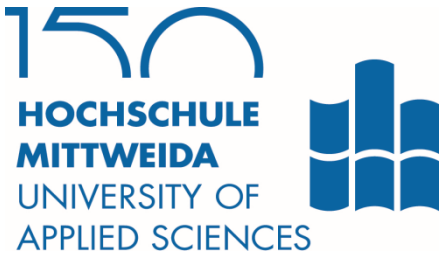


DIPLOMARBEIT

Herr

Markus Strohmaier

Fehlerdatenauswertung in Java

Mittweida, 2017

Fakultät Computer- und Biowissenschaften

DIPLOMARBEIT

Fehlerdatenauswertung in Java

Autor:
Herr Markus Strohmaier

Studiengang:
Technische Informatik

Seminargruppe:
KT12wWA-F

Erstprüfer:
Prof. Dr.-Ing. Rolf Hiersemann

Zweitprüfer:
M.Sc. Jan Roloff

Einreichung:
Mittweida, 23.09.2017

Verteidigung/Bewertung:
Mittweida, 2017

Bibliografische Angaben:

Strohmaier, Markus:
Fehlerdatenauswertung in Java, Mittweida, Hochschule Mittweida (FH), University of Applied Sciences, Fakultät Computer- und Biowissenschaften, Diplomarbeit, 2017

Referat:

Ziel der Diplomarbeit ist die Erstellung einer Software zum Auslesen und Aufbereiten von Fehlerdaten aus einer Datenbank einer Automatischen Optischen Inspektion (AOI). Des Weiteren sollen die Daten in Tabellenform und als Diagramm darstellbar und auch exportierbar sein .
Diese Arbeit befasst sich zunächst mit den Grundlagen von Java und Datenbanken sowie mit der Abfragesprache SQL. In den darauffolgenden Kapiteln werden die Anforderungen präzisiert und es wird näher auf die Installation der erforderlichen Komponenten sowie die Implementierung eingegangen. Im letzten Kapitel werden die Ergebnisse kurz zusammengefasst und ein Ausblick für Funktionserweiterungen wird dargestellt.

Inhaltsverzeichnis

Abkürzungsverzeichnis

AOI	Automatische Optische Inspektion
CAD	Computer-Aided Design
CSV	Comma-Seperated Values
DBMS	Database Management System
DLL	Dynamic Link Library
FPY	First Pass Yield
GUI	Graphical User Interface
Id	Identität
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
JAR	Java Archive
JDBC	Java Database Connectivity
JDK	Java Development Kit
JPEG	Joint Photographic Experts Group
JRE	Java Runtime Environment
MEZ	Mitteleuropäische Zeit
ODBC	Open Database Connectivity
PNG	Portable Network Graphics
SQL	Structured Query Language
SSMS	SQL Server Management Studio
VM	Virtuelle Maschine

Abbildungsverzeichnis

Tabellenverzeichnis

Einleitung

In beinahe allen Bereichen der Produktion von Gütern werden heutzutage automatische optische Inspektionen (AOI) zur Überprüfung der gefertigten Waren eingesetzt. Der häufigste Einsatzbereich von AOI-Systemen liegt in der Fertigung von bestückten Leiterplatten. Um schnell auf Fehler und Abweichungen in der Produktion aber auch auf Reklamationen reagieren zu können, ist der Einsatz von geeigneten Auswertungstools unerlässlich.

In den meisten Fällen sind solche Tools über den AOI-Hersteller erhältlich, wobei es hier wiederum verschiedene Ausführungsvarianten gibt.
Die einfachsten Auswertungsprogramme sind meist kostenlos und beinhalten nur eine Grundvariante zur Abfrage der Fehlerdaten. Um anschließend visuelle Auswertungen wie Diagramme und ähnliches erstellen zu können, müssen die Daten in einem anderen Programm, wie z. B. Microsoft Excel, weiterverarbeitet werden. Somit ist es relativ aufwändig, die Daten in der gewünschten Weise darzustellen.
Komplexere Auswertungstools werden ebenfalls von beinahe allen AOI-Herstellern angeboten. Diese Tools beinhalten sehr viele Funktionen, mit denen nahezu alle Details zu bestimmten Produkten und Produktionslinien abgefragt und auch visuell dargestellt werden können. Der einzige Nachteil solcher Lösungen ist meist der sehr hohe Anschaffungspreis.

Aus diesem Grund im Zuge dieser Diplomarbeit eine eigene Software zur Fehlerdatenauswertung erstellt werden. Der große Vorteil einer solchen Eigenlösung besteht darin, dass nur die relevanten Funktionen integriert werden und somit die Benutzerfreundlichkeit leichter erhalten bleibt.

Nach längerer Überlegung und Abwägung der Vor- und Nachteile wurde die Programmiersprache Java zur Erstellung der Auswertungssoftware gewählt. Mit dieser Hochsprache kann relativ einfach auf die AOI-Datenbank zugegriffen werden und es können mit Hilfe von Zusatzbibliotheken auch Diagramme ohne großen Aufwand erstellt und angezeigt werden. Außerdem ist eine Java-Software ohne grobe Anpassungen auf beinahe jeglicher Art von Betriebssystem lauffähig.

Kapitelübersicht

Das erste Kapitel befasst sich mit der Begriffserläuterung und den Grundlagen der AOI. Hier werden zunächst die einzelnen Betriebsarten und die AOI-Technologien dargestellt. Des Weiteren werden die interne Bildauswertung und die Beurteilung der Fehler kurz erläutert. Das letzte Unterkapitel beschreibt kurz die derzeitige Methode zum Abruf der Fehlerdaten und auch eine kostenpflichtige Alternativ-Software vom AOI-Hersteller.

Im nächsten Kapitel wird auf die Grundlagen von Datenbanken eingegangen, wobei hier als erstes der Einsatzzweck von Datenbanken erläutert wird. Anschließend wird näher auf die häufigste Form der Datenbanken, die sogenannten relationalen Datenbanken, eingegangen. Der nächste Teil dieses Kapitels beschreibt die Zugriffsarten und deren Vor- und Nachteile näher. Im letzten Abschnitt ist eine Übersicht der gesamten AOI-Datenbank mit den darin enthaltenen Tabellen und Beziehungen ersichtlich.

Das dritte Kapitel erläutert die Abfrage der Datenbanken mittels der Structured Query Language (SQL). Nach einer kurzen Begriffsdefinition erfolgt ein kleiner Einblick in die SQL-Syntax, wobei hier nur die, für die Diplomarbeit, relevanten Punkte enthalten sind. Der letzte Teil dieses Kapitels enthält eine Auflistung und Beschreibung der wichtigsten Microsoft SQL-Datentypen.

Der vierte Teil der Diplomarbeit enthält eine kurze Einführung in die Programmiersprache Java. Hier wird zunächst auf die Vor- und Nachteile von Java im Vergleich zu anderen Programmiersprachen eingegangen. Anschließend werden die Kernkomponenten von Java und die Entwicklungsumgebung Eclipse näher erläutert. Die nächsten beiden Unterkapitel befassen sich mit den grundlegenden Datentypen sowie mit den Grundelementen der Programmiersprache.

Im fünften Kapitel wird auf die Anforderungen an das System eingegangen, wobei hier zunächst die allgemeine Definition und die Unterteilung der Anforderungen erläutert wird. Anschließend erfolgt eine detaillierte Beschreibung der Anforderungen an die Fehlerdaten-Software.

Der vorletzte Teil der Arbeit befasst sich zunächst mit dem Grobentwurf bzw. der Unterteilung der einzelnen Anforderungen in Funktionsbausteine. Anschließend wird auf die extern eingebundenen Bibliotheken eingegangen, welche bereits vorgefertigte Funktionen zur Erleichterung der Programmierung bereitstellen. Im letzten Unterabschnitt, dem sogenannten Feinentwurf, werden die entwickelten SQL-Abfragen sowie die einzelnen Java-Klassen detailliert beschrieben.

Im letzten Kapitel erfolgt eine kurze Zusammenfassung der einzelnen Abschnitte. Außerdem wird auf die den Nutzen und die Anwendung der Arbeit eingegangen. Des Weiteren ist ein kleiner Ausblick für zukünftige Funktionserweiterungen enthalten.

1 Automatische Optische Inspektion

Definition:

> Als automatische optische Inspektion (AOI) werden Systeme beschrieben, welche über Bildverarbeitungsverfahren Fehler und Prozessveränderungen in der Produktion von Gütern finden, melden und speichern können. AOI-Systeme sind auch unter der Bezeichnung Vision-Systeme bekannt und deren Einsatzgebiet erstreckt sich über beinahe alle Bereiche der industriellen Produktion von Gütern. Diese sind beispielsweise die Lebensmittel- und Pharmaproduktion oder die Elektronik-, Automobil-, Luft- und Raumfahrtindustrie.

Das häufigste Einsatzgebiet ist die Inspektion bei der Produktion elektronischer Baugruppen (Leiterplattenbestückung). Durch die Komplexität der vorangehenden Prozesse (Lötpastendruck, Bestückung, Lötung) ist eine Überprüfung des Produktionsergebnisses zwingend notwendig, um Produktionsfehler schnell erkennen und mit geeigneten Maßnahmen entgegenwirken zu können
Des Weiteren können Prozessfehler und instabile Prozesse durch Analysen der aufgezeichneten Fehler korrigiert werden. Somit ist es möglich, die Prozesssicherheit zu erhöhen und die Qualität der gefertigten Produkte zu verbessern.

Die Informationen der bestückten Leiterplatte werden der Maschine über CAD- oder Gerber-Daten übergeben. Durch diese Standard-Datenformate für die Leiterplattenfertigung weiß die AOI an welcher Position, welches Bauelement bestückt werden muss und auch welche Drehung und Polarität dieses haben muss.

1.1 Betriebsarten

Die automatische optische Inspektion kann im Wesentlichen über zwei Arten erfolgen:

- *Inline:*
  Das AOI-System ist direkt in die Produktionslinie integriert, somit wird die fertig bestückte und gelötete Leiterplatte direkt über Transportbänder zur AOI weitertransportiert und sofort inspiziert.
- *Stand Alone:*
  Nur die AOI-Maschine und wahlweise ein Ein- und/oder Ausgabeband ist vorhanden. Jede Leiterplatte muss von Hand eingelegt und entnommen werden.

Die nachfolgende Abbildung zeigt eine schematische Darstellung eines typischen Produktionsablaufs für die Leiterplattenbestückung, angefangen vom Lötpastendruck mittels Siebdruckverfahren über die Bestückautomaten bis hin zum Inline-AOI-System bzw. zum anschließenden AOI-Reparaturplatz.

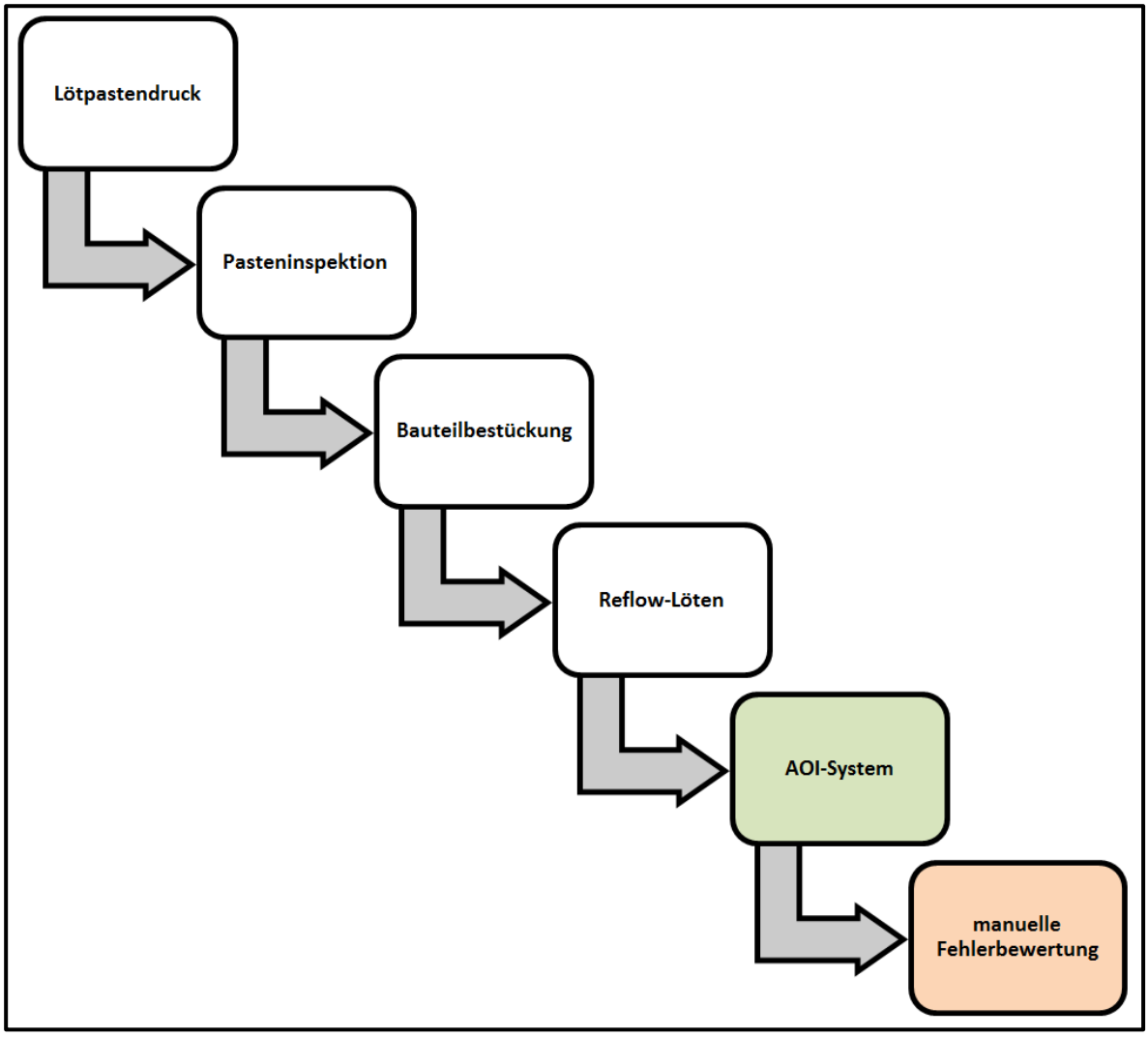


Abbildung 1-1: AOI im Inline-Modus

1.2 Bilderfassung

Die Aufnahme des Bildes bzw. der Bilder ist im Grunde genommen in zwei Technologien eingeteilt. So kann die Erfassung entweder mit *Scannern* oder mit *Kameras* erfolgen. Letztere können wiederum in zwei Kategorien unterteilt werden, nämlich in Kamerasysteme mit *Normalobjektiven* bzw. *telezentrischen Objektiven*.

1.2.1 Scanner-Systeme

Scanner-Systeme besitzen einen zentral angebrachten Scanner, der die komplette Leiterplatte auf einmal scannt. Durch diese Methode entsteht ein Geschwindigkeitsvorteil gegenüber AOI-Systemen mit Kameras. Der große Nachteil ist aber der entstehende Parallaxenfehler im Bereich der Randzonen der Leiterplatte. Deshalb können diese Systeme nur bei flachen Objekten eingesetzt werden.

Die nachstehende Skizze zeigt ein AOI-System mit einem zentralen Scanner zur Erfassung der kompletten Leiterplatte.

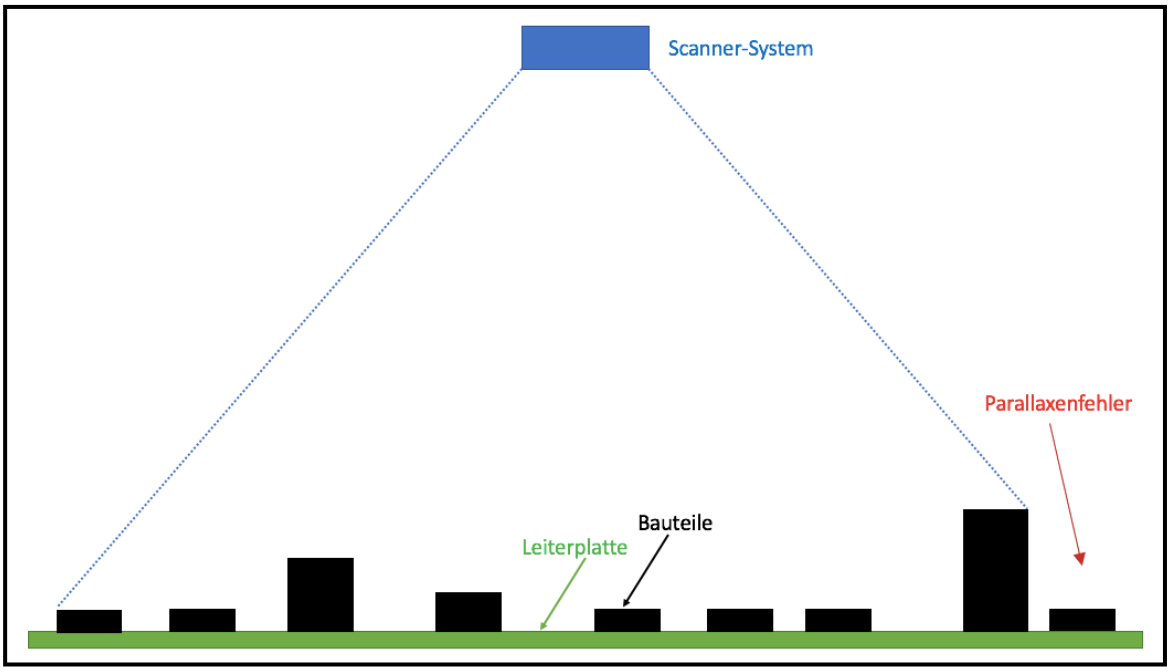


Abbildung 1-2: AOI-System auf Scanner-Basis[1]

[1] Vgl. Berg, Dietmar: Lohnt sich ein AOI-System? online im Internet

1.2.2 Kamera-Systeme

Die weitaus häufiger eingesetzte Methode zur Bilderfassung ist jene mit Kameras. Durch das eingeschränkte Sichtfeld kann meistens nicht die komplette Leiterplatte auf einmal erfasst werden. Deshalb sind die Kameras auf einem Schlitten montiert und können so mit Hilfe von Linearmotoren in der x- und y-Achse bewegt werden.

Der größte Vorteil von Kamerasystem gegenüber Systemen mit Scannern liegt in der deutlich höheren Tiefenschärfe und der Flexibilität der Bilderfassung. So kann bei jedem aufgezeichneten Bild die Helligkeit und Auflösung und teilweise sogar die Farbe variiert werden, wodurch die Erfassung und Auswertung unterschiedlicher Bauteile deutlich vereinfacht wird.

Wie bereits erwähnt, können Kameras mit Normalobjektiven oder telezentrischen Objektiven zum Einsatz kommen. Der Nachteil bei Normalobjektiven ist hier wieder der Parallaxenfehler, welcher aber bei weitem nicht so ausgeprägt wie bei Scanner-Systemen auftritt und deshalb in den meisten Anwendungen vernachlässigt werden kann. Durch den Einsatz von telezentrischen Objektiven ist es aber auch möglich, diesen geringen Parallaxenfehler zu eliminieren.

In der nachfolgenden Abbildung sind die beiden Varianten von AOI-Systemen mit Kameraerfassung dargestellt.

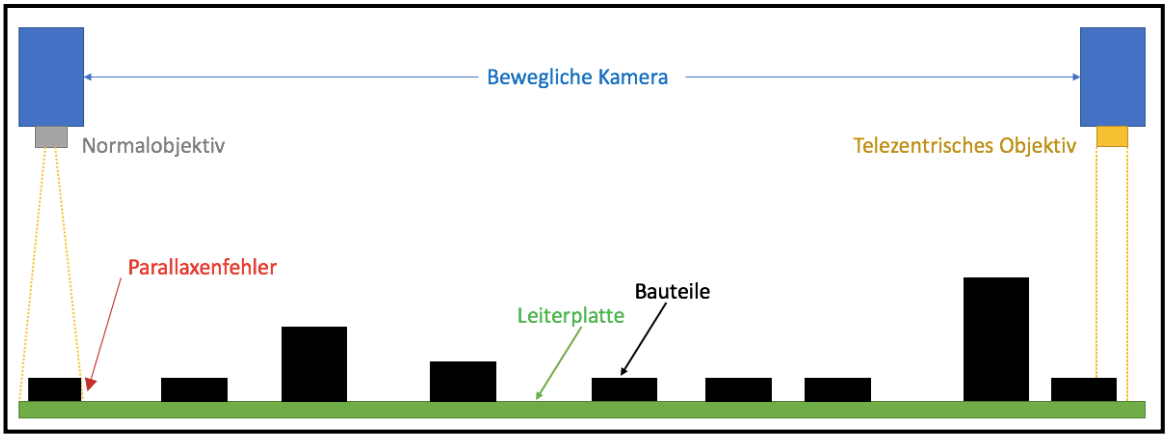


Abbildung 1-3: AOI-System auf Kamera-Basis[2]

[2] Vgl. Berg, Dietmar: Lohnt sich ein AOI-System? online im Internet

1.3 Datenverarbeitung

Die Verarbeitung der aufgenommenen Bilddaten ist abhängig von der eingesetzten AOI-Technologie. Bei Scanner-Systemen wird das aufgenommene Gesamtbild mit einer Vorlage verglichen und die Unterschiede werden, je nach Toleranzvorgaben, als 'Fehler' oder 'kein Fehler' deklariert.
Die Auswertung der erfassten Bilder von Kamerasystemen erfolgt ebenfalls über einen Vergleichstest. Jedoch werden hier die einzelnen Bauteile mit den Vorlagen aus der Bauteilebibliothek verglichen und beurteilt. Über diesen Vergleichstest ist es möglich die geforderten Merkmale (Position, Drehung, Polarität, Aufdruck, etc.) auszuwerten. Eine besondere Herausforderung ist die Inspektion der Verlötung eines Bauteils, weil die Pins (Bauteilanschlüsse) und das ausgehärtete Lötzinn beinahe die gleiche Farbe (silberglänzend) besitzen. Zur Überprüfung der Lötung wird hier mit dem Kontrast an der Lötstelle gearbeitet. Durch den Hell-Dunkel-Übergang vom Lötzinn zum Anschluss des Bauteils kann festgestellt werden, ob es sich um eine gute oder schlechte Lötstelle handelt.

