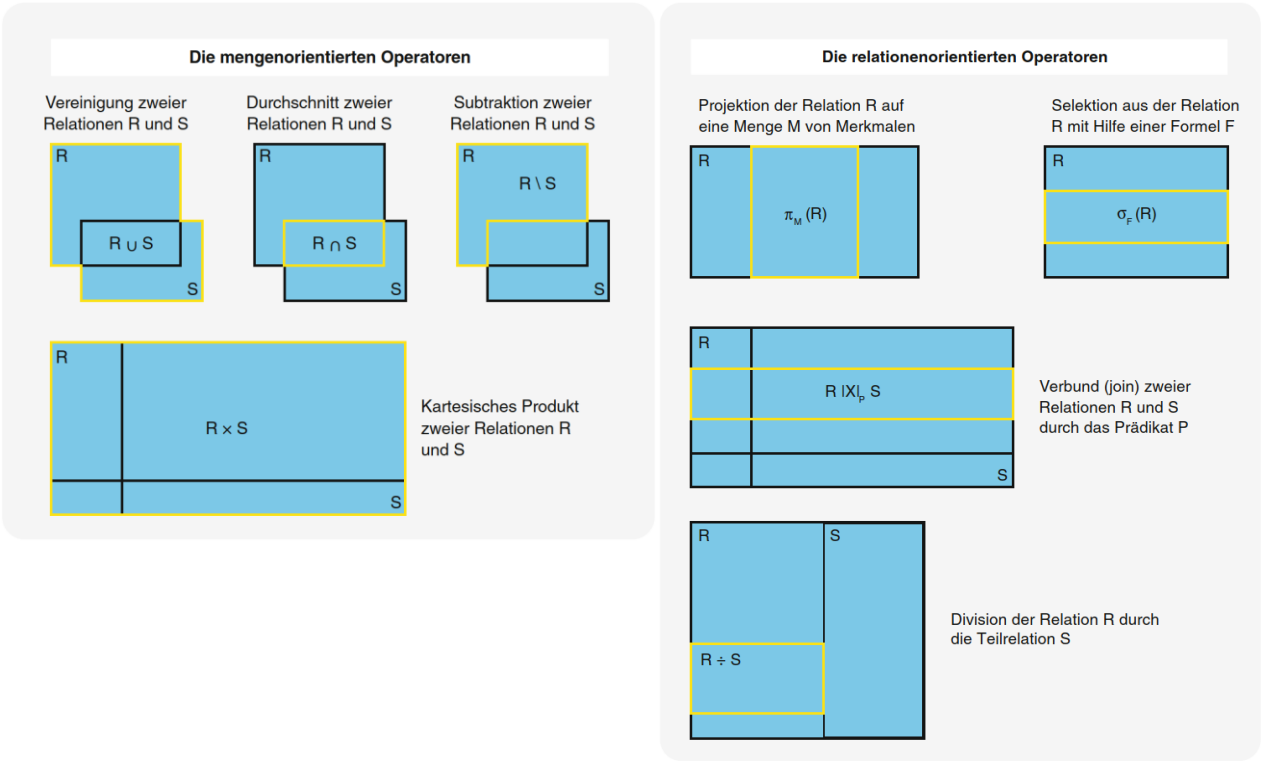

Abbildung 17: Beispiele mengenorientierter und relationenorientierter Operatoren, Quelle: Meier/Kaufmann (2016), S. 94 f.

3.6.2 SQL

Die „Structured Query Language“ (SQL) findet vor allem Verwendung in relationalen Datenbanken, um die Funktionen einer Datenbank zu definieren. Der Ursprung von SQL wurde im Jahr 1974 als sogenanntes „SEQUEL“ durch IBM entwickelt und weiterentwickelt.[39]

„SEQUEL“ wurde auf dem Prinzip generiert, Datenbankabfragen basierend auf englischen Wörtern wie z.B.: „SELECT“, „FROM“, „WHERE“, „COUNT”, “GROUP BY”, etc. zu ermöglichen und nicht mathematische Symbole verwenden zu müssen. Mit der Weiterentwicklung zu SQL wurde diese zur führenden Datenbanksprache und durch ANSI und in weiterer Folge ISO normiert. SQL gilt als relational vollständig, was bedeutet, dass alle Operationen der relationalen Algebra unterstützt werden. „SELECT“ kongruiert mit dem relationalen Operator Projektion π, „FROM“ führt notwendige Tabellen auf. „WHERE“ steht für den Selektionsoperator σ. Ein Beispiel für diese SQL-Operatoren zeigt der Code-Ausschnitt in Abbildung 18. SQL besitzt, zusätzlich zu den bereits genannten Relationenoperatoren, eingebaute Funktionen wie „COUNT“, „SUM“, „AVG“, „MAX“, und „MIN“, welche zur Zählung, Summenbildung, Mittelwertberechnung sowie der Berechnung des Maximal- oder Minimalwertes dienen. Diese Funktionen werden auch Aggregatfunktionen genannt. Zur Löschanweisung wird der Befehl „DELETE“ verwendet. Die genannten Beispiele stellen die unzähligen Möglichkeiten in SQL auf sehr komprimierte Weise dar.[40]

[39] Vgl. Hartmut/Jochen/Gerd (2016), S. 347.

[40] Vgl. Meier/Kaufmann (2016), S. 104 ff.

```
SELECT  M#,Name,Strasse,Ort,Unt,A#,Bezeichnung
FROM    MITARBEITER, ABTEILUNG
WHERE   Unt=A#
```

Abbildung 18: Beispiel für SQL Operatoren, Quelle: Meier/Kaufmann (2016), S. 106.

Mit der Einführung von SQL:1999 wurde die Möglichkeit geschaffen, Algorithmen in SQL auszudrücken. Es wird mittels Anweisungen festgelegt, was der Anwender beabsichtigt und wie ein Zugriff erfolgen soll. SQL ist somit eine deklarative und keine prozedurale Datenbanksprache und bietet den Vorteil, dass ein Programm kürzer, leichter lesbar und fehlerfreier wird als beispielsweise ein funktional vergleichbares Programm in der Programmiersprache C. Im Jahr 2011 wurde SQL durch objektorientierte Möglichkeiten erweitert und ermöglicht seither die Erstellung objektrelationaler Datenbanken.[41]

3.6.3 QBE

Eine weitere Form der relational vollständigen Sprachen ist „Query-by-Example“ (QBE). Hier werden Auswertungsvorstellungen der Benutzer interaktiv in Tabellen ausgeführt. Mit Hilfe des Operators „P.“ („print“) können alle Daten in einer Tabelle oder bestimmten Spalte angezeigt werden. QBE bietet die Möglichkeit weiterer Selektionen, indem die gewünschten Selektionsbedingungen in der jeweiligen Spalte befüllt werden. Dies kann mittels „AND“ oder „OR“ Verknüpfungen geschehen. Abbildung 19 zeigt ein Beispiel für eine QBE Tabelle mit „OR“ verknüpften Selektionsbedingungen. Wie in anderen Abfragesprachen gibt es hier auch die Option einen Verbund von Tabellen zu generieren, Informationen einzufügen oder Daten zu verändern oder zu löschen. Anders als bei SQL können bei QBE weder eine Tabelle definiert, noch Benutzerberechtigungen vergeben bzw. verwaltet werden. „Query-by-Example“ dient hauptsächlich der Bearbeitung und Abfrage von Tabelleninhalten.[42]

```
MITARBEITER | M# | Name | Strasse | Ort      | Unt
-------------------------------------------------
            |    | P.   |         |'Liestal'|
            |    | P.   |         |         |'A6'
```

Abbildung 19: Beispiel für eine QBE Tabelle mit „OR“ verknüpften Selektionsbedingungen, Quelle: Meier/Kaufmann (2016), S. 109.

3.6.4 Graphbasierte Sprachen

Graphabfragesprachen (engl. „graph query languages“) wurden insbesondere durch die immer größer werdenden Datenmengen der sozialen Medien entwickelt und arbeiten ebenso wie relationale Sprachen mit Mengen. Graphen werden in Graphdatenbanken mittels Graphstrukturen gespeichert und als eine Menge von Knoten, Kanten und Pfaden betrachtet. Auf diese Weise wird die Verarbeitung der Strukturen in Graphdatenbanken realisiert. Mittels eigener Sprachkonstrukte wird bei der Verarbeitung der Daten direkt auf Strukturen der Graphen zugegriffen. Graphbasierte Sprachen ermöglichen sowohl eine Filterung (engl. „conjunctive query“), welche als Ergebnis einen Teilgraphen ausgibt, als auch die Aufteilung der

[41] Vgl. Hartmut/Jochen/Gerd (2016), S. 347.

[42] Vgl. Meier/Kaufmann (2016), S. 108 ff.

Grapheninformationen in skalare Werte wie Anzahl, Summe oder Minimum. Diese Sprachen unterscheiden sich zu relationalen Systemen durch erweiterte Funktionsweisen wie einer Analyse der Pfade oder der Pfadmuster und einer Deklaration mit Standardbegriffen zur Suche entsprechender Datensätze in der Datenbank (engl. „regular path query“).[43]

Ein Beispiel einer graphbasierten Abfragesprache ist Cypher, welche mit Musterabgleichen (engl. „pattern matching“) operiert. Cypher beruht auf Datenmanipulation (engl. „data manipulation language“, DML) und Datendefinitionssprache (engl. „data definition language“, DDL). Die Grundkonstruktion besteht aus den drei Teilen „MATCH“, „WHERE“ und „RETURN“. „MATCH“ beschreibt die Suchmuster, mit „WHERE“ werden Bedingungen für die Filterung vorgegeben und „RETURN“ gibt an, welche Informationen ausgegeben werden sollen. Die zusätzliche Funktion „ORDER BY“ bietet die Möglichkeit der Sortierung von Ergebnissen. In Abbildung 20 ist ein Code-Beispiel für die Grundkonstruktion eines Cypher-Codes dargestellt. Cypher kennt Befehle für das Verbinden von Knoten und Kanten, Bilden von Teilmengen, Verändern oder Löschen von Datenwerten etc., die selbst komplizierte Datenbankabfragen ermöglichen.[44]

```
MATCH (p:Product)
WHERE p.unitPrice > 55
RETURN p.productName, p.unitPrice
ORDER BY p.unitPrice
```

Abbildung 20: Beispiel einer Abfrage in Cypher, Quelle: Meier/Kaufmann (2016), S. 114.

3.6.5 Eingebettete Sprachen

Relationale Manipulations- und Abfragesprachen werden oft in bestehende Programmierumgebungen eingebettet. Programme arbeiten mit ihrer eigentlichen Programmiersprache sowie relationalen Funktionen. Diese müssen beispielsweise die Option bieten, auf einzelne Tupel mittels „CURSOR“ (Zeiger) zugreifen zu können. Dieser Zeiger ermöglicht die Bearbeitung einzelner Tabellenzeilen und die Abarbeitung Tupel für Tupel. Durch die Verwendung dieses Cursorkonzeptes können Wirtssprachen relationale Datenbanksprachen in sich einbetten und Tabellenwerte auf einfache Weise z.B.: sortieren, bearbeiten und auswerten. SQL kann beispielsweise in datenbankinternen Funktionen sowie Prozeduren seit der Einführung des Standards SQL:1999 eingebettet werden. Umgesetzt wird dies auf dem Server mit Hilfe von gespeicherten Prozeduren (engl. „stored procedures“) und gespeicherten Funktionen (engl. „stored functions“). So existiert beispielsweise eine Schnittstelle zur Programmiersprache Java genannt „Java Database Connectivity“ (JDBC). Auch graphbasierte Sprachen können in diverse Wirtssprachen eingebettet werden. Hier kommt ebenfalls das Cursorkonzept zur Anwendung.[45]

[43] Vgl. Meier/Kaufmann (2016), S. 111 ff.

[44] Vgl. Meier/Kaufmann (2016), S. 113 ff.

[45] Vgl. Meier/Kaufmann (2016), S. 118 ff.

3.7 Modellierung einer Datenbank

Bevor mit der Erstellung einer Datenbank begonnen werden kann, muss diese modelliert werden (bedarfsorientierte Anwendungsentwicklung). Es sollten beteiligte Stakeholder wie Mitarbeiter und Abteilungsleiter einbezogen sowie betroffene Geschäftsprozesse analysiert und berücksichtigt werden. Es muss Klarheit über die unterschiedlichen Arten von Daten und Informationen herrschen und wie mit diesen umgegangen werden soll. In der Folge wird ein Datenbankmodell erstellt und realisiert sowie die notwendigen Daten erfasst und eine Ausgabe- und Dokumentationsmöglichkeit geschaffen. Schlussendlich gilt es das neue System zu testen – sowohl während der Entwicklungsphase als auch beim finalisierten Produktivsystem. Erfahrungen und notwendige Korrekturen werden im Zuge eines kontinuierlichen Verbesserungsprozesses (KVP) implementiert.[46]

3.7.1 Datenbanklebenszyklus

Ähnlich des Ablaufs bei der Software-Entwicklung, müssen beim Entwurf einer Datenbank mehrere Phasen durchlaufen werden. Ein Beispiel eines solchen Phasenmodells ist in Abbildung 21 ersichtlich.

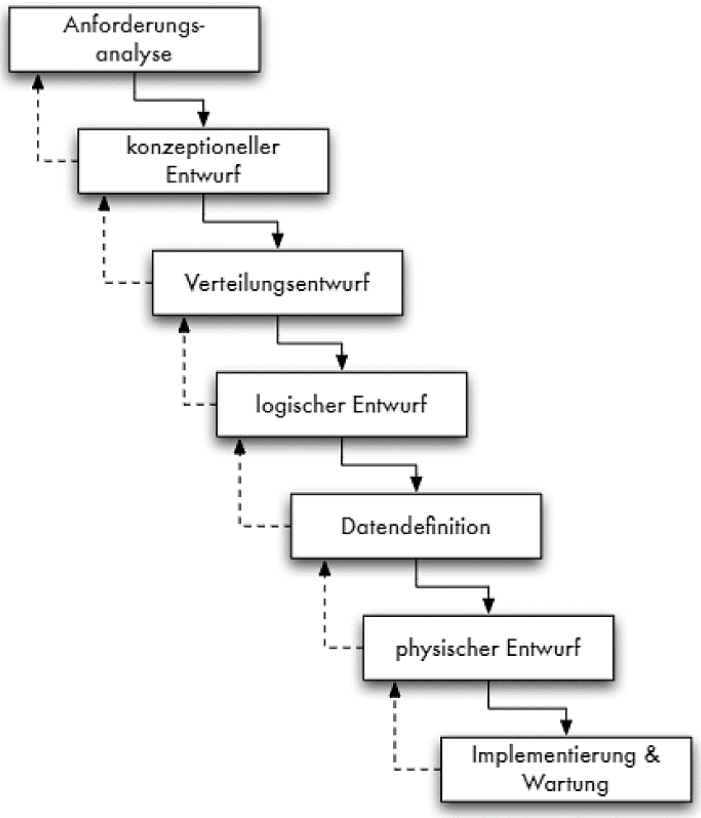

Abbildung 21: Phasenmodell eines Datenbank-Entwurfs, Quelle: Grunert/Heuer/Meyer/Saake/Sattler (2020), S. 61.

Im ersten Schritt müssen alle Anforderungen an die Funktionen und Daten der zukünftigen Datenbank evaluiert und analysiert werden. Auf Basis dieser Informationen wird ein konzeptioneller Entwurf der Datenbank erstellt. Das später verwendete System wird in diesem Schritt noch außer Acht gelassen. Im

[46] Vgl. Theis (2021), S. 31 f.

Falle einer verteilten Realisierung der Anwendung ist die Erstellung eines Verteilungsentwurfes sinnvoll. Stehen diese Schritte fest, geht man über zur detaillierten Generierung des logischen Entwurfes inklusive aller Prozeduren. In dieser Phase wird, unter Berücksichtigung der Erkenntnisse des konzeptionellen Entwurfes, das passende Datenbankmodell sowie das Schema gewählt und optimiert. Auf Basis des logischen Entwurfes wird möglichst präzise unter Verwendung der gewünschten Datenbanksprache das Schema für das zukünftige Datenbankmanagementsystem inklusive Benutzersichten definiert. Darauffolgend werden beim physischen Entwurf die erforderlichen Zugriffsstrukturen festgelegt und realisiert. In der letzten Phase wird die generierte Datenbankanwendung installiert und entsprechend einem kontinuierlichen Verbesserungsprozesses angepasst und gewartet. Dieses Phasenmodell eines Datenbank-Entwurfes ist, wie bei Entwicklungsprozessen üblich, nicht streng sequenziell zu betrachten. Revisionen und Schleifen erhöhen die Effizienz des gesamten Prozesses.[47]

3.7.2 Entitäten-Beziehungsmodell

Das Entitäten-Beziehungsmodell (engl. „entity-relationship model“, ER-Modell) kommt sowohl in der konzeptionellen Entwurfsphase als auch in der Phase der Implementierung zum Einsatz. Eine Entität (engl. „entity“) beschreibt ein individuelles reales oder abstraktes Objekt oder Ereignis und kann zu sogenannten Entitätsmengen zusammengefügt werden. Im ER-Modell werden die Beziehungen zwischen den Entitäten betrachtet und mit Hilfe von Assoziationen (engl. „association“) nach ihrer Bedeutung kategorisiert. Diese Assoziationen geben an, wie umfangreich eine Beziehung ist und treten immer paarweise auf. Diese können: „genau einer“, „keiner oder einer“, „einer oder mehreren“ oder „keiner, einer oder mehreren“ Entitäten zugeordnet sein. Ziel des Entitäten-Beziehungsmodells ist es, den realen Kontext bei der Datenmodellierung grafisch, wie beispielsweise in Abbildung 22 ersichtlich, darzustellen.[48]

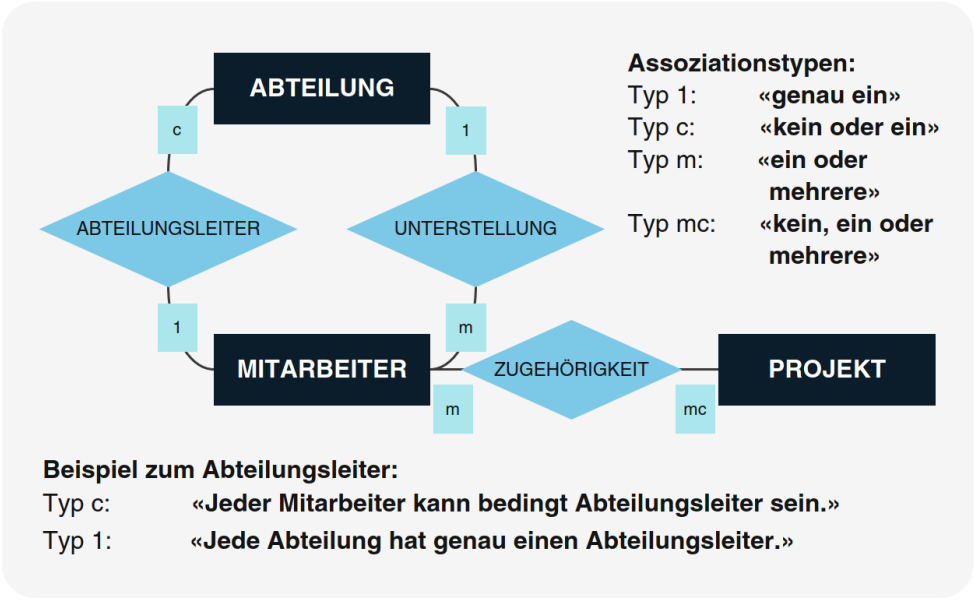

Abbildung 22: Entitäten-Beziehungsmodell mit Assoziationstypen, Quelle: Meier/Kaufmann (2016), S. 31.

[47] Vgl. Grunert/Heuer/Meyer/Saake/Sattler (2020), S. 58 ff.

[48] Vgl. Meier/Kaufmann (2016), S. 28 ff.

4 SOFTWARELÖSUNGEN ZUR DATENHALTUNG

Auf Basis der ermittelten Informationen wurde die Entscheidung für eine SQL-Lösung getroffen. Diese Form der relationalen Datenbanken ist für den gewünschten Zweck ausreichend und im Unternehmen konnte in der Vergangenheit, aufgrund einiger erfolgreich implementierter SQL-Projekte, bereits Know-how generiert werden.

4.1 Microsoft Lösungen

Microsoft Corporation bietet diverse Möglichkeiten Datenbanken zu implementieren. Die beiden bekanntesten und etabliertesten sind Microsoft SQL Server und Microsoft Access.

4.1.1 Microsoft SQL Server

Microsoft SQL Server zählt zu den führenden RDBMS weltweit. Die erste Version dieser relationalen Datenbank wurde im Jahr 1989 veröffentlicht. Es handelt sich hierbei um ein klassisches relationales DBMS mit vielen zusätzlichen Funktionen wie verbesserter Fehlerbehandlung, erweiterter Zeilenverarbeitung und Transaktionssteuerung. Die aktuellste Version ist Microsoft SQL Server 2019. Das Transaktionskonzept erfüllt alle ACID Eigenschaften. Microsoft SQL Server besitzt eine hohe Leistungsfähigkeit, eine gute Skalierbarkeit und ist sehr gut administrierbar. Unterschiedliche Redundanzmechanismen gewährleisten die Verfügbarkeit und es können ebenso „Big-Data“ und „Business-Intelligence-Projekte“ damit umgesetzt werden. Microsoft SQL Server bietet erweiterte Sicherheitsfunktionen und ist mit allen aktuellen SQL Versionen kompatibel. Es unterstützt In-Memory-Funktionen und ist sowohl für Windows, Linux als auch Container-Umgebungen geeignet.[49]

Microsoft bietet SQL Server in unterschiedlichen Versionen an und ihre starke Marktposition verspricht längerfristige Sicherheit. Neben einer kostenlosen, aber eingeschränkten Testversion, kann zwischen einer Standard Edition sowie einer Enterprise Edition gewählt werden. Hier gibt es unterschiedliche Lizenzmodelle und die Gebühren sind für die kommerzielle Verwendung teuer. Lizenziert wird nach Server oder Cores. Der Produktlebenszyklus beträgt 5 Jahre mit monatlichen bzw. zweimonatlichen kumulativen Updates.[50]

Mit Microsoft SQL Server 2019 können Daten aus MongoDB, Oracle- und Teradata-Datenbanken ausgelesen werden. Die transparente Datenverschlüsselung (engl. „transparent data encryption“, TDE) garantiert verbesserte Sicherheit und die intelligente Abfragenbearbeitung (engl. „intelligent query processing“, IQP) ermöglicht beschleunigte Abfragen.[51]

[49] Vgl. Luber/Ehneß (2021), Online-Quelle [10.11.2021].

[50] Vgl. Software-Express GmbH & Co. KG (2021), Online-Quelle [10.11.2021].

[51] Vgl. Joos/Ehneß (2019), Online-Quelle [10.11.2021].

Microsoft SQL Server unterstützt die Programmiersprachen:[52]

- C++
- C#
- Delphi
- Go
- Java
- JavaScript
- PHP
- Python
- R
- Ruby
- Visual Basic

4.1.2 Microsoft Access

Der wesentlichste Vorteil von Microsoft Access ist, dass es sich um einen Bestandteil des Office Paketes handelt. Aufgrund der Tatsache, dass es eine Microsoft Applikation ist, sind viele der Funktionen selbsterklärend und kleinere Datenbankprojekte können einfach und unkompliziert erstellt werden. Online können zudem viele Vorlagen und Hilfen abgerufen werden. Microsoft Access kombiniert Backend- und Frontend-Funktionen in einem Programm und es können sowohl PC als auch Browser Anwendungen umgesetzt werden. Darüber hinaus ist die Verwendung mehrerer Datenquellen möglich und bestehende Access Daten können in Microsoft SQL Server integriert werden. Das Transaktionskonzept entspricht den ACID Prinzipien.

Größter Nachteil von Microsoft Access ist das fehlende Benutzerberechtigungskonzept. Access kann nur auf Windows-Rechnern verwendet werden und es enthält eine relativ starre Datenbankstruktur. Komplexe Aufgabenstellungen sind schwer umsetzbar. Je größer die Datenbank wird, umso langsamer und instabiler wird das System. Aus diesem Grund ist Microsoft Access nur für kleinere relationale Datenbankprojekte geeignet.

Microsoft Access unterstützt folgende Programmiersprachen:[53]

- C
- C++
- C#
- Delphi
- Java

[52] Vgl. solid IT GmbH (1) (2021), Online-Quelle [10.11.2021].

[53] Vgl. solid IT GmbH (2) (2021), Online-Quelle [10.11.2021].

- VBA
- Visual Basic.NET

4.2 Cloud und Open Source Lösungen

Cloud-Datenbanken kennzeichnen sich durch die Speicherung der Daten in einem Onlinespeicher („Cloud“). Die vier größten Anbieter für Cloud-Datenbanken sind Microsoft Corporation, Oracle Corporation, Amazon Web Services und Google LLC.

Unterschieden wird zwischen zwei Cloud-Datenbank Modellen. Traditionelle Cloud-Datenbank Modelle speichern Daten auf einer virtuellen Maschine auf einer internen Infrastruktur. Datenbankverwaltung und -überwachung werden durch Mitarbeiter der Organisation durchgeführt. Beim „Database as a Service“ (DBaaS) Modell handelt es sich um ein gebührenpflichtiges Abonnement, bei dem die Infrastruktur eines Serviceanbieters genutzt wird. Klassischerweise verwaltet der Kunde bzw. die Kundin die Inhalte der Datenbank und nutzt dabei die Infrastruktur und die Datenbank des Cloud-Computing-Anbieters.[54]

Als Open Source werden Programme bezeichnet, deren Source- bzw. Quell-Code öffentlich und kostenlos zur Verfügung gestellt wird. Es darf ohne Einschränkungen weitergegeben und verändert werden. Häufig gibt es allerdings, vor allem im Enterprise-Bereich, Skepsis bezüglich der Sicherheit von Open Source Programmen.[55]

Es wird eine Vielzahl an unterschiedlichen Cloud- und Open Source Lösungen angeboten. Betrachtet werden im Folgenden die gebräuchlichsten SQL-Datenbanken.

4.2.1 MySQL

MySQL wurde im Jahr 1995 von der Oracle Corporation entwickelt und ist eine Open Source Datenbank. Es entspricht hinsichtlich des Transaktionskonzeptes den ACID Kriterien und wird häufig für Webanwendungen verwendet. MySQL kann auf über 20 Plattformen verwendet werden, darunter unter anderem Microsoft Windows, Linux, Apple macOS und UNIX. Die Struktur entspricht einem Client-Server-Modell was bedeutet, zentraler Kern ist der Server, dieser verarbeitet die Datenbankbefehle, während der Client am Rechner die Befehle an den Server sendet. Unterstützt werden große Datenmengen und viele Datentypen. MySQL ermöglicht eine Benutzerverwaltung mit verschlüsselten Kennwörtern und bietet diverse Analysetools.[56]

[54] Vgl. Sheldon/Shore (2021), Online-Quelle [10.11.2021].