Die folgende Abbildung zeigt eine Aufnahme mit einem Kamerasystem mit telezentrischem Objektiv. Zu sehen ist ein Lötfehler (nicht verlöteter Anschluss) mit unterschiedlichen Licht- und Farbeinstellungen. Alle sauber verlöteten Pins besitzen den oben beschriebenen Hell-Dunkel-Übergang, welcher jedoch beim nicht verlöteten Anschluss so gut wie nicht vorhanden ist.

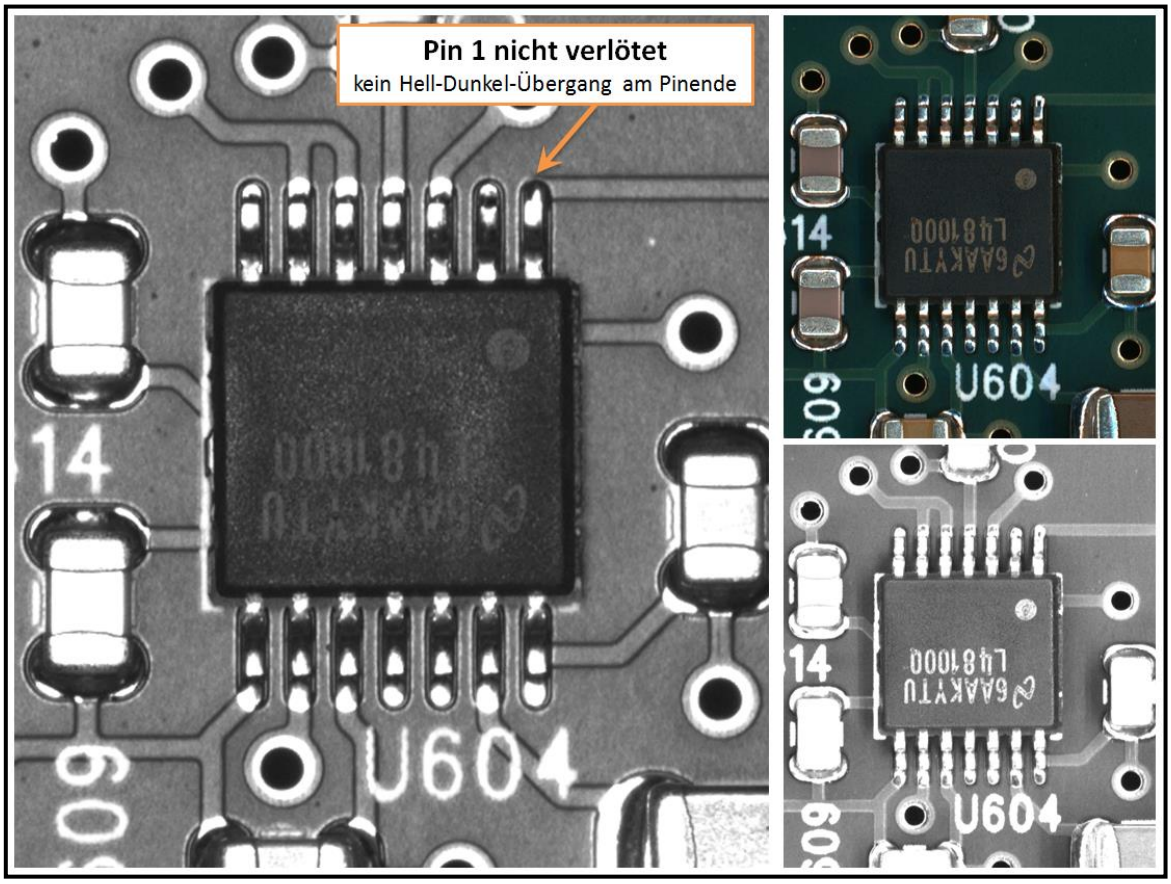


Abbildung 1-4: Fehlerbild einer AOI mit Kamerasystem

1.4 Fehlerbeurteilung

Das AOI-System speichert alle Differenzen zur Vorlage, welche nicht in die Toleranzvorgaben fallen, in der Datenbank als Fehler ab. Alle diese Fehler müssen nach der AOI vom Fertigungspersonal noch einmal optisch inspiziert und beurteilt werden, wobei das Ergebnis dieser Beurteilung ebenfalls in der Datenbank eingetragen wird.

Da die AOI-Toleranzen tendenziell sehr klein eingestellt sind, um keine Fehler zu übersehen, und die Fehlererkennung von vielen verschiedenen Faktoren wie Lichteinfall, Oberfläche und Farbe des zu prüfenden Objekts, etc. abhängt, entstehen sogenannte Pseudofehler.
Diese temporären Fehler sind keine echten Fehler und müssen somit nicht nachgearbeitet werden. Das Prüfpersonal muss diese Fehler mit einem eigenen Fehlercode deklarieren, damit bei anschließenden Auswertungen sofort erkennbar ist, ob es sich um einen echten Fehler oder einen Pseudofehler handelt.
Durch die Analyse der Pseudofehlerhäufigkeit ist es möglich, Bauteile mit beispielsweise glatten oder glänzenden Oberflächen, also mit erhöhter Wahrscheinlichkeit für temporäre Fehler, schnell herauszufiltern.

1.5 Datenauswertung

Die aktuell verwendete Lösung zur Analyse der Fehlerdaten ist ein kostenlos bereitgestelltes Tool vom AOI-Hersteller. Mit diesem wird ein Webserver generiert, auf welchen mir einem gewöhnlichen Internet Browser zugegriffen werden kann. Die Daten können als Tabelle oder in Diagrammen angezeigt und auch gefiltert oder exportiert werden. Die größten Nachteile dieser Lösung bestehen aber darin, dass zum einen immer nur ein Benutzer auf den Service zugreifen kann und zum anderen in den eingeschränkten Anpassungsmöglichkeiten bei der Filterung und Diagrammdarstellung.

Die zweite externe Variante wäre ein Erwerb einer Zusatzsoftware, welche ebenfalls vom AOI-Hersteller bereitgestellt wird. Dieses Tools bietet extrem viele Funktionen und Informationen wie z.B. Live-Abfragen der einzelnen Maschinenstatus, Wartungszustände, etc. Der Hersteller stellte die Software für drei Monate zum Testen zur Verfügung. Nach mehrmaligem, intensivem Arbeiten mit der Anwendung ist die Erkenntnis, dass zwar viele hilfreiche Funktionen vorhanden sind, jedoch nur ein Bruchteil davon genutzt wird und viele Optionen sehr versteckt in der Benutzeroberfläche integriert sind.

Aus diesem Grund und auch wegen der sehr hohen Anschaffungskosten der Softwarelizenzen wurde hier entschieden, dass eine Eigenlösung mit den relevanten Kernfunktionen besser geeignet ist und diese zudem übersichtlicher aufgebaut werden kann.

2 Datenbanken

Definition:

> „Eine Datenbank ist eine Sammlung von Daten, die untereinander in einer logischen Beziehung stehen und von einem Datenbankverwaltungssystem (Database Management System, DBMS) verwaltet werden.“[3]

Ohne diese Datenorganisation müsste jedes Anwendungsprogramm, das Informationen abfragt, die gesamte Datenstruktur und deren Aufbau exakt kennen. Da diese Daten aber oft unterschiedlich abgespeichert sind und sich häufig auch auf verschiedenen Hardware-Plattformen befinden, muss jedes Programm an die jeweilige Hardware angepasst sein.

Die nachfolgende Abbildung soll einen solchen Betrieb schematisch darstellen. Hier ist sofort erkennbar, dass die benötigten Anwendungsprogramme sehr schnell eine hohe Komplexität erreichen.

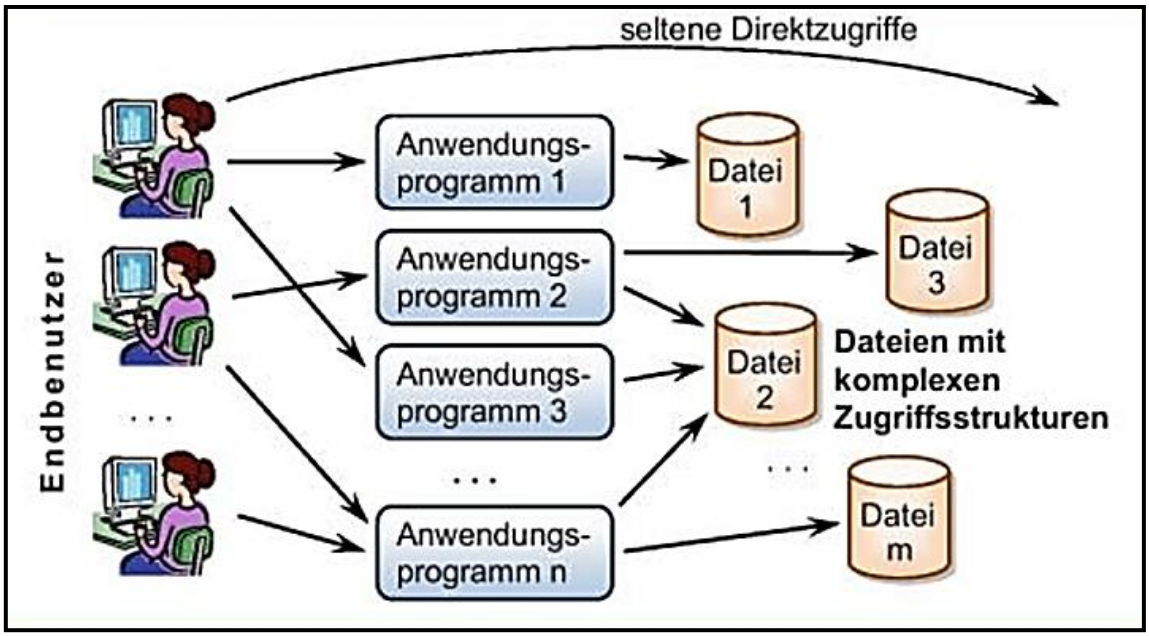


Abbildung 2-1: Mehrfachzugriff ohne DBMS[4]

[3] Schicker, Edwin: Datenbanken und SQL, S. 3.

[4] Schicker, Edwin: Datenbanken und SQL, S. 2.

Aus diesem Grund wird, wie in der nächsten Abbildung ersichtlich, ein DBMS eingesetzt. Dieses ermöglicht eine strukturierte Datenverwaltung über eine logische Schnittstelle, das heißt, der Benutzer bzw. das Anwendungsprogramm greift lediglich auf die Schnittstelle zu. Diese logischen Zugriffe werden vom DBMS in physische umgesetzt und das Verwaltungssystem liest bzw. schreibt die Daten.

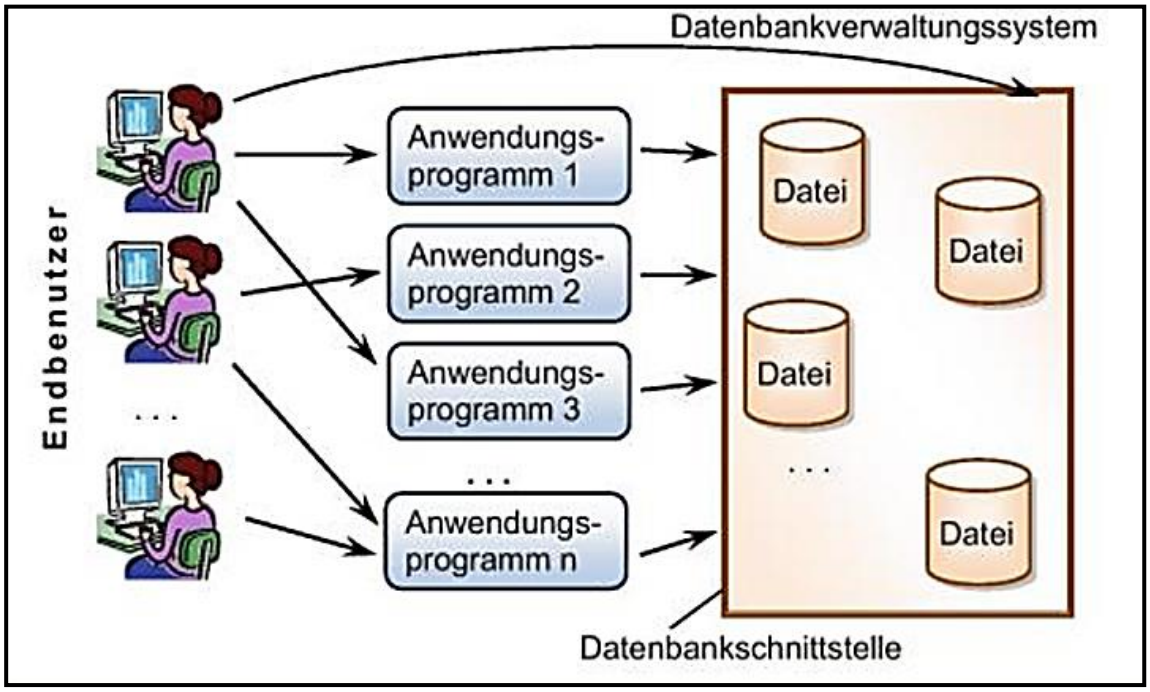


Abbildung 2-2: Mehrfachzugriff mit DBMS[5]

[5] Schicker, Edwin: Datenbanken und SQL, S. 3.

2.1 Relationale Datenbanken

Da es sich bei der AOI-Datenbank um eine sogenannte *relationale Datenbank* handelt, also ein System, dass seine Daten in Tabellen speichert und verwaltet, wird in dieser Arbeit nur auf diese Form der Datenbanken eingegangen.

> „Es gibt zahlreiche Datenbanksysteme von den unterschiedlichsten Anbietern, die SQL unterstützen und Daten bzw. Informationen in Tabellen speichern. Datenbanksysteme, die Informationen in Tabellen speichern, werden als *relationale Datenbanksysteme* bezeichnet.“[6]

Diese Tabellen, auch Relationen genannt, enthalten Spalten und Zeilen. Eine Zeile in der Tabelle wird auch als Tupel oder Satz (engl. record) bezeichnet. Ein Feldelement, also ein einzelner Wert eines Tupels, wird Attribut genannt.

> „Die Zusammenhänge zwischen den einzelnen Tabellen werden über Beziehungen hergestellt. Diese Beziehungen sind in den Tabellen mit abgespeichert.“[7]

Zur Vorbereitung dieser Abspeicherung müssen alle Tupel in der Tabelle indexiert werden. Dies geschieht mit dem sogenannten Primärschlüssel, somit besitzt jede Zeile eine eindeutige Identität (kurz Id).
In den nachfolgenden, einfachen Beispieltabellen ist ersichtlich, dass lediglich die Id vom Studenten übertragen wird und es somit nicht notwendig ist, die Vor- und Nachnamen noch einmal in der Telefonnummern-Tabelle zu speichern. Durch diese Vorgehensweise wird eine logische Beziehung zwischen den beiden Tabellen hergestellt. Außerdem werden sogenannte Redundanzen (doppelte oder mehrfache Informationseinträge) vermieden, wodurch wiederum Speicherplatz gespart wird.

'studenten'		
id	**vorname**	**name**
1	Markus	Strohmaier
2	Max	Mustermann
3	…	…

Tabelle 2-1: Beispieltabelle 'studenten'

'telefon'		
id	**stud_id**	**telefon_nr**
1	3	0123456789
2	1	9876543210
3	2	…

Tabelle 2-2: Beispieltabelle 'telefon'

[6] Laube, Michael: Einstieg in SQL, S. 20.

[7] Schicker, Edwin: Datenbanken und SQL, S. 12.

2.2 Zugriff

Zur Verwaltung und Konfiguration eines Microsoft SQL Servers ist das Tool *SQL Server Management Studio (SSMS)* von Microsoft von Vorteil, welches auch aus dem Internet unter *https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms* heruntergeladen werden kann. Mit diesem Programm kann über die ausgewählte Authentifizierungsmethode auf alle Datenbanken, die am Server verfügbar sind, zugegriffen werden. Somit können die Datenbanken administriert werden und es können auch alle dazugehörigen Details, wie etwa Tabellen oder Beziehungen angezeigt und bearbeitet werden. Außerdem ist es auch möglich Daten aus den Tabellen über einen integriertes SQL-Tool abzufragen.

Um den Inhalt einer Datenbank vor fremden Zugriffen zu schützen, muss das Datenbankmanagementsystem Sicherheitsmechanismen zur Zugriffskontrolle zu Verfügung stellen.

Die Art dieses *Authentifizierungsmodus* kann entweder bei der Installation der Datenbank oder auch im Nachhinein durch den Datenbank-Administrator eingestellt werden. Der Benutzer 'sa' hat Administratorrechte und wird automatisch bei der Erstellung des SQL Servers angelegt.

Bei Microsoft SQL Server stehen die folgenden drei Optionen zur Zugriffskontrolle zur Verfügung[8]:

- Die *Windows-Authentifizierung* ist die sicherste Methode zur Anmeldung und deshalb auch der Standard-Authentifizierungsmodus in Microsoft SQL Server. In diesem Modus werden die Windows-Anmeldedaten eines Benutzers automatisch über das Kerberos-Sicherheitsprotokoll abgefragt und der Zugriff erfolgt mit diesen Daten. Außerdem wird bei dieser Auswahl der Standardbenutzer 'sa' deaktiviert.
- Bei der *SQL Server-Authentifizierung* werden die Benutzeranmeldungen beim Anlegen des SQL-Servers erstellt und gespeichert. Hier müssen die Anmeldeinformationen (Benutzer und Kennwort) bei jedem Verbindungsaufbau angegeben werden.
- Im *gemischten Modus* sind beide oben genannten Varianter zur Anmeldung verfügbar. Somit können sich berechtigte Windows-Benutzer einloggen oder es kann eine Anmeldung mit Benutzername und Passwort erfolgen.

Für die Fehlerdatenauswertungs-Software wurde die Windows-Authentifizierung genutzt, weil diese den großen Vorteil der Domänenverwaltung zur Benutzersteuerung besitzt. Außerdem muss so kein Passwort in die Software integriert oder eingegeben werden.

[8] Vgl. Microsoft Developer Network, Stichwort: Auswählen eines Authentifizierungsmodus, online im Internet.

2.3 AOI-Datenbank

Die Datenbank, in der die AOI-Maschinen ihre aufgenommenen Daten speichern, ist eine relationale Microsoft SQL Datenbank. Das bedeutet, die Informationen sind in Tabellen abgelegt, die miteinander verknüpft sind und auf die Datenbank kann mit der Abfragesprache Microsoft SQL zugegriffen werden.

Es sind insgesamt 19 Tabellen in der AOI-Datenbank vorhanden, welche in der nachstehenden Übersichtstabelle aufgelistet sind. Alle Tabellen, die für die Fehlerdatenauswertungssoftware relevant sind, wurden hervorgehoben und sind kurz beschrieben

Tabellenname	Kurzbeschreibung des Inhalts
CARDS	Enthält die Seriennummern der Einzelplatinen und deren Status (vor Reparatur bzw. nach Reparatur)
FEEDER	-
JEDEC	Beinhaltet alle Gehäuseformen der Bauteilbibliothek
JOINT_BRIDGE	-
LIBRARY	-
LOG_PROD	-
MACHINE	Enthält alle verfügbaren AOI-Systeme (Vision) und auch alle Reparaturstationen (Repair) für das Prüfpersonal
OBJECT_TYPE	-
OPERATOR	Beinhaltet die registrierten Benutzer (Prüfpersonal) bzw. deren Personalnummern
PANELS	Enthält die Seriennummern der Gesamtleiterplatten, deren Status (vor Reparatur bzw. nach Reparatur) und auch den Inspektionszeitpunkt
PART_NUMBER	Enthält alle Bauteilnummern
PIXEL_SIZE	-
PRODUCT	-
RECIPE	Beinhaltet alle Produktnamen
SPC	-
SPC_OBJECT	-
TESTED_OBJECT	Enthält alle fehlerhaften Positionen aller Einzelplatinen inkl. Reparaturstatus und Reparaturzeit
TOLERANCE	-
VERSION	-

Tabelle 2-3: Auflistung der AOI-Datenbank-Tabellen

Beziehungen:

Alle Zeilen der oben genannten Tabellen sind indexiert. Diese eindeutige Id ist in jeder Tabelle in einer separaten Spalte (z.B. Panel_Id) gespeichert und wird auch Primärschlüssel genannt. Der gleiche Schlüssel wird dann auch in der zu verknüpfenden Tabelle integriert, hier wird dieser aber als Fremdschlüssel bezeichnet. Über diese beiden Schlüssel werden schlussendlich die Tabellenbeziehungen hergestellt.

Nachfolgend sind die Verknüpfungen der relevanten Tabellen dargestellt. Diese visuelle Ansicht der Beziehungen kann mit dem Tool Microsoft SQL Server Management Studio generiert werden. Zur besseren Übersicht wurde die Darstellung in zwei Abbildungen aufgeteilt.

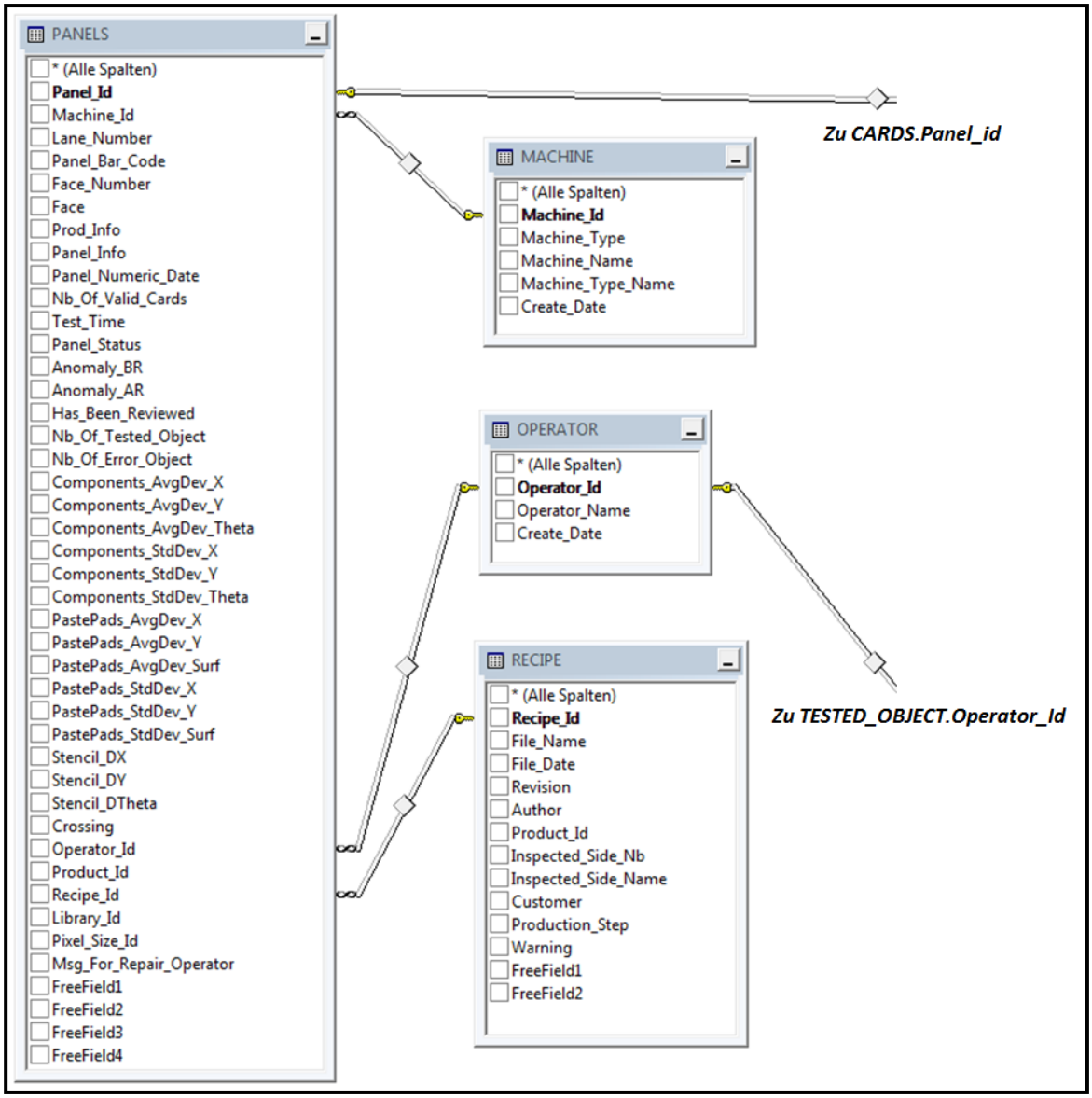


Abbildung 2-3: Tabellenbeziehungen (Teil 1/2)

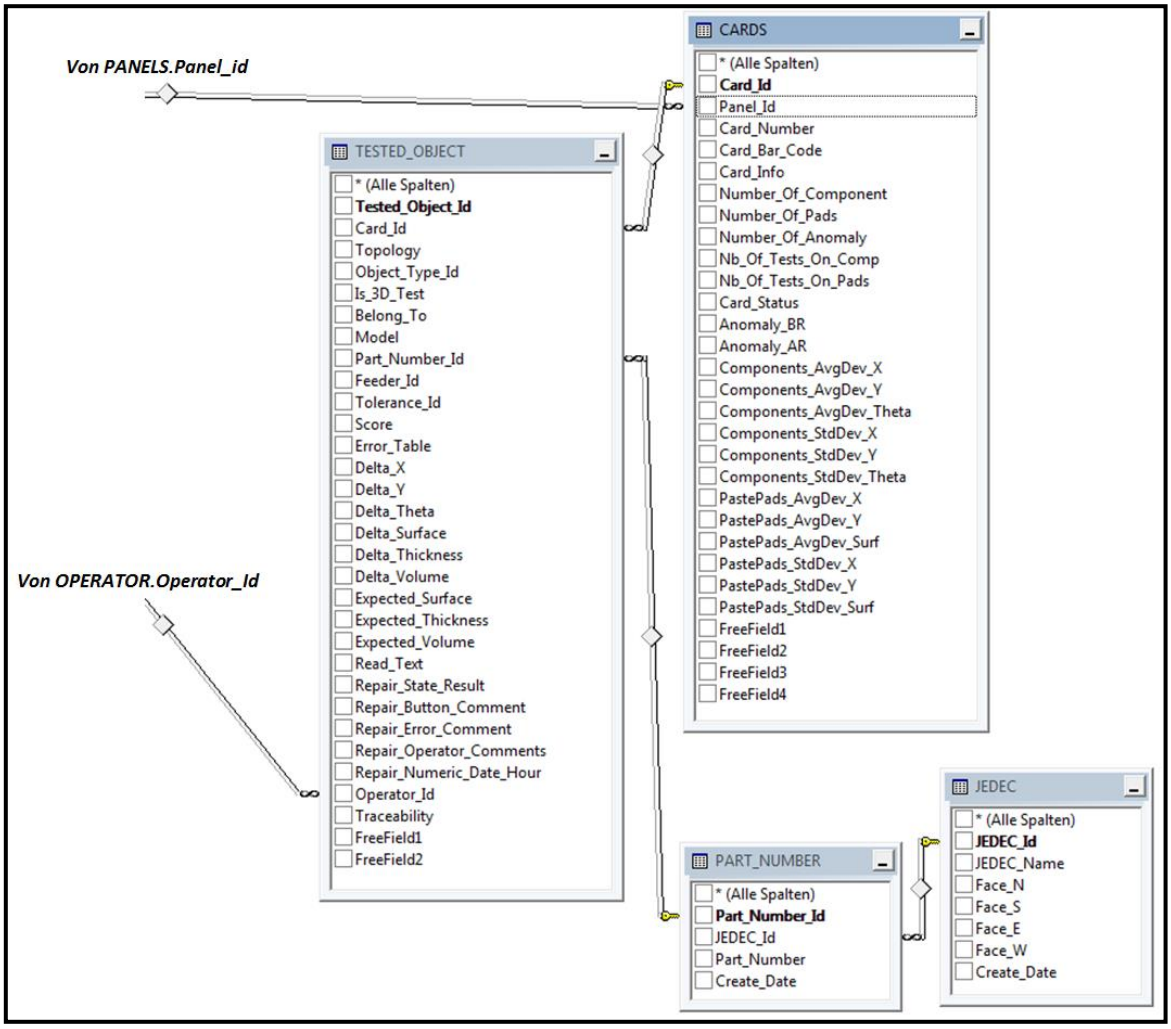


Abbildung 2-4: Tabellenbeziehungen (Teil 2/2)

Mit der Kenntnis dieser Tabellen und deren Beziehungen zueinander können nun eigene Ergebnis-Tabellen für die Weiterverarbeitung in der Auswertungssoftware mit Hilfe der Abfragesprache SQL generiert werden.
Dazu werden im nächsten Kapitel einige Grundlagen von SQL sowie die verwendeten Funktionen näher beschrieben.

3 SQL

Definition:

Die *Structured Query Language* (engl. für strukturierte Abfragesprache) ist eine

> „relationale Datenbanksprache, die sich grob in zwei Teile gliedern lässt: Mit der Datenbeschreibungssprache werden Tabellen, in denen die Daten in der Datenbank abgelegt werden, angelegt und gewartet; mit der Datenmanipulationssprache werden die Datensätze angelegt, gelöscht, verändert oder über Datenbankabfragen ausgewertet. Zur Auswertung wird entgegen der allg. üblichen Programmierung nicht der Weg zum Erhalt des Ergebnisses beschrieben, sondern die Struktur der Daten definiert (deklarative Programmierung), indem zunächst die im Ergebnis gewünschten Attribute der Datensätze, dann die anzusprechenden Tabellen und zum Schluss die gewünschten Einschränkungen, denen alle Ergebnisdatensätze unterliegen müssen, deklariert werden. Weiterhin stellt SQL Mechanismen für den Entwurf, die Einrichtung und die Zugriffssicherung von kompletten Datenbanksystemen zur Verfügung.“[9]

Über den Standard ISO/IEC 9075-1 ist die Spezifikation der Abfragesprache SQL festgelegt. Dennoch gibt es, je nach verwendeter Datenbank, Abweichungen vom Standard, welche als SQL-Dialekte bezeichnet werden.[10]

Den Kern von SQL bilden Datenbanktabellen, auf denen folgende Hauptfunktionen anwendbar sind:

- Daten abfragen
- Daten einfügen
- Daten löschen
- Daten aktualisieren

Mit diesen Funktionen können in der SQL-Abfrage Berechnungen, Einschränkungen, Tabellenverknüpfungen etc. implementiert werden.

[9] Gabler Wirtschaftslexikon, Stichwort: SQL, online im Internet

[10] Vgl. Laube, Michael: Einstieg in SQL, S. 21f.

3.1 Syntax

In dieser Kurzbeschreibung der SQL-Syntax werden nur jene Grundfunktionen und Standard-Schlüsselwörter näher beschrieben, welche für die Erstellung der Fehlerdatenauswertungs-Software relevant waren. Außerdem wird hier speziell auf den Dialekt Microsoft SQL Server eingegangen, weil es sich beim AOI-Server um einen Microsoft-Datenbankserver handelt.
Natürlich existiert eine viel höhere Anzahl an Schlüsselwörtern und Funktionen in der Datenbankabfragesprache SQL, alle diese aufzuführen und detailliert zu erläutern, würde aber den Rahmen der Arbeit sprengen.

Wie bereits in der Erstdefinition beschrieben nutzt SQL die deklarative Programmierung und es werden zuerst die zur Anzeige gewünschten Spalten, anschließend die Ursprungstabelle und zu guter Letzt die Einschränkungen angegeben. Die Kurzbeispiele auf den nächsten Seiten basieren auf den Tabellen von Seite 11 und sollen zur besseren Verständlichkeit beitragen.

```
SELECT name, vorname            /* Auswahl der anzuzeigenden Spalten */
FROM studenten                  /* Ursprungstabelle */
WHERE name LIKE 'Stroh%'        /* Einschränkung: nur Namen, die mit
                                „Stroh“ beginnen */
```

Kommentare werden in den meisten Editoren grün eingefärbt. Grundsätzlich wird zwischen einzeiligen und mehrzeiligen Kommentaren unterschieden.
Einzeilige Kommentare werden mit einem doppelten Bindestrich eingeleitet und gelten bis zum Ende einer Zeile.
Mehrzeilige Kommentare beginnen mit einem **/*** und sind solange aktiv, bis der Abschluss mit ***/** erfolgt.

Schlüsselwörter wie **SELECT**, **FROM**, **WHERE**, etc. sind vordefinierte Wörter, die zur Erstellung, Bearbeitung und Abfrage von Datenbanken dienen. Diese Wörter werden in den SQL-Tools blau eingefärbt. Zusätzlich wurden diese im obigen Beispiel in Großbuchstaben geschrieben. Dies dient aber nur zur besseren optischen Darstellung, diese Wörter können auch klein geschrieben werden.

> „Die Abfragesprache SQL ist *case insensitive*. Das heißt, es spielt überhaupt keine Rolle, ob Sie eine SQL-Anweisung in Großbuchstaben, in Kleinbuchstaben oder auch gemischt in Klein- und Großbuchstaben notieren.“[11]

[11] Laube, Michael: Einstieg in SQL, S. 55.

Um alle Spalten einer Tabelle im Ergebnis anzeigen zu lassen, müssen diese nicht alle explizit angegeben werden. Hier reicht der *Asterisk* (*) als Stellvertreterzeichen.[12]

Außerdem können doppelte Datensätze mit dem Zusatzbefehl **DISTINCT** zusammengefasst werden.

```
SELECT DISTINCT * FROM studenten       /* Alle Spalten aus der Tabelle
                                       'studenten' anzeigen und doppelte
                                       Werte herausfiltern */
```

Über den Befehl **GROUP BY** ist es möglich, identische Ergebnisdaten einer oder mehrerer Spalten in Gruppen anzuordnen.

Eine *Filterung von Zeichenketten* wird immer in einzelnen *Apostrophen* angeführt. Solche Zeichenketten sind, wie im vorhin angeführten Beispiel, rot eingefärbt. Ein zusätzliches, nachfolgendes *Prozentzeichen* gibt an, dass ab dieser Stelle alle folgenden Zeichen irrelevant sind. Steht dieses Zeichen zum Beispiel vorne, werden voranstehenden Zeichen ignoriert.

Bei einem *Vergleich von numerischen Werten* auf bestimmte Vorgabewerte, sind die Apostrophen wegzulassen und es ist einfach ein mathematischer Standardoperand (=, <, >, etc.) statt dem Schlüsselwort **LIKE** einzufügen (z.B. `WHERE baujahr > 1980`).

Alle diese Vergleiche und Einschränkungen können beliebig oft mit anderen Filterungen verknüpft werden und zwar mit den logischen Operatoren **AND** (**NOT**) und **OR** (**NOT**). So ist es möglich, anstatt mehrerer verknüpfter, komplizierter Anfragen, lediglich eine Anfrage mit mehreren Bedingungen zu erstellen.

[12] Vgl. Laube, Michael: Einstieg in SQL, S. 56.

Eine Sortierung des Ergebnisses kann am Schluss der Abfrage mit dem Befehl **ORDER BY** hinzugefügt werden.

```
SELECT *                          /* Alle Spalten auswählen */
FROM studenten                    /* Ursprungstabelle */
ORDER BY Vorname ASC              /* Sortiere aufsteigend nach Vorname */
```

Je nach enthaltenen Werten (Textwerte, Datumswerte oder numerische Werte) werden die Daten mit unterschiedlichen Sortierregeln angeordnet[13]:

- Alphabetische Sortierung
- Kalendarische Sortierung
- Numerische Sortierung

Die Standard-Sortierung erfolgt in aufsteigender Reihenfolge, diese kann aber auch mit dem Schlüsselwort **ASC** (ascending) explizit deklariert werden. Ist es notwendig die Sortierkriterien in absteigender Reihenfolge anzuordnen, so muss das Schlüsselwort **DESC** (descending) eingefügt werden.

Um zwei oder mehr Abfragen bzw. deren Ausgaben miteinander zu verknüpfen, kann das Schlüsselwort **UNION** eingesetzt werden. Die Spalten der Einzelabfragen müssen aber untereinander kompatibel sein, das heißt die Anzahl, Reihenfolge und enthaltenen Datentypen müssen für eine erfolgreiche Verknüpfung zu einer Ausgabetabelle identisch sein. Bei Abweichungen wird für jede Einzelabfrage ein separates Ergebnis erzeugt und ausgegeben.

[13] Laube, Michael: Einstieg in SQL, S. 123.

3.2 Der JOIN-Befehl

Um Tabellen für die Ausgabe temporär zu verknüpfen, dient das Schlüsselwort **JOIN**. Es können immer nur zwei Tabellen mit einem JOIN verknüpft werden, somit muss für mehrere Tabellenverknüpfungen eine Verschachtelung der einzelnen JOINs erfolgen.

Außerdem muss bei jedem JOIN-Befehl (außer CROSS JOIN) eine ON-Klausel hinzugefügt werden, um anzugeben, auf welche beiden Spalten die Verknüpfung angewendet werden soll.

Der JOIN-Befehl wird in fünf verschiedene Varianten unterteilt, wobei die letzten beiden nur sehr selten zur Anwendung kommen. Diese sind auch in der Fehlerdatenauswertung nicht eingesetzt worden:

- ***INNER JOIN:***
  Es werden jene Zeilen verbunden, die Wertübereinstimmungen im Primär- und Fremdschlüssel enthalten.

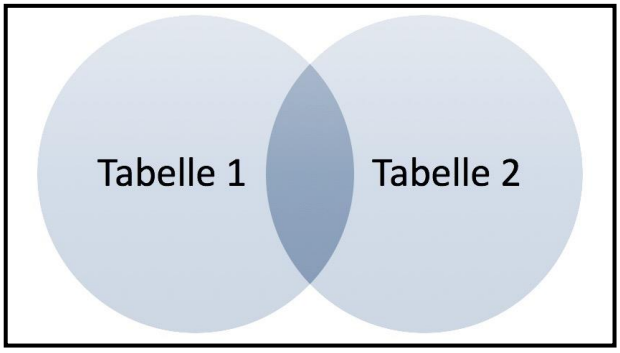


Abbildung 3-1: INNER JOIN[14]

- ***LEFT JOIN:***
  Es werden alle Zeilen der linken Tabelle und übereinstimmende Fremdschlüsselwerte der rechten Tabelle ausgegeben.

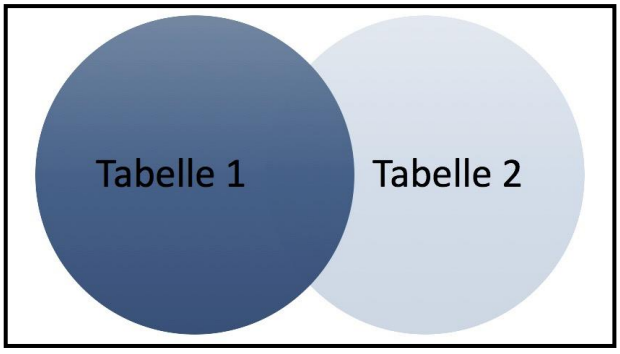


Abbildung 3-2: LEFT JOIN[14]

[14] Vgl. Laube, Michael: Einstieg in SQL, S. 353.

- ***RIGHT JOIN:***
  Es werden alle Zeilen der rechten Tabelle und übereinstimmende Fremdschlüsselwerte der linken Tabelle ausgegeben.

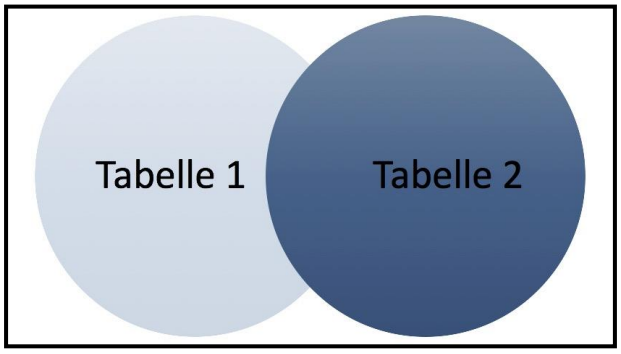


Abbildung 3-3: RIGHT JOIN[15]

- ***FULL OUTER JOIN:***
  Es werden alle Zeilen beider Tabellen verbunden, auch jene die keine übereinstimmenden Schlüssel besitzen. Dieser Join ist eine Kombination aus Inner, Left und Right Join.

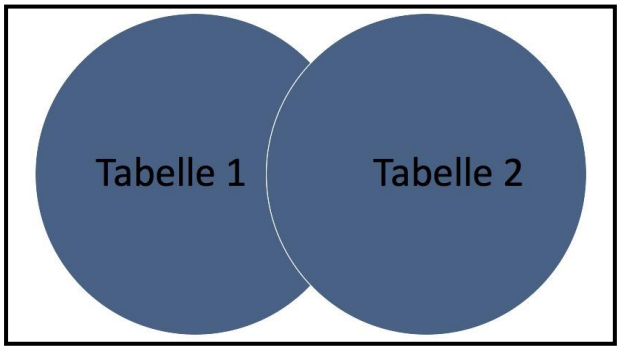


Abbildung 3-4: FULL OUTER JOIN[15]

- ***CROSS JOIN:***
  Alle Werte der linken Tabelle werden mit allen Werten der rechten Tabelle verknüpft. Hier ist ausnahmsweise keine ON-Klausel notwendig.

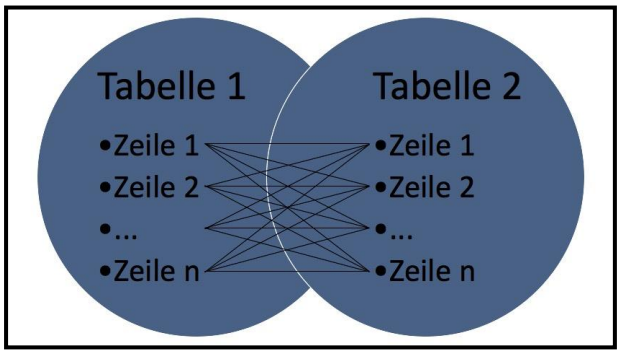


Abbildung 3-5: CROSS JOIN[15]

[15] Vgl. Laube, Michael: Einstieg in SQL, S. 354.