[55] Vgl. Tech Target (2014), Online-Quelle [10.11.2021].

[56] Vgl. Moore (2020), Online-Quelle [10.11.2021].

Unterstützt werden die folgenden Programmiersprachen:[57]

- Ada
- C
- C++
- C#
- D
- Delphi
- Eiffel
- Erlang
- Haskell
- Java
- JavaScript
- Objective-C
- OCaml
- Perl
- PHP
- Python
- Ruby
- Scheme
- Tcl

4.2.2 PostgreSQL

Die Open Source Datenbank PostgreSQL wurde im Jahr 1989 durch die PostgreSQL Global Development Group entwickelt. Sie ist eine objektrelationale Datenbank und findet hauptsächlich Verwendung für dynamische Webseiten. PostgreSQL entspricht ebenfalls den ACID Kriterien und besitzt eine Server-Client-Architektur. Diese Datenbank kann plattformunabhängig verwendet werden, bietet hohe Sicherheit sowie viele Schnittstellen und ist sehr anpassungsfähig. Eine Benutzerverwaltung mit Berechtigungskonzept ist mit PostgreSQL möglich. Als nachteilig ist der allgemein erhöhte Aufwand im Vergleich zu anderen Datenbankmanagementsystemen zu betrachten und nachträgliche Designänderungen gestalten sich umständlich.[58]

PostgreSQL unterstützt die Programmiersprachen:[59]

- C
- C++

[57] Vgl. solid IT GmbH (3) (2021), Online-Quelle [11.11.2021].

[58] Vgl. Datenbanken-verstehen.de (2021), Online-Quelle [11.11.2021].

[59] Vgl. solid IT GmbH (4) (2021), Online-Quelle [11.11.2011].

- Delphi
- Java
- JavaScript
- .Net
- Perl
- PHP
- Python
- Tcl

4.2.3 MariaDB

MariaDB entstand 2009 als Abspaltung von MySQL und ist mittlerweile ein eigenständiges Open-Source-Datenbanksystem der MariaDB Corporation. Der Fokus liegt auf einer offenen Entwicklung. Die vollständige Kompatibilität zur Datenbanksprache SQL ist gegeben und MariaDB bietet eine große Auswahl an kompatiblen Datenbank-Engines, welche zur Erstellung, Bearbeitung und Lösung von Daten in der Datenbank dienen. MariaDB gewährleistet volle ACID Konformität und ist hochverfügbar. Inaktive Daten können verschlüsselt werden und die Benutzerverwaltung erfolgt mit rollenbasierter Zugriffskontrolle. Diese Datenbank ist kompatibel mit diversen Betriebssystemen wie Microsoft Windows, Apple macOS, Linux, Ubuntu etc. Große Web-Services wie Google, Mozilla, Wikimedia Foundation, TeamSpeak und XAMPP setzen auf MariaDB.[60]

Die in MariaDB unterstützten Programmiersprachen sind:[61]

- Ada
- C
- C++
- C#
- D
- Eiffel
- Erlang
- Go
- Haskell
- Java
- JavaScript
- Objective-C
- OCaml
- Perl

[60] Vgl. Ionos SE (2020), Online-Quelle [11.11.2021].

[61] Vgl. solid IT GmbH (5) (2021), Online-Quelle [11.11.2021].

- PHP
- Python
- Ruby
- Scheme
- Tcl

4.2.4 Oracle Database

Die Oracle Corporation ist einer der Marktführer im Bereich der kommerziellen Datenbanken und die aktuellen Versionen der Oracle Database sind 19c und 21c. C steht hierbei für Cloud was bedeutet, dass kein eigener Speicher notwendig ist. Die Administration ist einfach und die Datenbank ist hochverfügbar, verschlüsselt und skalierbar. Oracle Database erfüllt die ACID Anforderungen und es können sowohl relationale als auch objektrelationale Daten verarbeitet werden. Anwendbar ist sie unter anderem auf den Betriebssystemen Microsoft Windows, Linux, Solaris, Unix und Apple macOS. Eine eingeschränkte kostenlose Version ist verfügbar, für umfangreichere Anwendungen ist jedoch eine lizenzierte Version notwendig, welche mit hohen Gebühren verbunden ist. Lizenziert wird bei Oracle Database nach Benutzern oder Prozeduren und die Gebühren sind höher als bei Microsoft SQL Server. Diese Datenbank verwendet eine prozedural erweiterte Version von SQL die „Procedural Language / Structured Query Language“ (PL/SQL). Eine Benutzerverwaltung ist umsetzbar inklusive einer möglichen Einbindung von „Active Directory“ zur Benutzerauthentifizierung. Oracle Database bietet zudem eine sogenannte „autonome Datenbank“, welche Automatismen zur Sicherung und Reparatur besitzt.[62]

Es wird eine Vielzahl an Programmiersprachen unterstützt:[63]

- C
- C++
- C#
- Clojure
- Cobol
- Delphi
- Eiffel
- Erlang
- Fortran
- Groovy
- Haskell
- Java
- JavaScript
- Lisp

[62] Vgl. Oracle Corporation (2019), Online-Quelle [11.11.2021].

[63] Vgl. solid IT GmbH (6) (2021), Online-Quelle [11.11.2021].

- Objective-C
- OCaml
- Perl
- PHP
- PL/SQL
- Python
- R
- Ruby
- Scala
- Tcl
- Visual Basic

4.2.5 IBM Db2

Ursprünglich 1983 von der IBM Corporation für ihr eigenes IBM Mainframe System entwickelt, wurde Db2 später für weitere Betriebssysteme wie Linux, Unix und Microsoft Windows erweitert. IBM bietet eine kostenlose, eingeschränkte Version von Db2, kommerzielle Versionen sind jedoch kostenpflichtig und es erfolgt eine Lizenzierung auf Basis Processor Value Unit (PVU), das bedeutet die Lizenzgebühren fallen pro Kern an. Diese Datenbank bietet eine effiziente Speicherung und Analyse von Daten und ist hochverfügbar und verschlüsselt. Es werden objektorientierte Funktionen, numerische Datentypen sowie XML und JSON unterstützt. Db2 entspricht den ACID Kriterien und bietet eine administrierbare Benutzerverwaltung.[64]

Die folgenden Programmiersprachen können in Db2 angewendet werden:[65]

- C
- C++
- C#
- Cobol
- Delphi
- Fortran
- Java
- Perl
- PHP
- Python
- Ruby
- Visual Basic

[64] Vgl. Craig S. Mullins (2017), Online-Quelle [11.11.2021].

[65] Vgl. solid IT GmbH (7) (2021), Online-Quelle [11.11.2021].

4.2.6 SQLite

Bei SQLite handelt es sich um eine prozessinterne Bibliothek, welche eine eigenständige, serverlose Open Source SQL-Datenbank-Engine implementiert. Es wurde 2000 entwickelt und besitzt keinen separaten Serverprozess, sondern liest und schreibt direkt auf normale Festplattendateien. SQLite ist eine Einzeldatei-Datenbank, was bedeutet, dass sich der gesamte Quellcode in einer Datei befindet. Sie enthält eine Bibliothek mit Standard-SQL-Funktionen und funktioniert plattformübergreifend. Je mehr Speicher SQLite zur Verfügung gestellt wird, umso schneller funktioniert sie. SQLite-Datenbanken sind sehr zuverlässig und erfüllen das ACID Transaktionskonzept. Größter Nachteil ist die fehlende Möglichkeit der Benutzerverwaltung.[66]

SQLite unterstützt die Programmiersprachen:[67]

- Actionscript
- Ada
- Basic
- C
- C++
- C#D
- Delphi
- Forth
- Fortran
- Haskell
- Java
- JavaScript
- Lisp
- Lua
- MatLab
- Objective-C
- OCaml
- Perl
- PHP
- PL/SQL
- Python
- R
- Ruby
- Scala

[66] Vgl. SQLite Consortium (2021), Online-Quelle [11.11.2021].

[67] Vgl. solid IT GmbH (8) (2021), Online-Quelle [11.11.2021].

- Scheme
- Smalltalk
- Tcl

4.2.7 Amazon Aurora

Vom kommerziellen System Aurora von Amazon Web Services gibt es keine kostenlose Version. Die Kosten fallen je GB Speicherverbrauch bzw. je Anzahl an Eingabe/Ausgabe-Anforderungen an. Wird nicht der richtige Anwendungsfall gewählt, ist dieses System sehr teuer. Amazon Aurora ist mit MySQL und PostgreSQL kompatibel und eine Migration dieser Daten ist möglich. Amazon Web Services bietet eine hohe Sicherheit, Zuverlässigkeit sowie Verfügbarkeit gemäß ACID Konzept. Aurora verfügt über einen hohen Verschlüsselungsgrad und automatisierte Sicherungen, die Verwaltung von Hardware sowie einer einfachen Datenbankeinrichtung. Das Patching wird durch Amazon Web Services übernommen. Ein Berechtigungskonzept für die Benutzerverwaltung kann angewendet werden.[68]

Die folgenden Programmiersprachen werden in Amazon Aurora unterstützt:[69]

- Ada
- C
- C++
- C#
- D
- Delphi
- Eiffel
- Erlang
- Haskell
- Java
- JavaScript
- Objective-C
- OCaml
- Perl
- PHP
- Python
- Ruby
- Scheme
- Tcl

[68] Vgl. Amazon Web Services Inc. (2021), Online-Quelle [11.11.2021].

[69] Vgl. solid IT GmbH (9) (2021), Online-Quelle [11.11.2021].

4.2.8 Google Cloud Spanner

Google LLC bietet mehr als ein Dutzend Datenbanken an, eine davon ist Cloud Spanner. Von dieser kommerziellen Datenbank ist keine kostenlose Version verfügbar. Entwickelt wurde Cloud Spanner, um die SQL-Semantik, Online-Transaktionsverarbeitung (engl. „Online Transaction Processing“, OLTP), transaktionale Konsistenz und hochverfügbare horizontale Skalierung zu unterstützen. Diese Datenbank kombiniert SQL und NoSQL Eigenschaften und wird aus diesem Grund als „NewSQL“ Datenbank klassifiziert.[70]

Strikte transaktionale Konsistenz ist eines der wesentlichen Eigenschaften des ACID Transaktionskonzeptes in Cloud Spanner. Es bietet eine unbegrenzte Skalierung und eine automatische Fragmentierung der Daten auf Basis der Datengröße sowie der Anfragelast. Diese Datenbank ist hochverfügbar, verschlüsselt und bietet sehr gute Sicherungs- und Wiederherstellungsmöglichkeiten. Die Daten können in Echtzeit erfasst werden und der Zugriff mit Hilfe von Benutzerverwaltung kontrolliert werden.[71]

Google Cloud Spanner unterstützt die Programmiersprachen:[72]

- C++
- C#
- Go
- Java
- Node.js
- PHP
- Python
- Ruby

4.3 Auswahl der Datenbank

Zur zukünftigen Umsetzung der Datenbank im Unternehmen fällt die Wahl auf Microsoft SQL Server bzw. alternativ auf die Open Source Lösungen MySQL oder MariaDB. Diese erfüllen alle erforderlichen Anforderungen des Unternehmens an die Datenbanklösung, sowie die ACID Transaktionskriterien und es ist bereits Know-how im Umgang mit diesen Lösungen vorhanden.

[70] Vgl. Redaktion ComputerWeekly.de (2021), Online-Quelle [11.11.2021],

[71] Vgl. Google (2021), Online-Quelle [11.11.2021].

[72] Vgl. Google (2021), Online-Quelle [11.11.2021].

4.4 Vor- und Nachteile von Microsoft Excel Lösungen

Bei Microsoft Excel handelt es sich um ein Kalkulationsprogramm und es ist vom Grundprinzip nicht als Datenbank geschaffen worden. Diese Tatsache hat maßgeblich Einfluss auf seine Anwendungsmöglichkeiten und seine Grenzen.

Die wesentlichen Vorteile in der Anwendung von Microsoft Excel für die Assetdaten im Life Cycle Management liegen in der Tatsache, dass sämtliche bisherigen Daten bereits in Microsoft Excel erfasst wurden. Die Bedienung ist einfach und für die jeweiligen Benutzer ist keine Umstellung notwendig, da es sich bei Microsoft Excel um eine weit verbreitete und im Unternehmen etablierte Software handelt. Zudem kommen auf das Unternehmen keine zusätzlichen Lizenzgebühren zu, da es Bestandteil des bestehenden Office 365 Pakets ist. Microsoft Excel bietet gute grafische Auswertungsmöglichkeiten und bei geringen Datenmengen ist es in jedem Fall ausreichend.

Signifikante Nachteile in der Anwendung von Microsoft Excel als Datenbankersatz liegen vor allem in der schwierigen Berechtigungsvergabe und der damit verbundenen fehlenden Nachvollziehbarkeit von Änderungen. Anwendenden können lediglich Berechtigungen zum Lesen oder Ändern nur für das gesamte Dokument vergeben werden, nicht jedoch für einzelne Abschnitte oder einzelne Daten innerhalb des Dokumentes. Datenbanken hingegen ermöglichen einzelnen Benutzergruppen unterschiedliche Zugriffsmöglichkeiten sowie eine exakte Nachvollziehbarkeit von wem welche Änderungen vorgenommen wurden. Ein paralleles Bearbeiten desselben Dokumentes von mehreren Benutzer*innen ist in Microsoft Excel unmöglich und schränkt das Unternehmen somit in der Arbeitsweise maßgeblich ein.

Im Gegensatz zu Datenbanken gerät Microsoft Excel sehr schnell an seine Belastungsgrenzen und je höher die zu verarbeitenden Datenmengen sind, desto größer ist die Wahrscheinlichkeit für Performanceprobleme bis hin zum Absturz des gesamten Programmes. Die Verknüpfung von unterschiedlichen Datenquellen ist kaum möglich und Schnittstellen zu anderen Applikationen wie beispielsweise einem Enterprise Ressource Planning (ERP) System sind schwierig bis gar nicht umsetzbar. Komplexe Abfragen und Auswertungen sind nicht möglich. Sicherungen und Wiederherstellungen der Daten sind nur in geringem Ausmaß durchführbar.

5 MÖGLICHKEITEN ZUR VISUALISIERUNG VON DATEN

Ein „User Interface“ (UI) ist die visuelle Schnittstelle eines Produktes und dient als Bindeglied zwischen den Anwender*innen und der Funktionalität eines Produktes. Das Frontend ist die Benutzeroberfläche, welche die Möglichkeit bietet, auf die im Backend gespeicherten Daten zuzugreifen. Dies geschieht durch eine Reihe von Mensch-Maschine-Interaktionen. Eine Trennung zwischen Frontend und Backend ist empfohlen, um beispielsweise Änderungen am Frontend durchführen zu können, ohne dabei auf das Backend zugreifen zu müssen. Es handelt sich um eine Reihe von Texten, Formen und Grafiken, welche durch ihre Kombination eine flüssige und natürliche Interaktion ermöglichen.[73]

Das folgende Fischgrätendiagramm in Abbildung 23 stellt die Korrelation zwischen Anwender*in und User Interface grafisch dar. Kommunikation und Interaktion zwischen Mensch und Maschine müssen aufeinander abgestimmt werden. Die Darstellung des User Interfaces wird durch die anwendende Person wahrgenommen und interpretiert. Je besser also die Darstellung der Applikation und je einfacher und selbstverständlicher die Kontrollelemente, desto besser wird das User Interface verstanden und als praktikabel bewertet.[74]

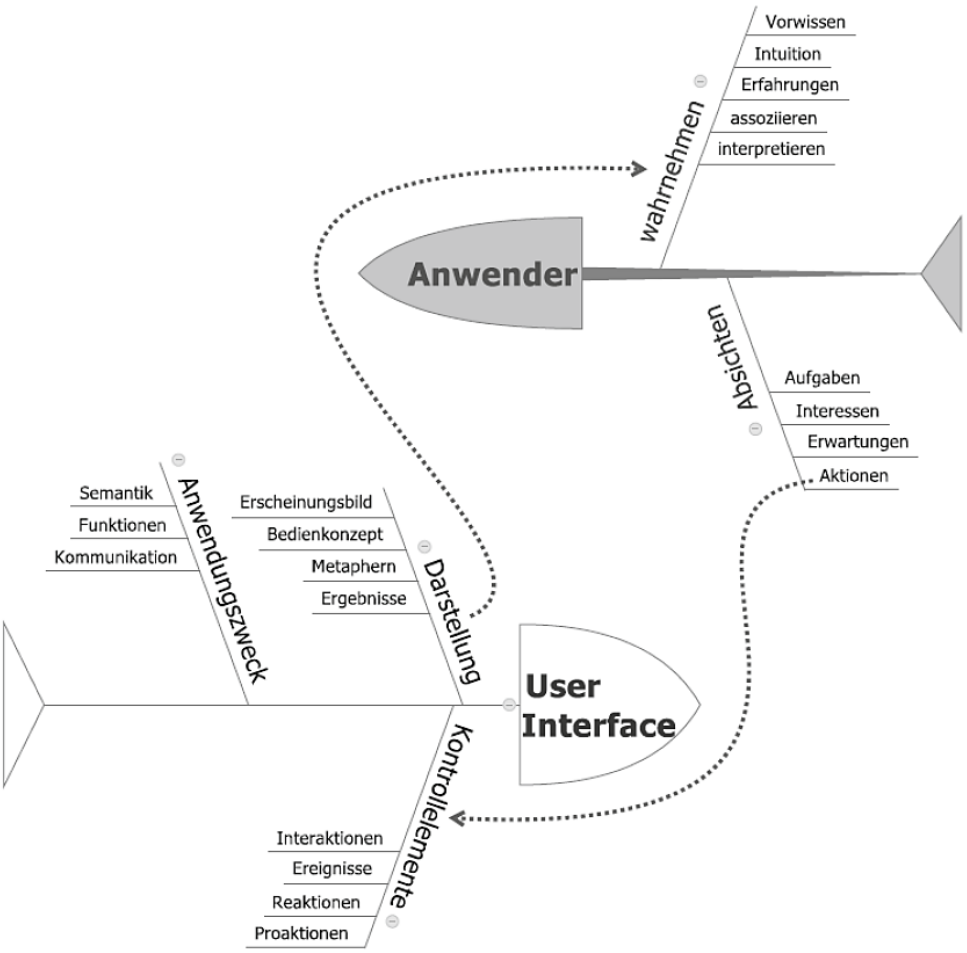

Abbildung 23: Fischgrätendiagramm Korrelation zwischen Anwender*in und User Interface, Quelle: Chlebek (2011), S. 29.

[73] Vgl. Malewicz/Malewicz (2020), S. 16.

[74] Vgl. Chlebek (2011), S. 29.

5.1 Integrierte Frontends

Während das Backend der Datenbank sämtliche Datentabellen und Prozeduren enthält, werden im Frontend sämtliche Datenbankobjekte verarbeitet wie Formulare, Masken, Berichte und Abfragen und die Prozesse und Daten des Backend vereinfacht und abstrahiert. Das Frontend dient somit der Visualisierung als grafische Benutzeroberfläche und dem benutzerfreundlichen Zugriff.[75]

Microsoft Access ist eine der Datenbanken, die ein integriertes Frontend enthält. Es bietet die Möglichkeit nicht nur die Datenbank selbst zu erstellen, sondern in wenigen Schritten auch das dazugehörige Frontend mit allen notwendigen Verknüpfungen zur Datenbank. In Abbildung 24 ist exemplarisch die Erstellung einer Benutzeroberfläche in Microsoft Access dargestellt.

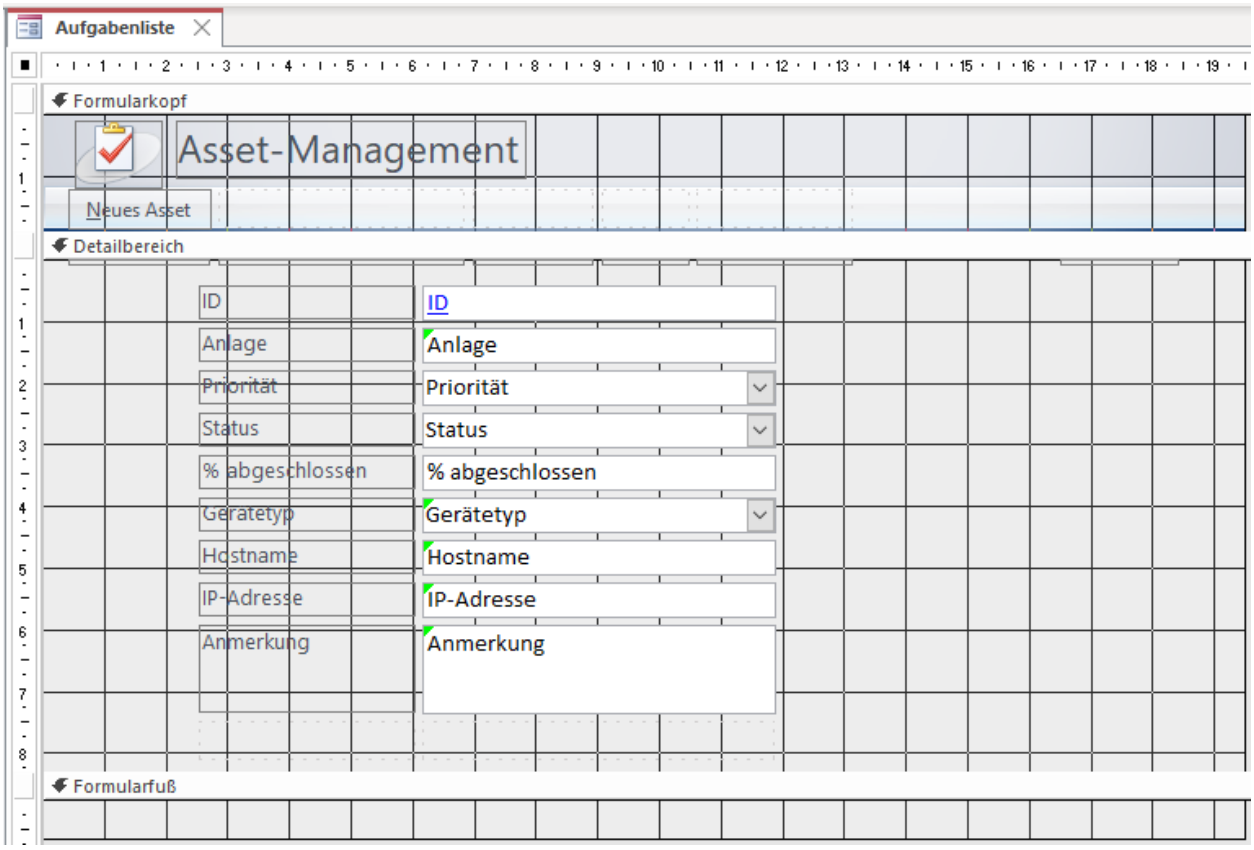

Abbildung 24: Erstellung eines Frontend in Microsoft Access, Quelle: Eigene Darstellung.

[75] Vgl. Geißler/Ostler (2018), Online-Quelle [12.11.2011].

5.2 Plattformübergreifende Lösungen

Ein großer Teil der angebotenen Datenbanken verteilt das Backend und das Frontend mit Hilfe eines Client-Server-Modells. Das bedeutet, dass der Client auf die bereitgestellten Dienste eines Servers zugreift. Der Ablauf der Kommunikation zwischen Client und Server wird mit Hilfe von Protokollen geregelt. Ein Server kann hierbei einen oder mehrere Clients bedienen und sämtliche Aufgaben und Funktionen sind unabhängig von einer physischen Hardware. Angestoßen wird die Interaktion durch den Client. Die wesentlichen Vorteile eines Client-Server-Modells sind die zentrale Datenhaltung, die zentrale Verwaltung von Services, Ressourcen und der Zugriffsberechtigungen und die standortunabhängige Bereitstellung des Servers, der mehrere Clients bedienen kann. Als nachteilig sind die Notwendigkeit von ausreichender Bandbreite und das Erfordernis von Redundanzen bei kritischen Diensten sowie der erhöhte finanzielle sowie zeitliche Aufwand beim Betrieb eines Servers zu betrachten.[76]

Die für das Unternehmen gewählte Datenbanken Microsoft SQL Server, MySQL oder MariaDB werden in Kombination mit einem Webserver wie beispielsweise „Apache HTTP Server“ die Daten zwischen dem Server und den Clients wie „MySQL Workbench“ und „phpMyAdmin“ mittels PHP-Script und JSON-Datenformat austauschen.

[76] Vgl. Luber/Donner (2020), Online-Quelle [12.11.2021].

5.3 Anforderungen an das Frontend

Zur Analyse der funktionalen Anforderungen an die Datenbanklösung wurde eine Mindmap (Abbildung 25) erstellt, um einen Überblick über die gewünschten Funktionsweisen zu erhalten. Ziel dieser Baumstruktur ist es, die geforderten Funktionen aller Stakeholder zu evaluieren, konkretisieren und ihre Umsetzung zu ermöglichen.