Das unten angeführte Beispiel eines INNER JOINs soll zur besseren Verständlichkeit dienen. Hier soll ebenfalls mit Hilfe der bereits auf Seite 11 (Kapitel Datenbanken) genannten Beispieltabellen 'studenten' und 'telefon' eine Ergebnistabelle erstellt werden, die den Namen, Vornamen und die Telefonnummer der Studenten enthält.

```
/* auszugebende Tabellen mit Select wählen */
SELECT vorname, name, telefon_nr

/* Ursprungstabelle angeben */
FROM studenten

/* Zieltabelle für die Join-Anweisung wählen */
INNER JOIN telefon

/* Primär- und Fremdschlüssel zur Gleichheitsprüfung angeben */
ON studenten.id = telefon.stud_id
```

vorname	name	telefon_nr
Markus	Strohmaier	9876543210
Max	Mustermann	0123456789
…	…	…

Tabelle 3-1: Ergebnistabelle nach INNER JOIN

3.3 Datentypen

„Ein Datentyp besteht aus Elementen, die einem Wertebereich zugeordnet sind, und Operatoren, die auf diese Elemente angewendet werden können. Mit Operatoren sind mathematische Operatoren wie +,-,*,/ gemeint, die auf numerische Wertebereiche angewendet werden können. Einer Spalte, die in einer Tabelle angelegt werden soll, muss immer ein Datentyp zugeordnet werden. Dieser Datentyp enthält Informationen darüber, welche Elemente die Spalte speichern kann.“[16]

Die Standarddatentypen sind in allen SQL Datenbanken vorhanden, diese werden noch durch spezifische Datentypen der einzelnen Datenbankarten ergänzt. In der nachfolgenden Tabelle sind die Datentypen von Microsoft SQL aufgeführt.

Kategorie	Name	Kommentar	Standard
Genaue numerische Werte	*bigint*	ganze Zahlen	x
	bit	ganze Zahlen (0 und 1)	x
	decimal	rationale Zahlen	x
	int	ganze Zahlen	x
	money	Währungen	
	numeric	rationale Zahlen	x
	smallint	ganze Zahlen	x
	smallmoney	Währungen	
	tinyint	ganze Zahlen	x
Ungefähr numerische Werte	*float*	rationale Zahlen	
	real	rationale Zahlen	
Datum und Uhrzeit	*date*	Datum	x
	datetime	Datum und Zeit	x
	datetime2	Datum und Zeit	
	datetimeoffset	Datum und Zeit	
	smalldatetime	Datum und Zeit	
	time	Zeit	x
Zeichenfolgen	*char*	Zeichenketten	x
	varchar	Zeichenketten	x
	text	Zeichenketten	
	nchar	Zeichenketten (Unicode)	
	nvarchar	Zeichenketten (Unicode)	
	ntext	Zeichenketten (Unicode)	
Binärzeichenfolgen	*binary*	Binärdaten (feste Länge)	
	varbinary	Binärdaten (variable Länge)	
	image	Binärdaten (variable Länge)	

Tabelle 3-2: Microsoft SQL Datentypen[17]

[16] Laube, Michael: Einstieg in SQL, S. 181.

[17] Vgl. Microsoft Developer Network, Stichwort: Datentypen (Transact-SQL), online im Internet

Datentypen können über SQL in andere Typen konvertiert werden. Dies geschieht entweder über eine *explizite* oder *implizite Typkonvertierung*.
Die explizite Umwandlung erfolgt mit dem Schlüsselwort CAST, somit wird der Datenbank ausdrücklich mitgeteilt, dass der Wert in einen bestimmten Typ konvertiert werden soll. Im unten angeführten Beispiel wird eine ganze Zahl explizit in eine Dezimalzahl umgewandelt.

```
SELECT CAST (4 AS DECIMAL(6, 3)) AS ergebnis
/* Umwandlung der ganzen Zahl 4 in eine dezimale Zahl (maximal 6-stellig und 3 Nach-
kommastellen) > ergebnis = 4.000 */
```

Bei der implizierten Typkonvertierung versucht das Datenbanksystem bei Vergleichen, Wertzuweisungen oder mathematischen Operationen automatisch eine Übereinstimmung bei unterschiedlichen Typen zu finden.[18]
Zur besseren Verständlichkeit soll wieder das nachstehende Beispiel dienen, in dem eine rationale Zahl mit einer Zeichenkette summiert wird.

```
SELECT (3.3 + '7') AS ergebnis
/* Die Zeichenkette 7 wird zur rationalen Zahl 3.3 addiert
      a) ergebnis in Microsoft SQL = 10.3 (rationale Zahl)
      b) ergebnis in MySQL = 10.300000000000001 */
```

Hier ist darauf zu achten, in welchem Datenbanksystem gearbeitet wird, denn wie im obigen Beispiel ersichtlich, unterscheiden sich die Ergebnisse zwischen Microsoft SQL und MySQL.

Aus diesem Grund sollte immer im Vorhinein abgeklärt werden, welche Art von Datenbank abgefragt wird und bei Unsicherheiten ist die explizite Typkonvertierung anzuwenden.

[18] Vgl. Laube, Michael: Einstieg in SQL, S. 200.

4 Java

Definition:

> „Java ist eine objektorientierte Sprache, die auch Konzepte aus der imperativen Programmierung besitzt. *Objektorientierung* bedeutet (ganz grob gesagt), dass man versucht, ein Programm wie in der echten Welt mit Hilfe von Objekten, die miteinander interagieren, zu modellieren. *Imperatives Programmieren* bedeutet, dass man dem Computer eine Reihe von Anweisungen gibt, die festlegen, in welcher Reihenfolge was gemacht wird.
>
> [...] Von anderen bekannten Sprachen wie C und C++ grenzt sich Java ab durch einen Verzicht auf einige fehleranfällige Konzepte wie manuelle Speicherverwaltung und Zeigerarithmetik.“[19]

Java wurde ursprünglich von Sun Microsystems (heute Oracle Group) entwickelt und zählt zu den höheren Programmiersprachen. Diese Sprachen heben sich von der Ebene der Maschinensprachen durch ihre Komplexität ab.

Der Hauptunterschied von Java zu anderen Hochsprachen ist die Umwandlung des Quellcodes zum Maschinencode. Diese erfolgt bei Java nämlich durch einen Compiler UND Interpreter, wohingegen andere Sprachen einen Compiler ODER einen Interpreter nutzen.

Der **Compiler** wandelt den Quellcode bereits auf dem Quellsystem vor der Ausführung der Anwendung komplett in Maschinencode um.

Vorteile:

- *sehr schnelle Ausführungsgeschwindigkeit*
- *keine direkte Weitergabe des Quellcodes ans Zielsystem*

Nachteile:

- *plattformabhängiger Maschinencode*
- *Debugging (Fehlersuche) sehr aufwändig*

[19] Lorig, Daniel: Java-Programmierung für Anfänger, S. 6.

Beim **Interpreter** wird direkt mit der Quelldatei gearbeitet. Die Anwendung wird erst auf dem Zielsystem Schritt für Schritt bzw. Anweisung für Anweisung abgearbeitet.

Vorteile:

- *leichtere Fehlersuche für den Entwickler*
- *Einzelschrittmodus (step-by-step) möglich*

Nachteile:

- *langsamere Ausführungsgeschwindigkeiten*
- *Quellcode wird immer direkt im Zielsystem benötigt*

Die nachfolgenden beiden Abbildungen zeigen eine schematische Darstellung des Compiler- und Interpreter-Systems.

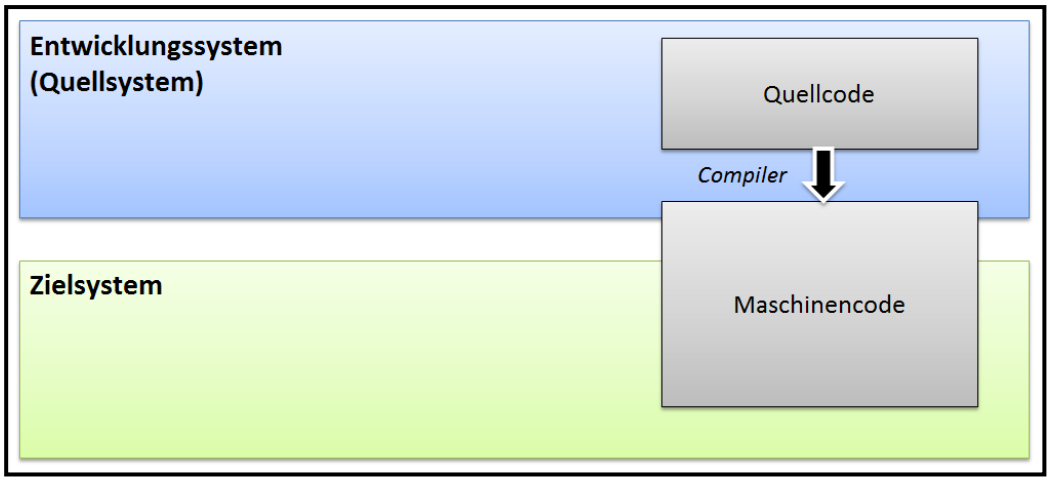


Abbildung 4-1: Hochsprache mit Compiler[20]

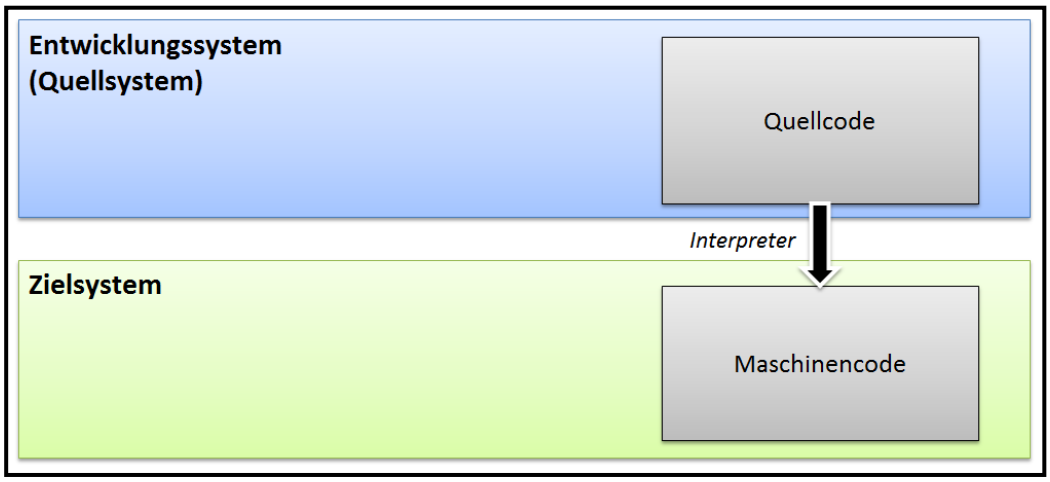


Abbildung 4-2: Hochsprache mit Interpreter[20]

[20] Vgl. Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 28.

In Java werden die beiden genannten Systeme kombiniert. Zunächst wird der Quellcode mittels eines Java-Compilers in einen sogenannten *Bytecode* umgewandelt. Anschließend wird dieser Bytecode über den Interpreter, auch *virtuelle Maschine (VM)* oder *Java Runtime Environment (JRE)* genannt, in Maschinencode konvertiert.

Vorteile:

- *plattformunabhängig*

> „Dadurch, dass für alle gebräuchlichen Betriebssysteme Java-Compiler und virtuelle Maschinen verfügbar sind, besteht ein wesentlicher Vorteil der Programmiersprache Java in der *Plattformunabhängigkeit*.“[21]

- *keine direkte Weitergabe des Quellcodes ans Zielsystem*

Nachteile:

- *etwas langsamere Ausführungsgeschwindigkeit (zweifache Umwandlung)*
- *deshalb nur bedingt für zeitkritische Anwendungen geeignet*

Die folgende Abbildung zeigt die schematische Übersetzung eines Java-Quellcodes in den Bytecode und anschließend in den Maschinencode.

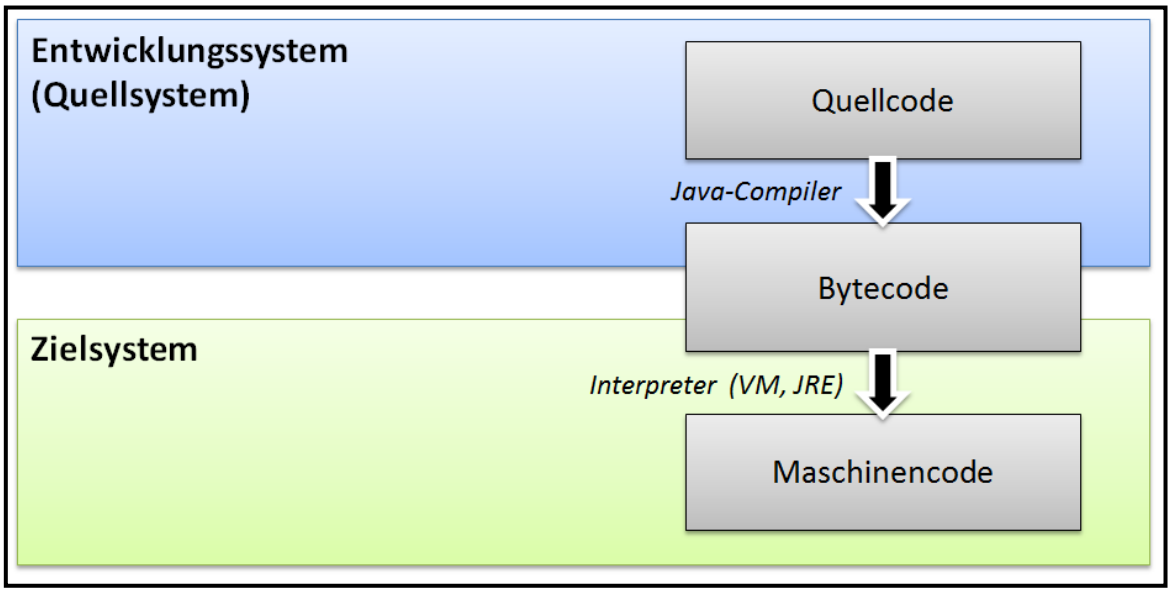


Abbildung 4-3: Compiler- und Interpreter-Kombination in Java[22]

[21] Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 33.

[22] Vgl. Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 33.

Der vorhin genannte Vorteil der Plattformunabhängigkeit war entscheidend für die Auswahl der Programmiersprache Java, weil die Software in Zukunft auch auf anderen Betriebssystemen außer Windows lauffähig sein soll.
Die einzige, nennenswerte Anpassung, die dann implementiert werden muss, ist die Art der Anmeldung an der AOI-Datenbank. Auf beispielsweise einem Linux-System kann die Windows Authentication logischerweise nicht zum Login genutzt werden. Eine temporäre, wenn auch etwas unsicherere, Lösung wäre hier die Standard-Anmeldung über einen Benutzernamen und ein Kennwort.

Der Nachteil der etwas geringeren Ausführungsgeschwindigkeit ist für die Anwendung unkritisch.

4.1 Java Development Kit

Grundsätzlich muss bei der Installation von Java zwischen *JRE (Java Runtime Environment)* und *JDK (Java Development Kit)* unterschieden werden.

Das JRE ist, wie bereits erwähnt, eine virtuelle Maschine (auch Laufzeitumgebung genannt), die den Java-Bytecode in Maschinencode umwandelt. Diese Laufzeitumgebung wird benötigt, um Java-Programme am jeweiligen System ausführen zu können.

Um jedoch Java-Programme erstellen zu können, ist das JDK erforderlich, welches eine Sammlung von Komponenten zur Entwicklung und Erstellung von Java-Programmen enthält. Die wichtigsten Bestandteile des Development Kits sind der Java-Compiler, eine hohe Anzahl von Bibliotheken, aber auch die Laufzeitumgebung. Das bedeutet, bei der Installation des JDK ist eine Minimalvariante des JRE enthalten.[23]

Sowohl das JRE als auch das JDK können für jedes gängige Betriebssystem über die Hersteller-Webseite *http://www.oracle.com/* heruntergeladen werden.

[23] Vgl. Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 35.

4.2 Integrated Development Environment

Nach der Installation des JDK ist es bereits möglich, Java-Programme zu erstellen und auszuführen. Jedoch kann dies nur sehr umständlich über einen Texteditor und die Kommandozeile durchgeführt werden.

Zur komfortableren Programmierung und Anwendungsausführung bzw. zur einfacheren Fehlersuche werden deshalb sogenannte *integrierte Entwicklungsumgebungen (kurz IDE, engl. für Integrated Development Environment)* zur Verfügung gestellt, weil diese besonders viele Funktionen zur Steigerung der Benutzerfreundlichkeit für den Programmierer anbieten.

Es existiert eine sehr hohe Anzahl an kommerziellen und frei verfügbaren Entwicklungsumgebungen für die Java-Programmierung, besonders beliebt ist hier die kostenlos verfügbare Software Eclipse, welche über die offizielle Webseite *http://www.eclipse.org/* heruntergeladen werden kann. Diese Entwicklungsumgebung bietet viele nützliche Features zur Unterstützung des Programmierers an, wie zum Beispiel das Syntax Highlighting, welches automatisch Schlüsselwörter farbig markiert oder den Debug-Modus zur Erleichterung der Fehlersuche. Außerdem wird der eingegebene Quellcode bereits während der Benutzereingabe im Hintergrund kompiliert und der Programmierer wird sofort über etwaige Syntax-Fehler informiert.[24]

Für die Fehlerdatenauswertung-Software wurde Eclipse in der Version 'Mars' verwendet. Zwischenzeitlich existieren bereits zwei neuere Releases ('Neon' und 'Oxygen') für Eclipse. Da die ältere Version aber bereits am Entwicklungsrechner vorhanden war, und diese für die Erstellung des Auswertungsprogramms völlig ausreichend ist, wurde hier keine neuere Version installiert.

[24] Vgl. Lorig, Daniel: Java-Programmierung für Anfänger, S. 7.

4.3 Datentypen

In Java sind acht sogenannte primitive Datentypen integriert, welche in der nachfolgenden Tabelle aufgelistet sind.

Datentyp	Verwendung
boolean	Wahrheitswert (0 oder 1 bzw. false oder true)
char	Einzelnes Zeichen (z.B. 'a')
byte	Ganzzahl
short	Ganzzahl
int	Ganzzahl
long	Ganzzahl
float	Kommazahl
double	Kommazahl

Tabelle 4-1: Primitive Datentypen in Java[25]

Diese Datentypen repräsentieren einfache Werte und nehmen nur wenig Speicherplatz in Anspruch. Außerdem erzeugen diese Geschwindigkeitsvorteile, weil sie nur einen geringen Aufwand für den Compiler und Interpreter bedeuten.[26]

Auch in Java ist es möglich Datentypen in andere zu konvertieren, was auch als Casting bezeichnet wird. So wie in SQL wird auch hier in implizites und explizites Casting unterteilt.
Bei der impliziten Umwandlung wird immer in den Datentyp mit dem größeren Wertebereich konvertiert.

```
short klein;                    // Wertebereich 16 Bit
long gross;                     // Wertebereich 64 Bit
gross = klein;                  // klein wird implizit auf 64 Bit umgewandelt
```

Über das explizite Casting ist es möglich, eine ausdrückliche Umwandlung durchzuführen. Diese Konvertierung wird angewandt, um beispielsweise einen Datentyp mit großem Wertebereich in einen mit kleinerem Wertebereich zu konvertieren. Hier muss aber beachtet werden, dass es zu einem möglichen Datenverlust kommen kann.

```
short klein;                    // Wertebereich 16 Bit
long gross;                     // Wertebereich 64 Bit
klein = (short) gross;          // gross wird explizit auf 16 Bit umgewandelt
                                // ACHTUNG: Möglicher Datenverlust
```

[25] Vgl. Lorig, Daniel: Java-Programmierung für Anfänger, S. 25f.
[26] Vgl. Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 156.

4.4 Objektorientierung

> „Das wichtigste Konzept in Java ist die Objektorientierung, diese ist dort allgegenwärtig. Objektorientierung bedeutet, dass man versucht, ein Programm als Reihe von interagierenden Objekten zu modellieren, wie in der realen Welt.“[27]

Klasse:

Durch Klassen werden neue Datentypen definiert, welcher der Programmierer auf seine Anwendung und Bedürfnisse individuell zuschneiden kann. Diese selbst generierten Datentypen sind weitaus komplexer und leistungsfähiger als die primitiven Datentypen. Sie können beispielsweise mehrere Werte speichern, auf Botschaften reagieren und auch selbst aktiv werden.[28]

> „Eine Klasse beschreibt den Aufbau eines komplexen Datentyps. Eine Klasse wird durch *Eigenschaften* (Datenelement oder Attribut) und ihre *Fähigkeiten* (Methoden) beschrieben.“[29]

Klassen werden durch das Schlüsselwort class eingeleitet und die Bezeichnung der Klasse beginnt im Regelfall mit einem Großbuchstaben.

Methode:

> „Eine Methode ist eine Ansammlung von Code-Anweisungen, die auf Befehl ausgeführt werden können. Wenn man z.B. denselben Code mehrmals an verschiedenen Stellen ausführen möchte, bietet es sich an, den Code in eine Methode zu stecken. Dann genügt es, die Methode dort, wo der Code benötigt wird, einfach nur noch über den Methodennamen aufzurufen.“[30]

Der Aufruf einer Methode in der eigenen Klasse erfolgt einfach über den Methodenbezeichner. Ist es erforderlich diese Methode in einer fremden Klasse abzurufen, so muss die sogenannte Punktnotation angewendet werden, das heißt, vor dem Methodennamen muss der Klassenbezeichner und ein Punkt eingefügt werden.

```
methode1();                    // Methoden-Aufruf in der eigenen Klasse
Klasse1.methode1();            // Aufruf in einer fremden Klasse über Punktnotation
```

Um die Bezeichnungen der einzelnen Methoden auf einen Blick von Klassennamen differenzieren zu können, sollten diese vorzugsweise mit einem Kleinbuchstaben beginnen.

---

[27] Lorig, Daniel: Java-Programmierung für Anfänger, S. 18.

[28] Vgl. Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 156.

[29] Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 158.

[30] Lorig, Daniel: Java-Programmierung für Anfänger, S. 47.

Konstruktor:

Der sogenannte Konstruktor stellt eine besondere Art von Methode dar. Dieser erstellt eine neue Instanz einer Klasse und dient zur Initialisierung (z.B. Startwerte für Variablen festlegen, etc.).[31]

Konstruktorennamen sind immer identisch mit den Klassennamen und es erfolgt kein void oder vorangestellter Datentyp wie bei Methoden. So ist es dem Compiler möglich zwischen Standard-Methode und Konstruktor zu unterscheiden.

Eine Klasse kann keinen oder einen Konstruktor enthalten oder aber auch mit mehreren Konstruktoren ausgestattet sein. Bei Klassen mit keinem Konstruktor wird automatisch ein Standard- oder Default-Konstruktor erstellt und allen Attributen werden Standardwerte zugewiesen. Der Konstruktor in der Standardausführung enthält keine Parameter und ist auch der am häufigsten eingesetzte.
Da alle Konstruktoren einer Klasse den gleichen Namen tragen, muss bei der Verwendung von mehreren Konstruktoren darauf geachtet werden, dass die Parameteranzahl und deren Typen eindeutig sind. Das heißt, jede Konstellation darf nur einmal vorkommen.Nur so ist es möglich, beim Aufruf eines Konstruktors, den richtigen anzusprechen.

Objekt:

> „Ein Objekt ist ein Exemplar (Instanz), das nach dem Bauplan einer Klassendefinition erstellt wurde. Die Klasse stellt den Bauplan dar [...]. Das Objekt ist eine Variable, die nach diesem Plan aufgebaut ist.“[32]

Um ein neues Objekt einer Klasse erstellen zu können, wird der bereits beschriebene Konstruktor benötigt.

Das folgende Kurzbeispiel zeigt die Erstellung eines neuen Objekts der Klasse String mit Hilfe des Schlüsselwortes new. Über diese Klasse ist es möglich, eine beliebig definierbare Zeichenkette in einer Variablen abzuspeichern. Außerdem können anschließend Zusatzfunktionen der String-Klasse über die bereits erwähnte Punktnotation aufgerufen werden.

```
String text = new String("Mein Text");        // neues Objekt der Klasse String
int laenge = text.lenght();                   // Speichern der Länge des Strings
                                              // im primitiven Datentyp 'int'
```

[31] Lorig, Daniel: Java-Programmierung für Anfänger, S. 51.

[32] Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 163.

5 Anforderungen

Definition:

„Eine Anforderung beschreibt, was der Kunde oder Benutzer von seinem Produkt erwartet (Bedingungen, Attribute, Ziele, Nutzen, etc.). Anforderungen sind definiert als [IEEE1990]:

- Eine Eigenschaft oder Bedingung, die von einem Benutzer (Person oder System) zur Lösung eines Problems oder zur Erreichung eines Ziels benötigt wird.
- Eine Eigenschaft oder Bedingung, die ein System oder eine Systemkomponente erfüllen muss, um einen Vertrag, eine Norm oder andere, formell vorgegeben Dokumente zu erfüllen.
- Eine dokumentierte Repräsentation einer Eigenschaft oder Bedingung wie in den ersten beiden Punkten beschrieben.“ [33]

Anforderungen präzisieren also Ziele und schaffen eine Basis für die Lösungsansätze und Lösung einer Aufgabenstellung. Ohne diese vereinbarten und definierten Ziele sind das Ergebnis und die Qualität eines Produkts und dessen Eigenschaften nicht vorhersehbar. Außerdem ist das Resultat hauptsächlich von den Vorlieben und Interpretationen der beteiligten Personen (z.B. Entwickler, Vertrieb, etc.) und gegebenenfalls auch von deren Beziehungen zueinander abhängig.[34]

Aus diesem Grund ist es auch sehr wichtig, Qualitätskontrollen von Anforderungen durchzuführen, weil Fehler, die während der Anforderungserstellung und Dokumentation auftreten, wesentlich einfacher und vor allem kostengünstiger beseitigt werden können als im Nachhinein.

[33] Ebert, Christof: Systematisches Requirements Engineering, S. 21

[34] Vgl. Ebert, Christof: Systematisches Requirements Engineering, S. 51

5.1 Anforderungsarten

Anforderungen werden im Wesentlichen in drei Hauptgruppen unterteilt:

- Funktionale Anforderungen
- Qualitätsanforderungen
- Randbedingungen

In der nachstehenden Abbildung ist die Unterteilung schematisch dargestellt und es sind einige zugehörige Beispiele aufgelistet.

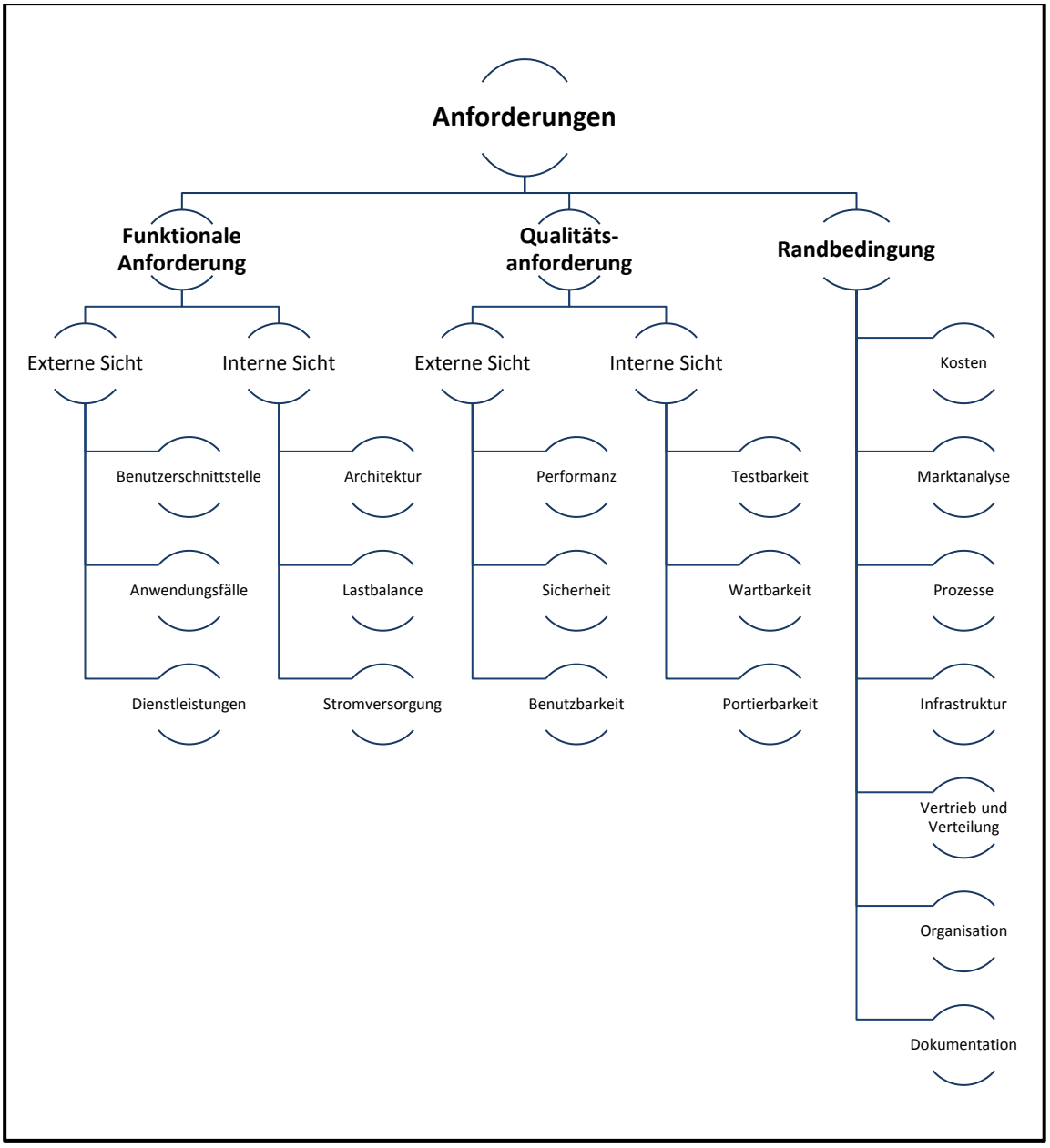


Abbildung 5-1: Anforderungsarten[35]

[35] Vgl. Ebert, Christof: Systematisches Requirements Engineering, S. 29

5.2 Produktspezifische Anforderungen

Die produktspezifischen, funktionalen Anforderungen werden aus der Aufgabenstellung abgeleitet. Die Titel der nachfolgenden Unterkapitel sind Schlagwörter zu den geforderten Grundfunktionen, wobei die ersten beiden nur die unterschiedlichen Modi zum Datenabruf darstellen. sollen

5.2.1 Einzelselektion

Mit dem Modus *Einzelselektion* soll der Benutzer in der Lage sein, einen bestimmten Fertigungsauftrag detailliert analysieren zu können. Nach der Eingabe der Auftragsnummer und Abfrage der Daten aus der Datenbank sollen die Daten in verschiedensten Formen weiterverarbeitet werden können.

Um eine fehlerhafte Abfrage zu verhindern muss das Eingabefeld so formatiert sein, dass nur gültige Zahlen eingegeben werden können. Der Nummernkreis für die Fertigungsaufträge liegt zwischen 2013800000 und 2017899999.

5.2.2 Mehrfachselektion

Über den Modus *Mehrfachselektion* muss die Software dem Benutzer Informationen in einem gewissen Zeitraum zur Verfügung stellen. Hierzu soll der Benutzer über ein Auswahlfeld ein Start- und Enddatum definieren können, um anschließend alle Fertigungsaufträge im gewünschten Zeitfenster abzurufen.

Durch die Größe dieser gewählten Zeitspanne steigt natürlich auch die Anzahl der darin enthaltenen Fertigungsaufträge und Daten. Um hier zu großen Datenverkehr zu vermeiden, und da das Feature *Tabellen* in diesem Modus ohnehin kein sinnvolles Ergebnis liefern kann, soll diese Funktion hier in diesem Modus nicht zu Verfügung stehen.

5.2.3 Tabellen

Wie bereits im vorigen Abschnitt erwähnt, ist die Funktion *Tabellen* wegen der Größe der Datenmengen nur im Einzelselektions-Modus verfügbar.

Bei Auswahl dieses Features müssen dem Benutzer alle Produktionsdaten zu seinem eingegebenen Auftrag in Tabellenform angezeigt werden. Anschließend soll in der Tabelle die Möglichkeit bestehen, Standard-Funktionen wie Sortieren und Filtern auszuführen.

Die angezeigten Daten dürfen nicht vom Benutzer verändert werden können, um Falscheingaben zu vermeiden. Das Markieren und Kopieren einzelner Zeilen soll jedoch möglich sein.

5.2.4 Fehlerbilder

Zu jedem aufgezeichneten Fehler wird ein Bild auf einem externen Rechner gespeichert. Diese *Fehlerbilder* sind jedoch in keinem Standard-Format gespeichert, sondern im *Hersteller-Format 'ois'*. Zum Öffnen dieses Dateityps muss das Zusatzprogramm OIS-Viewer vom AOI-Hersteller auf dem Zielsystem vorhanden sein.

Da in der Tabelle alle relevanten Daten vorhanden sind, ist es von Vorteil diese Option in der Tabellenübersicht unterzubringen.

5.2.5 Diagramme

Das Anzeigen von Diagrammen soll dem Benutzer die Möglichkeit bieten, die wichtigsten und ausschlaggebendsten Daten auf schnelle Weise visuell darstellen zu können.

Hierzu kann sich das Programm im Einzel- oder Mehrfachselektionsmodus befinden. Es werden hier entweder die Daten eines Fertigungsauftrages oder mehrerer Aufträge in einem Zeitraum aufgeschlüsselt.

Vor der Erstellung des Diagramms muss der Benutzer die Filtereinstellungen für die Diagrammberechnung anpassen. So können zum Beispiel alle Fehler, nur Fehler einer bestimmten Seite, nur Pseudofehler, etc. angezeigt werden.

Die Darstellung soll in einem Säulendiagramm erfolgen, wobei die Säulen in absteigender Reihenfolge sortiert anzuzeigen sind. Dies hat den Vorteil, dass die häufigsten Fehler am Anfang des Diagramms aufgelistet und somit sofort erkennbar sind.

5.2.6 First Pass Yield

Der First Pass Yield (kurz FPY) ist eine Qualitätskennzahl in der Qualitätssicherung und im Qualitätsmanagement. Diese Zahl gibt die Anzahl von Baugruppen oder Bauteilen wieder, welche beim ersten Produktionsdurchlauf ohne Nacharbeit, also ohne Fehler und Reparaturen, gefertigt wurden.
Die häufigste Angabe des FPY erfolgt in Prozent, das heißt es wird die Ausbeute in Bezug auf 100 Einheiten angegeben. Je höher somit der FPY ist, umso höher ist die Gesamtqualität eines Fertigungsauftrages bzw. umso stabiler und sicherer laufen die Produktionsprozesse.

Der Funktionsaufruf im Programm soll, je nach Selektionsmodus, die FPY-Daten eines bzw. mehrerer Aufträge berechnen und in einer Tabelle anzeigen.
Es sollen außerdem die Grunddaten der Berechnung angezeigt werden:

- Anzahl der inspizierten Baugruppen
- Anzahl von Baugruppen ohne Nacharbeit
- Anzahl der Baugruppen mit Pseudofehlern
- Anzahl der Baugruppen mit echten Fehlern

5.2.7 Export

Die Export-Funktion ist für die Weiterverwendung und -bearbeitung der Daten hilfreich. Beim Aufruf sollen, je nach Auswahl, die Daten der *Tabelle*, das *Diagramm* oder die *FPY*-Daten exportiert werden. Im Mehrfachselektionsmodus ist natürlich nur der Export des Diagramms oder des FPYs möglich.

Da Tabellen- und FPY-Daten bereits im praktischen Tabellenformat vorliegen, soll die Export-Funktion diese beiden Datensätze im *CSV-Format (Comma-Seperated Values)* mit der Dateiendung *.csv* und einem Semikolon als Trennzeichen speichern. Diese Dateien können dann mit einem gewöhnlichen Text-Editor oder Tabellenkalkulationsprogramm geöffnet und weiterbearbeitet werden.

Der Diagramm-Export soll das Säulendiagramm in einem gängigen Bildformat abspeichern. Zur Auswahl sollen die beiden wichtigsten und gebräuchlichsten Formate zur Verfügung stehen. Diese sind das *JPEG (Joint Photographic Experts Group)* und das *PNG (Portable Network Graphics)*.
Außerdem ist beim Speichern der Diagramme darauf zu achten, dass die Größe einer exportierten Bilddatei nicht übermäßig viel Speicherplatz einnimmt (< 1 Megabyte).

6 Entwurf und Implementierung

6.1 Grobentwurf

Der Grobentwurf soll den Lösungsansatz ohne einzelne Details in schematischer Form darstellen. Er enthält die Grundfunktionen der geforderten Anwendung und dient als Anhaltspunkt für das Projekt. Außerdem soll er die bereits erarbeiteten Anforderungen beinhalten und diese in Kategorien unterteilen.

Die Grundidee bestand darin, das Programm in zwei verschiedene Auswertungsmodi zu unterteilen, nämlich in die Einzelselektion und die Mehrfachselektion. Der Benutzer muss vor dem Start der Auswertung die Art der Selektion auswählen und hat dann die Möglichkeit auf die jeweiligen Einzelfunktionen zuzugreifen.

Im Wesentlichen wurde aus jeder Anforderung eine eigene Klasse gebildet, so wird beispielsweise die FPY-Funktion über eine eigene Klasse bzw. deren Methoden berechnet und in Tabellenform ausgegeben. Die genauen Objekte und Einzelfunktionen der Klassen werden im nächsten Kapitel (Feinentwurf) detaillierter beschrieben.
Als übergeordnete Klasse dient das Hauptfenster. Aus diesem kann, je nach Auswahlmodus (Einzel- bzw. Mehrfachselektion), über eingebaute Schaltflächen auf die einzelnen Klassen bzw. Methoden zugegriffen werden.

Auf der nachfolgenden Seite sind die grundlegende schematische Einteilung und die Grundfunktionen ersichtlich. Es handelt sich hierbei um einen der ersten Entwürfe.

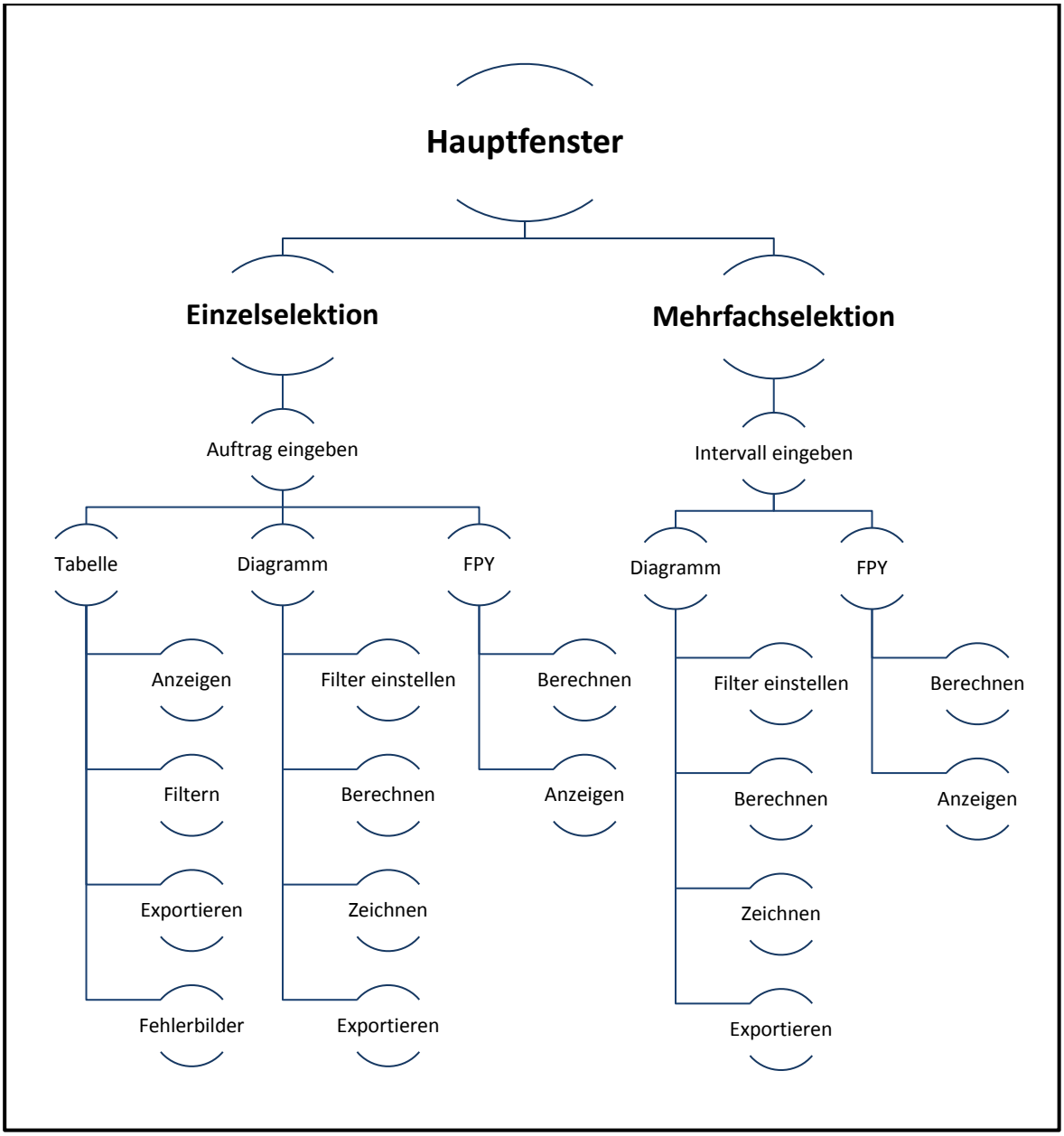


Abbildung 6-1: Einteilung der Funktionen (Grobentwurf)

6.2 Einbindung externer Java-Komponenten

Zur Unterstützung des Entwicklers ist bereits eine große Anzahl an Bibliotheken bei der Installation von Java enthalten. Zusätzlich stehen im Internet zahllose, kostenlose (Open Source) Zusatzbibliotheken für Java zur Verfügung, welche nahezu alle Sparten abdecken. Das heißt, es existiert beinahe für jede Anforderung bereits ein Grundmodul, welches vom Entwickler nur noch auf die eigene Anwendung angepasst werden muss.

Um diese Sonderbibliotheken zu integrieren, sollten die Bibliotheks-Dateien vorzugsweise im Projektordner abgelegt werden. In der Fehlerdatenauswertung wurde ein im Projektordner ein eigener Unterordner mit dem Namen 'lib' erstellt, in welchem diese Dateien gespeichert wurden.

Die benötigten Bibliotheksdateien müssen anschließend im Eclipse-Projekt eingebunden werden. Diese Einbindung geschieht mit Hilfe der Eclipse-Oberfläche über das Menü 'Project-Properties/Java Build Path'.[36]

Der nachfolgende Screenshot aus Eclipse zeigt den Projekt-Ordner 'lib' mit den enthaltenen und integrierten Zusatzbibliotheken.