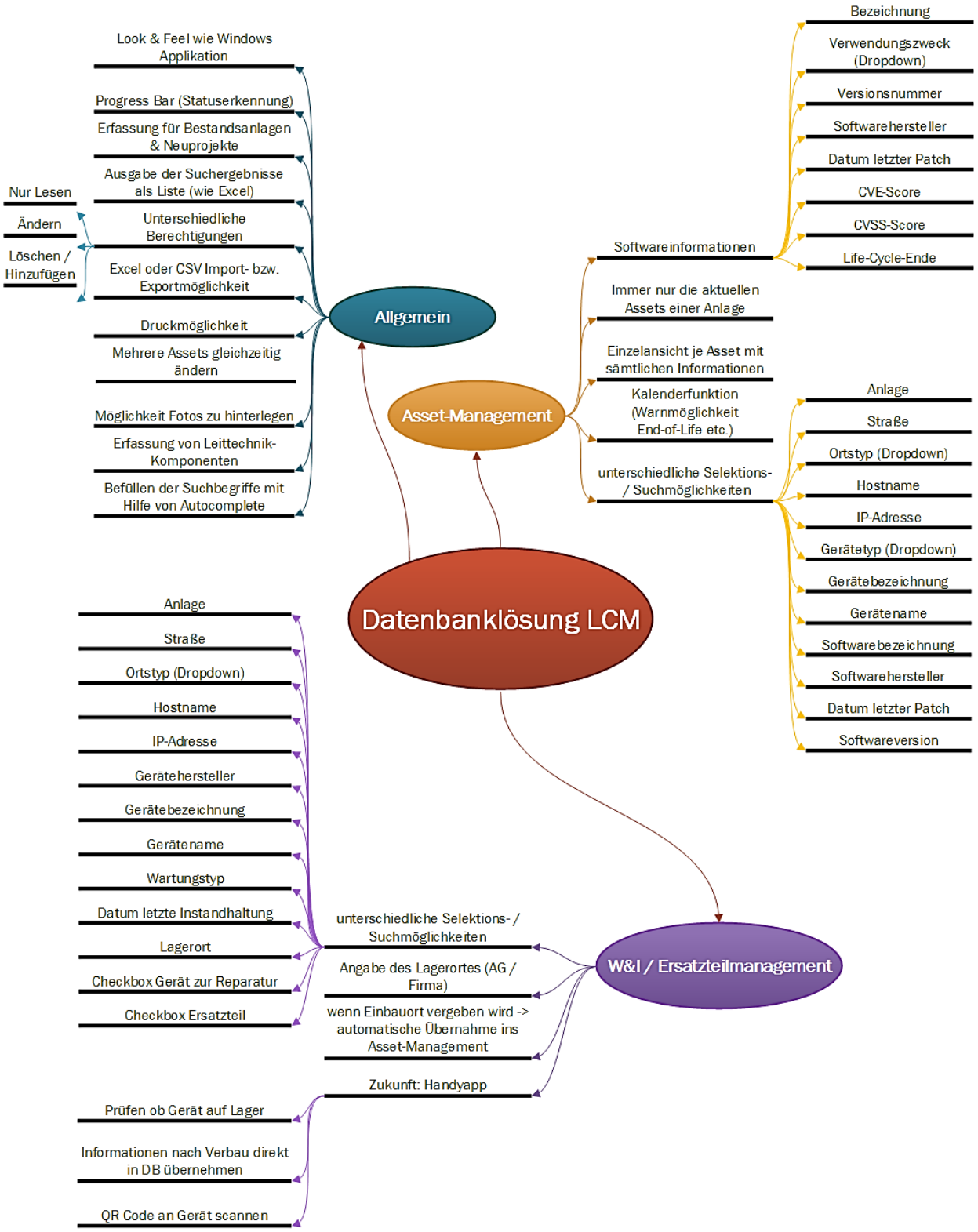

Abbildung 25: Mindmap Anforderungen an die Datenbanklösung, Quelle: Eigene Darstellung.

5.3.1 Anwendungssituation

Das Tool soll in erster Linie von den zuständigen Personen in der Leittechnikabteilung des Unternehmens genutzt werden. Sobald Teile für ein Projekt den Bestellprozess durchlaufen, sollen diese Assets bereits in der Datenbank erfasst werden. Wenn in weiterer Folge feststeht, wo die Komponenten verbaut werden sollen und welche IP-Adresse sie erhalten, werden diese und weitere Informationen sukzessive hinzugefügt und so der Datensatz vervollständigt. Es soll bereits beginnend mit der Projektphase ein Überblick über alle verbauten Leittechnik Assets gegeben sein. Im Zuge des Life Cycle Managements wird dann mit Hilfe der Datenbanklösung sicherheitsrelevanter Handlungsbedarf rasch erkannt und umgesetzt. Sie dient der Bedarfserkennung und Dokumentation bei sicherheitsrelevanten Updates und wenn Komponenten ihr End-of-Life erreicht haben.

Darüber hinaus soll die Datenbank zur Unterstützung von Service-, Wartungs- und Instandhaltungstätigkeiten von den zuständigen Mitarbeiter*innen eingesetzt werden. Es werden notwendige Informationen wie beispielsweise Gewährleistungszeiträume oder der Komponentenstatus (z.B. zur Reparatur beim Lieferanten, repariert, ausgetauscht) hinterlegt. Diese Informationen können zum Beispiel zur Nachverfolgung von ausständigen Reparaturen ausgewertet werden. Alle zur Verfügung stehenden Ersatzteile mit unterschiedlichen Lagerorten sind in der Datenbank ersichtlich und die Verfügbarkeit im Falle einer Störung oder eines Defektes der Komponente kann unmittelbar überprüft werden.

5.3.2 Anforderungen an die Bedienerführung und die Benutzerschnittstelle

Das gesamte „Look and Feel“ der Datenbanklösung soll an Microsoft Lösungen erinnern, um allen Anwendenden den Einstieg in das Tool zu erleichtern. Die Datenbanklösung soll sämtliche gespeicherten Assetinformationen gemäß Suchanfrage ausgeben. Sie bietet die Möglichkeit nach bestimmten Assets zu suchen, bestimmte Anlagenorte zu selektieren oder eine Liste gemäß bestimmter Softwareinformationen auszugeben. So soll zum Beispiel die Möglichkeit gegeben sein auszuwerten, welche Assets auf einer Anlage verbaut sind oder, wenn das End-of-Life einer Softwarekomponente angekündigt wird, sehr rasch feststellen zu können, auf welchen Anlagen diese Software zum Einsatz kommt und wo Handlungsbedarf besteht. Die Daten sollen in einem Tabellenformat ausgegeben werden und es soll direkt die Möglichkeit gegeben sein, einzelne oder mehrere Datensätze zu bearbeiten. Diese Funktion ermöglicht es, beispielsweise das Patchdatum oder den Komponententausch für eine gesamte Anlage in einem Arbeitsschritt zu bearbeiten und so die betroffenen Datensätze in wenigen Minuten zu aktualisieren.

Folgende Suchmöglichkeiten sollen gegeben sein:

- Was ist auf einer bestimmten Anlage bzw. Straße verbaut?
- Welcher Gerätetyp ist in welchen Anlagen verbaut?
- Ein bestimmter Hostname oder eine bestimmte IP-Adresse
- Auf welchen Komponenten ist die Software eines bestimmten Herstellers installiert?

- Auf welchen Komponenten ist eine bestimmte Software- bzw. Firmwareversion installiert?
- Welche Komponenten wurden zu einem bestimmten Zeitpunkt gepatcht?

Für jedes Asset steht eine detaillierte Übersicht zur Verfügung, in welcher überschaubar alle Asset-Daten angezeigt werden inklusive einer Progress-Bar, die den aktuellen Status des Assets aufzeigt.

Ausgegeben wird:

- Progress-Bar mit dem Status der Komponente
    - bestellt (20 %)
    - verbucht (40 %)
    - geliefert (60 %)
    - einbaufertig (80 %)
    - verbaut (100 %)
- Anlagenname
- Straße
- Kilometer
- Richtungsfahrbahn
- Ortstyp
- Geschoß
- Raumname
- Schranknummer
- Hostname
- IP-Adresse
- Gerätehersteller
- Gerätebezeichnung
- Gerätename
- Gerätetyp
- Redundanz zu
- Seriennummer
- Priorität
- ISMS-Dokumentation vorhanden
- Inbetriebnahmedatum
- Lieferdatum der Komponente im Unternehmen
- Lieferdatum der Komponente beim auftraggebenden Unternehmen (Beginn der Gewährleistung)
- Ende der Gewährleistung beim liefernden Unternehmen
- Ende der Gewährleistung beim auftraggebenden Unternehmen
- Datenblätter zur Komponente
- Interne Anmerkungen zur Komponente

Je Asset soll darüber hinaus eine Übersicht sämtlicher auf dem Asset installierten Softwares zur Verfügung stehen.

Softwareinformationen:

- Softwarebezeichnung
- Verwendungszweck
- Versionsnummer
- Softwarehersteller
- Datum letzter Patch
- CVE-Score
- CVSS-Score
- Life-Cycle-Ende

Um den großen Teil des Lebenszyklus einer Komponente leiten und dokumentieren zu können, sollen in der Datenbanklösung auch Informationen des Service sowie der Wartung und Instandhaltung ersichtlich sein.

Folgende Suchmöglichkeiten sollen gegeben sein:

- Was ist auf einer bestimmten Anlage bzw. Straße verbaut?
- Welcher Gerätetyp ist in welchen Anlagen verbaut?
- Ein bestimmter Hostname oder eine bestimmte IP-Adresse
- Datum der letzten Instandhaltung
- Wurde die Komponente zur Reparatur eingesendet?
- Welche Ersatzteile sind auf welchen Lagerorten verfügbar?

Die Übersicht der Ersatzteilverfügbarkeit mit unterschiedlichen Lagerorten, sowie die Nachvollziehbarkeit von Wartungs- und Instandhaltungstätigkeiten stehen im Fokus. Wird eine Ersatzteilkomponente auf einer Anlage verbaut, kann diese in wenigen Schritten in die Asset-Verwaltung übernommen werden.

Die Datenbank soll darüber hinaus als Unterstützung zur Nachverfolgung von eingesendeten Geräten dienen. In der jeweiligen Komponentenansicht ist der Status ersichtlich. Hierbei wird unterschieden, ob sich die Komponente noch im Haus befindet, ob sie eingesendet wurde, ob sie erneut auf einer Anlage verbaut oder ob sie verschrottet wurde. Es soll die Möglichkeit gegeben sein, Dateien wie Lieferscheine oder Fehlerberichte in der Datenbank zu hinterlegen, um den Return-Material-Authorization-Prozess (RMA-Prozess) zu unterstützen.

Ein großer Teil der Daten zu den bestehenden Anlagen wurde im Vorfeld in einer Microsoft Excel Datei erfasst. Die Daten wurden geprüft und freigegeben. Diese Daten werden als Basis in die Datenbank übernommen. Bei der Erweiterung bzw. Änderung der Daten soll die Datenintegrität mit Hilfe von Integritätsbedingungen sichergestellt werden.

5.3.3 Zusätzliche Funktionalitäten der Lösung

Neben den Funktionen zum Ändern oder Hinzufügen von Datensätzen sowie dem Anzeigen der Assetinformationen soll die Listenansicht der Datenbanklösung weitere Standardfunktionen aufweisen. Dazu gehören zum Beispiel das Zurückspringen von der Listenansicht in die Suchansicht sowie das Speichern von Änderungen. Aufgrund der Tatsache, dass mehrere Personen gleichzeitig an der Datenbank arbeiten sollen, ist eine automatische Aktualisierung notwendig. Ähnlich den Standard-Windowsfunktionen soll den Anwendenden die Möglichkeit gegeben werden, Handlungen rückgängig zu machen oder wiederherzustellen. Grundsätzlich können ganze Zeilen oder einzelne Zellen in der Liste markiert werden, analog zu den Funktionen einer Microsoft Excel Liste. Ein Button in der Listenansicht soll die Möglichkeit geben, einen Datensatz zu löschen. Dieser wird jedoch nur für Benutzer*innen aktiv sein, welche die jeweiligen Rechte dazu besitzen. Ein Suchfeld ermöglicht innerhalb der ausgegebenen Liste nach einem bestimmten Begriff oder Wert zu suchen.

Darüber hinaus soll in der Listenansicht der Asset-Verwaltung die Möglichkeit gegeben sein, sich ebenfalls detaillierte W&I Informationen anzeigen zu lassen und vice versa in der Listenansicht der W&I Informationen detaillierte Assetinformationen. Auf diese Weise wird die Option geboten, sämtliche vorhandenen Daten ohne zusätzlichen Aufwand einzusehen. Ein Auswertebutton gibt mit Hilfe von Pivot Charts und Tabellen eine Übersicht über die markierten Assets oder über die gesamte Liste, wenn keine Zeilen markiert wurden. Diese Funktion ist ebenso analog zur Datenanalyse im Microsoft Excel.

Eine Sortierfunktion soll den Benutzer*innen die Option bieten, die ausgegebene Liste aufsteigend oder absteigend zu sortieren. Darüber hinaus soll eine Filterfunktion gegeben sein, welche die Möglichkeit bereitstellt, nach bestimmten Begriffen oder Werten zu filtern. Eine Druckfunktion ermöglicht das Drucken oder Ausgeben als PDF-Datei und mit Hilfe der Upload- und Downloadfunktion können Daten von einer Microsoft Excel- oder CSV-Datei auf die Datenbank hochgeladen werden bzw. als Microsoft Excel- oder CSV-Datei heruntergeladen werden. Der Upload ist kongruent zur Löschfunktion abhängig von den Benutzerrechten.

5.3.4 Systemgrenzendiagramm

Das Systemgrenzendiagramm in Abbildung 26 zeigt auf der einen Seite sämtliche Eingangsdaten der Datenbank, auf der anderen Seite die Ausgangsdaten bzw. welche Ergebnisse die Datenbank liefern wird. In der Mitte wird illustriert, welche Informationen in welcher Form im System verarbeitet werden.

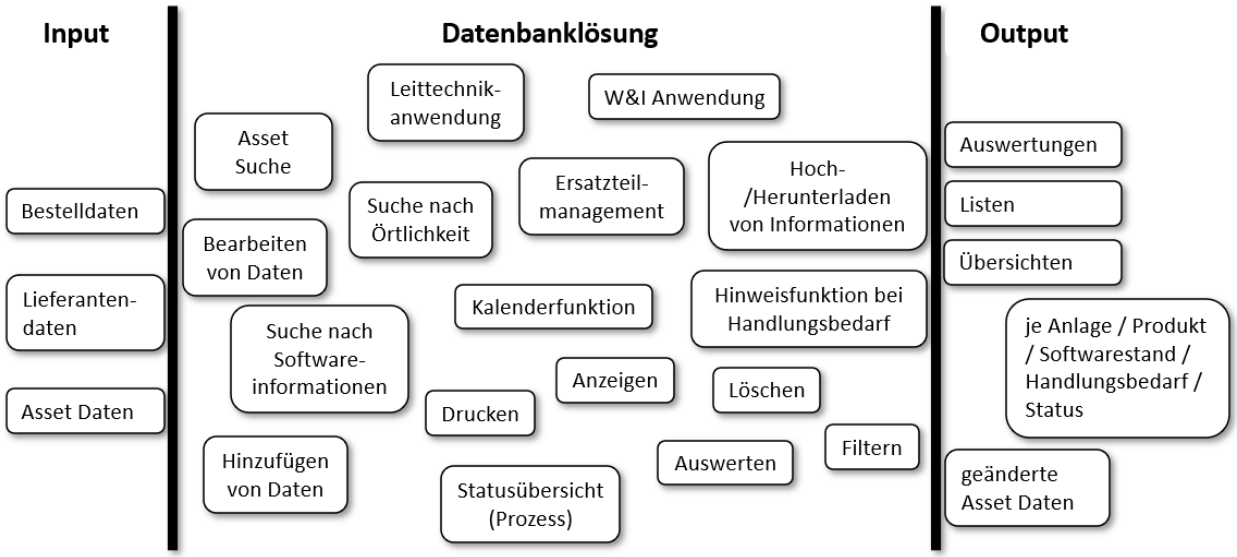

Abbildung 26: Systemgrenzendiagramm, Quelle: Eigene Darstellung.

5.3.5 Einbinden der Prozessorganisation

Durch das Erfassen im Zuge der Bestellung, Statusupdate bei Anlieferung, Überarbeitung der Daten, wenn das Gerät vorbereitet wird, bis hin zum Verbau sind mehrere Prozesse der Organisation mit dem Tool beschäftigt und in die Datenpflege eingebunden. Sämtliche W&I Handlungen und zusätzliche Informationen, welche im Zuge von Wartungs- und Instandhaltungstätigkeiten auftreten (z.B. Austausch eines Gerätes, Einsenden zum Lieferanten und Tracken von Geräten innerhalb der Gewährleistung) sind ebenfalls Bestandteil der Datenbanklösung.

5.3.6 Technische Voraussetzungen

Die Datenbanklösung soll auf allen aktuellen Windows Rechnern im Unternehmen mit Verbindung zum internen Netz funktionieren. Da im Unternehmen keine Apple-Geräte im Einsatz sind, ist eine Abstimmung auf das macOS Betriebssystem derzeit nicht notwendig. Die Speicherung der Daten wird auf dem unternehmenseigenen Server stattfinden und dort parallel zu sämtlichen Unternehmensdaten täglich gesichert.

6 MODELLIERUNG EINES PROTOTYPS DER DATENBANK (UI-MOCKUP)

Unter Berücksichtigung sämtlicher bisher evaluierten Anforderungen findet die Modellierung des Prototyps als UI-Mockup statt. Als Software kommt hierfür Adobe XD von Adobe Systems zum Einsatz. Diese vektorbasierte Grafiksoftware ist optimal geeignet, das User Interface grafisch darzustellen inklusive eines Tests der Funktionsweise mit Hilfe von Verknüpfungen zu den verschiedenen Ansichten.

Die erstellten Ansichten stellen die Berechtigungen eines Administrators dar. Einschränkungen zur reinen Leseberechtigungen werden in den einzelnen Anwendungsfällen gesondert beschrieben. Im Falle von Einschränkungen werden bestimmte Menüfunktionen in grauer anstelle von schwarzer Farbe dargestellt.

Es wird je Anwendungsfall ein Bereich des UI-Mockups erstellt. Zu diesen gehören unter anderem:

- Startansicht
- Suchansicht Asset-Management
- Suchansicht W&I / Ersatzteilmanagement
- Listenansicht Asset-Management
- Listenansicht W&I / Ersatzteilmanagement
- Einfügen eines Datensatzes
- Bearbeiten von Datensätzen
- Detaillierte Assetinformationen in unterschiedlichen Varianten
- Softwareinformationen
- Suchen, Filtern und Drucken in der Listenansicht
- Up- und Download von Informationen

Um den Benutzer*innen den Einstieg in die Software so einfach wie möglich zu gestalten, sollte der Umgang mit der Datenbank ähnlich zu Microsoft Applikationen gestaltet sein. Die Farben sollen in Graustufen gehalten werden, um die Darstellung für das Auge so angenehm wie möglich zu realisieren. Abbildung 27 illustriert diese Vorgaben anhand eines Styleguides für die Arbeits- bzw. Listenansicht.

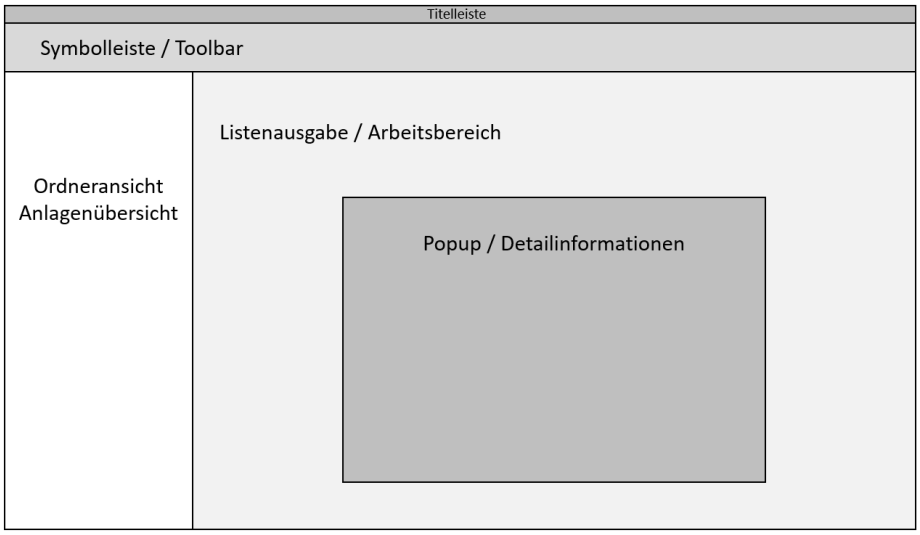

Abbildung 27: Styleguide Arbeits- bzw. Listenansicht, Quelle: Eigene Darstellung.

6.1 Startansicht

In der Startansicht ist, wie in Abbildung 28 ersichtlich, die Auswahlmöglichkeit zwischen Asset-Management und W&I / Ersatzteilmanagement gegeben. Das Asset-Management findet hauptsächlich in der Leittechnikabteilung Anwendung, um im Zuge der Projekterrichtung neue Assets zu erfassen oder sämtliche verbauten Assets inklusive Informationen bestehender Anlagen auszugeben. Für Wartungs- und Instandhaltungstätigkeiten wird der Bereich W&I / Ersatzteilmanagement geschaffen.

Der grundsätzliche Aufbau sowie die Funktionsweisen beider Bereiche sind ähnlich und sie unterscheiden sich hauptsächlich in Suchmöglichkeiten sowie den ausgegebenen Assetinformationen.

Abbildung 28: UI-Mockup Startansicht, Quelle: Eigene Darstellung.

6.2 Suchansicht Asset-Management

Die Suchansicht für das Asset-Management ermöglicht die Asset-Suche nach bestimmten, definierten Suchwerten. Die Auswahl dieser Werte fällt auf jene, die am aussagekräftigsten sind bzw. die am häufigsten benötigt werden.

Folgende Suchbegriffe können einzeln oder in Kombination gewählt werden:

- Anlagenname
- Straße
- Ortstyp

- Hostname
- IP-Adresse
- Gerätehersteller
- Gerätetyp
- Gerätebezeichnung
- Gerätename
- Softwarebezeichnung
- Softwarehersteller
- Datum letzter Patch
- Software- / Firmwareversion

In jedem Suchfeld ist ein grauer Mustertext hinterlegt. Dieser soll den Anwender*innen bei der Eingabe als Unterstützung dienen. Bei der Datumsangabe kann das Datum händisch im Format TT.MM.JJJJ eingegeben oder alternativ in einem Pop-up-Kalender ausgewählt werden. Die Standardsuchansicht ist in Abbildung 29 ersichtlich, Abbildung 30 illustriert die Datumsauswahl mit Hilfe der Kalenderfunktion. Die Befüllung der restlichen Werte erfolgt mit Hilfe von Drop-Down Feldern oder unterstützt durch eine Autocomplete-Funktion, um die Suche nach nicht vorhandenen Schreibweisen der Suchbegriffe zu vermeiden. Es wird ebenfalls die Möglichkeit gegeben, zurück zum Startbildschirm zu wechseln.

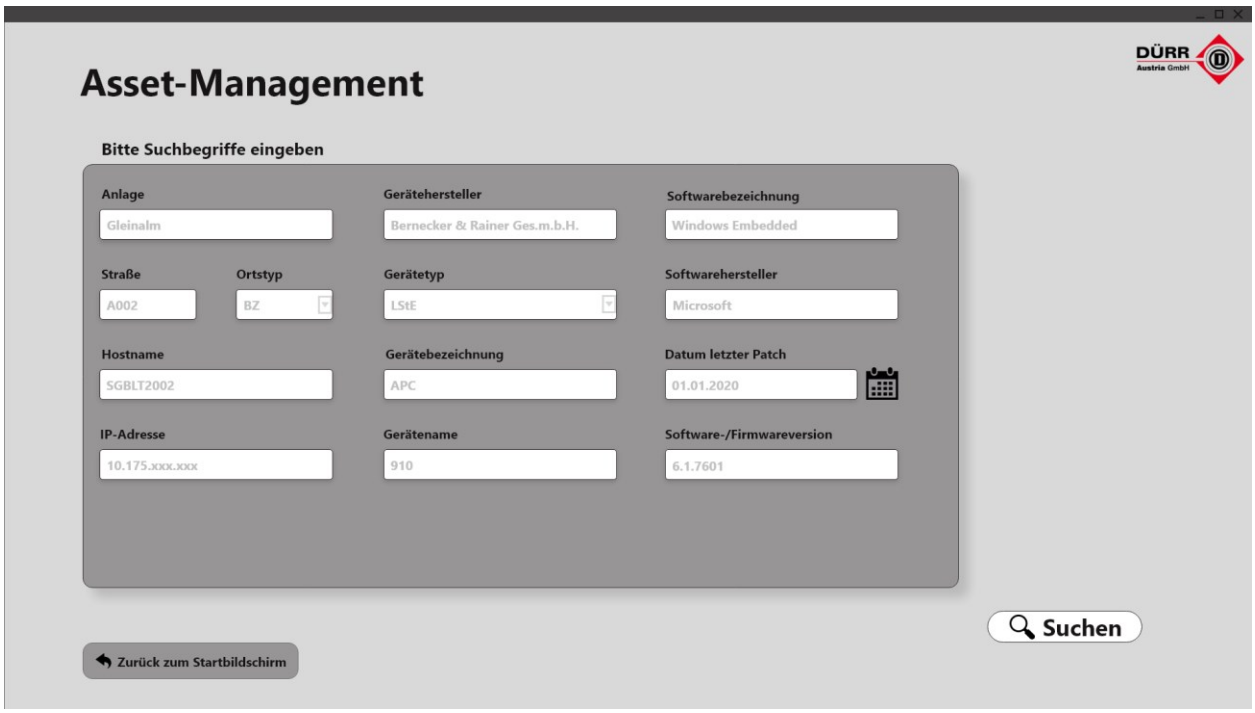

Abbildung 29: UI-Mockup Asset-Management Suchansicht, Quelle: Eigene Darstellung.

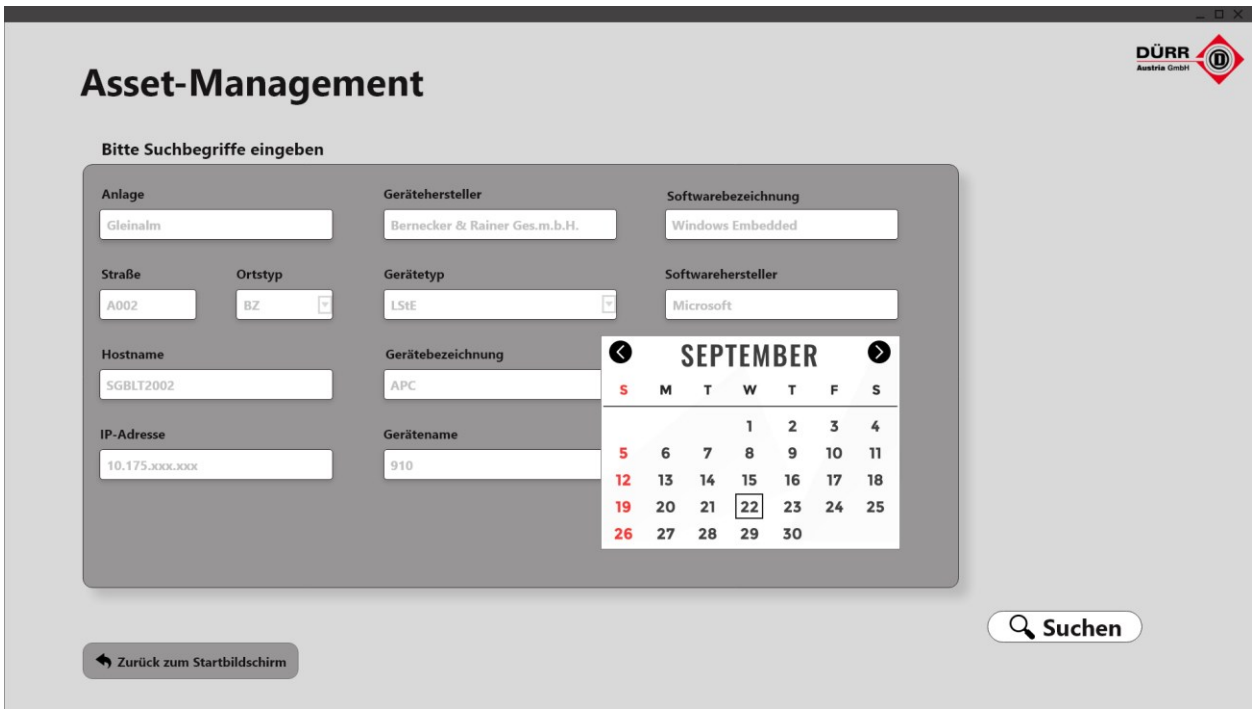

Abbildung 30: UI-Mockup Suchansicht Datumsauswahl, Quelle: Eigene Darstellung.