Abbildung 6-2: Zusatzbibliotheken in Eclipse

Die nähere Erläuterung und Funktion der einzelnen Bibliotheksdateien ist auf den nachfolgenden Seiten ersichtlich.

[36] Vgl. Eclipse documentation, Stichwort: Java Build Path, online im Internet

6.2.1 Java Database Connectivity (JDBC)

Für die Anbindung einer Java-Applikation an eine SQL-Datenbank wird die sogenannte JDBC-Schnittstelle benötigt. Diese wurde bereits in der Java Version 1.1 von Sun Microsystems in Anlehnung an Microsofts ODBC (Open Database Connectivity) entwickelt, um jegliche Art von relationalen Datenbankmanagementsystemen über Java ansprechen zu können. Bei JDBC und ODBC handelt es sich um standarisierte Datenbankschnittstellen, welche die Datenbanksprache SQL zur Kommunikation verwenden. Ohne einen Treiber muss die Anwendung direkt mit der Datenbankschnittstelle kommunizieren, das bedeutet, dass für jede unterschiedliche Datenbank ein eigener Quellcode zur Kommunikation benötigt wird.[37]

Der JDBC-Treiber für Microsoft SQL Server Datenbanken kann kostenlos über die Microsoft-Homepage unter *https://www.microsoft.com/de-DE/download/details.aspx?id=11774* heruntergeladen werden.

Im Download-Paket sind mehrere Ordner und Dateien enthalten, so ist beispielsweise eine JDBC-Datei für Java 7 und eine andere für Java 8 enthalten. Da alle Rechner, auf denen die Fehlerauswertung ausgeführt werden soll, aber bereits Java 8 enthalten, wurde hier die aktuellere JDBC-Version 4.2 in Eclipse eingebunden (Datei 'sqljdbc42.jar').

6.2.2 Windows Authentication

Zusätzlich zur Schnittstellen-Datei ist beim Download von JDBC für SQL Server auch eine Bibliotheksdatei für die Windows Authentication enthalten (Datei 'sqljdbc_auth.dll'). Diese dient, wie bereits im Kapitel 2 (Datenbanken) detailliert beschrieben, zum sicheren Verbinden mit der Microsoft SQL Datenbank über die Anmeldedaten des Windows-Benutzers.

Diese Bibliotheksdatei ist aber nicht, wie alle anderen Zusatzbibliotheken, eine JAR-Datei, sondern eine Dynamic Link Library (DLL). Um diese Datei und die damit verbundene Windows Authentication mit Java nutzen zu können, gibt es zwei Möglichkeiten:

- Kopieren der Datei ins System-Verzeichnis 'C:/Windows/System32/'
- Verweis auf den Pfad in dem sich die Datei befindet ('native Library' im Eclipse Build Path)

Für die Fehlerdatenauswertung wurde die zweite Variante gewählt. Die DLL-Datei wurde gemeinsam mit allen anderen Zusatzbibliotheks-Dateien im Projekt-Verzeichnis 'lib' abgelegt. Die Datei wird bei der schlussendlichen Erstellung der ausführbaren JAR-Datei zwar nicht mitexportiert, diese muss sich aber nur im gleichen Pfad wie die ausführbare JAR-Datei befinden, damit die Kommunikation mit der Datenbank funktioniert.

---

[37] Vgl. Habelitz, Hans-Peter: Programmieren Lernen mit Java, S. 503.

6.2.3 JFreeChart

Zur Erstellung der Diagramme wurde die frei verfügbare Bibliothek JFreeChart in der Version 1.0.19 von der Internetseite *http://www.jfree.org/jfreechart/* heruntergeladen und eingebunden (Datei 'jfreechart-1.0.19.jar'). Mit dieser können die verschiedensten Diagrammarten (Balken-, Säulen-, Torten-Diagramme, etc.) erstellt werden. Außerdem ist eine Speicherfunktion für Bilddateien ebenfalls integriert, welche ja, wie im Kapitel Anforderungen bereits erwähnt, benötigt wird.

Für die Benutzung von JFreeChart mussten die enthaltenen Funktionen und Methoden der Bibliothek analysiert werden. Eine detaillierte Originaldokumentation muss kostenpflichtig über die vorhin genannte Webseite erworben werden.
Da die Diagramme in der Fehlerdatenauswertungssoftware aber eher schlicht und ohne viele Extras auskommen, wurde hier auf den folgenden Internetseiten recherchiert, um einige Beschreibungen und Beispiele als Anhaltspunkte für die Diagrammgenerierung zu finden.

http://www.jfree.org/forum/

http://www.tutorialspoint.com/jfreechart/

http://www.java2s.com/Code/Java/Chart/Bar-Chart.htm

Nachfolgend ein Beispiel eines Säulendiagramms erstellt in JFreeChart. Beim Download der Bibliotheksdateien sind einige dieser Beispiele als Demo enthalten.

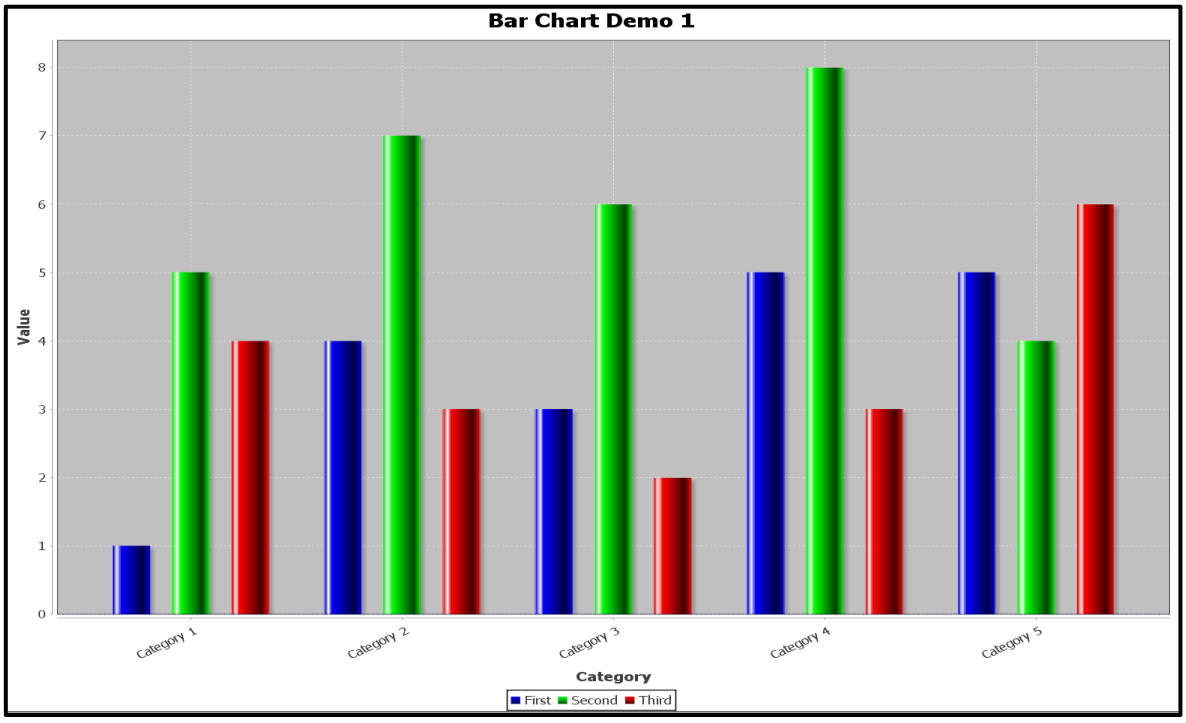


Abbildung 6-3: JFreeChart-Beispiel[38]

[38] JFreeChart-Webseite, Stichwort: Samples, online im Internet

6.2.4 JCommon

Zusätzlich nutzt JFreeChart einige Funktionen von der ebenfalls frei verfügbaren JCommon-Bibliothek. Letztere wurde in der Version 1.0.23 auf der Webseite *http://www.jfree.org/jcommon/* heruntergeladen (Datei 'jcommon-1.0.23.jar'). Sie enthält Grundfunktionen wie beispielsweise Textanpassung, Layout Manager, Kalender zur Datumsauswahl, usw.

JCommon wird aktuell nicht mehr weiterentwickelt und die relevanten Funktionen werden in naher Zukunft direkt in die JFreeChart-Bibliothek integriert.[39]

6.2.5 JDatePicker

Da die manuelle Eingabe des Start- und End-Datums im Mehrfachselektionsmodus für den Anwendungsbenutzer mit sehr viel Aufwand verbunden ist und außerdem auch viele Falscheingaben entstehen können, wurde die frei verfügbare Zusatzbibliothek JDatePicker im Projekt integriert. Der Download-Link und die Beschreibung sind auf der Internetseite *https://jdatepicker.org/* zu finden, wobei für das Projekt die Version 1.3.4 heruntergeladen und eingebunden wurde (Datei 'jdatepicker-1.3.4.jar').

Die Bibliothek ermöglich es, dem Benutzer das Datum über eine vorgefertigte grafische Oberfläche auszuwählen. Die Auswahl wird dann direkt in ein nebenstehendes Textfeld übertragen, welches nur nach dessen Inhalt abgefragt werden muss.

Für die Abfrage eines Datumsintervalls müssen natürlich zwei JDatePicker-Module eingefügt werden. Die eine Komponente dient zum Eintrag des Startdatums und die andere für die Eingabe des Enddatums.

[39] Vgl. JFreeChart-Webseite, Stichwort: JCommon, online im Internet

6.3 Feinentwurf

Im Feinentwurf werden die Klassen und deren Objekte (Methoden, Variablen, etc.) definiert. Er basiert auf den Grundlagen der Anforderungen und wird im Grunde genommen direkt vom Grobentwurf abgeleitet.

Im Fall der Fehlerauswertungs-Software wurden als erstes die SQL-Abfragen entwickelt und verfeinert, anschließend wurden die bereits erwähnten externen Java-Komponenten im Projekt eingefügt und zu guter Letzt wurden die einzelnen Klassen ausprogrammiert.

6.3.1 SQL-Abfragen

Vor der Implementierung der einzelnen SQL-Abfragen in das Java-Programm, wurden diese im Microsoft SQL Server Management Studio getestet und so angepasst, dass der Ergebnissatz zur optimalen Weiterverarbeitung zur Verfügung steht.
Alle nachfolgenden Abfragen wurden auf den in Kapitel 3 (SQL) genannten Grundlagen entwickelt.

Detailtabelle

Die erste dieser Abfragen dient zur Ermittlung der Daten für die Detailtabelle eines einzelnen Fertigungsauftrags. Es werden hier insgesamt 16 Spalten mit allen möglichen Auftragsdetails abgefragt.

Hier bestand die besondere Herausforderung darin, die in der Datenbank enthaltenen Inspektions- und Reparaturzeiten in ein Datumsformat umzuwandeln. Die Daten dieser beiden Spalten sind nämlich als ganze Zahlen gespeichert, welche die Sekunden angeben, die seit dem 01.01.1970 00:00 Uhr vergangen sind (UNIX Timestamp).

Da Microsoft SQL Server bei Datumsangaben aber vom 01.01.1900 00:00 Uhr ausgeht, muss hier der 01.01.1970 als Datumsoffset angegeben werden.
Außerdem ist die Zeitverschiebung von '+1 Stunde' der mitteleuropäischen Zeitzone (MEZ) zu berücksichtigen.

Eine zusätzliche Abweichung entsteht durch die Umstellung auf die sogenannte Daylight Saving Time (Sommerzeit). Durch diese vergrößert sich die Zeitverschiebung um eine weitere Stunde, also insgesamt um '+2 Stunden'.
Diese Ungenauigkeit wurde derzeit noch nicht behoben, weil die Lösung dieses Problems mit sehr viel Aufwand verbunden ist. Hier müsste für jeden abgefragten Zeitstempel, eine Prüfung erfolgen, ob dieser in der Sommerzeit oder Winterzeit erstellt wurde.

```
/* --- Detailtabellendaten abfragen ---
-- Auswählen der anzuzeigenden Tabellen */
SELECT Panel_Bar_Code AS Panel, Card_Bar_Code AS Card, Card_Number,
      /* Umwandeln der Inspektionszeit von 'Sekunden seit 01.01.1970 00:00':
      a) ins Datumsformat 'dd mon yyyy hh:mi:ss:mmm' (Code 13)
      mit 1 Stunde Offset (MEZ)
      b) Umwandlung der ersten 17 Zeichen des Datums in einen String (varchar(17))
      Sekunden und Millisekunden werden abgeschnitten */
      CONVERT(VARCHAR(17), DATEADD(SECOND, Panel_Numeric_Date,
      '1970-01-01 01:00'), 13) AS Inspection_Time,
      Machine_Name AS Machine, File_Name AS Product,
      Face, Test_Time, CARDS.Anomaly_BR AS Before_Repair, CARDS.Anomaly_AR
      AS After_Repair, Topology, Part_Number, JEDEC_Name AS Jedec,
      Repair_Error_Comment AS Failure_Code, Operator_Name AS Operator,
      /* Umwandeln der Inspektionszeit von 'Sekunden seit 01.01.1970 00:00':
      a) ins Datumsformat 'dd mon yyyy hh:mi:ss:mmm' (Code 13)
      mit 1 Stunde Offset (MEZ)
      b) Speichern der ersten 17 Zeichen des Datums in einen String (varchar(17))
      Sekunden und Millisekunden werden abgeschnitten */
      CONVERT(VARCHAR(17), DATEADD(SECOND, Repair_Numeric_Date_Hour,
      '1970-01-01 01:00'), 13) AS Repair_Time

-- Ursprungstabelle
FROM TESTED_OBJECT

-- Verknüpfungen zu anderen Tabellen über JOINs herstellen
RIGHT JOIN CARDS ON CARDS.Card_Id = TESTED_OBJECT.Card_Id
INNER JOIN PANELS ON PANELS.Panel_Id = CARDS.Panel_Id
INNER JOIN MACHINE ON MACHINE.Machine_Id = PANELS.Machine_Id
INNER JOIN RECIPE ON RECIPE.RECIPE_ID = PANELS.RECIPE_ID
LEFT JOIN OPERATOR ON TESTED_OBJECT.Operator_Id = OPERATOR.Operator_Id
LEFT JOIN PART_NUMBER ON TESTED_OBJECT.Part_Number_Id = PART_NUMBER.Part_Number_Id
LEFT JOIN JEDEC ON PART_NUMBER.JEDEC_Id = JEDEC.JEDEC_Id

-- Einschränkung: nur nach dem eingegebenen String Filtern
WHERE Card_Bar_Code LIKE 'eingegebener Auftrag'

-- Sortierung
ORDER BY Panel_Bar_Code ASC
```

Diagramm

Die nächste Abfrage beschäftigt sich mit den erforderlichen Daten für die Erstellung des Diagramms. Je nach Auswahlmodus wird hier nach einem bestimmten Zeitraum oder nach einem Fertigungsauftrag abgefragt.

Bei der Abfrage der Daten eines bestimmten Intervalls ist auch hier ist die Berechnung des richtigen Datums und die Berücksichtigung der Zeitzone erforderlich, um diese Angaben als Einschränkung zu definieren.

Im Einzelselektionsmodus hingegen, muss nur der eingegebene Fertigungsauftrag als Einschränkung angegeben werden, somit werden alle Daten mit dieser Filterung

Außerdem kommen noch Zusatzfilter hinzu, die der Benutzer vor der Diagrammerstellung auswählen kann, wie zum Beispiel die Fehlerart (Pseudofehler, echte Fehler), usw.

Die nachfolgende SQL-Abfrage zeigt nur eine der vielen Varianten für eine Diagrammabfrage. Durch das Java-Programm werden hier zusätzlich noch der Modus und die Filtereinstellungen überprüft und je nach Ergebnis wird die SQL-Abfrage dann angepasst.

```
/* --- Diagrammdaten abfragen ---
-- Auswählen der anzuzeigenden Tabellen */
SELECT Count(Topology) AS 'Counter', Topology, Face

-- Ursprungstabelle
FROM TESTED_OBJECT

-- Verknüpfungen zu anderen Tabellen über JOINs herstellen
RIGHT JOIN CARDS ON CARDS.Card_Id = TESTED_OBJECT.Card_Id
INNER JOIN PANELS ON PANELS.Panel_Id = CARDS.Panel_Id
INNER JOIN MACHINE ON MACHINE.MACHINE_ID = PANELS.MACHINE_ID
INNER JOIN RECIPE ON RECIPE.RECIPE_ID = PANELS.RECIPE_ID
LEFT JOIN PART_NUMBER ON TESTED_OBJECT.Part_Number_Id = PART_NUMBER.Part_Number_Id
LEFT JOIN JEDEC ON PART_NUMBER.JEDEC_Id = JEDEC.JEDEC_Id

/* Einschränkungen: nur Werte zwischen 01. Juli 2017 00:00_00 und 23:59:59 Uhr
      Umwandeln der Inspektionszeit von 'Sekunden seit 01.01.1970 00:00':
      a) ins Datumsformat 'dd mon yyyy hh:mi:ss:mmm' (Code 13) mit 1h Offset (MEZ)
      b) Umwandlung der ersten 17 Zeichen des Datums in einen String (varchar(17))
      Sekunden und Millisekunden werden abgeschnitten */
WHERE DATEADD(SECOND, Panel_Numeric_Date, '1970-01-01 01:00')
      BETWEEN '2017-07-01 00:00:00.000' AND '2017-07-01 23:59:59.000'
      -- und keine Pseudofehler (Fehlercode 810) und nur Top-Seite
      AND Repair_Error_Comment  NOT LIKE '810' AND Face Like 'TOP'

GROUP BY Topology, Face -- Gruppieren nach Position und Seite

-- absteigende Sortierung nach Anzahl der Positionen
ORDER BY COUNT(Topology) DESC
```

First Pass Yield

Die letzte und mit Abstand umfangreichste Abfrage ist jene für die Aufbereitung der Daten zur FPY-Berechnung. Die Berechnung selbst erfolgt mit Hilfe der Java-Software.

Grundsätzlich werden hier drei Einzelabfragen mit dem SQL-Schlüsselwort UNION miteinander kombiniert. Wie bereits in Kapitel 3 (SQL) erwähnt, musste hier auf die Kompatibilität der Spalten und derer Werte geachtet werden.

Auch hier ist entweder die Einzel-Abfrage oder die Abfrage über ein Datumsintervall möglich. Die Eruierung des aktuellen Auswahlmodus und weiterer Filtereinstellungen wird wieder über die Java-Software durchgeführt, welche dann die SQL-Abfrage abändert bzw. erweitert.

Im folgenden Beispiel ist wieder nur eine Version dieser Einstellungsmöglichkeiten angeführt. Hier wird nach einem einzelnen Fertigungsauftrag ohne zusätzliche Filtereinstellungen gefiltert.

Die erste Unterabfrage soll als Ergebnis die Anzahl aller geprüften Leiterplatten ausgeben, die dem Auswahlkriterium entsprechen. Zusätzlich soll in der Spalte Code ein eindeutiger Wert (in diesem Fall 'all') hinzugefügt werden, damit die einzelnen Spalten bei der Auswertung des Ergebnissatzes in Java dann identifiziert werden können.

```
/* --- FPY-Schritt 1: Anzahl der geprüften Leiterplatten ermitteln ---
-- Auswählen der anzuzeigenden Tabellen */
SELECT COUNT(DISTINCT Card_Bar_Code) AS 'Counter', Face, 'all' AS Code,
CONVERT (varchar(10), Card_Bar_Code) AS Order_Nr

-- Ursprungstabelle
FROM TESTED_OBJECT

-- Verknüpfungen zu anderen Tabellen über JOINs herstellen
RIGHT JOIN CARDS ON CARDS.Card_Id = TESTED_OBJECT.Card_Id
INNER JOIN PANELS ON PANELS.Panel_Id = CARDS.Panel_Id
INNER JOIN MACHINE ON MACHINE.MACHINE_ID = PANELS.MACHINE_ID
INNER JOIN RECIPE ON RECIPE.RECIPE_ID = PANELS.RECIPE_ID

-- Einschränkung: nur gewählten Auftrag berücksichtigen
WHERE Card_Bar_Code LIKE 'zu berechnender Auftrag'

-- Gruppieren nach Seite und Auftragsnummer
GROUP BY Face, CONVERT (varchar(10), Card_Bar_Code)

--- Ende von Schritt 1 ---

UNION -- Verknüpfung mit nächster Unterabfrage
```

In der nächsten Unterabfrage wird die Anzahl aller Leiterplatten ermittelt, die keine Fehler enthalten. Diese müssen also nicht vom Prüfpersonal inspiziert und bewertet werden und erhalten somit auch keinen Reparaturstatus.

Aus diesem Grund ist der Reparaturstatus dieser Zeilen mit NULL versehen. Dieser NULL-Wert in die Spalte Code eingefügt, und dient wieder zu Erkennung in der Java-Software.

```
/* --- FPY-Schritt 2: Leiterplatten ohne Fehler ermitteln ---
-- Auswählen der anzuzeigenden Tabellen */
SELECT COUNT(DISTINCT Card_Bar_Code), Face, Repair_Error_Comment AS Code,
CONVERT (varchar(10), Card_Bar_Code) AS Order_Nr

-- Ursprungstabelle
FROM TESTED_OBJECT

-- Verknüpfungen zu anderen Tabellen über JOINs herstellen
RIGHT JOIN CARDS ON CARDS.Card_Id = TESTED_OBJECT.Card_Id
INNER JOIN PANELS ON PANELS.Panel_Id = CARDS.Panel_Id
INNER JOIN MACHINE ON MACHINE.MACHINE_ID = PANELS.MACHINE_ID
INNER JOIN RECIPE ON RECIPE.RECIPE_ID = PANELS.RECIPE_ID

-- Einschränkung: nur gewählter Auftrag und nicht fehlerhafte Leiterplatten
WHERE Card_Bar_Code LIKE 'zu berechnender Auftrag'
      AND Repair_Error_Comment IS NULL

-- Gruppieren nach Seite und Auftragsnummer
GROUP BY Face, Repair_Error_Comment, CONVERT (varchar(10), Card_Bar_Code)

--- Ende von Schritt 2 ---

UNION --Verknüpfung mit nächster Unterabfrage
```

Im letzten Schritt der Dreifach-Abfrage soll die Anzahl der Leiterplatten ausgegeben werden, auf denen echte Fehler aufgetreten sind. Diese Leiterplatten mussten also alle an mindestens einer Position vom Personal repariert werden.

Bei der WHERE-Klausel muss hier beachtet werden, dass die NULL-Werte und die Pseudofehler (Fehlercode 810) weggefiltert werden. Somit werden nur mehr jene Zeilen summiert, die Fehlercodes von echten Fehlern beinhalten.

Für die FPY-Berechnung ist die Art der echten Fehler belanglos. Deshalb wurde in der Spalte Code in dieser Unterabfrage nur der Wert '_real' gesetzt.

Die letzte Zeile der Abfrage dient zur Sortierung der Ergebnisdaten. Hierbei werden alle Einzelergebnisse der Unterabfragen auf einmal sortiert. Somit ist es möglich, durch die Sortierreihenfolge die Daten bereits optimal für die Weiterverarbeitung im Java-Programm vorzubereiten.

```
/* --- FPY-Schritt 3: Leiterplatten mit Fehlern aber ohne Pseudofehler ermitteln ---
-- Auswählen der anzuzeigenden Tabellen */
SELECT COUNT(DISTINCT Card_Bar_Code), Face, '_real' AS Code,
CONVERT (varchar(10), Card_Bar_Code) AS Order_Nr

-- Ursprungstabelle
FROM TESTED_OBJECT

-- Verknüpfungen zu anderen Tabellen über JOINs herstellen
RIGHT JOIN CARDS ON CARDS.Card_Id = TESTED_OBJECT.Card_Id
INNER JOIN PANELS ON PANELS.Panel_Id = CARDS.Panel_Id
INNER JOIN MACHINE ON MACHINE.MACHINE_ID = PANELS.MACHINE_ID
INNER JOIN RECIPE ON RECIPE.RECIPE_ID = PANELS.RECIPE_ID

-- Einschränkung: nur gewählter Auftrag und keine Pseudofehler und nur fehlerhafte
WHERE Card_Bar_Code LIKE 'zu berechnender Auftrag'
      AND Repair_Error_Comment NOT LIKE '810'
      AND Repair_Error_Comment IS NOT NULL

-- Gruppieren nach Seite und Auftragsnummer
GROUP BY Face, CONVERT (varchar(10), Card_Bar_Code)

--- Ende von Schritt 3 ---

-- Sortierung des Ergebissatzes (alle 3 Schritte auf einmal)
ORDER BY Order_Nr ASC, Face DESC, Code ASC
```

6.3.2 Klassendetails

Wie bereits im Grobentwurf ersichtlich, bestand der Lösungsansatz darin, die einzelnen Anforderungen in Klassen und in weiterer Folge Methoden und Attribute einzuteilen. Im nachfolgenden Bildausschnitt sind die einzelnen Klassendiagramme und deren Abhängigkeiten schematisch dargestellt.

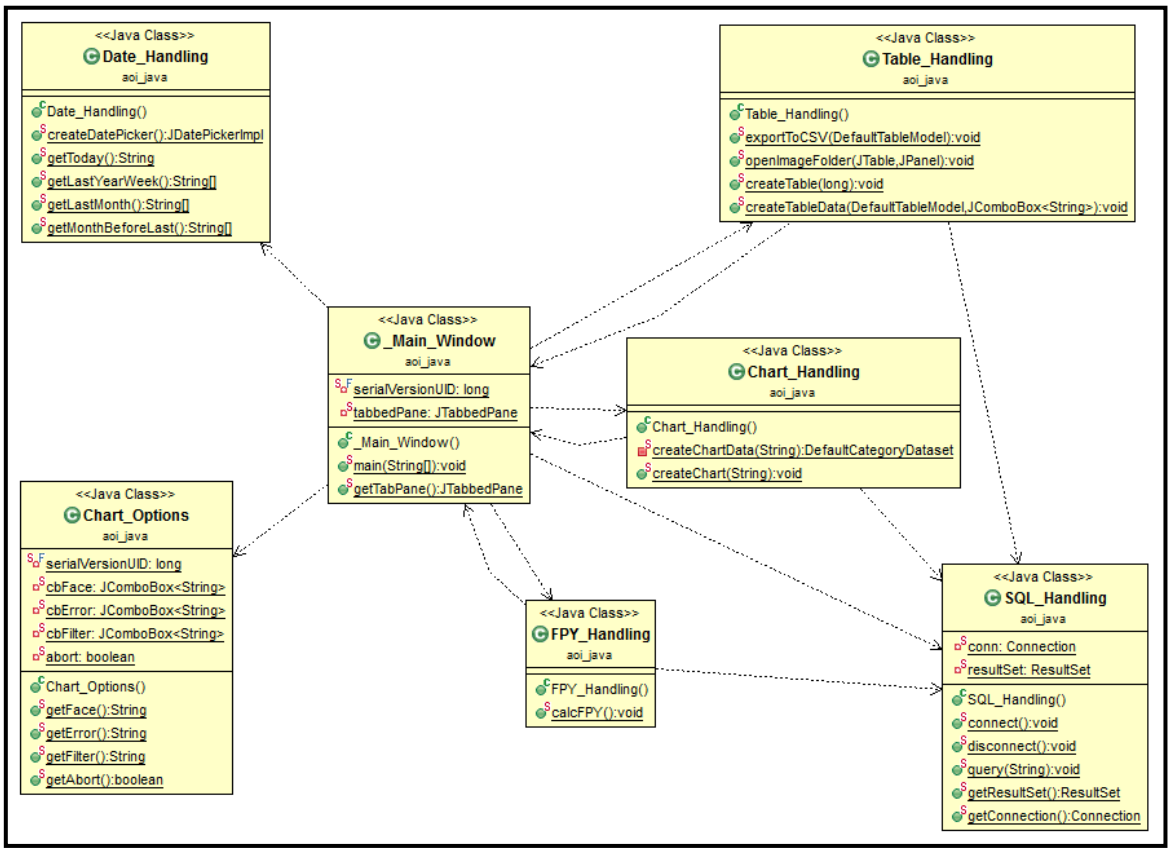


Abbildung 6-4: Klassendiagramme inkl. Abhängigkeiten

Die nachfolgenden Unterkapitel beschreiben diese Klassen und deren Inhalte näher.

Für eine noch detailliertere Einsicht der Klassen und Objekte ist auch der komplette Quellcode aller Klassen inklusive ausführlicher Kommentierung auf der beigefügten CD abgelegt.

_Main_Window.java

Die Klasse _Main_Window enthält, wie der Name schon verrät, die grafische Benutzeroberfläche (GUI, Graphical User Interface), über der die gesamten Unterfunktionen mittels Schaltflächen, Eingabefeldern, usw. abgerufen werden können. Im Anhang der Diplomarbeit sind einige Screenshots dieser GUI eingefügt.

Der grundlegende Aufbau des Fensters erfolgt über Tabs, wobei das Fenster immer mit dem Home-Tab startet. Dieser befindet sich immer an der ersten Tab-Position und enthält die bereits erwähnten Komponenten zur Ausführung der Funktionen für die Fehlerauswertung. Beim Aufruf einer Funktion wird das Ergebnis (z.B. die Tabelle, das Diagramm, etc.) in einem neuen Tab angezeigt. So können auch mehrere Datenbankabfragen in einem Fenster durchgeführt werden und zwischen deren Ergebnissen kann auch relativ einfach über die Tab-Schaltflächen hin- und hergewechselt werden.

Im Home-Tab ist es möglich zwischen dem Mehrfach- und Einzelselektions-Modus auszuwählen. Alle Funktionen, die im aktuell selektierten Modus nicht zur Verfügung stehen, werden automatisch deaktiviert, so wird beispielsweise die Schaltfläche 'Tabelle' und das Textfeld für den Einzelauftrag im Mehrfachselektionsmodus ausgegraut.

Im Mehrfachselektionsmodus kann der Benutzer einen Auswertungs-Zeitraum auswählen und diverse Filter für die FPY-Berechnung und Diagramm-Anzeige setzen.

Per Einzelselektionsmodus kann ein einzelner Fertigungsauftrag in ein Textfeld eingegeben werden und über die Schaltfläche Tabelle werden alle Details des Auftrags angezeigt. Außerdem kann auch hier die FPY- und Diagramm-Funktion genutzt werden.

Die komplette Erstellung der GUI wird im bereits in Kapitel 4 (Java) beschriebenen Konstruktor der Klasse durchgeführt. Beim Aufruf dieses Konstruktors wird ein neues Objekt der Klasse _Main_Window erstellt und somit wird das Fenster generiert.

Um dieses Fenster ausführbar zu machen, muss der Klasse _Main_Window eine sogenannte 'main-Methode' hinzugefügt werden. Somit weiß der Compiler, dass es sich um eine ausführbare Java-Klasse handelt. In dieser main-Methode wird der eben genannte Konstruktor aufgerufen. Außerdem sind hier auch Zusatzoptionen wie zum Beispiel Fenstergröße, Fensterposition, Anpassung der GUI an die Systemoptik usw. integriert.

Chart_Handling.java

In der Chart_Handling-Klasse befinden sich alle Funktionen, die für das Aufbereiten der Diagramm-Daten und Zeichen der Diagramme zuständig sind.

Die Daten für das zu erstellende Diagramm werden über die Methode 'createChartData' aufbereitet. Hier werden die Ergebniszeilen aus der SQL-Abfrage nacheinander abgerufen und zum Datensatz hinzugefügt.

Über die Methode 'createChart' wird das Säulendiagramm nach Abfrage der Daten aus createChartData erstellt und gezeichnet. Außerdem ist hier zusätzlich die Schaltfläche zum Speichern des Diagramms als Bilddatei und auch deren Funktion integriert. Am Ende der Methode wird ein neuer Tab im Hauptfenster (_Main_Window) erstellt, in welchem das gezeichnete Diagramm eingefügt und somit für den Benutzer sichtbar gemacht wird.

Chart_Options.java

Die Klasse Chart_Options ist im Grunde genommen nur ein Dialogfeld, welches dem Benutzer ermöglicht, die Filtereinstellungen für ein zu zeichnendes Diagramm anzupassen. Beim Aufruf der Diagrammschaltfläche im Hauptfenster wird der Konstruktor von Chart_Options ausgeführt. Dieser öffnet ein Dialogfeld, welches diese Optionen zur Filterung (Fehlerart, Seite, Fehlertyp) beinhaltet. Nach erfolgreicher Eingabe wird die SQL-Abfrage direkt an die ausgewählten Optionen angepasst und somit werden nur die relevanten Daten abgefragt und an die Klasse Chart_Handling weitergegeben.

Date_Handling.java

Über die Klasse Date_Handling werden alle Funktionen, die mit Datumsangaben in Verbindung stehen, abgehandelt. In der Methode 'createDatePicker' wird der JDatePicker initialisiert und es erfolgt die Einstellung des Datumsformats. In der Fehlerauswertung wurde das Format yyyy-MM-dd, also Jahr-Monat-Tag, gewählt.

Außerdem wurden hier noch Methoden für die Schnellauswahl eines Zeitraums implementiert. Die Namen und Funktionen dieser Methoden und jeweils ein Beispiel des generierten Start- bzw. Enddatums sind in der nachfolgenden Tabelle ersichtlich.

Name der Methode	**Kurzbeschreibung der Funktion**	**Startdatum**	**Enddatum**
getToday()	heutiges Datum	2017-07-31	2017-07-31
getLastYearWeek()	letzte Kalenderwoche	2017-07-24	2017-07-30
getLastMonth()	letzter Monat	2017-06-01	2017-06-30
getMonthBeforeLast()	vorletzter Monat	2017-05-01	2017-05-31

Tabelle 6-1: Datums-Schnellauswahl-Methoden

FPY_Handling.java

Die Klasse FPY_Handling enthält nur die Methode 'calcFPY', in welcher die gesamte Berechnung und auch die Erstellung der Ausgabetabelle stattfinden. Zur Berechnung eines First Pass Yields wird die Anzahl der produzierten und fehlerhaften Leiterplatten (mit und ohne Pseudofehler) benötigt. Durch diese drei Werte kann der FPY vor der Reparatur (inkl. Pseudofehler) und der FPY nach der Reparatur (Pseudofehler herausgefiltert) errechnet werden.

Alle oben genannten Werte werden in eine Tabelle gespeichert. Als letzte Zeile der Tabelle wird schlussendlich noch die Summe der Werte bzw. bei den FPY-Angaben der Durchschnitt hinzugefügt.

Diese Tabelle wird, ähnlich wie bei der Klasse Chart_Handling, in einem neuen Tab im Hauptfenster ausgegeben und somit für den Benutzer sichtbar gemacht.

SQL_Handling.java

In der SQL_Handling-Klasse sind alle Funktionen, die mit der Verbindung und Kommunikation der Datenbank zu tun haben, implementiert.
Die Methode 'connect' stellt eine Verbindung zur Datenbank über den JDBC-Treiber und den übergebenen Parametern her. Diese Parameter sind Servername, Datenbankname und Authentifizierungsmodus.
Die Methode 'disconnect' dient zum Trennen der Datenbankverbindung. Diese wird, sofern eine Verbindung besteht, automatisch beim Beenden des Programms aufgerufen, um unnötige Datenbankverbindungen zu vermeiden.
Zum Durchführen von SQL-Abfragen wird die Methode 'query' aufgerufen. Der SQL-Befehl wird der Methode als String übergeben und dieser wird über mit Hilfe der Funktion 'executeQuery' an die Datenbank weiterübermittelt. Das Ergebnis dieses Funktionsaufrufs wird in einem sogenannten ResultSet (Ergebnissatz) zurückgegeben und dieser kann anschließend weiterverarbeitet werden (z.B. Speicherung in Tabellen, etc.).

Table_Handling.java

Die Klasse Table_Handling ist für die Erstellung der Detailtabelle erforderlich. Diese Funktion steht, wie bereits in den Anforderungen beschrieben, nur im Einzelselektionsmodus zur Verfügung.

Um die Daten aus der SQL-Abfrage in eine geeignete Tabellenform zu bringen, wurde die Methode 'createTableData' eingefügt. Diese arbeitet jede Zeile des Ergebnissatzes ab und fügt diese zur Tabelle hinzu.

Nach der erfolgreichen Aufbereitung der Daten kommt die Methode 'createTable' zum Einsatz, welche zur grafischen Erstellung der Tabelle dient und diese wiederum in einem neuen Tab im Hauptfenster darstellt.
Außerdem enthält diese Methode Grundfunktionen, wie das Filtern und Sortieren der Tabelle und auch die Schaltflächen zum Exportieren als CSV-Datei und Öffnen der Fehlerbilder.
Die Filterung erfolgt über ein separates Textfeld und eine Auswahlliste. Zunächst kann die zu filternde Spalte über die Auswahlbox selektiert werden und anschließend ist es möglich über das Textfeld zu filtern.
Die Sortierung erfolgt, wie in jeder üblichen sortierbaren Tabelle, über einen Klick auf den gewünschten Spaltennamen, wobei bei jedem Klick die Sortierreihenfolge umgekehrt wird.

Das Exportieren und die Fehlerbild-Anzeige wurden in eigenen Methoden implementiert. Ersteres wird über die Unterfunktion 'exportToCSV' durchgeführt, wobei sich nach einem Klick auf die Schaltfläche ein Dialog öffnet, welcher dem Benutzer den Speicherort und Namen der zu speichernden Datei auswählen lässt.
Die zweite Methode 'openImageFolder' fragt die aktuell ausgewählte Zeile der Tabelle ab und zeigt, wenn vorhanden, Fehlerbilder an, welche auf einem separaten Rechner im Netzwerk gespeichert sind. Voraussetzung für die Anzeige der Fehlerbilder ist die Installation des AOI-Hersteller-Programms 'OIS-Viewer' auf dem lokalen Rechner.

7 Zusammenfassung

Ziel dieser Diplomarbeit war die Erstellung einer Software zur Auswertung von AOI-Fehlerdaten. Da diese Daten auf einem Microsoft SQL Server liegen, musste als erstes die Datenbankstruktur analysiert werden. Hier wurden zunächst die wichtigsten Tabellen aus der Datenbank und deren Beziehungen zueinander herausgefiltert.

Im nächsten Schritt wurden die Anforderungen an die Software aufgenommen, präzisiert und in Funktionsbausteine unterteilt. Des Weiteren wurde die Programmiersprache Java nach Abwägung der Vor- und Nachteile zur Erstellung der Auswertungssoftware gewählt.

Der Abruf der Daten aus der Datenbank erfolgt über die Abfragesprache SQL, wobei hier zu beachten ist, dass ein sogenannter SQL-Dialekt, nämlich Microsoft SQL, zur Kommunikation mit dem Datenbankserver eingesetzt wird.

Der erste Schritt im Feinentwurf war die Entwicklung der einzelnen SQL-Abfragen für die Hauptfunktionen. Hier wurde mit dem Tool Microsoft SQL Server Management Tool gearbeitet, welches außer der Durchführung von SQL-Abfragen auch die Verwaltung der Datenbank bzw. des Datenbankservers unterstützt.

Anschließend wurde nach Zusatzbibliotheken für Java gesucht, welche die benötigten Funktionen wie z. B. das Zeichnen von Diagrammen oder die Auswahl eines Datums mittels grafischer Hilfsmittel bereitstellen. Nach etwas Recherche wurden hier einige hilfreiche und auch kostenlose Bibliotheken im Internet gefunden, wobei in den meisten Fällen auch Anleitungen zur Implementierung zur Verfügung standen.

Der letzte und mit Abstand aufwändigste Teil der Diplomarbeit, war die Erstellung des Java-Programms. Hier wurden die Anforderungen im Grunde genommen in einzelne Klassen und Methoden unterteilt. Die gesamten Funktionen sind über eine relativ schlichte und selbsterklärende Benutzeroberfläche aufrufbar.

Diese Benutzeroberfläche ist anhand von Screenshots im Anhang ersichtlich. Da der Java-Quelltext durch die komplexen SQL-Abfragen sehr umfangreich wurde, ist dieser nicht in gedruckter Form im Anhang enthalten, sondern befindet sich auf der beigelegten CD.

7.1 Ergebnisse

Da sich die Auswertungssoftware noch einem frühen Entwicklungsstadium befindet, wurde diese bisher nur an einige ausgewählte Benutzer zum Testen weitergegeben. Die ersten Rückmeldungen dieser Benutzer sind aber durchaus positiv. Nach bisherigen Erkenntnissen sind zwei große Vorteile im Vergleich zur Abfrage über das bisher verwendete Webinterface entstanden.

Zum einen ist der Geschwindigkeitsvorteil durch das Arbeiten mit der Eigenlösung enorm. Dadurch, dass Auswertungen jetzt komplett mit einer Software durchgeführt werden können, wird nicht nur die Produktivität der Mitarbeiter erhöht, sondern es kann bei Produktionsproblemen auch schneller und somit effektiver reagiert und werden. Dies bringt in weiterer Folge natürlich auch eine Qualitätssteigerung des gesamten Produktionsprozesses mit sich.

Der zweite große Vorteil besteht im Multi-User-Betrieb, das heißt, mehrere Benutzer sind in der Lage zeitgleich auf die AOI-Datenbank zuzugreifen und Fehlerdaten zu analysieren bzw. Auswertungen zu erstellen. Damit blockieren sich die Benutzer nicht mehr gegenseitig und die Mitarbeiter-Produktivität und Prozess- bzw. Produkt-Qualität wird wiederum gesteigert.

7.2 Ausblick

Prinzipiell wurden alle erfassten Anforderungen in der Fehlerdatenauswertungs-Software implementiert. Selbstverständlich gibt es aber auch Wünsche für Verbesserungen und Optimierungen von den Benutzern.

Der erste Optimierungsschritt wird die Integration zur richtigen Auswertung der Sommerzeit betreffen. Hier muss erst analysiert und recherchiert werden, ob es komfortabler ist, diese Zeitabweichung im Java-Programm zu berechnen oder ob die Abweichung bereits direkt über den Microsoft SQL Server eruiert werden kann.

Die nächste Verbesserung soll weitere Anpassungsmöglichkeiten im Bereich der Diagrammanzeige in Angriff nehmen. Da die Diagramme derzeit nur mit bestimmten Benutzereingaben berechnet und gezeichnet werden, sind diese im Moment sehr statisch und können nicht nachträglich angepasst bzw. verändert werden. Hier wäre es weitaus komfortabler Diagramme auch im Nachhinein anpassen zu können, um nicht immer neue Diagramme generieren zu müssen.

Einer der größten offenen Punkte, die bereits während der Programmentwicklung aufgekommen sind, ist die Überführung der gesamten Auswertungssoftware ins Englische. Somit können auch die anderssprachigen Tochterfirmen, welche die gleichen AOI-Systeme besitzen, auf diese Software zur Datenanalyse zugreifen, um die Prozesse und die Qualität zu optimieren.