Durch Klick auf den Suchbutton öffnet sich die Asset-Listenansicht mit den Ergebnissen aus der Suche. Konnten keine übereinstimmenden Datensätze gefunden werden, erscheint ein Feld mit dem Hinweis: „Keine Datensätze ermittelt.“. Diese Fehlermeldung ist in Abbildung 31 dargestellt. Mit Schließen dieses Pop-ups kehrt man wieder zur vorherigen Suchansicht zurück und die gewählten Parameter können adaptiert werden.

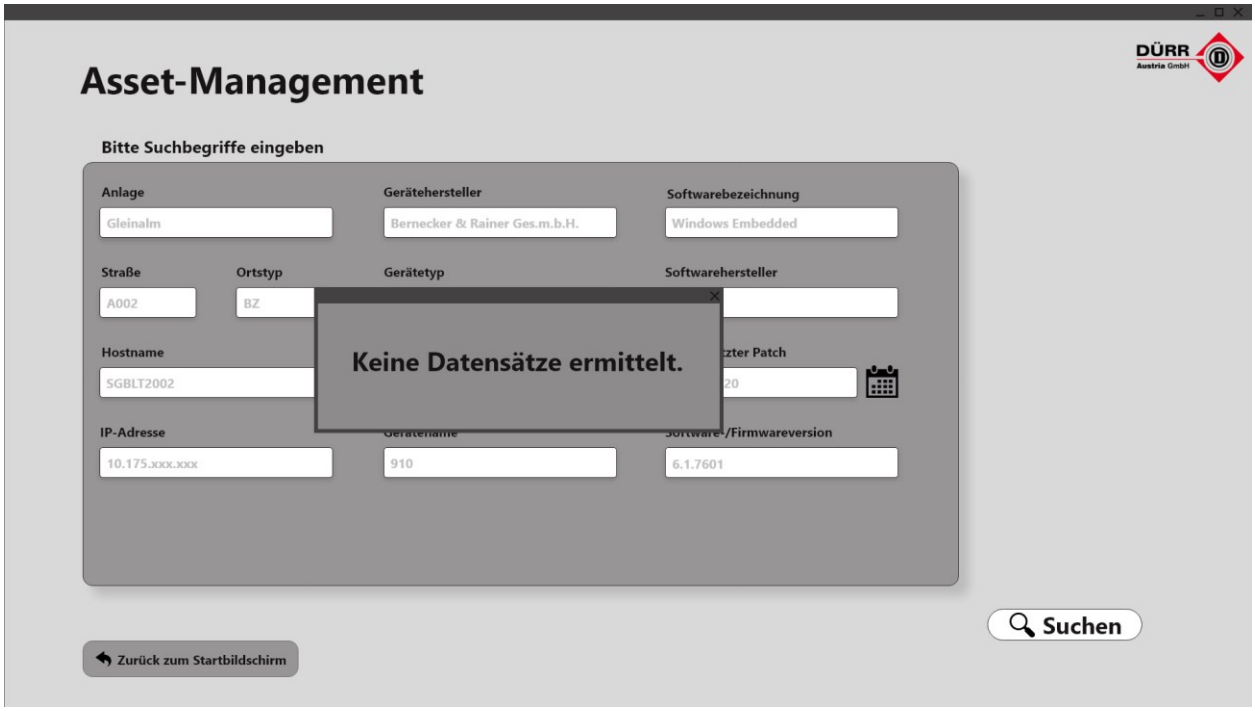

Abbildung 31: UI-Mockup Fehlermeldung Suchansicht, Quelle: Eigene Darstellung.

In Abbildung 32 ist ersichtlich, dass für die Suche ein einzelner Suchbegriff im Feld „Anlage“ gewählt wurde. Erkennbar ist dies durch die Änderung der Textfarbe von grau in schwarz.

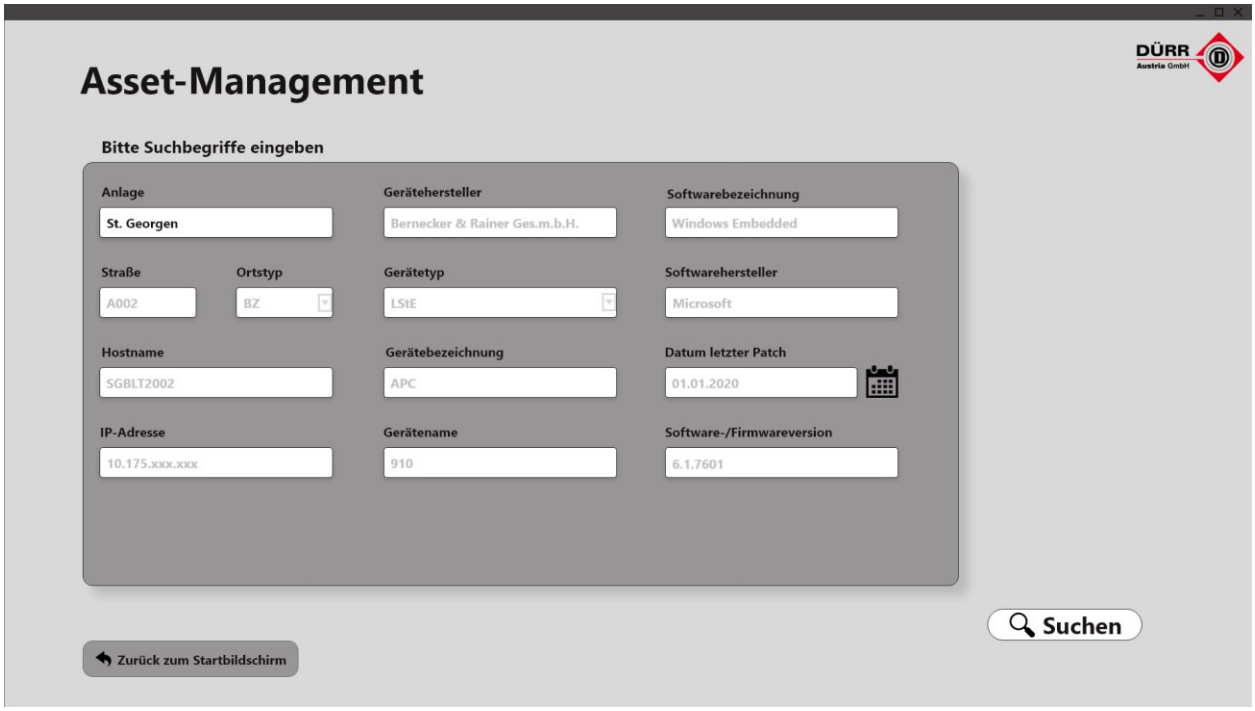

Abbildung 32: UI-Mockup Suchansicht Anlagenauswahl, Quelle: Eigene Darstellung.

6.3 Listenansicht nach Suchanfrage Asset-Management

Die Listenansicht ist dreiteilig aufgebaut. Am oberen Rand befindet sich ein Menüband mit folgenden Menüfeldern:

- Zurück zur Suche
- Speichern
- Datensatz einfügen
- Bearbeiten
- Mehrere Datensätze bearbeiten
- Aktualisieren
- Rückgängig
- Wiederherstellen
- Löschen
- Suchen
- Assetinformation
- W&I Informationen
- Auswerten
- Sortieren
- Filtern
- Drucken

- Upload
- Download

Die einzelnen Funktionen dieser Befehle werden in den nachfolgenden Abschnitten detailliert beschrieben und gelten sowohl für das Asset-Management als auch für das W&I / Ersatzteilmanagement.

Direkt unter dem Menüband befindet sich auf der linken Seite eine Ordneransicht, in der alle verfügbaren Anlagen angezeigt werden und in welcher direkt zur vollständigen Liste einer anderen Anlage gewechselt werden kann. Auf der rechten Seite wird die Liste der Datensätze gemäß Suchanfrage ausgegeben. Die Spalten und Zeilen ähneln der Ansicht in Microsoft Excel. Es können einzelne Datensätze mit Hilfe eines Auswahlfeldes am linken Rand ausgewählt werden. Hierbei wird die gesamte Zeile markiert und der Datensatz kann, abhängig von der jeweiligen Benutzerberechtigung, bearbeitet oder gelöscht werden. Einzelne Felder können mittels einfachem Klick, mehrere Felder mittels gedrückter Maustaste oder gedrückter Umschalttaste in Kombination mit den Pfeiltasten markiert bzw. ausgewählt werden. Bildlaufleisten an den Rändern veranschaulichen den Bildlauf und geben die Möglichkeit die Listen vollständig einsehen zu können. Bei gedrückter STRG-Taste in Kombination mit dem Mausrad kann die Ansicht vergrößert oder verkleinert werden. Abbildung 33 zeigt einen Überblick über die erläuterte Listenansicht. Die Lesbarkeit der einzelnen Datensätze in den angezeigten Listen ist wissentlich schlechter gewählt, um die Sicherheit der enthaltenen Daten nicht zu gefährden.

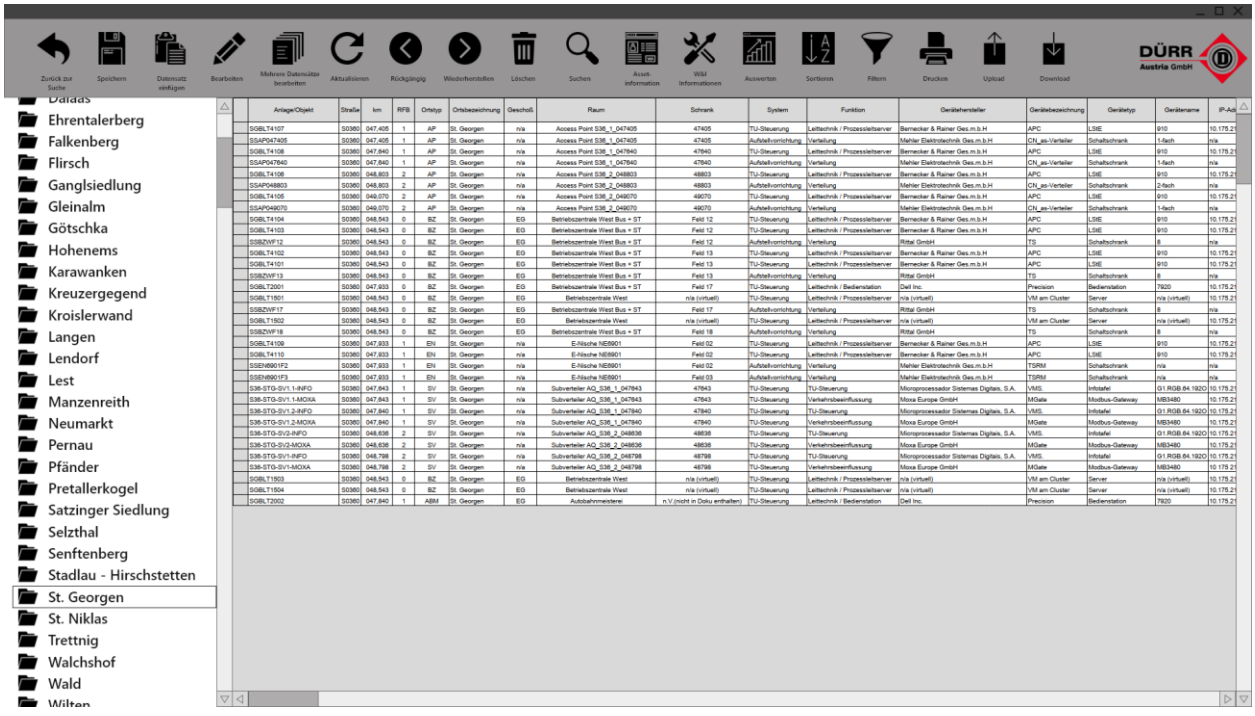

Abbildung 33: UI-Mockup Listenansicht, Quelle: Eigene Darstellung.

Mit Klick auf den **„Zurück zur Suche“**-Button springt das System zurück in die Suchansicht und die Suchparameter können angepasst werden. Der Klick auf den **„Speichern“**-Button speichert etwaige Änderungen in den Datensätzen der Liste. Wird die Listenansicht verlassen oder das Programm geschlossen und es werden ungespeicherte Änderungen erkannt, erscheint ein Pop-up mit der Rückfrage,

ob die Änderungen gespeichert werden sollen. Ausgewählt werden kann hier analog zur Standard-Windows-Rückfragemeldung „Speichern“, „Nicht Speichern“ oder „Abbrechen“. Der Menüpunkt **„Aktualisieren“** aktualisiert den Suchvorgang bzw. die erzeugte Liste. Diese Funktion ist insbesondere deshalb notwendig, da mehrere Personen gleichzeitig Zugriff auf die Datenbank haben und somit sichergestellt werden kann, dass alle aktuellen Daten angezeigt werden. Mittels **„Rückgängig“** können Änderungen oder Löschungen rückgängig gemacht werden, **„Wiederherstellen“** kehrt eine rückgängig gemachte Aktion wieder um. Diese Aktionen sind analog zu den in Windows üblichen Funktionen.

Aufgrund von prozessübergreifenden Verbindungen zwischen dem Asset-Management und dem W&I / Ersatzteilmanagement, ist es in beiden Bereichen möglich, sich die Daten des jeweils anderen anzeigen zu lassen.

6.4 Einfügen eines Datensatzes innerhalb der Listenansicht

Wählt man **„Datensatz einfügen“**, wird eine leere Zeile der Liste hinzugefügt und ermöglicht die Befüllung mit Informationen. Abbildung 34 demonstriert die der Liste hinzugefügte Zeile. Diese Funktion ist im Falle einer Leseberechtigung grau hinterlegt und kann nicht ausgewählt werden. Bei der Befüllung mit Informationen werden jene Felder, die ebenfalls in der Suchansicht ersichtlich sind mittels Drop-Down Möglichkeiten befüllt, Ausnahmen sind hierfür Hostnamen, IP-Adressen und Software- bzw. Firmwareversionen. Benutzer*innen mit Administratorrechten können diese Listen an vorgeschlagenen Elementen erweitern, jene mit reinen Schreibberechtigungen können lediglich vorhandene Parameter auswählen. Diese Vorgehensweise ermöglicht es, ein hohes Maß an Datenkonsistenz sicherzustellen.

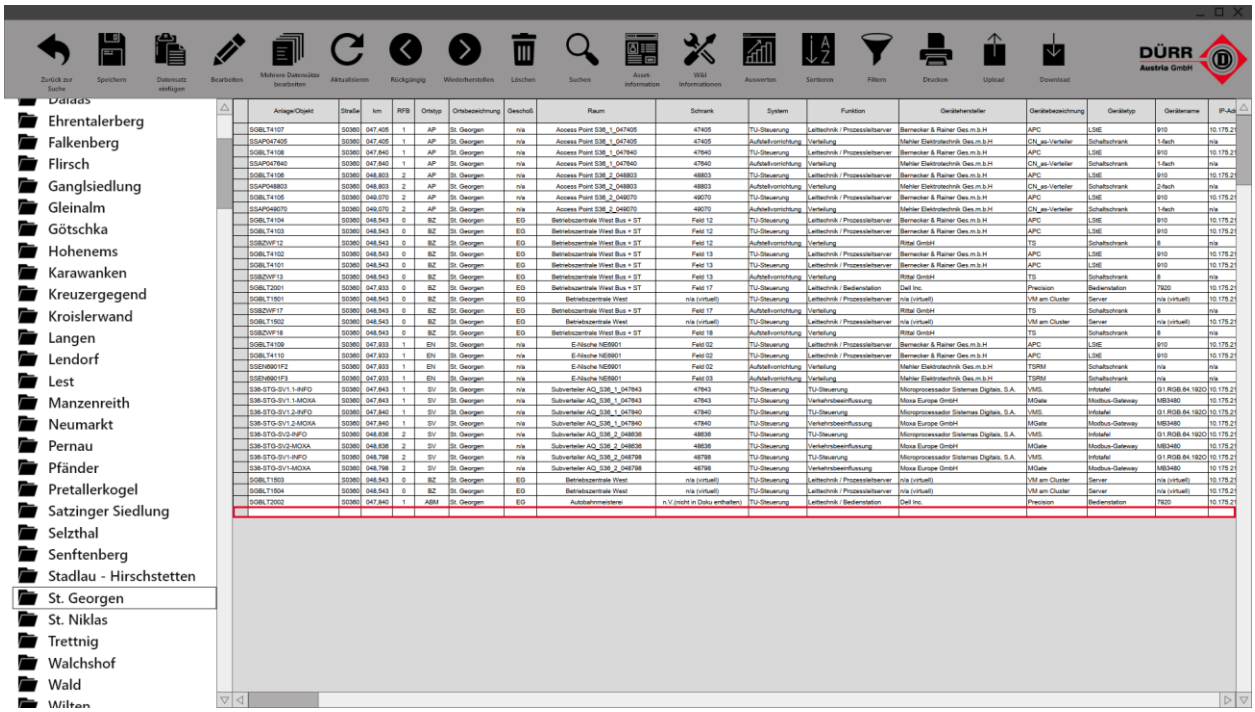

Abbildung 34: UI-Mockup Datensatz zur Liste hinzufügen, Quelle: Eigene Darstellung.

6.5 Bearbeiten eines einzelnen Datensatzes

Wird ein einzelner Datensatz mittels Doppelklick markiert oder die Zelle mit einem einfachen Klicken aktiviert und der Button **„Bearbeiten“** gewählt, kann der Wert dieser Zelle geändert und mittels Enter-Taste bestätigt werden. Wie in Abbildung 35 ersichtlich, wurde für die Änderung eine Zelle ausgewählt. Bei leseberechtigten Benutzer*innen ist dieser Menüpunkt grau hinterlegt und es ist trotz Doppelklick keine Änderung des Zelleninhaltes möglich.

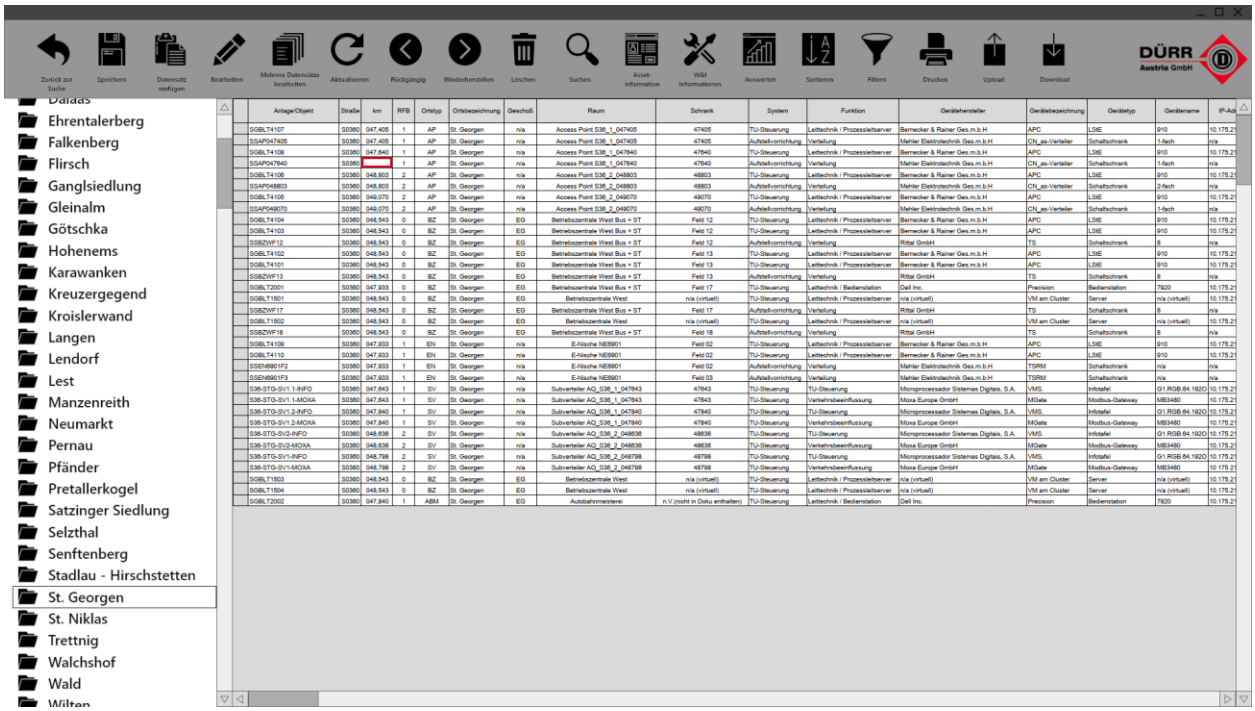

Abbildung 35: UI-Mockup Listenansicht Datensatz bearbeiten, Quelle: Eigene Darstellung.

6.6 Bearbeiten mehrerer Datensätze gleichzeitig

Um mehrere Felder gleichzeitig zu bearbeiten, können diese Zellen zeilenübergreifend markiert werden und die Funktion mittels Klick auf **„Mehrere Datensätze bearbeiten“** aktiviert werden. Die Felder erscheinen in weißer Farbe und wird nun ein neuer Wert eingegeben und mittels Enter-Taste bestätigt, wird dieser Wert für alle markierten Zellen übernommen und so mehrere Datensätze gleichzeitig verändert. Abbildung 36 stellt diese Form der Änderung grafisch dar. Diese Funktion bietet sich insbesondere im Falle von Aktualisierungen, Patches oder Umbauten an, da hier meist mehrere Assets gleichzeitig bearbeitet oder getauscht werden. Diese Funktion erleichtert es den Anwender*innen, die Datensätze mit möglichst geringem Aufwand aktuell zu halten. Voraussetzung für die Anwendung dieser Funktion ist eine Administrator- oder Schreibberechtigung. Bei der lesenden Benutzerberechtigung ist diese Funktion mit grauer Farbe hinterlegt.

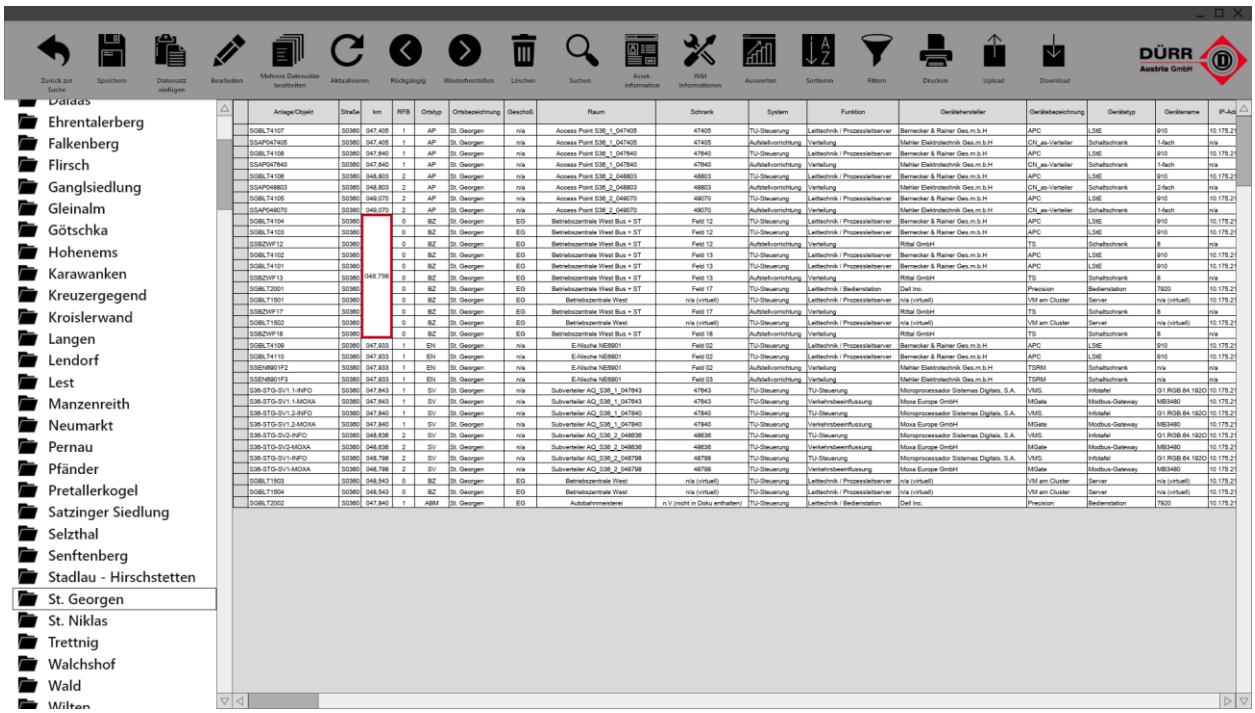

Abbildung 36: UI-Mockup Listenansicht mehrere Datensätze bearbeiten, Quelle: Eigene Darstellung.

6.7 Löschen von Daten bzw. Datensätzen in der Listenansicht

Um einzelne Werte oder ganze Datensätze löschen zu können, muss die Zelle mittels Doppelklick oder die Zeile mittels Klick auf das Zeilenauswahlfeld am linken Rand markiert werden. Benutzer*innen mit Administrator- oder Schreibrechten können diese Daten dann mit Hilfe von **„Löschen“** entfernen. Bei der lesenden Benutzerberechtigung ist diese Funktion ebenfalls mit grauer Farbe hinterlegt.

6.8 Suchanfrage in der Listenansicht

Mit dem Menüpunkt **„Suchen“** können bestimmte Begriffe und Werte innerhalb der Listenansicht gesucht und angezeigt werden. Diese werden dann in der Liste farblich markiert und mit Pfeilbuttons kann zwischen den Suchergebnissen hin und her gesprungen werden. Die Suchanfrage wird nicht Case-Sensitiv gestaltet, was bedeutet, dass die Groß-/Kleinschreibung der Begriffe unerheblich für das Ausführen von Suchanfragen ist. Gesucht werden kann sowohl nach einzelnen Begriffen oder auch nach Wortgruppen.

Die Abbildungen 37 und 38 veranschaulichen den Suchvorgang nach dem Begriff „LStE“.

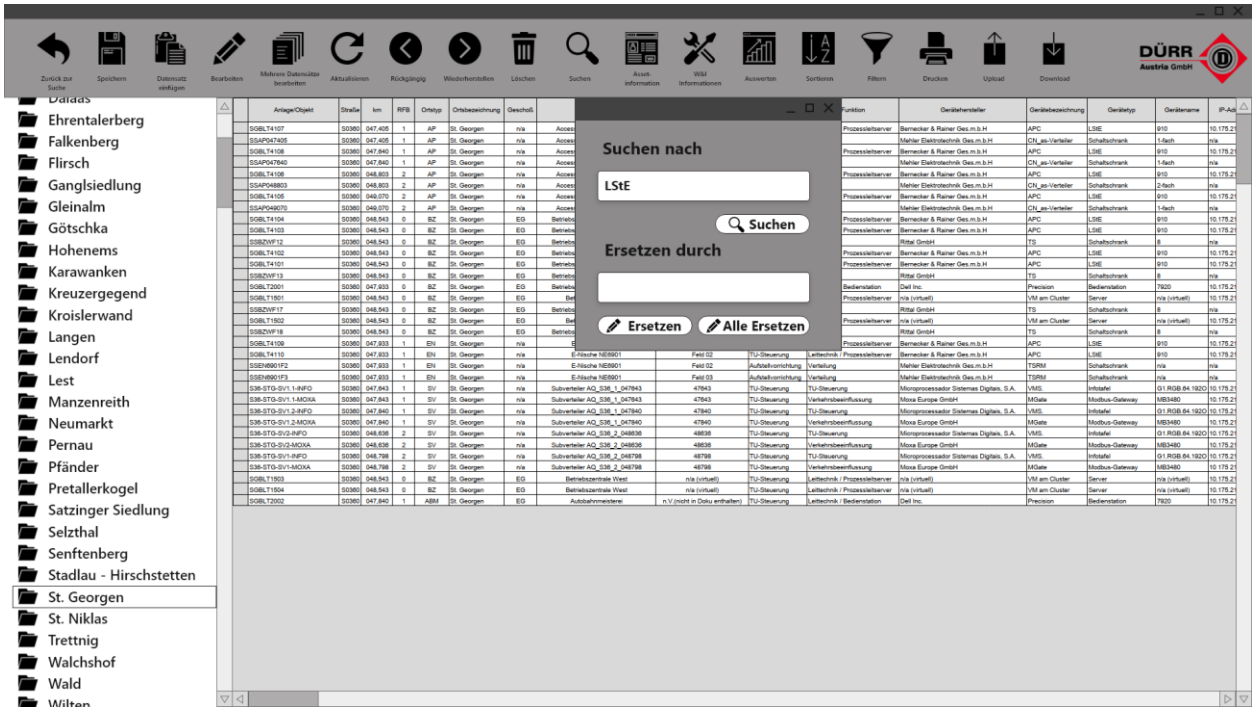

Abbildung 37: UI-Mockup Suchen/Ersetzen in Listenansicht, Quelle: Eigene Darstellung.

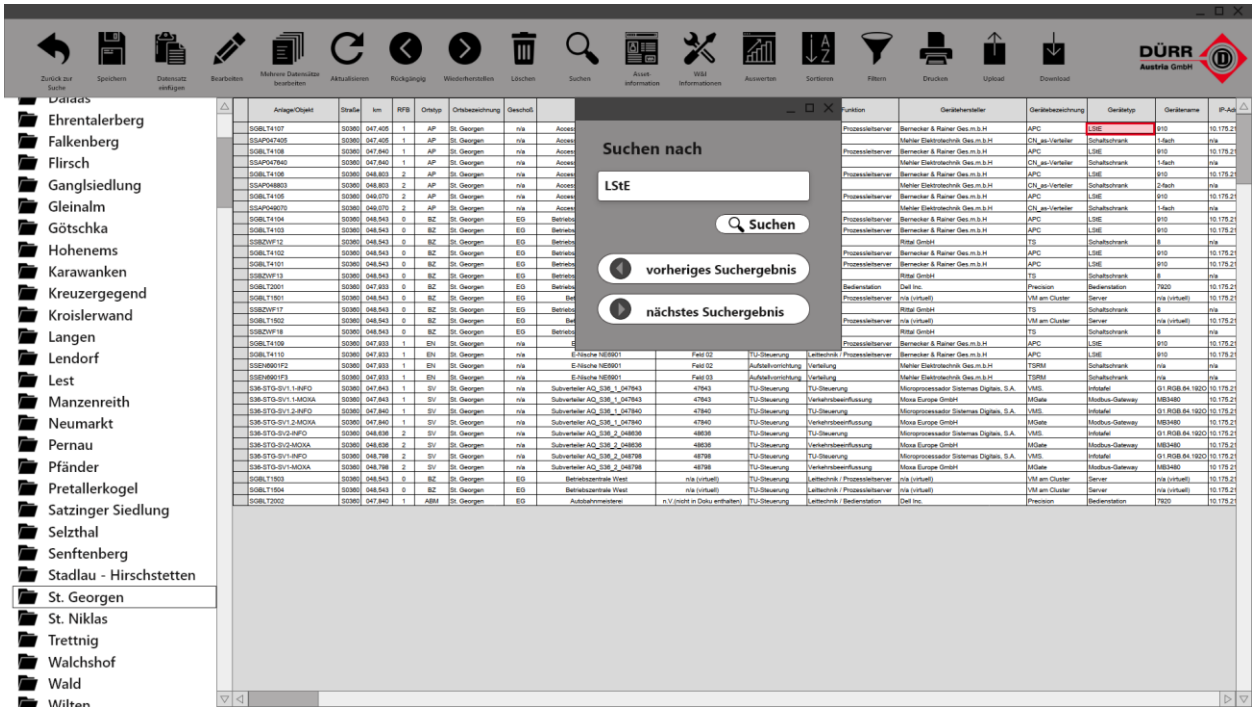

Abbildung 38: UI-Mockup Suchen in Listenansicht, Quelle: Eigene Darstellung.

Die Suchansicht ermöglicht ebenso das Ersetzen von Begriffen in markierten Bereichen oder der gesamten Liste. Adminstrator- oder Schreibrechte sind für die Ersetzen-Funktion Voraussetzung, bei Leserechten ist dieser Bereich grau hinterlegt und lediglich die Suchfunktion ist aktiv. Wie in Abbildung 39 dargestellt, soll der Begriff „LStE“ durch „Lokale Steuereinheit“ ersetzt werden. Eine Unterscheidung zwischen dem Ersetzen von einzelnen Begriffen oder allen mit der Suche übereinstimmenden Begriffen ist gegeben. Hierbei wird ein Suchbegriff gewählt sowie der Begriff vorgegeben, gegen den dieser Begriff ersetzt werden

soll. Durch Klick auf „Suchen“ wird die Suche gestartet und einzelne Suchergebnisse farblich markiert. Soll dieser Begriff ersetzt werden, wird „Ersetzen“ ausgewählt und das System springt automatisch zum nächsten Suchergebnis. Soll dieser markierte Begriff nicht ersetzt werden, wird mittels erneuter Auswahl von Suchen auf das nächste Suchergebnis verwiesen. Diese Funktion kann durch Auswahl des „Schließen“ Buttons am rechten oberen Rand des Pop-ups beendet werden.

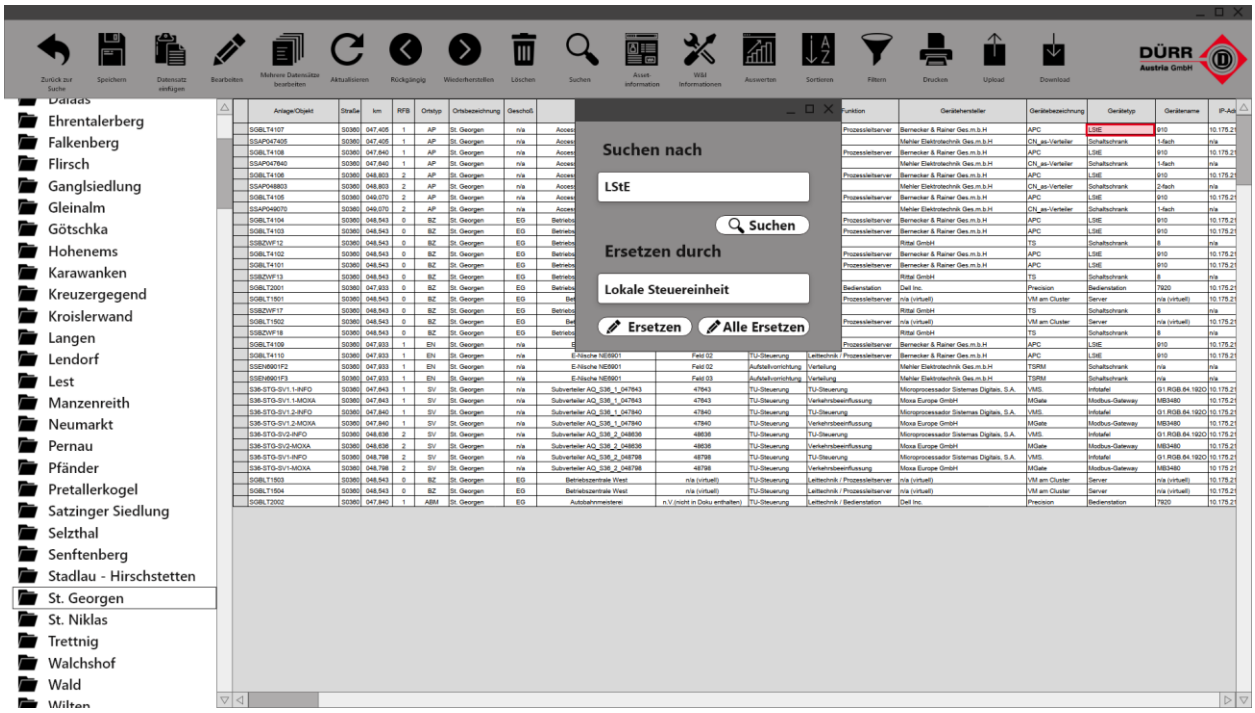

Abbildung 39: UI-Mockup Ersetzen in Listenansicht, Quelle: Eigene Darstellung.

6.9 Anzeige detaillierter Assetinformationen

Mit Klick in eine Zelle des gewünschten Assets wird das Asset gewählt und daraufhin der Menüpunkt **„Assetinformationen“** ausgewählt. Ein Pop-up erscheint mit allen verfügbaren Informationen zum jeweiligen Asset:

- Assetstatus (bestellt / verbucht / geliefert / einbaufertig / verbaut)
- Anlagenname
- Straße
- Kilometer
- Richtungsfahrbahn (RFB)
- Ortstyp
- Geschoß
- Raum
- Schrank
- Hostname
- IP-Adresse
- Gerätehersteller
- Gerätebezeichnung

- Gerätename
- Gerätetyp
- Redundanz zu
- Seriennummer
- Priorität
- ISMS-Dokumentation vorhanden
- Inbetriebnahmedatum
- Beginn Gewährleistung Kunde
- Ende Gewährleistung Kunde
- Anmerkungen
- Datenblätter
- Softwareinformationen

Aufgrund der Menge an Informationen, werden diese auf zwei Seiten aufgeteilt. Mittels Pfeiltaste im Pop-up kann zwischen diesen Seiten hin und her gesprungen werden. In den Abbildungen 40 und 41 sind diese Assetinformationen-Ansichten dargestellt. Statusänderungen können mit Hilfe eines Drop-down Menüs vorgenommen werden.

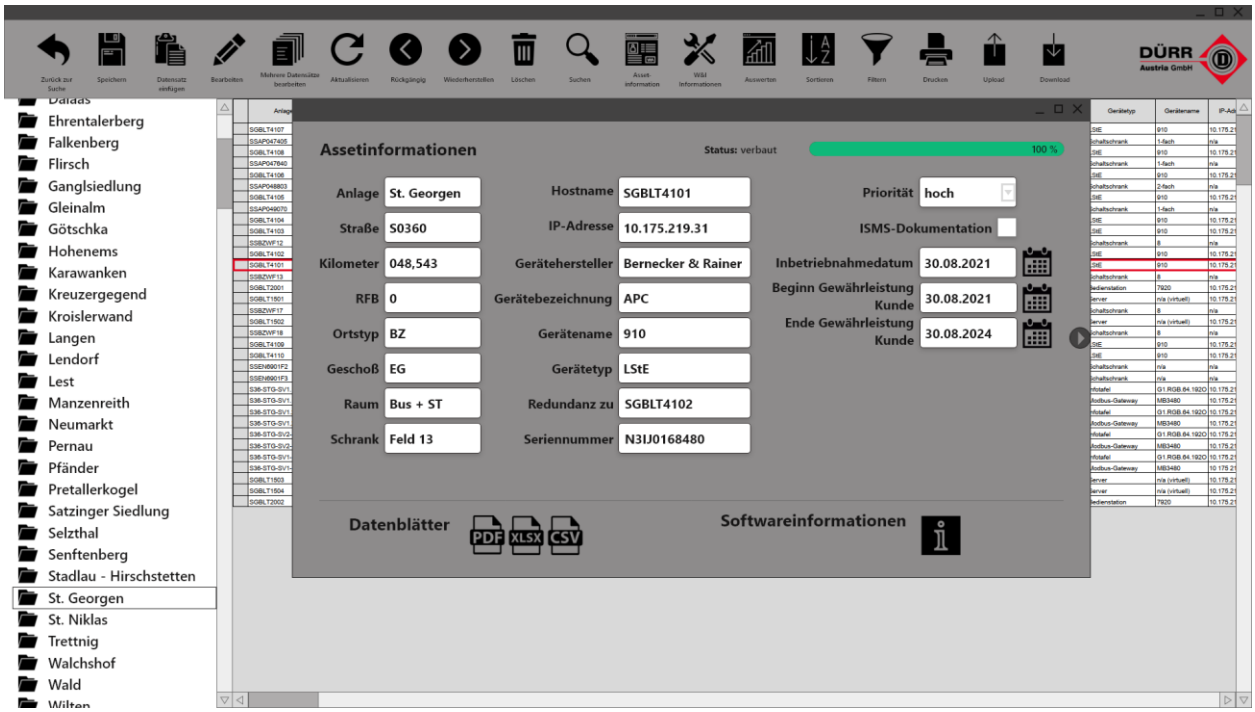

Abbildung 40: UI-Mockup detaillierte Assetinformationen Seite 1, Quelle: Eigene Darstellung.

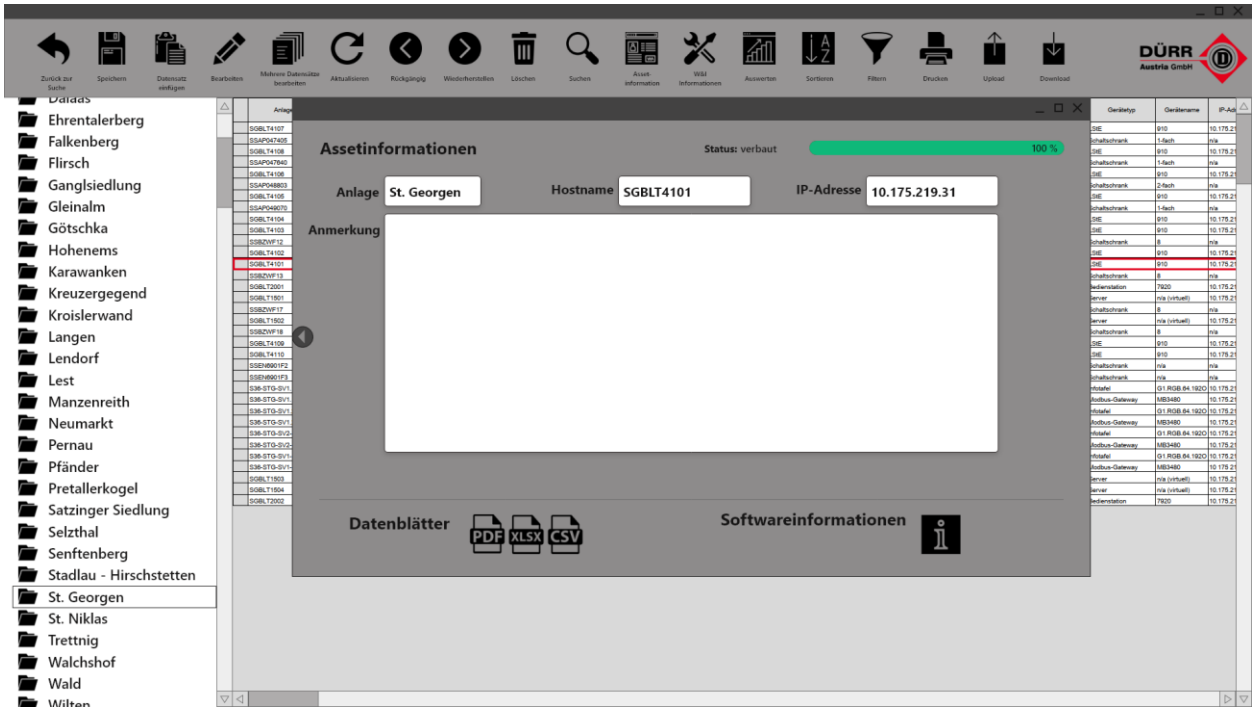

Abbildung 41: UI-Mockup detaillierte Assetinformationen Seite 2, Quelle: Eigene Darstellung.

Werden Datenblätter ausgewählt, öffnen sich diese in der jeweiligen auf dem Rechner installierten Software gemäß Dateityp. Softwareinformationen werden in einem gesonderten Pop-up, wie im nachfolgenden Kapitel 6.10 beschrieben, angezeigt.

6.10 Anzeige der Softwareinformationen

Werden die Softwareinformationen in der Assetinformationen-Ansicht gewählt, öffnet sich ein weiteres Pop-up mit Informationen sämtlicher am Asset installierter Software, welches in Abbildung 42 ersichtlich ist. Benutzer*innen mit Administrator- und Schreibrechten können direkt in dieser Ansicht Änderungen vornehmen sowie zusätzliche Softwareinformationen hinterlegen. Die Softwarebezeichnung, der Verwendungszweck und der Softwarehersteller werden mit Hilfe eines Dropdown-Menüs befüllt bzw. geändert. Ein Doppelklick auf die jeweilige Softwarezeile öffnet ein weiteres Pop-up mit detaillierten Informationen zu etwaigen Softwareproblemen, Bugs etc. sowie zusätzlichen Dokumenten, die der Dokumentation und Information dienen wie z.B. der PDF des Schwachstellen-Scans, der unter anderem detaillierte Informationen über den CVE bzw. CVSS Score enthält. Dieses Pop-up ist in Abbildung 43 dargestellt.

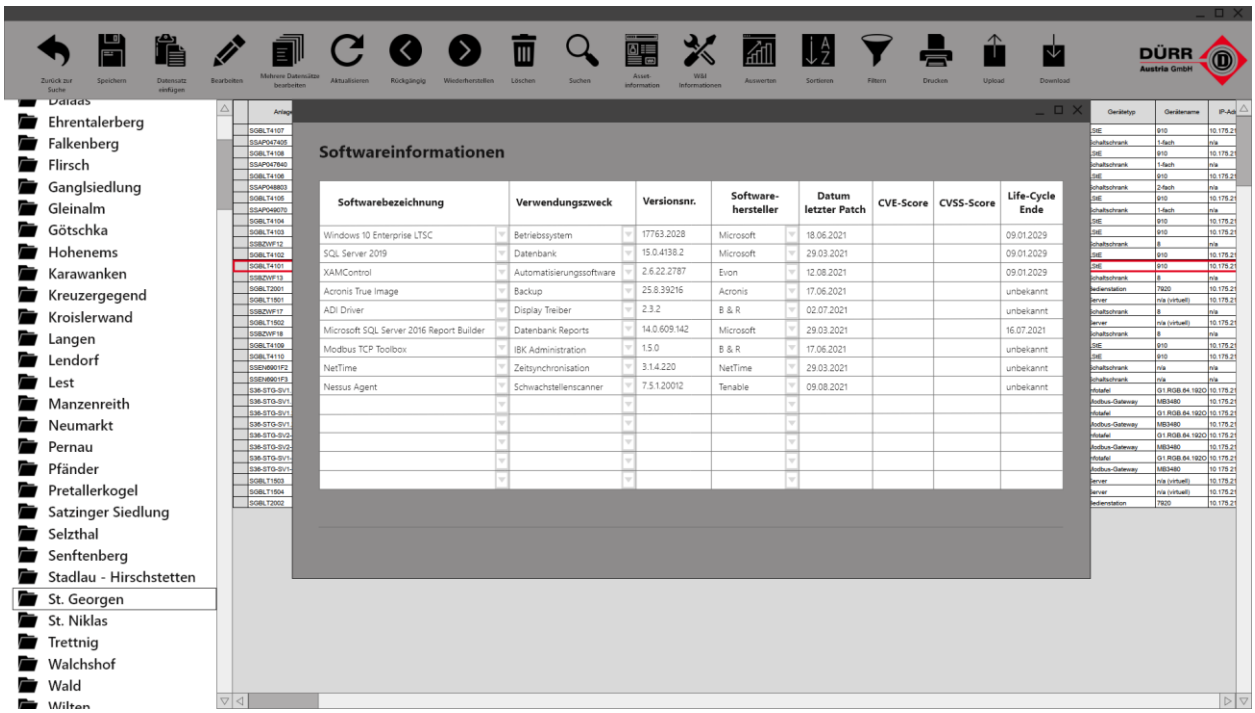

Abbildung 42: UI-Mockup Softwareinformationen, Quelle: Eigene Darstellung.

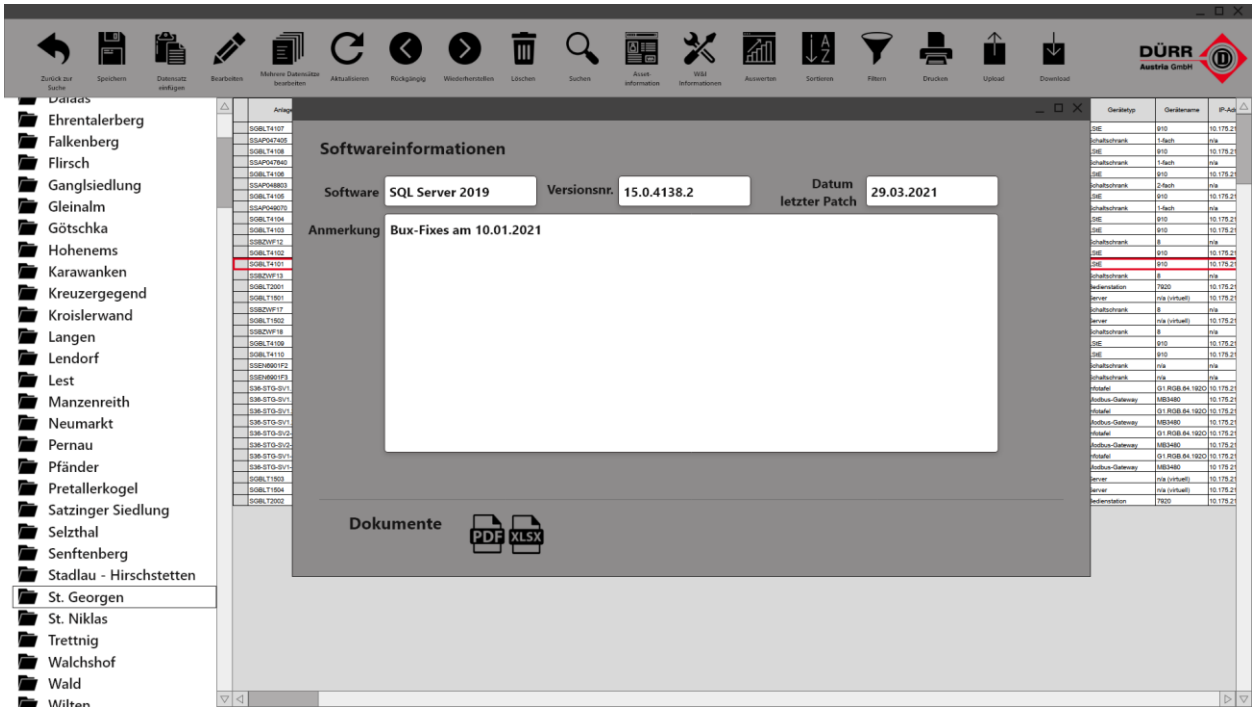

Abbildung 43: UI-Mockup detaillierte Softwareinformation, Quelle: Eigene Darstellung.

6.11 Auswerten der Listenergebnisse

Ähnlich der Datenanalysefunktion in Microsoft Excel kann eine beliebige Anzahl an Zeilen oder Spalten in der Listenansicht, oder auch alle Zeilen der Listenansicht gewählt werden, um mittels **„Auswerten“** eine Analyse der gewählten Daten vorzunehmen. Die Auswahl ganzer Zeilen findet über die Auswahlfelder am linken Rand statt, die Auswahl ganzer Spalten über die Auswahlfelder am oberen Rand. Durch gedrückt halten der Umschalttaste bzw. STRG-Taste können mehrere Zeilen und Spalten gewählt werden. „Auswerten“ erstellt auf Basis der gewählten Daten Pivottabellen und gibt diese in einem Pop-up aus. Je mehr Daten ausgewählt werden, umso mehr Tabellen werden erstellt und es kann, wie bei den Assetinformationen, zwischen mehreren Seiten hin und her gesprungen werden. Abbildung 44 zeigt beispielhaft eine solche Seite mit diversen Auswertungen.

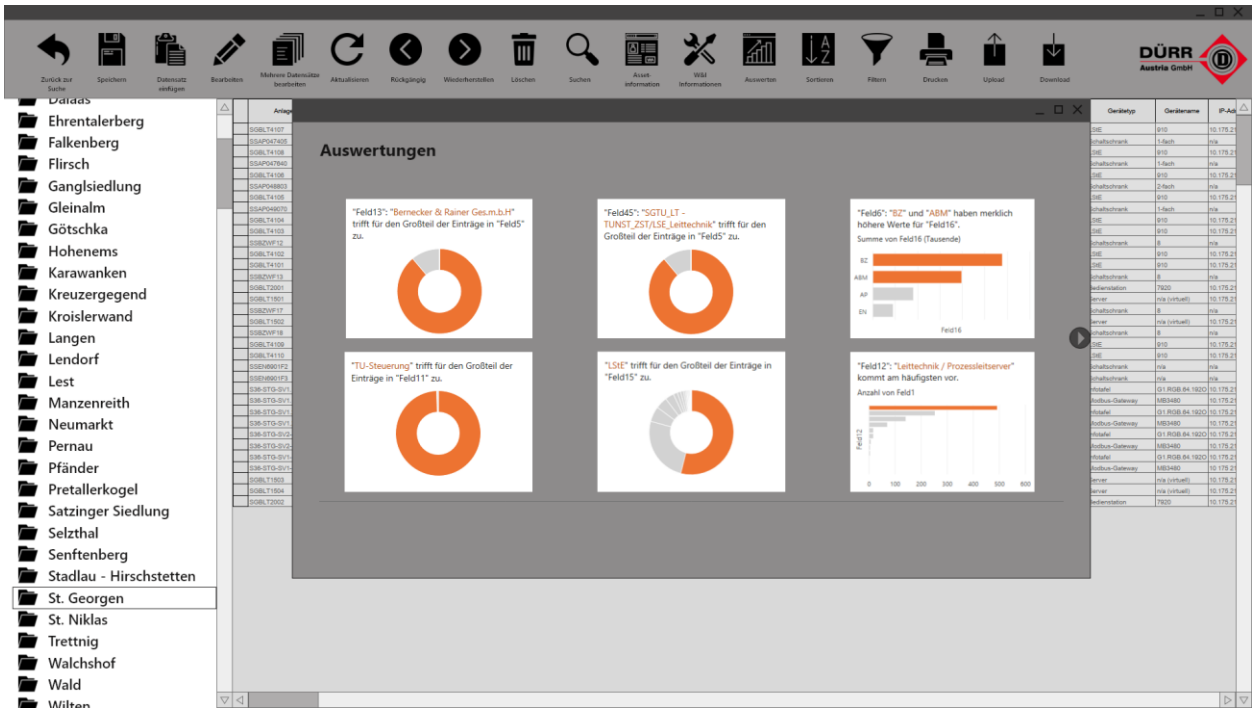

Abbildung 44: UI-Mockup Auswertungen, Quelle: Eigene Darstellung.

6.12 Sortieren der Listenergebnisse

Die Sortierfunktion der Listenergebnisse, ausgewählt mittels **„Sortieren“**, ermöglicht durch Markieren einzelner Spalten eine aufsteigende Sortierung von A bis Z (bzw. bei Zahlenwerten von 0 aufsteigend) sowie eine absteigende Sortierung von Z bis A (bzw. bei Zahlenwerten vom höchsten Wert absteigend bis 0). Ebenso kann in der Liste eine Sortierung nach einzelnen Spalten erfolgen. Diese Sortiermöglichkeit wird mit Hilfe eines Dropdown Menüs dargestellt, mit welchem die Spaltenüberschrift gewählt werden kann, nach der die Sortierung der Liste stattfinden soll. Sämtliche Spalten in der Datenbank stehen in diesem Dropdown Menü zur Verfügung. Gemäß Auswahl wird das Listenergebnis neu ausgegeben und die Liste entsprechend sortiert. Mit der Rückgängig-Funktion kann die Sortierung danach rückgängig gemacht und die Liste in ihren Urzustand zurückversetzt werden. In Abbildung 45 ist die Markierung der Spalte „Gerätetyp“ ersichtlich. Abbildung 46 zeigt die unterschiedlichen Sortiermöglichkeiten auf.

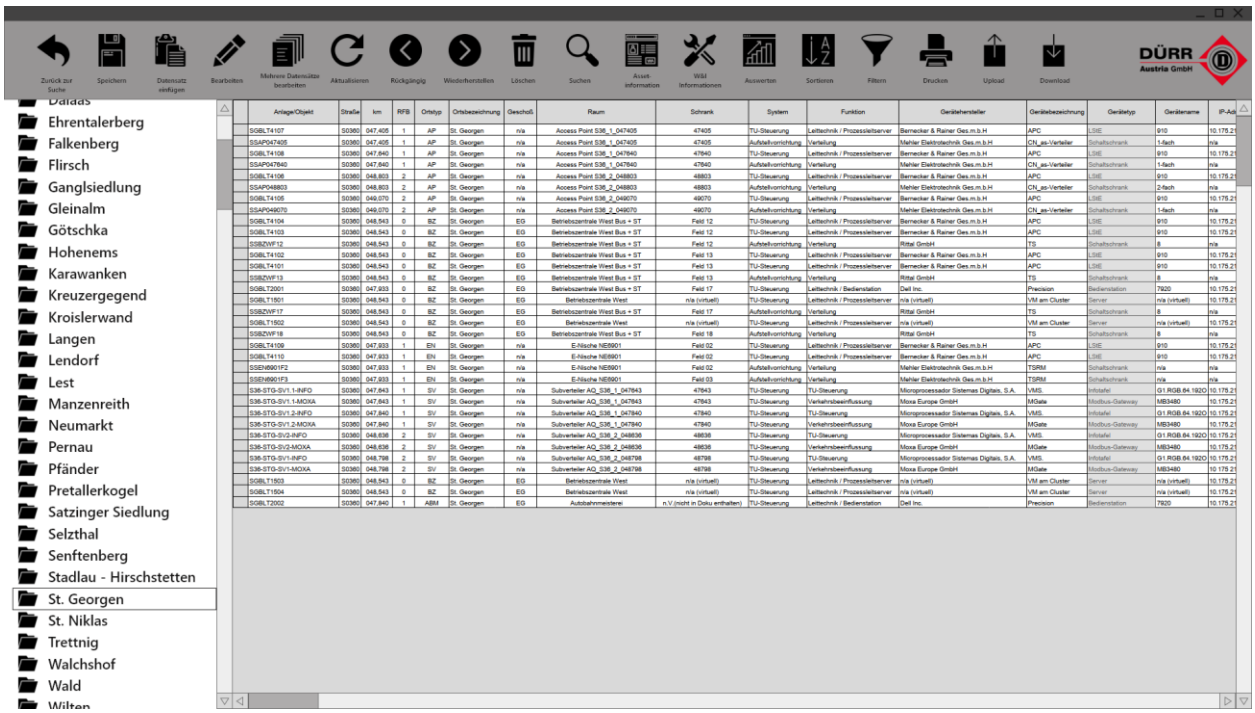

Abbildung 45: UI-Mockup Auswahl zur Sortierung, Quelle: Eigene Darstellung.

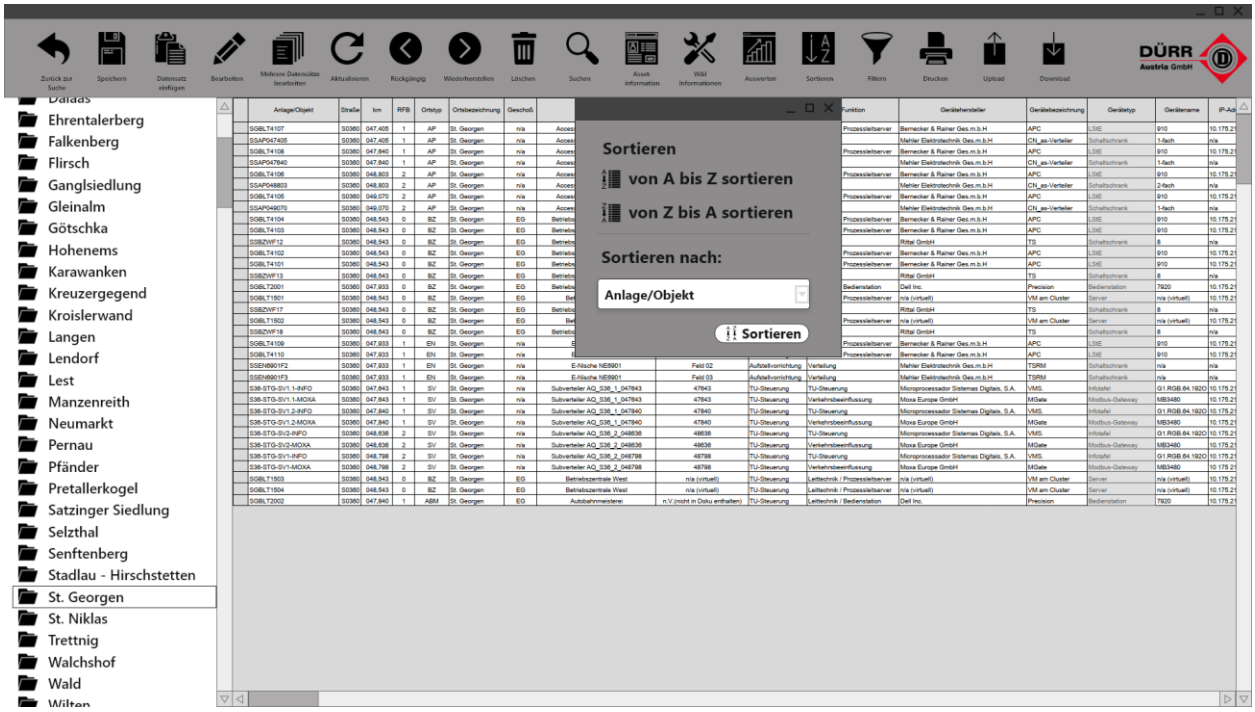

Abbildung 46: UI-Mockup Sortiermöglichkeiten, Quelle: Eigene Darstellung.

6.13 Filtern in der Listenansicht

Die „**Filtern**“-Funktion bietet die Möglichkeit, die Listenansicht nach bestimmten Parametern zu filtern. Zur Auswahl stehen die folgenden Filtereigenschaften:

- Entspricht
- Entspricht nicht
- Ist größer als
- Ist größer oder gleich
- Ist kleiner als
- Ist kleiner oder gleich
- Beginnt mit
- Beginnt nicht mit
- Endet mit
- Endet nicht mit
- Enthält
- Enthält nicht

Diese Filter können einzeln angewendet werden und alternativ mit einer „und“- bzw. einer „oder“-Bedingung miteinander verknüpft werden. Werden keine Datensätze gefunden, die den jeweiligen Filterkombinationen entspricht, wird die Listenansicht leer dargestellt. Mittels „Rückgängig“ kann die ursprüngliche Listenansicht wieder hergestellt werden. In Abbildung 47 wurde die Spalte „Gerätetyp“ markiert und der Filter „entspricht“ mit dem Suchbegriff „LStE“ gewählt. Die Filterfunktion wird ebenso wie die Suchfunktion nicht Case-Sensitiv gestaltet.

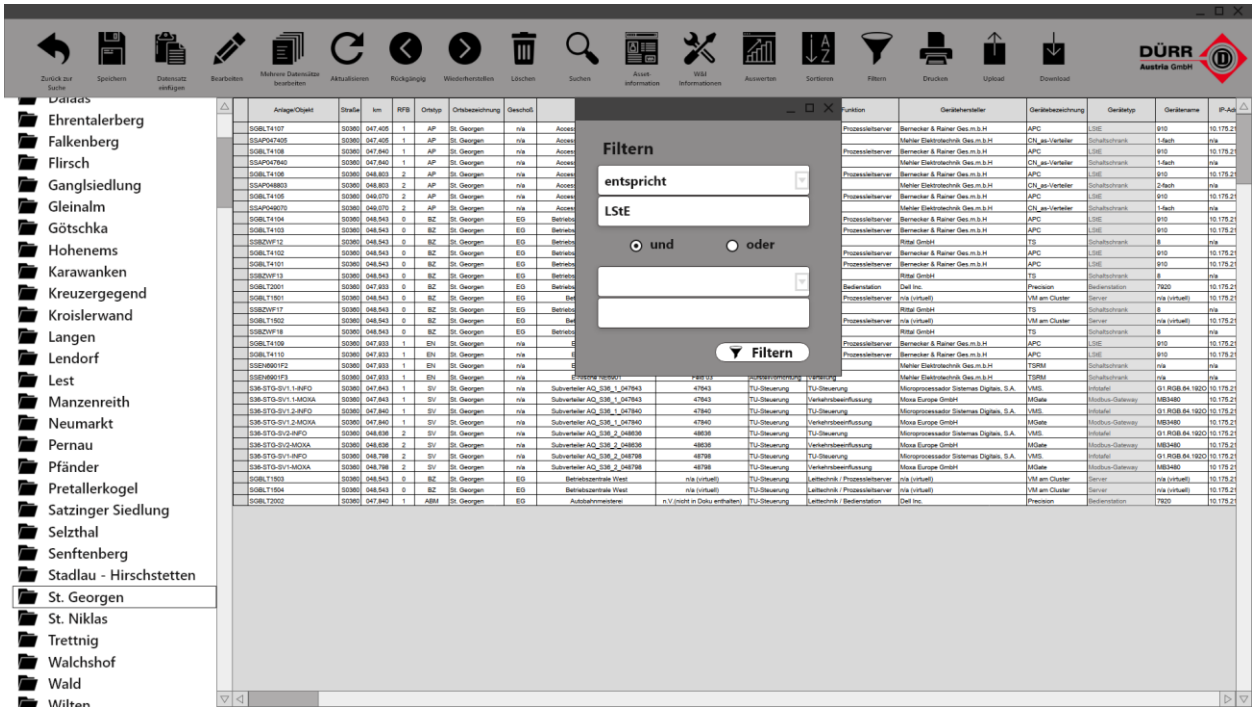

Abbildung 47: UI-Mockup Filtermöglichkeiten, Quelle: Eigene Darstellung.

6.14 Drucken der Listenansicht

Der Klick auf das Drucksymbol **„Drucken“** öffnet das Druckmenü des Druckers, der am jeweiligen Rechner als Standarddrucker installiert ist. Abbildung 48 stellt exemplarisch eine Druckeransicht dar. Abhängig von diesem Drucker sind auch die jeweiligen Druckfunktionen, die in der Datenbank ausgeführt werden können. Ist auf dem Rechner ein PDF-Drucker installiert, kann auch dieser ausgewählt werden, um eine PDF-Datei der jeweiligen Liste zu erstellen.

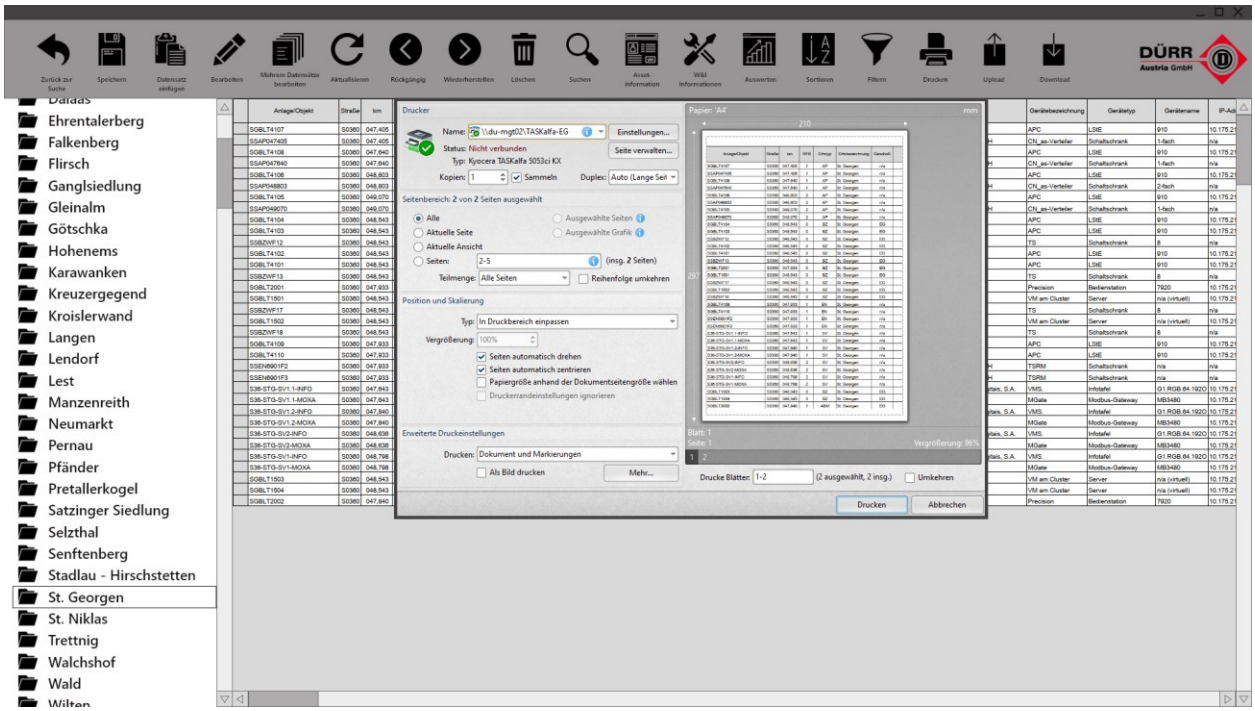

Abbildung 48: UI-Mockup Druckansicht, Quelle: Eigene Darstellung.

6.15 Upload von Dateien und Daten

Die Upload Funktion von Dateien und Daten ist nur für Benutzer*innen mit Administrator- oder Schreibrechten aktiv. Wird ein bestimmtes Asset mittels Auswahlfeld am linken Rand ausgewählt, kann zu diesem Asset mit **„Upload“**, wie in Abbildung 49 dargestellt, ein Dokument hochgeladen werden, welches darauffolgend in der Assetansicht angezeigt wird bzw. von dort geöffnet werden kann. Hinzugefügt werden können zu einzelnen Assets Dokumente in den Formaten:

- DOC bzw. DOCX
- XLS bzw. XLSX
- ODT bzw. ODF
- ODS
- PPT bzw. PPTX bzw. PPS
- CSV

Diese Funktion soll die Möglichkeit bieten, Datenblätter und Informationen der Hersteller, sowie interne Dateien wie Lieferscheine, Schwachstellenanalysen, Information Security Management System-Dokumente (ISMS-Dokumente) etc. zu hinterlegen.

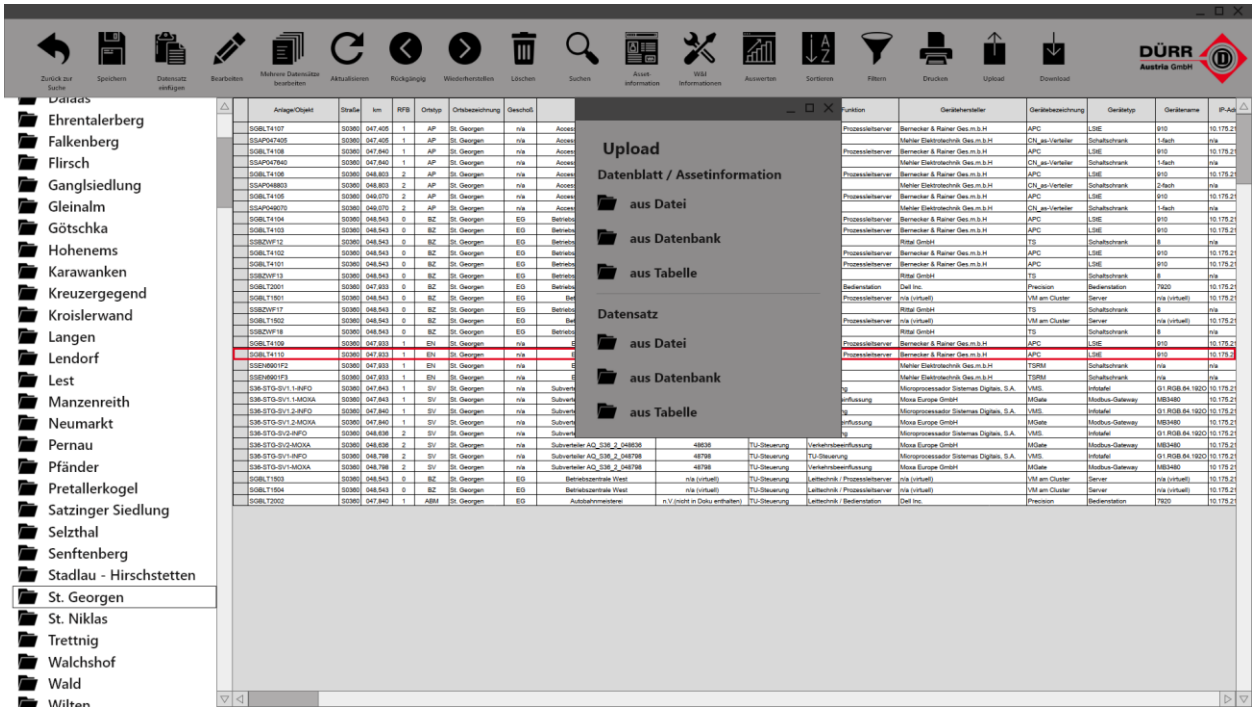

Abbildung 49: UI-Mockup Upload, Quelle: Eigene Darstellung.

Wird beispielsweise im Abschnitt Datenblatt / Assetmanagement „Upload aus Datei“ gewählt, öffnet sich, wie in Abbildung 50 ersichtlich, die Windows Ordneransicht, mit der Möglichkeit das gewünschte Dokument zu wählen. Wird dieses Pop-up mit „Öffnen“ bestätigt, wird das gewählte Dokument dem ausgewählten Asset hinzugefügt. Wird die Auswahl abgebrochen, kehrt das System zur Uploadansicht zurück.

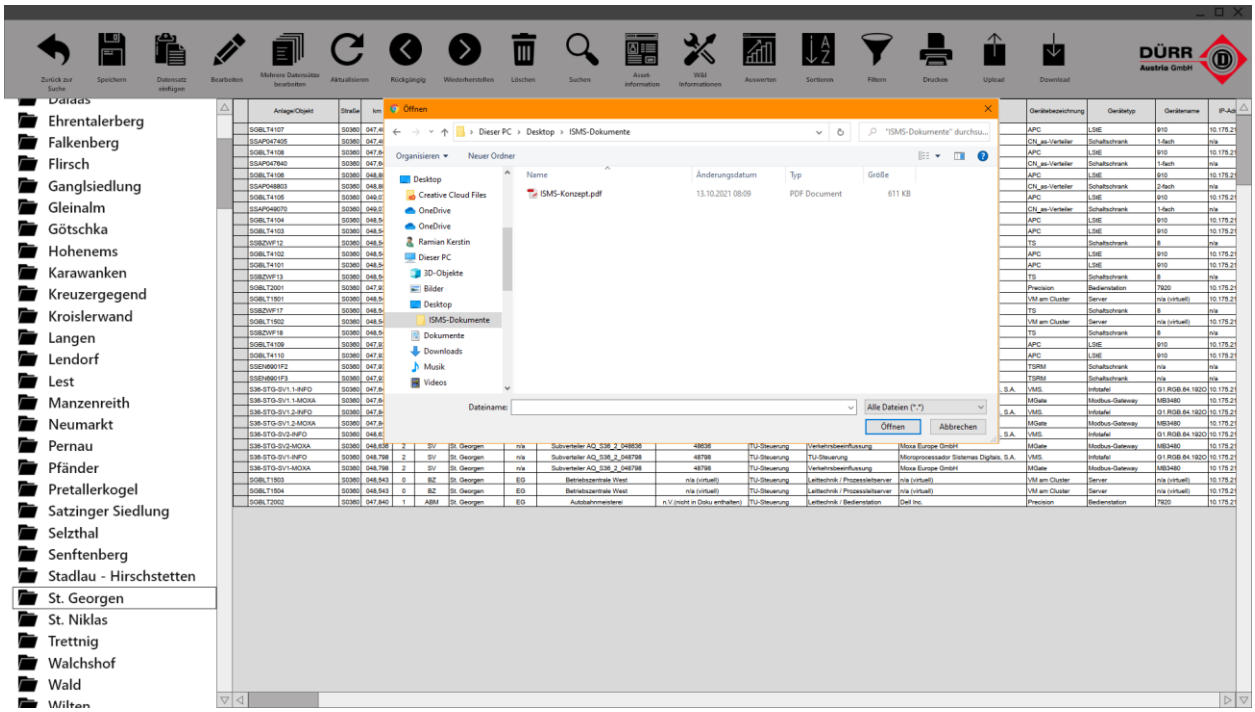

Abbildung 50: UI-Mockup Upload aus Datei, Quelle: Eigene Darstellung.

Wird beispielsweise im Abschnitt Datensatz „Upload aus Tabelle“ gewählt, öffnet sich, wie in Abbildung 51 dargestellt, die Windows Ordneransicht, mit der Möglichkeit die gewünschte Tabelle zu wählen. Wird dieses Pop-up mit „Öffnen“ bestätigt, werden die Daten des gewählten Dokumentes der geöffneten Anlage hinzugefügt. Wird die Auswahl abgebrochen, kehrt das System zur Uploadansicht zurück. Um die Daten korrekt hinzufügen zu können ist es notwendig, dass das gewählte Dokument in den folgenden Formaten gespeichert wurde:

- XLS oder XLSX
- CSV

Die Referenzen zu den einzelnen Spaltenüberschriften müssen übereinstimmen. Es wird zur Sicherstellung der Konformität im Unternehmen eine vorgegebene Vorlage für die Erfassung der Daten im Dokument verwendet.

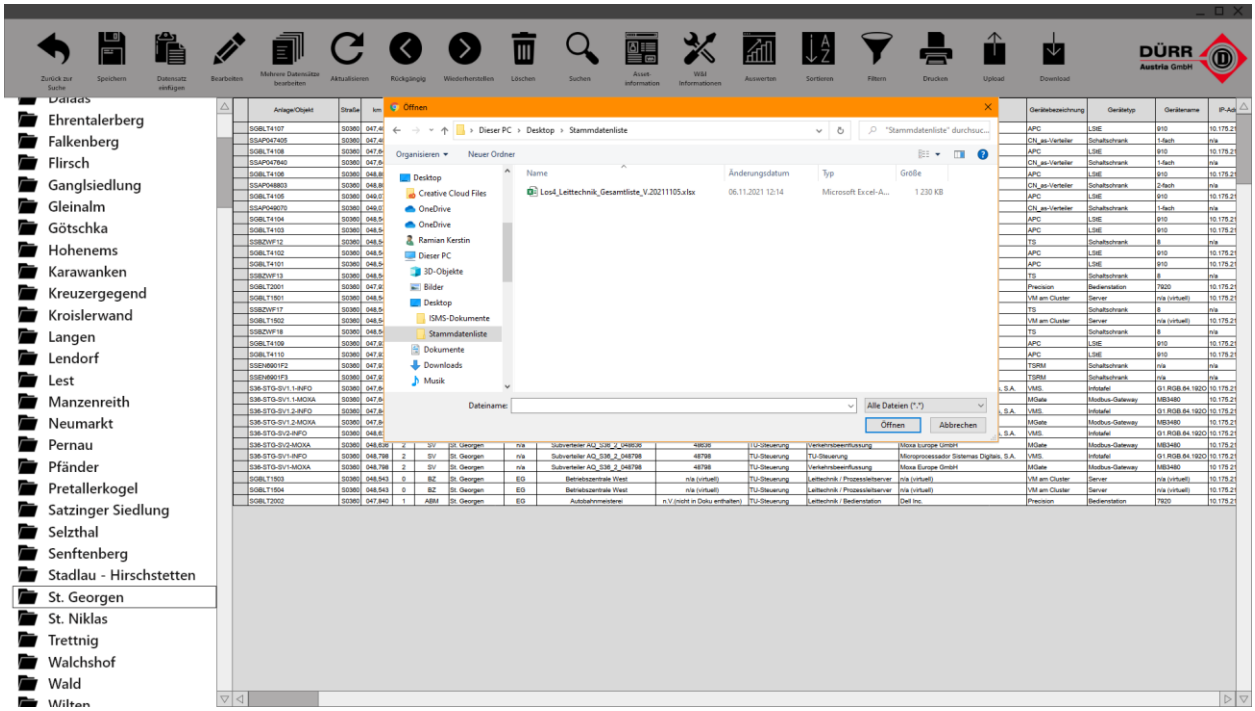

Abbildung 51: UI-Mockup Upload aus Tabelle, Quelle: Eigene Darstellung.

Werden Datensätze einer Anlage hinzugefügt, findet im Hintergrund eine Objektidentifizierung und Duplikaterkennung statt. Wird ein Duplikat erkannt, wird dieser Datensatz farblich markiert und es kann, wie in Abbildung 52 ersichtlich, einzeln die Entscheidung getroffen werden, ob der Datensatz geändert, oder der aktuelle Datensatz beibehalten werden soll. Ebenso können alle Datensätze gesammelt aktualisiert oder beibehalten werden.

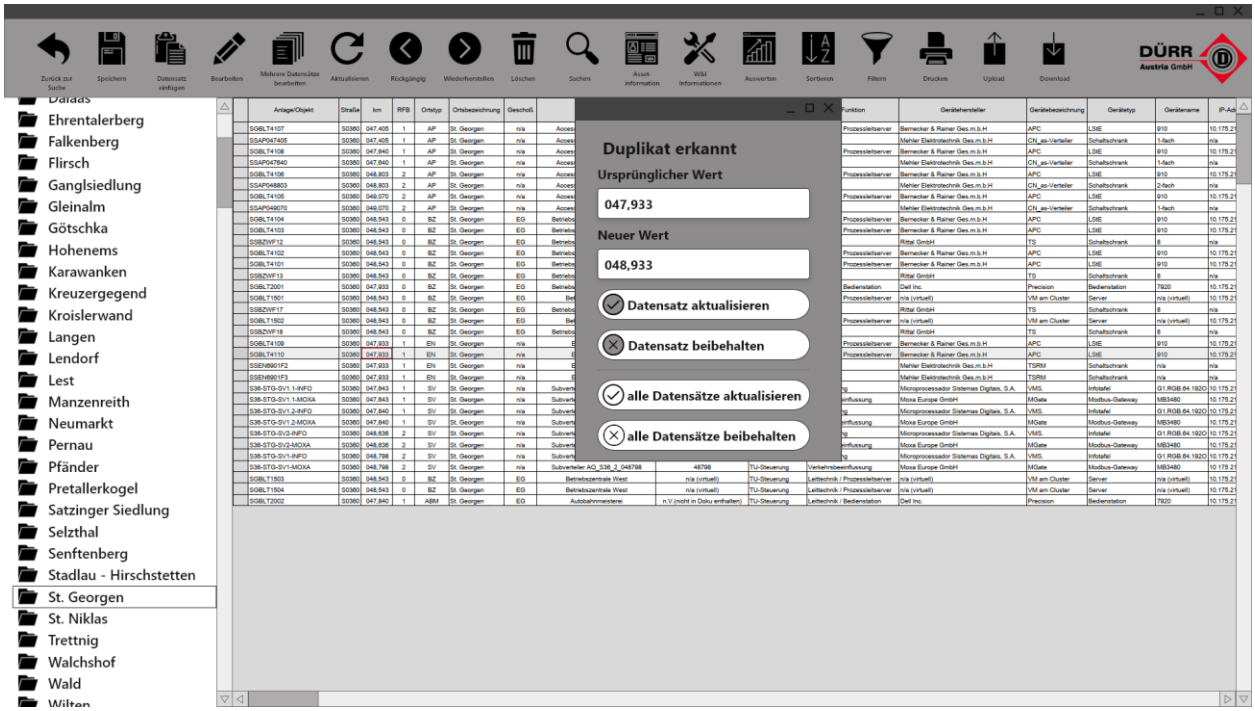

Abbildung 52: UI-Mockup Duplikaterkennung, Quelle: Eigene Darstellung.

Wesentliche Referenz ist hierfür die Anlagenbezeichnung in Kombination mit dem Hostnamen und der IP-Adresse. Diese sind je Anlage einzigartig und kommen in dieser Zusammensetzung nur einmalig vor.

6.16 Download der Ergebnisliste

Mit der **„Download“**-Funktion ist die Möglichkeit gegeben, die aktuell ausgegebene Liste der Listenansicht als Dokument zu speichern. Hierfür kann ein Bereich der Liste ausgewählt werden oder, wenn keine Auswahl getroffen wurde, die gesamte Liste exportiert werden. Wie in Abbildung 53 ersichtlich, bietet diese Funktion die folgenden drei Möglichkeiten die Liste zu speichern:

- PDF erzeugen
- Microsoft Excel Dokument erzeugen
- CSV Dokument erzeugen

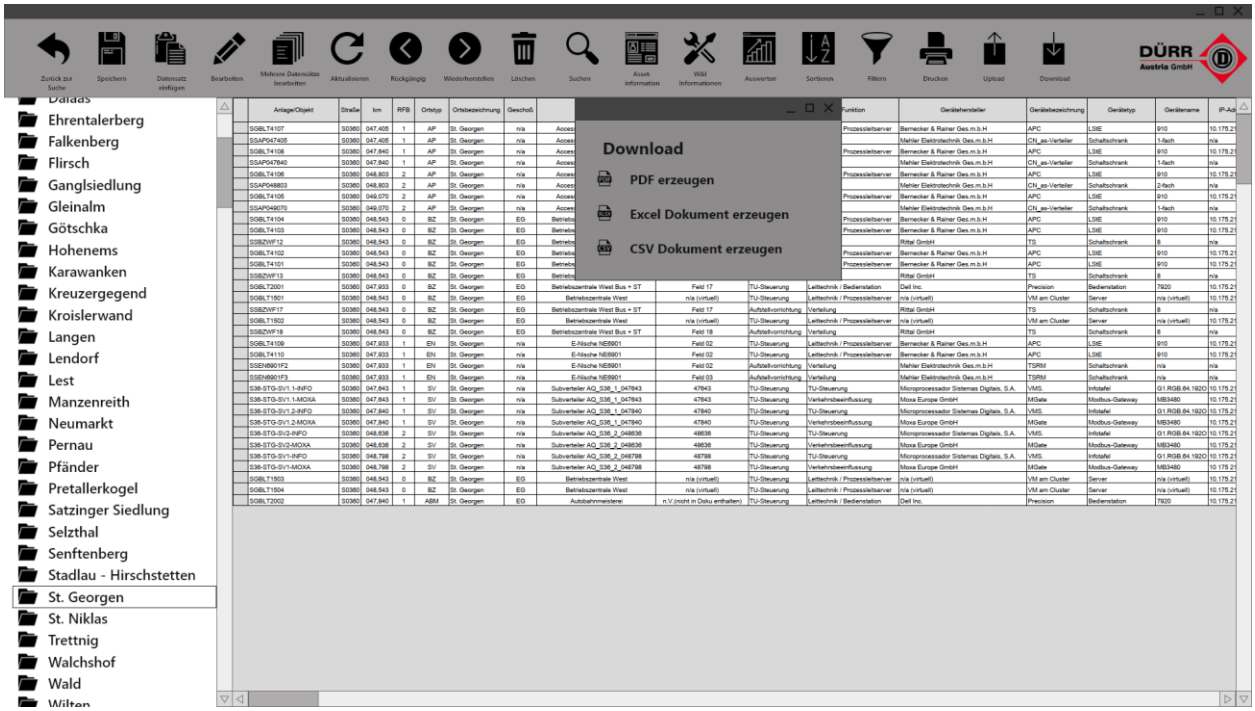

Abbildung 53: UI-Mockup Download, Quelle: Eigene Darstellung.

Nachdem der gewünschte Dateityp gewählt wurde, öffnet sich die Windows Ordneransicht, wie sie exemplarisch in Abbildung 54 dargestellt ist. Hier wird der gewünschte Speicherort, sowie der Dateiname festgelegt und „Speichern“ gewählt. Daraufhin kehrt das Programm wieder zur Listenansicht zurück. Wird die Ordneransicht stattdessen geschlossen, kehrt das Programm zum Downloadmenü zurück.

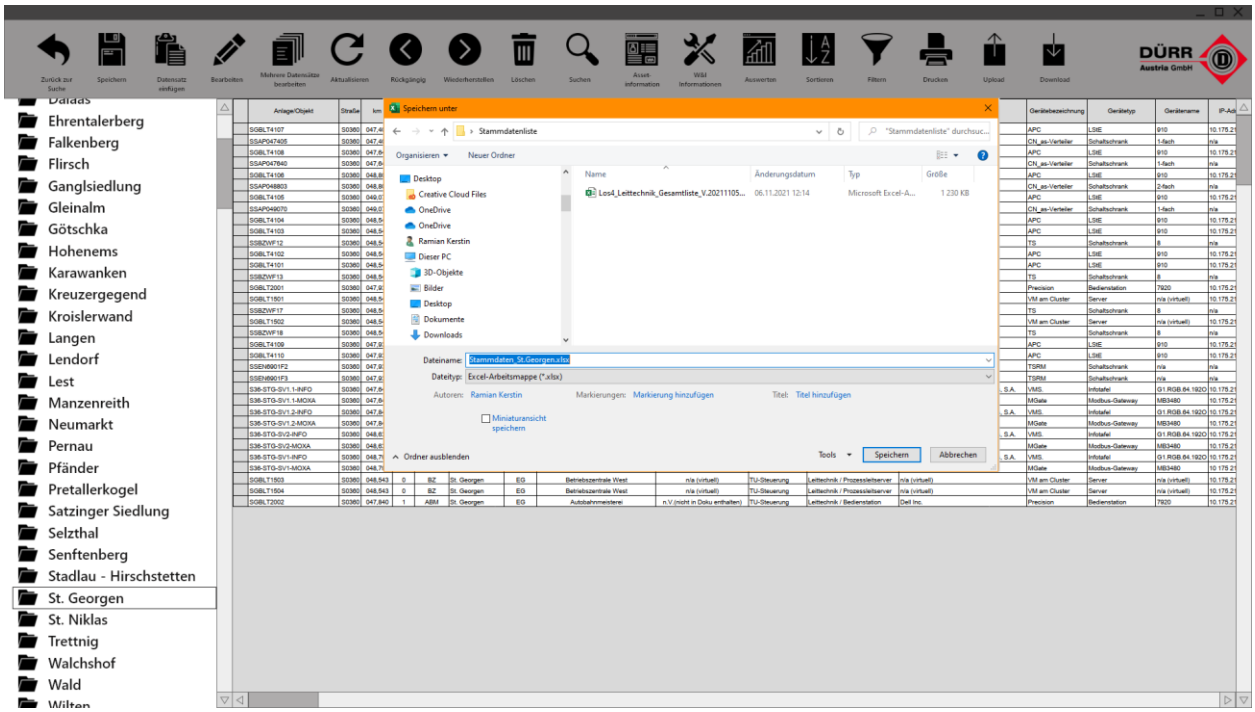

Abbildung 54: UI-Mockup Download Speicherort und Dateiname festlegen, Quelle: Eigene Darstellung.

6.17 Wechseln der Anlage innerhalb der Listenansicht

Am linken Rand der Listenansicht befindet sich eine vollständige Liste aller in der Datenbank befindlichen Anlagen. Der Aufbau ähnelt der Ordneransicht im Windows Betriebssystem. An dieser Stelle kann mittels Doppelklick auf die Anlagenbezeichnung direkt auf die vollständige Liste einer beliebigen Anlage gesprungen werden. Es werden in der erzeugten Liste alle in der ausgewählten Anlage verbauten Assets angezeigt. In Abbildung 55 ist der Wechsel zur Anlage „Gleinalm“ grafisch dargestellt.

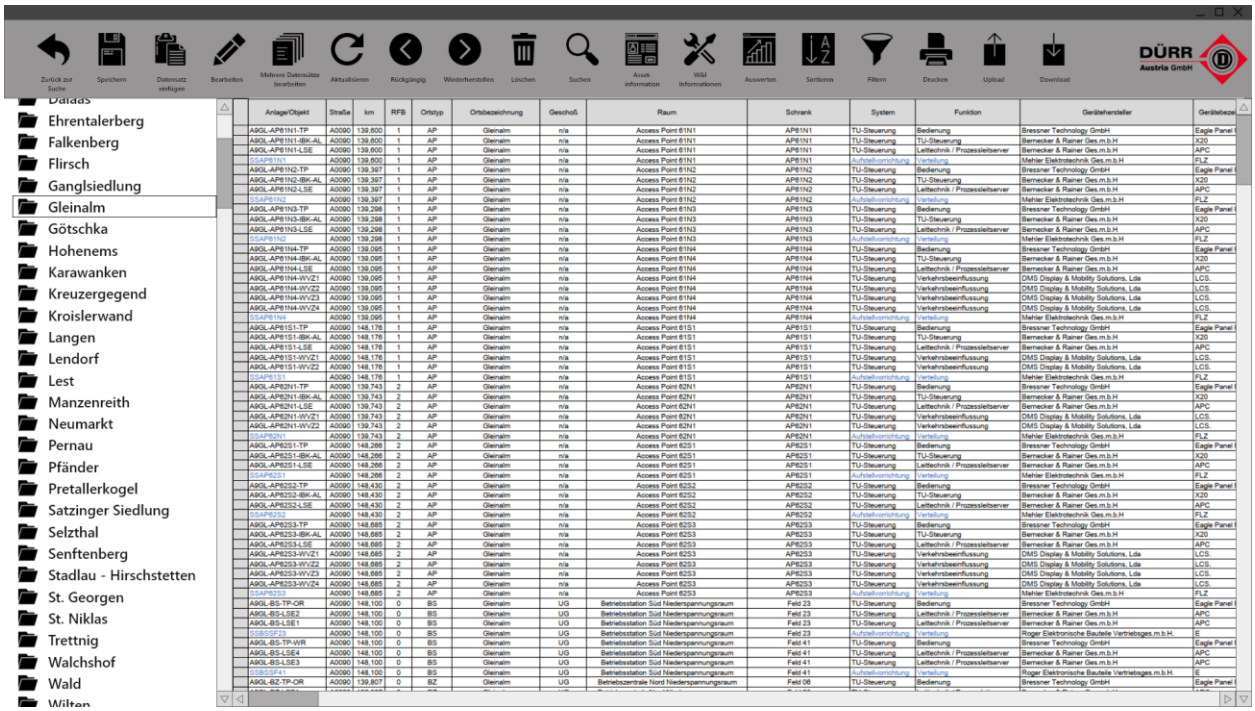

Abbildung 55: UI-Mockup Anlagenwechsel, Quelle: Eigene Darstellung.

6.18 Suchansicht W&I / Ersatzteilmanagement

Die Suchansicht für das W&I / Ersatzteilmanagement ermöglicht die Asset-Suche nach bestimmten, für die Wartung und Instandhaltung sowie das Ersatzteilmanagement essenziellen Suchwerten. Gewählt werden hier ebenfalls jene, die am aussagekräftigsten sind bzw. die am häufigsten benötigt werden.

Folgende Suchbegriffe können einzeln oder in Kombination gewählt werden:

- Anlagenname
- Straße
- Ortstyp
- Hostname
- IP-Adresse
- Gerätehersteller
- Gerätetyp
- Gerätebezeichnung

- Gerätename
- Wartungstyp
- Datum letzte Instandhaltung
- Ersatzteil
- Reparatur

Die Datumsangabe kann, wie bei der Suchansicht des Asset-Managements, händisch im Format TT.MM.JJJJ eingegeben werden oder alternativ in einem Pop-up-Kalender ausgewählt werden. Die Befüllung der restlichen Werte erfolgt, analog zur Suchanfrage im Asset-Management, mit Hilfe von Drop-Down Feldern oder unterstützt durch eine Autocomplete-Funktion, um die Suche nach nicht vorhandenen Schreibweisen der Suchbegriffe zu vermeiden. Es ist ebenfalls die Möglichkeit gegeben, zum Starbildschirm zurückzukehren. Abbildung 56 veranschaulicht die W&I / Ersatzteilmanagement Suchansicht.

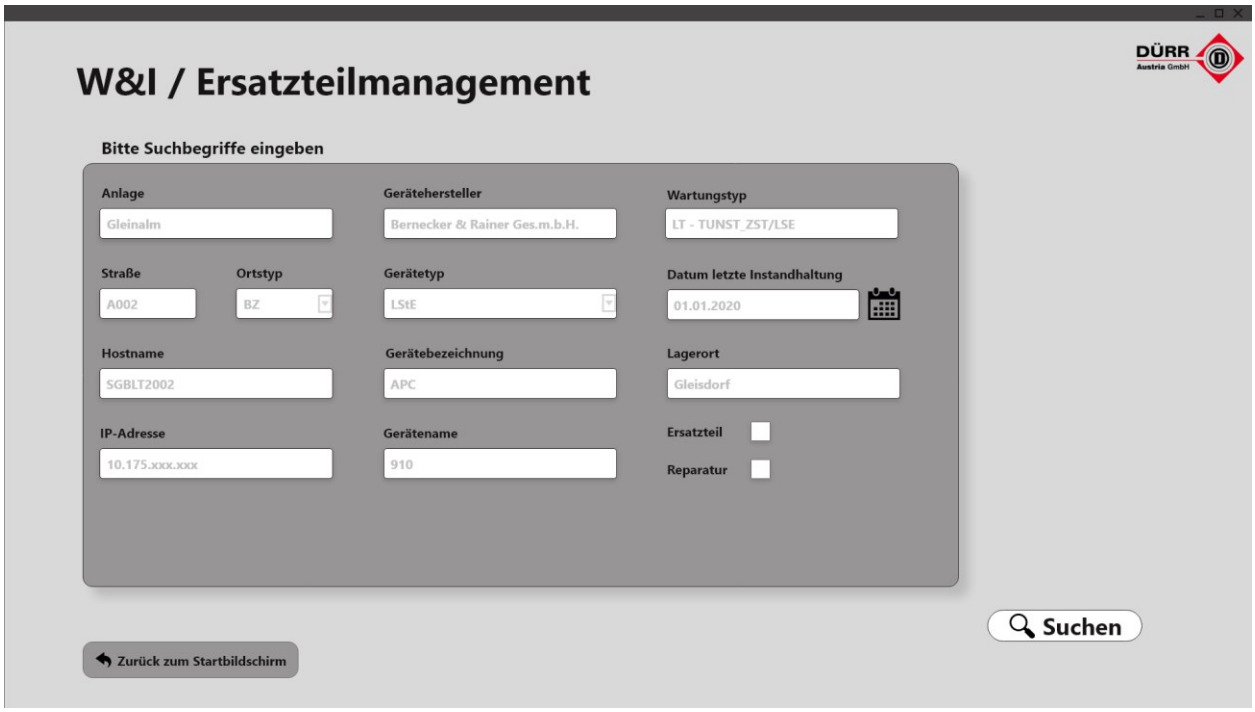

Abbildung 56: UI-Mockup W&I Suchansicht, Quelle: Eigene Darstellung.

Durch Klick auf den Suchbutton öffnet sich die W&I / Ersatzteil-Listenansicht mit den Ergebnissen aus der Suche. Konnten keine übereinstimmenden Datensätze gefunden werden, erscheint ein Pop-up mit dem Hinweis: „Keine Datensätze ermittelt.“, welches in Abbildung 57 ersichtlich ist.

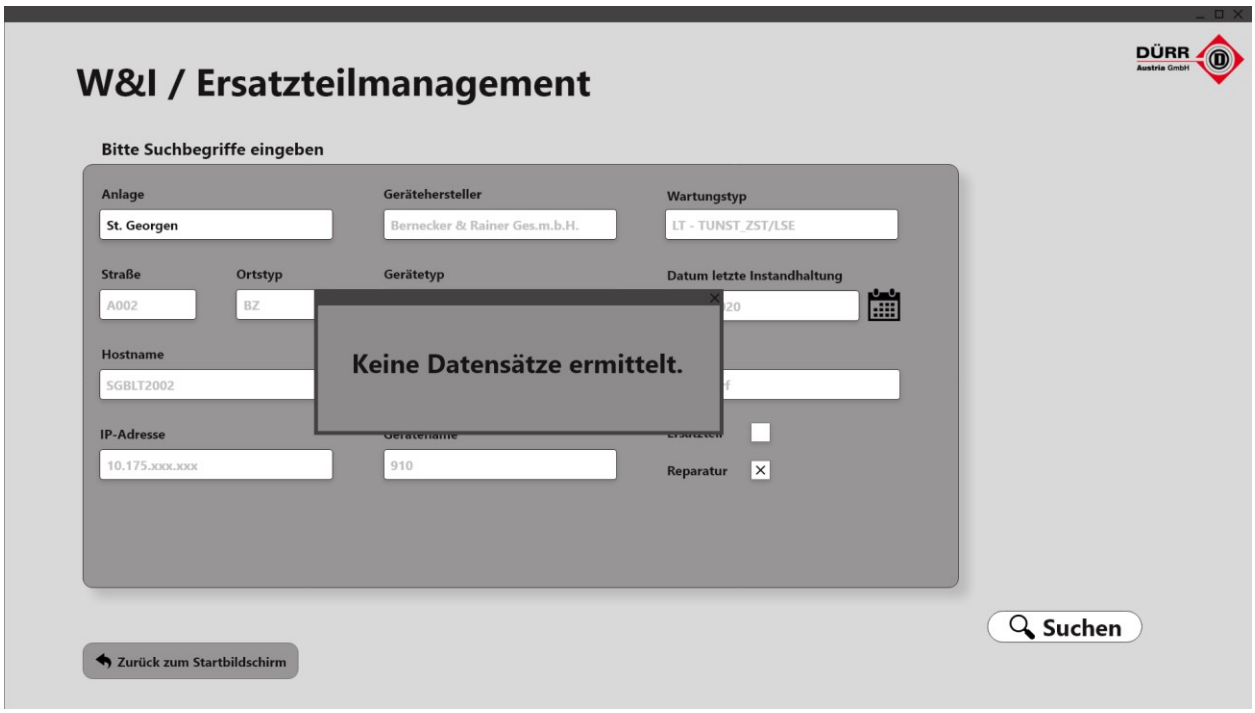

Abbildung 57: UI-Mockup W&I Suchansicht Fehlermeldung, Quelle: Eigene Darstellung.

6.19 Listenansicht nach Suchanfrage W&I

Der Aufbau der Listenansicht nach einer Suchanfrage im W&I / Ersatzteilmanagement ist, wie in Abbildung 58 erkennbar ist, identisch mit jenem des Asset-Managements. Sämtliche Menüpunkte in der Menüleiste haben dieselben Funktionen abhängig von der jeweiligen Benutzerberechtigung.

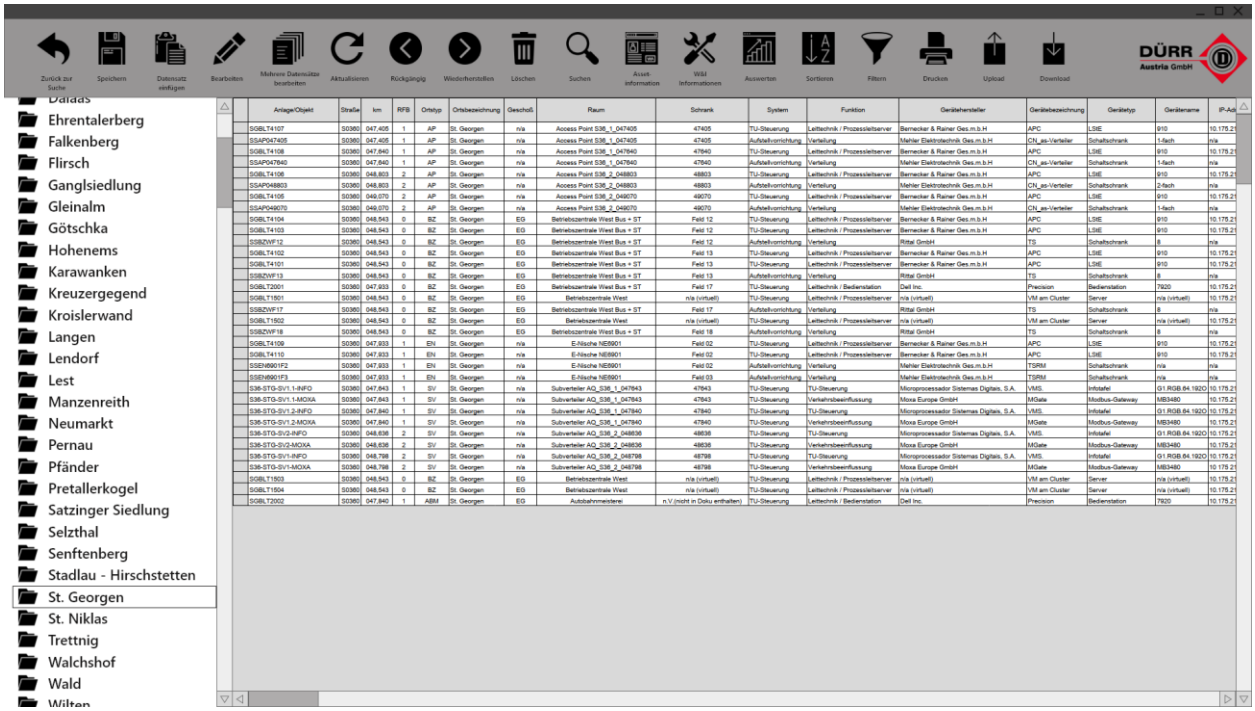

Abbildung 58: UI-Mockup W&I Listenansicht, Quelle: Eigene Darstellung.

Aufgrund von prozessübergreifenden Verbindungen zwischen dem Asset-Management und dem W&I / Ersatzteilmanagement, ist es in beiden Bereichen möglich, sich die Daten des jeweils anderen anzeigen zu lassen.

6.20 Detaillierte W&I Informationen

Mit Klick in eine Zelle oder die Zeile des gewünschten Assets wird das Asset gewählt und daraufhin der Menüpunkt **„W&I Informationen“** ausgewählt. Ein Pop-up erscheint mit allen verfügbaren Informationen zum jeweiligen Asset:

- Assetstatus (bestellt / verbucht / geliefert / einbaufertig / verbaut)
- Anlagenname
- Straße
- Kilometer
- Richtungsfahrbahn (RFB)
- Ortstyp
- Geschoß
- Raum
- Schrank
- Hostname
- IP-Adresse
- Gerätehersteller
- Gerätebezeichnung
- Gerätename
- Gerätetyp
- Redundanz zu
- Seriennummer
- Wartungstyp
- Ersatzteil
- Reparatur
- Inbetriebnahmedatum
- Beginn Gewährleistung Lieferant
- Ende Gewährleistung Lieferant
- Beginn Gewährleistung Kunde
- Ende Gewährleistung Kunde
- Anmerkungen
- Datenblätter / Wartungskarten

Aufgrund der Menge an Informationen, werden diese auf zwei Seiten aufgeteilt. Mittels Pfeiltaste im Pop-up kann zwischen diesen Seiten hin und her gesprungen werden. Statusänderungen können mit Hilfe eines Drop-down Menüs vorgenommen werden. Bei Reparaturen wird der Status auf „verbucht“ zurückversetzt.

Abbildung 59 und Abbildung 60 veranschaulichen die zwei Seiten der detaillierten W&I Informationen.

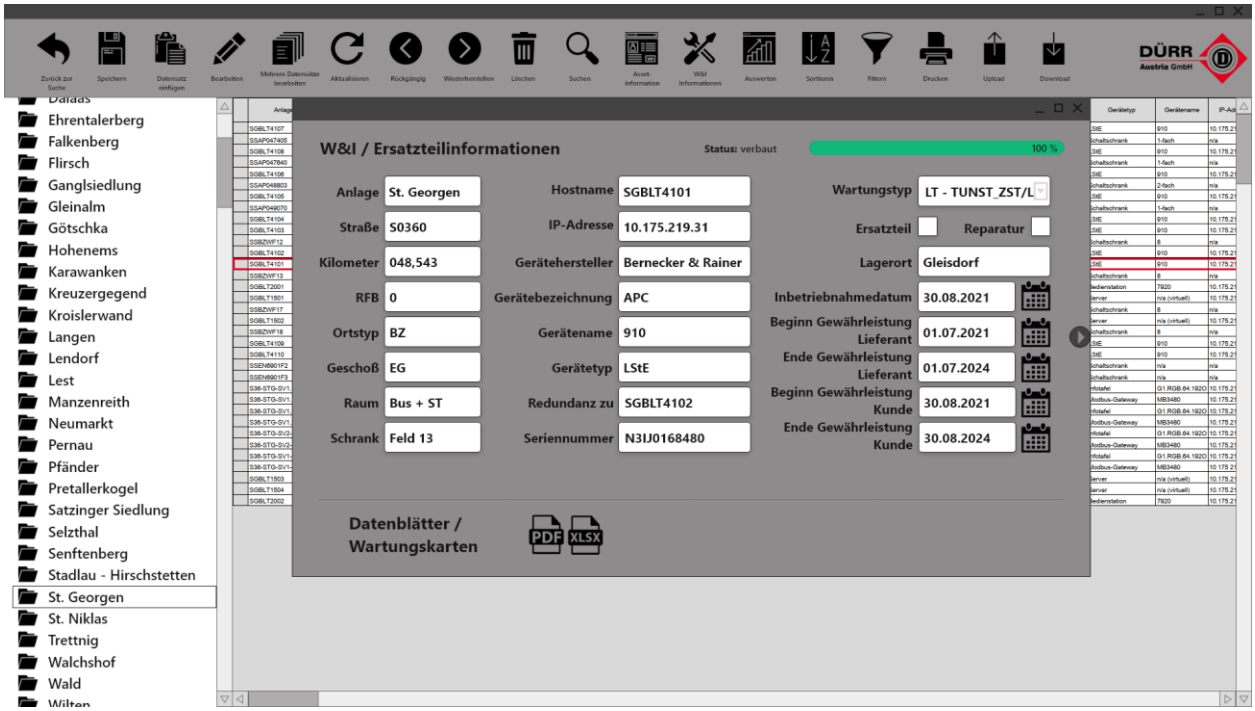

Abbildung 59: UI-Mockup detaillierte W&I Informationen Seite 1, Quelle: Eigene Darstellung.

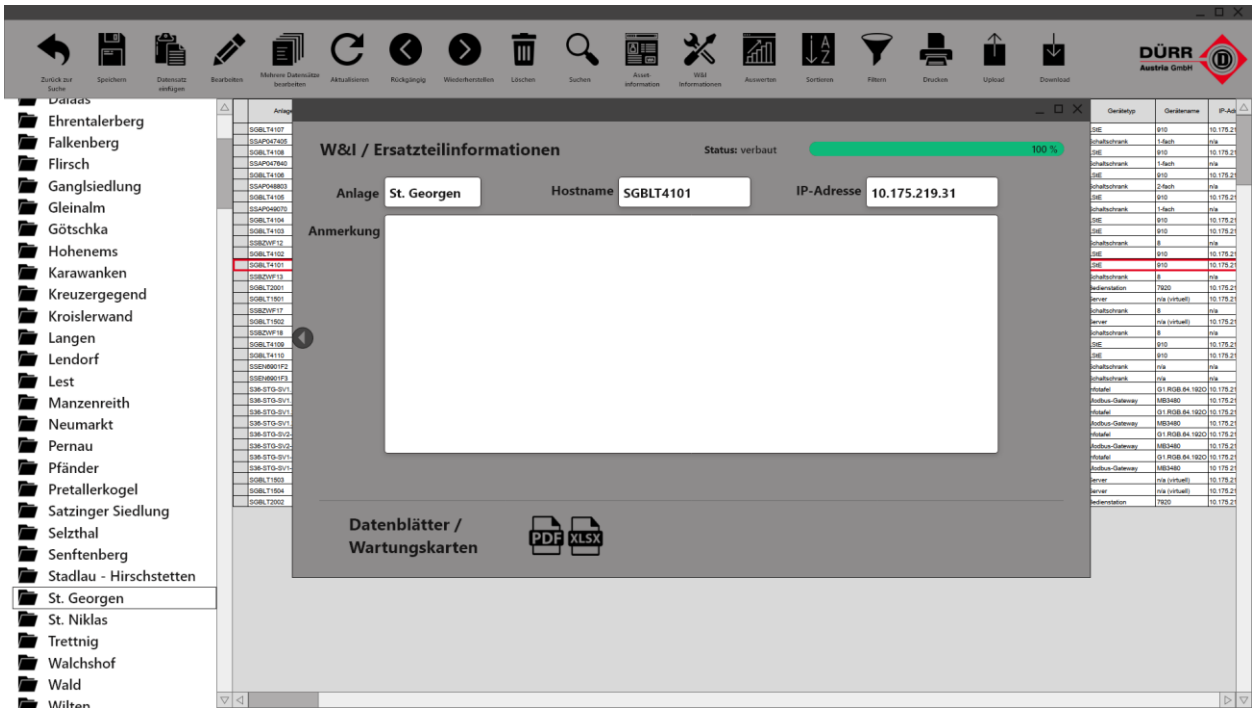

Abbildung 60: UI-Mockup detaillierte W&I Informationen Seite 2, Quelle: Eigene Darstellung.

In Abbildung 61 wird der Unterschied zwischen vollständigen W&I Informationen und Ersatzteilinformationen verdeutlicht. Aufgrund der Tatsache, dass das Asset als Ersatzteil dient, fehlt ein großer Teil der Daten. Werden diese anlagenspezifischen Daten ergänzt und die Auswahl bei „Ersatzteil“ entfernt, geht das Asset automatisch in das Asset-Management über. Wesentlich für die Übernahme ist die Befüllung der Felder Anlage, Hostname und IP-Adresse. Werden Datenblätter ausgewählt, öffnen sich diese in der jeweiligen, auf dem Rechner installierten, Software gemäß Dateityp.

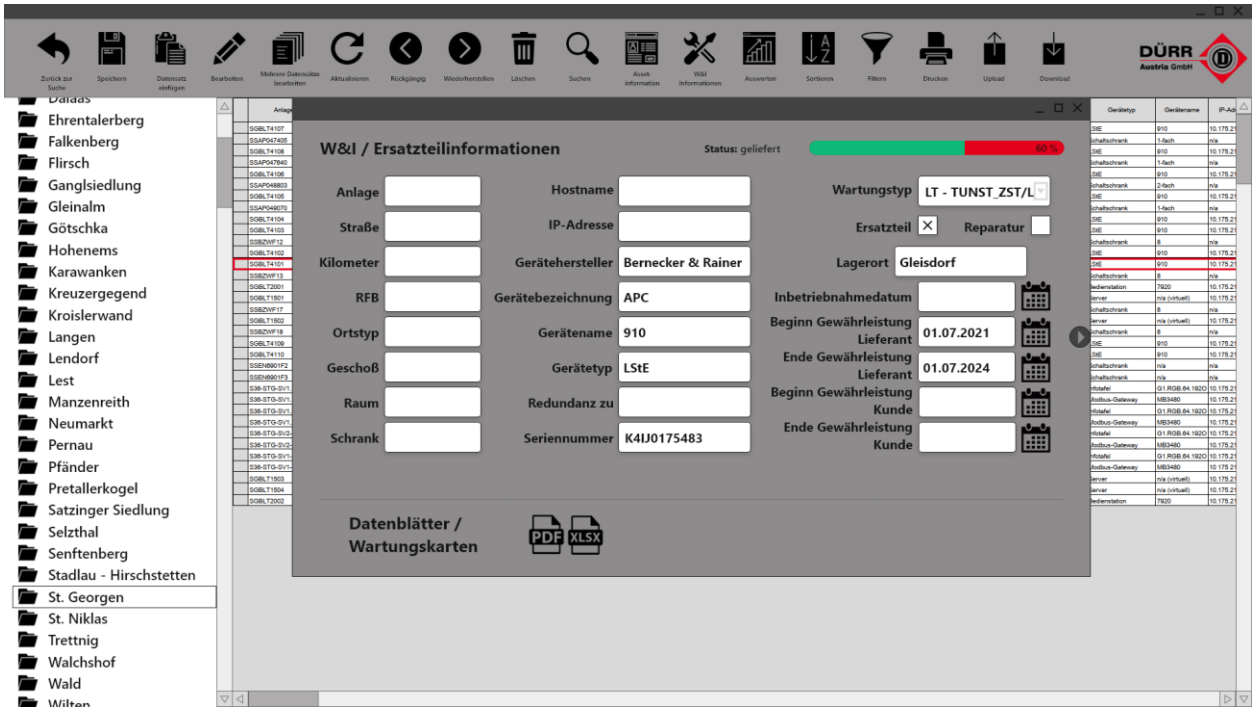

Abbildung 61: UI-Mockup Ersatzteilinformationen Seite 1, Quelle: Eigene Darstellung.

7 RESÜMEE UND AUSBLICK

Die Datenbanklösung dient hauptsächlich der Qualitätssicherung von internen Prozessen und der einfacheren Nachvollziehbarkeit und Berichterstattung zum Auftraggeber von Wartungs- und IT-Serviceleistungen. Sie soll als Unterstützung dienen, das Life Cycle Management im Unternehmen NISG-konform umzusetzen.

Eine Amortisationsrechnung für die Erstellung der Datenbank ist aus Sicht des Unternehmens nicht zielführend. Aufgrund der Tatsache, dass ein großer Teil der Daten von bestehenden Anlagen bereits im Zuge eines Projektes erfasst wurde, ist hierfür kein zusätzlicher Aufwand notwendig. Die Erfassung zusätzlicher leittechnischer Assets wird zu einem späteren Zeitpunkt erfolgen.

7.1 Anwendbarkeit des Datenbankmodellierungsansatzes

Dank der Verwendung von Adobe XD für die Erstellung des UI-Mockups, konnten mit Hilfe von einzelnen Verknüpfungen die Anwendung und die einzelnen Prozessschritte praxisnah simuliert werden. Jede Funktionalität erhielt ein eigenes Layout und mehrere Layouts konnten miteinander in Beziehung gebracht werden. Auf diese Weise konnte jeder Klick auf eine Schaltfläche nachgestellt werden. Sämtliche Ansichten und die Menüführung wurden auf diese Weise gemeinsam mit einigen Kollegen erprobt und ein großer Teil der Anmerkungen wurde bereits während der Erstellung des Konzeptes der Benutzerschnittstelle eingearbeitet.

Aus gegenwärtiger Sicht kann davon ausgegangen werden, dass der gewählte Ansatz und die Anwendungsmöglichkeiten anwendbar ist und das User Interface in eine funktionierende Datenbanklösung übergeleitet werden kann.

Ob es sich bei der Datenbank um eine Open-Source-Lösung oder Microsoft SQL Server handeln wird, wurde noch nicht entschieden. Der Sicherheitsaspekt muss hierbei in jedem Fall berücksichtigt werden, da es sich bei den erfassten Komponenten um wesentliche Bestandteile der kritischen Infrastruktur handelt und eine Sicherheitslücke in diesem System maßgeblichen Einfluss auf die Sicherheit der betroffenen Anlagen haben kann. Der Schutz der gespeicherten Daten vor unzulässigem Zugriff hat oberste Priorität.

Im Zuge der Anwendung der fertiggestellten Datenbanklösung in der Praxis wird mit Hilfe eines kontinuierlichen Verbesserungsmanagements regelmäßig Verbesserungspotential evaluiert und sukzessive implementiert werden. Welche exakten Auswertungen zukünftig benötigt werden, wird sich im Zuge der Verwendung herauskristallisieren.

7.2 Ausblick

Die Programmierung und Implementierung der Datenbanklösung wurden bereits parallel zur Erstellung des Konzeptes und des UI-Mockups initiiert. Der Start der Umsetzung steht unmittelbar bevor. Es soll ein Webinterface geschaffen werden, das den Mitarbeiter*innen die Möglichkeit gibt, auf die Datenbank zuzugreifen.

Zukünftig soll die Datenbanklösung Verknüpfungen bzw. Schnittstellen zu anderen im Unternehmen genutzten Lösungen ermöglichen. Insbesondere zum verwendeten ERP-System der KWP Informationssysteme GmbH sowie zum Building Information Modeling (BIM) System soll eine Schnittstelle geschaffen werden. Eine Verbindungsmöglichkeit zur Baudokumentationssoftware „Plan Radar“, welche insbesondere bei der Wartung und Instandhaltung eingesetzt wird, wird zukünftig ebenfalls geprüft werden. Die Datenbanklösung soll sich als Mittelpunkt dieser einzelnen Anwendungen als das Tool zum strukturierten und lückenlosen Asset-Management etablieren.

Eine Kalenderfunktion im Hintergrund soll die Möglichkeit schaffen, ein Pop-up anzeigen zu lassen, dass bestimmte Komponenten gepacht oder gewartet werden müssen mit beispielsweise einem Monat Vorlaufzeit, um interne Ressourcen rechtzeitig planen zu können.

Darüber hinaus ist zukünftig die Erstellung einer Handyapplikation geplant, die auf der Anlage die Möglichkeit bietet, mittels QR-Code, welcher bereits in der Vorbereitungsphase für jedes Asset generiert und auf den Geräten angebracht wird, die Verwaltung insbesondere bei Wartungs- und Instandhaltungstätigkeiten sowie IT-Betriebsleistungen zu erleichtern. Der QR-Code wird gescannt und auf der Handyapplikation erscheinen automatisch sämtliche gespeicherten Asset-Daten. Wird ein Gerät gewartet oder gepatcht, kann dies ausgewählt werden und das jeweilige Wartungs- oder Patchdatum wird in der Datenbank aktualisiert. Sofern ein Gerät gegen ein anderes ersetzt wird, ist dies ebenfalls mittels Scan des alten sowie neuen QR-Codes möglich und in der Datenbank werden die zugehörigen Daten aktualisiert.

Jede Information in einem Asset soll eine eigene Archivfunktionalität erhalten, bei der jederzeit rekonstruiert werden kann, welche Änderungen bei dem Asset durchgeführt und durch wen diese Änderungen ausgeführt wurden. So sollen beispielsweise die letzten fünf Änderungen der Softwareversion oder des Instandhaltungsdatums lückenlos nachvollziehbar sein.

LITERATURVERZEICHNIS

Gedruckte Werke (14)

Boerger, Jeremy (2021): *Rethinking Information Technology Asset Management*, Business Expert Press LLC,

Chlebek, Paul (2011): *Praxis der User Interface-Entwicklung, Informationsstrukturen, Designpatterns, Vorgehensmuster*, Vieweg + Teubner Verlag

Crespo, Márquez; Macchi, Macchi; Kumar, Parlikad (2020): *Value Based and Intelligent Asset Management, Mastering the Asset Management Transformation in Industrial Plants and Infrastructures*, Springer Nature Switzerland AG

Gronwald, Klaus-Dieter (2020): *Integrierte Business-Informationssysteme, Ganzheitliche, geschäftsprozessorientierte Sicht auf die vernetzte Unternehmensprozesskette ERP, SCM, CRM, BI, Big Data Analytics*, 3. Auflage, Springer-Verlag GmbH Deutschland

Grunert, Hannes; Heuer, Andreas; Meyer, Holger; Saake, Gunter; Sattler, Kai-Uwe (2020): *Datenbanken, Kompaktkurs*, mitp Verlags GmbH & Co KG, Frechen

Hartmut, Ernst; Jochen, Schmidt; Gerd, Beneken (2016): *Grundkurs Informatik, Grundlagen und Konzepte für die erfolgreiche IT-Praxis - Eine umfassende, praxisorientierte Einführung*, 6. Auflage, Springer Fachmedien Wiesbaden

Kersken, Sascha (2019): *IT-Handbuch für Fachinformatiker, Der Ausbildungsbegleiter*, 9. Auflage, Rheinwerk Verlag GmbH, Bonn

Kleppmann, Martin (2019): *Datenintensive Anwendungen designen, Konzepte für zuverlässige, skalierbare und wartbare Systeme*, dpunkt.verlag GmbH

Kurowski, Oliver (2012): *NoSQL Einführung, CouchDB, MongoDB und Redis*, Software & Support Media GmbH

Malewicz, Michal; Malewicz, Diana (2020): *Designing User Interfaces*

Meier, Andreas; Kaufmann, Michael (2016): *SQL- & NoSQL-Datenbanken*, Springer-Verlag

Piepmeyer, Iothar (2011): *Grundkurs Datenbanksysteme, Von den Konzepten bis zur Anwendungsentwicklung*, Carl Hanser Verlag München

Schneider, Markus (2004): *Implementierungskonzepte für Datenbanksysteme*, Springer-Verlag Berlin Heidelberg GmbH

Theis, Thomas (2021): *Microsoft Access für Einsteiger, Datenbanken entwerfen und entwickeln*, 2. Auflage, Rheinwerk Verlag GmbH, Bonn

Online-Quellen (32)

Amazon Web Services Inc. (2021): *Amazon Aurora* https://aws.amazon.com/de/rds/aurora/?aurora-whats-new.sort-by=item.additionalFields.postDateTime&aurora-whats-new.sort-order=desc [Stand: 11.11.2021]

Begerow Beratungsgesellschaft mbH & Co. KG (1) (2021): *Key/Value-Datenbanksysteme*
https://datenbanken-verstehen.de/lexikon/key-value-datenbanksysteme/ [Stand: 16.09.2021]

Begerow Beratungsgesellschaft mbH & Co. KG (2) (2021): *Graphdatenbanken*
https://www.datenbanken-verstehen.de/lexikon/graphdatenbanken/ [Stand: 16.09.2021]

Craig S. Mullins (2017): *Die Funktionen des relationalen Datenbank-Management-Systems (RDBMS) IBM DB2*
https://www.computerweekly.com/de/ratgeber/Die-Funktionen-des-relationalen-Datenbank-Management-Systems-RDBMS-IBM-DB2 [Stand: 11.11.2021]

Datenbanken-verstehen.de (2021): *PostgreSQL*
https://www.datenbanken-verstehen.de/datenbankarten/postgresql/ [Stand: 11.11.2021]

Dürr Austria GmbH (2021): *www.duerr-gruppe.at*
https://www.duerr-gruppe.at/de/firmenphilosophie.html [Stand: 15.11.2021]

Geißler, Otto; Ostler, Ulrike (2018): *Was ist ein Frontend und ein Backend?*
https://www.datacenter-insider.de/was-ist-ein-frontend-und-ein-backend-a-714429/ [Stand: 12.11.2011]

Google (2021): *Cloud Spanner*
https://cloud.google.com/spanner#all-features [Stand: 11.11.2021]

Ionos SE (2020): *MariaDB vs. MySQL*
https://www.ionos.at/digitalguide/hosting/hosting-technik/mariadb-vs-mysql/ [Stand: 11.11.2021]

Ionos SE (2019): *NoSQL - Funktion und Vorteile von NoSQL-Datenbanken*
https://www.ionos.at/digitalguide/hosting/hosting-technik/nosql/ [Stand: 15.09.2021]

Joos, Thomas; Ehneß, Jürgen (2019): *Der intelligenteste SQL Server aller Zeiten*
https://www.storage-insider.de/der-intelligenteste-sql-server-aller-zeiten-a-889536/ [Stand: 10.11.2021]

Luber, Stefan; Donner, Andreas (2020): *Was ist das Client-Server-Modell?*
https://www.ip-insider.de/was-ist-das-client-server-modell-a-940627/ [Stand: 12.11.2021]

Luber, Stefan; Ehneß, Jürgen (2021): *Was ist Microsoft SQL Server?*
https://www.storage-insider.de/was-ist-microsoft-sql-server-a-992556/ [Stand: 10.11.2021]

Moore, Lindsay (2020): *MySQL*
https://www.computerweekly.com/de/definition/MySQL [Stand: 10.11.2021]

Oracle Corporation (2019): *Oracle Database 19c*
https://www.oracle.com/a/tech/docs/database19c-wp.pdf [Stand: 11.11.2021]

Redaktion ComputerWeekly.de (2021): *Google Cloud Spanner*
https://whatis.techtarget.com/de/definition/Google-Cloud-Spanner [Stand: 11.11.2021]

Sheldon, Robert; Shore, Joel (2021): *Cloud-Datenbank*
https://www.computerweekly.com/de/definition/Cloud-Datenbank [Stand: 10.11.2021]

Software-Express GmbH & Co. KG (2021): *Microsoft SQL Server 2019*
https://www.software-express.de/hersteller/microsoft/sql-server/ [Stand: 10.11.2021]

solid IT GmbH (1) (2021): *Microsoft SQL Server System Properties*
https://db-engines.com/en/system/Microsoft+SQL+Server [Stand: 10.11.2021]

solid IT GmbH (2) (2021): *Microsoft Access Systemeigenschaften*
https://db-engines.com/de/system/Microsoft+Access [Stand: 10.11.2021]

solid IT GmbH (3) (2021): *MySQL Systemeigenschaften*
https://db-engines.com/de/system/MySQL [Stand: 11.11.2021]

solid IT GmbH (4) (2021): *PostgreSQL Systemeigenschaften*
https://db-engines.com/de/system/PostgreSQL [Stand: 11.11.2011]

solid IT GmbH (5) (2021): *MariaDB Systemeigenschaften*
https://db-engines.com/de/system/MariaDB [Stand: 11.11.2021]

solid IT GmbH (6) (2021): *Oracle Systemeigenschaften*
https://db-engines.com/de/system/Oracle [Stand: 11.11.2021]

solid IT GmbH (7) (2021): *IBM Db2 Systemeigenschaften*
https://db-engines.com/de/system/IBM+Db2 [Stand: 11.11.2021]

solid IT GmbH (8) (2021): *SQLite Systemeigenschaften*
https://db-engines.com/de/system/SQLite [Stand: 11.11.2021]

solid IT GmbH (9) (2021): *Amazon Aurora Systemeigenschaften*
https://db-engines.com/de/system/Amazon+Aurora [Stand: 11.11.2021]

SQLite Consortium (2021): *About SQLite*
https://www.sqlite.org/about.html [Stand: 11.11.2021]

Tech Target (2014): *Open Source*
https://whatis.techtarget.com/de/definition/Open-Source [Stand: 10.11.2021]

Technische Hochschule Mittelhessen (1) (2015): *Dokumentenorientierte Datenbanken*
http://wi-wiki.de/doku.php?id=bigdata:dokumentdb [Stand: 16.09.2021]

Technische Hochschule Mittelhessen (2) (2015): *Key-Value-Datenbanken*
http://wi-wiki.de/doku.php?id=bigdata:keyvaluedb [Stand: 16.09.2021]

Technische Hochschule Mittelhessen (3) (2015): *Konsistenz*
http://wi-wiki.de/doku.php?id=bigdata:konsistenz [Stand: 02.09.2021]

Normen (2)

ISO copyright office (Hrsg.) (2014): *ISO 55000: Asset management - Overview, principles and terminology*

Institute, Austrian (Hrsg.) (2015): *ÖNORM EN16646: Instandhaltung im Rahmen des Anlagenmanagements*

ABBILDUNGSVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

1NF	Erste Normalform
2NF	Zweite Normalform
3NF	Dritte Normalform
4NF	Vierte Normalform
5NF	Fünfte Normalform
ACID	Atomicity, Consistency, Isolation, Durability
ASFiNAG	Autobahnen- und Schnellstraßen-Finanzierungs-Aktiengesellschaft
BASE	Basically Available, Soft-State, Eventual Consistency
BCNF	Boyce-Codd-Normalform
CI	Configuation Item
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DBaaS	Database as a Service
DBMS	Datenbankmanagementsystem
DDL	Data Definition Language
DML	Data Manipulation Language
ER	Entity Relationship
ERP	Enterprise Ressource Planning
GB	Gigabyte
HTTP	Hypertext Transfer Protokoll
ID	Identifikator
IQP	Intelligent Query Processing
ISMS	Information Security Management System
JDBC	Java Database Connectivity
JSON	JavaScript Object Notation
KVP	Kontinuierlicher Verbesserungsprozess
LCM	Life Cycle Management
LStE	Lokale Steuereinheit
NAS	Network Attached Storage
NISG	Netz- und Informationssystemsicherheitsgesetz

ODL	Object Definition Language
OLTP	Online Transaction Processing
OOP	Objektorientierte Programmierung
OQL	Object Query Language
ORM	Objektrelationales Mapping
PL/SQL	Procedural Language / Structured Query Language
PVU	Processor Value Unit
QBE	Query by Example
RDBMS	Relationales Datenbankmanagementsystem
RMA	Return Material Authorization Process
SPS	Speicherprogrammierbare Steuerung
SQL	Structured Query Language
TDE	Transparent Data Encryption
TuKo	Tunnelkopfrechner
UI	User Interface
W&I	Wartung und Instandhaltung
XML	Extensible Markup Language