Literaturverzeichnis

Ebert, Christof: Systematisches Requirements Engineering: Anforderungen ermitteln, dokumentieren, analysieren und verwalten.
Heidelberg: dpunkt.verlag GmbH, 5. Auflage, überarb. Auflage, 2014

Habelitz, Hans-Peter: Programmieren lernen mit Java: Aktuell zu Java 8 und mit dem WindowBuilder.
Bonn: Rheinwerk Computing, 4. Auflage, 2016

Laube, Michael: Einstieg in SQL.
Bonn: Rheinwerk Computing, 2017

Lorig, Daniel: Java-Programmierung für Anfänger: Programmieren lernen ohne Vorkenntnisse.
North Charleston, South Carolina, USA: CreateSpace Independent Publishing Platform, 2015

Schicker, Edwin: Datenbanken und SQL.
Wiesbaden: Springer Vieweg, 4. Auflage, 2014.

Berg, Dietmar: Lohnt sich ein AOI-System?
Verfügbar unter *http://www.plastverarbeiter.de/ai/resources/a4300bce5b4.pdf*
Letzter Zugriff am 02.08.2017

Eclipse Documentation. Previous Release.
Verfügbar unter *http://help.eclipse.org/mars/index.jsp*
Letzter Zugriff am 16.07.2017

Gabler Wirtschaftslexikon. Stichwort: Datenbank.
Verfügbar unter *http://wirtschaftslexikon.gabler.de/Archiv/55473/datenbank-v12.html*
Letzter Zugriff am 18.08.2017

Gabler Wirtschaftslexikon. Stichwort: SQL.
Verfügbar unter *http://wirtschaftslexikon.gabler.de/Archiv/74908/sql-v11.html*
Letzter Zugriff am 18.07.2017.

java2s.com. BarChart.
Verfügbar unter *http://www.java2s.com/Code/Java/Chart/Bar-Chart.htm*
Letzter Zugriff am 26.06.2017

JDatePicker. Java Swing Date Picker.
Verfügbar unter *https://jdatepicker.org/*
Letzter Zugriff am 05.07.2017

JFree Forum. JFreeChart, JCommon.
Verfügbar unter *http://www.jfree.org/forum/*
Letzter Zugriff am 22.06.2017

Microsoft Developer Network. Datentypen (Transact-SQL).
Verfügbar unter *https://msdn.microsoft.com/de-de/library/ms187752(v=sql.120).aspx*
Letzter Zugriff am 12.07.2017

Microsoft Docs. Auswählen eines Authentifizierungsmodus.
Verfügbar unter *https://docs.microsoft.com/de-de/sql/relational-databases/security/choose-an-authentication-mode*
Letzter Zugriff am 10.07.2017

Tutorials Point. JFreeChart.
Verfügbar unter *http://www.tutorialspoint.com/jfreechart/*
Letzter Zugriff am 22.06.2017

Tutorials Pont. JDBC.
Verfügbar unter *https://www.tutorialspoint.com/jdbc/*
Letzter Zugriff am 17.06.2017

Unix Time Stamp. Epoch Converter.
Verfügbar unter *https://www.unixtimestamp.com/*
Letzter Zugriff am 02.07.2017

W3Schools. SQL Tutorial.
Verfügbar unter *https://www.w3schools.com/sql/*
Letzter Zugriff am 14.06.2017

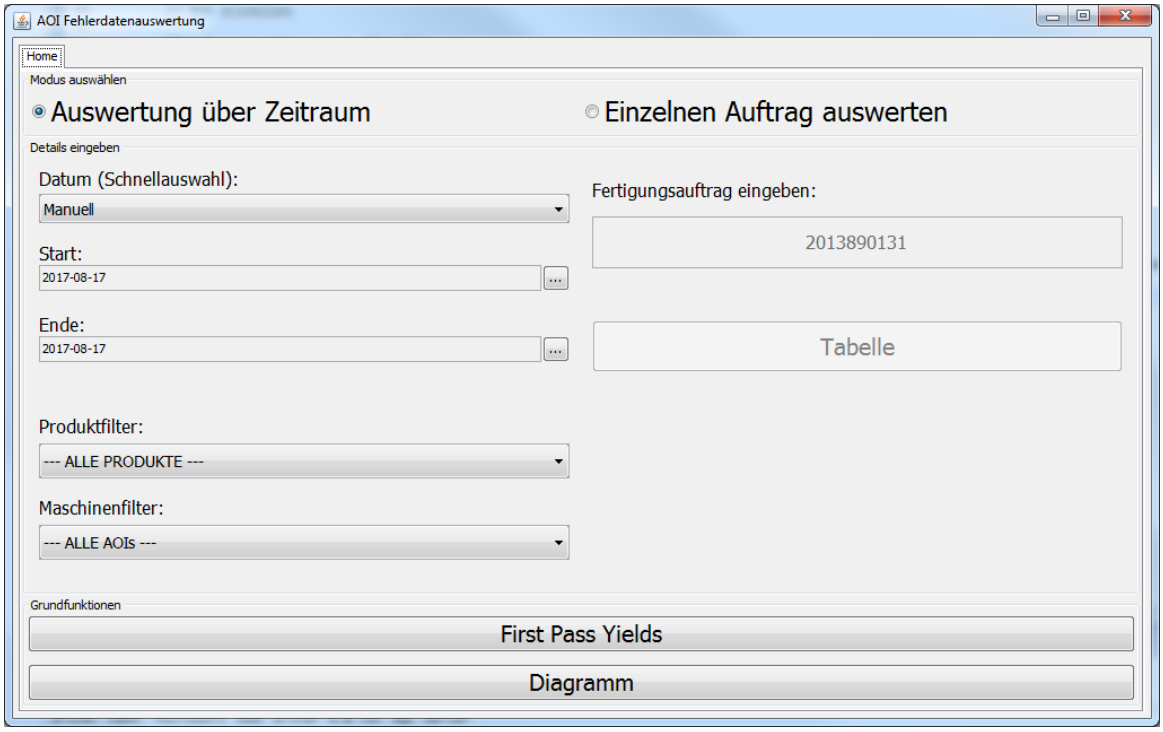
Anhang

a) Grafische Benutzeroberfläche (GUI)

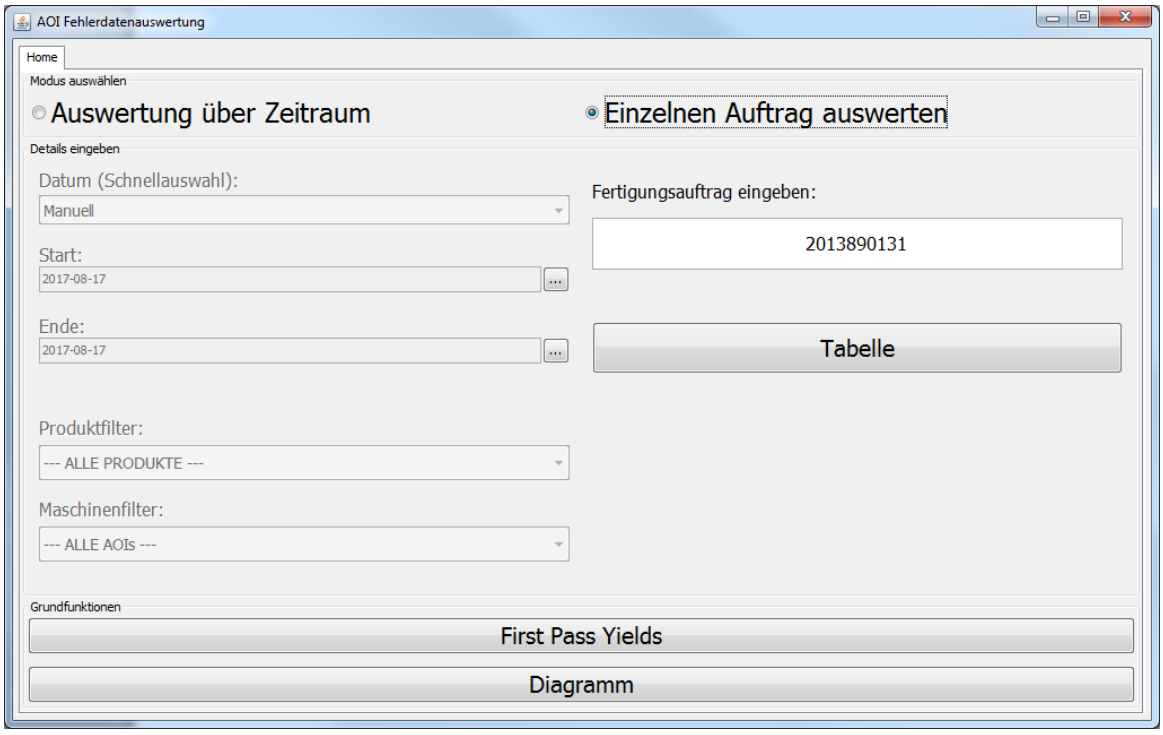
b) Inhalt der CD-ROM

- Ordner *'dokumentation'*
  - Datei *'diplomarbeit_strohmaier_2017.pdf'*
    Digitale Fassung der Diplomarbeit „Fehlerdatenauswertung in Java“
  - Datei *'quelltext_strohmaier.pdf'*
    Gesamter Quelltext der erstellten Software inkl. Kommentierung

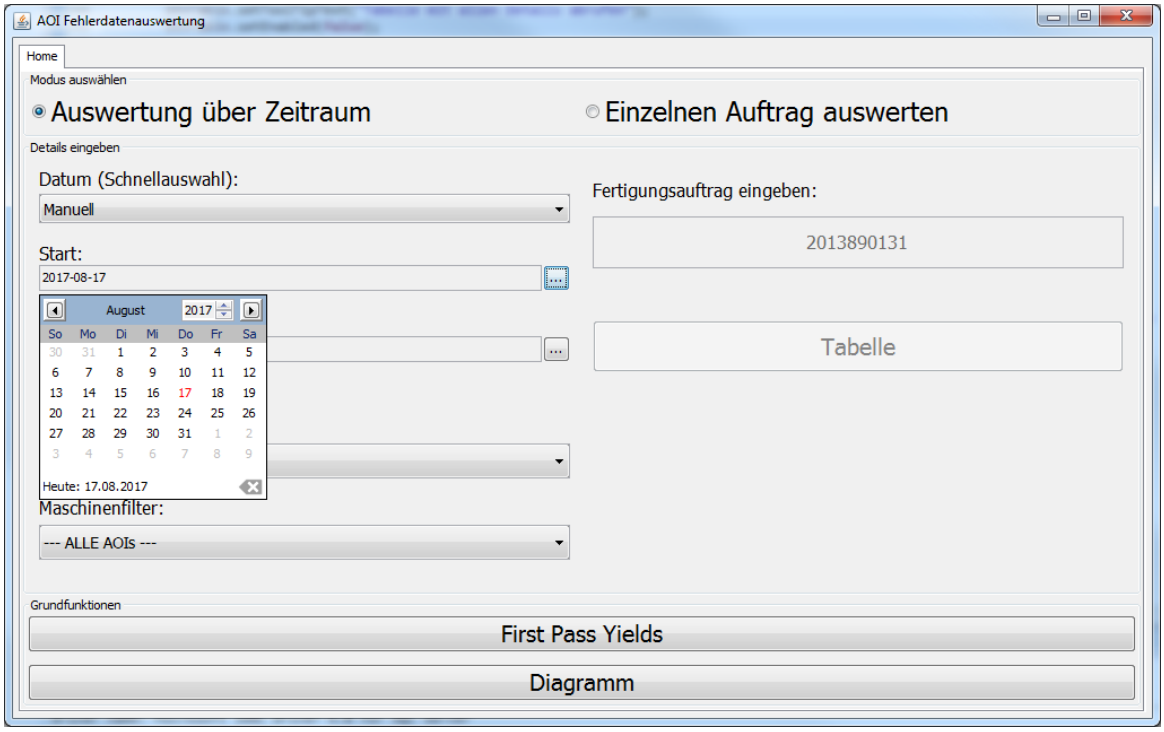
- Ordner *'programm'*
  - Datei *'fehlerdatenauswertung.jar'*
    Ausführbare Programmdatei zum Starten der Auswertungssoftware
  - Datei *'sqljdbc_auth.dll'*
    Bibliotheksdatei zur Windows Authentifizierung mit dem Server
  - Ordner *'aoi_java'*
    Projektordner von Eclipse mit allen Klassen und Bibliotheken



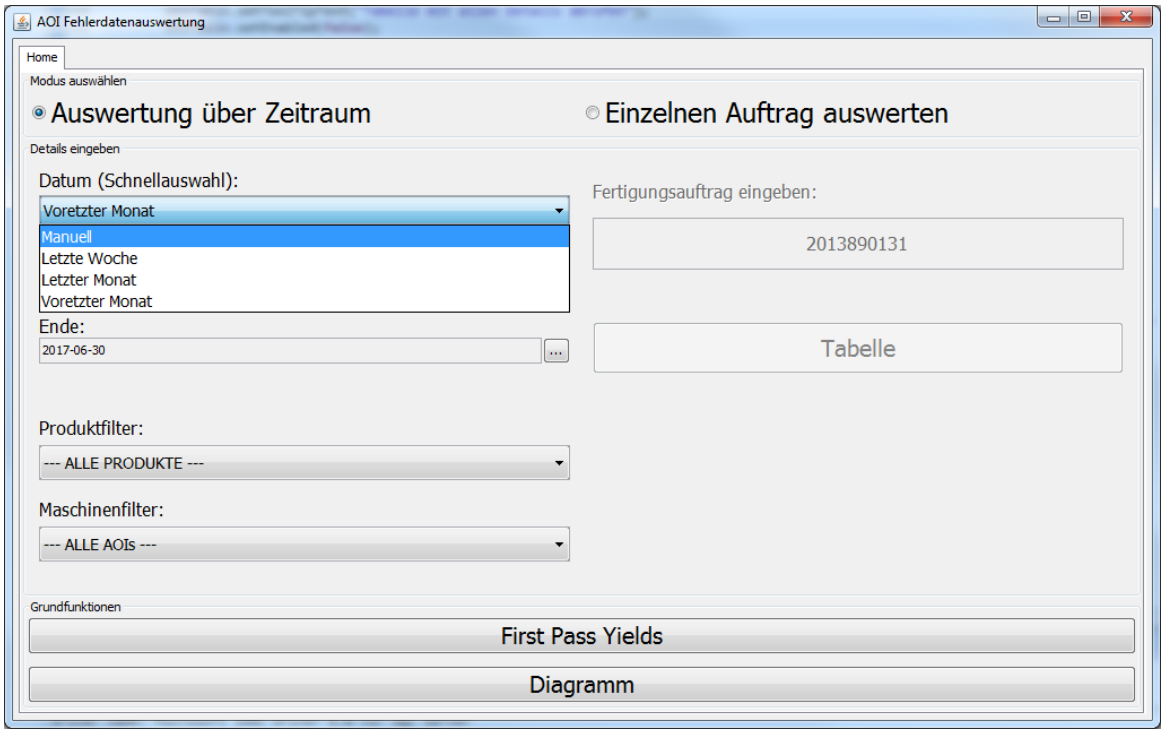
Screenshot 1: Hauptfenster (Home) – Mehrfachselektions-Modus



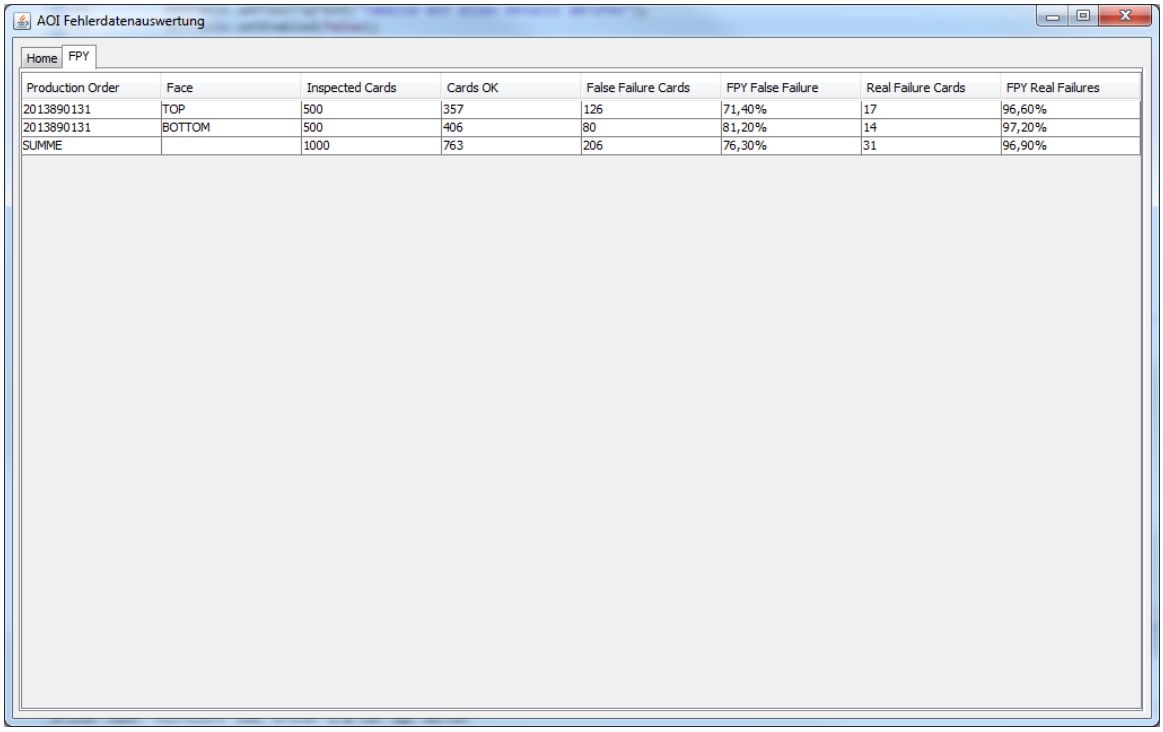
Screenshot 2: Hauptfenster (Home) – Einzelselektions-Modus



Screenshot 3: Manuelle Datumsselektion über JDatePicker

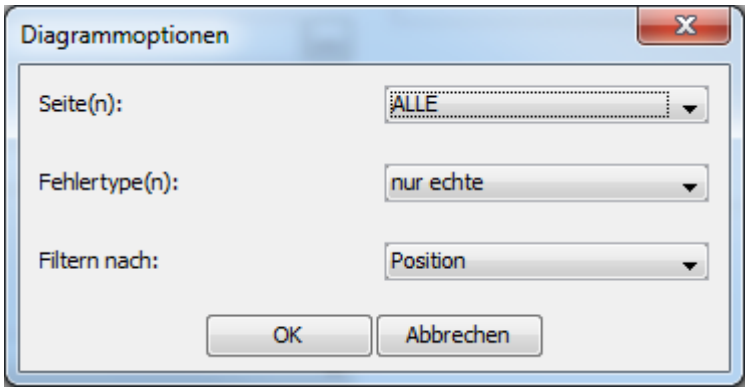


Screenshot 4: Schnellauswahl eines Zeitraums

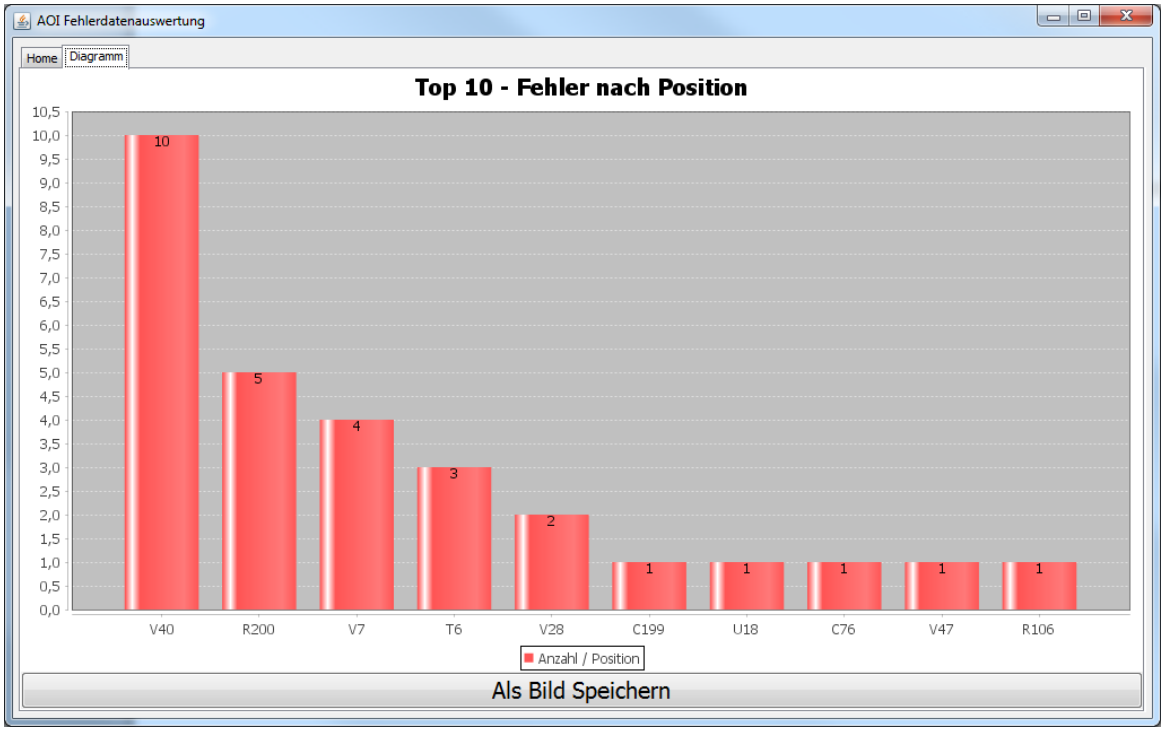


Production Order	Face	Inspected Cards	Cards OK	False Failure Cards	FPY False Failure	Real Failure Cards	FPY Real Failures
2013890131	TOP	500	357	126	71,40%	17	96,60%
2013890131	BOTTOM	500	406	80	81,20%	14	97,20%
SUMME		1000	763	206	76,30%	31	96,90%

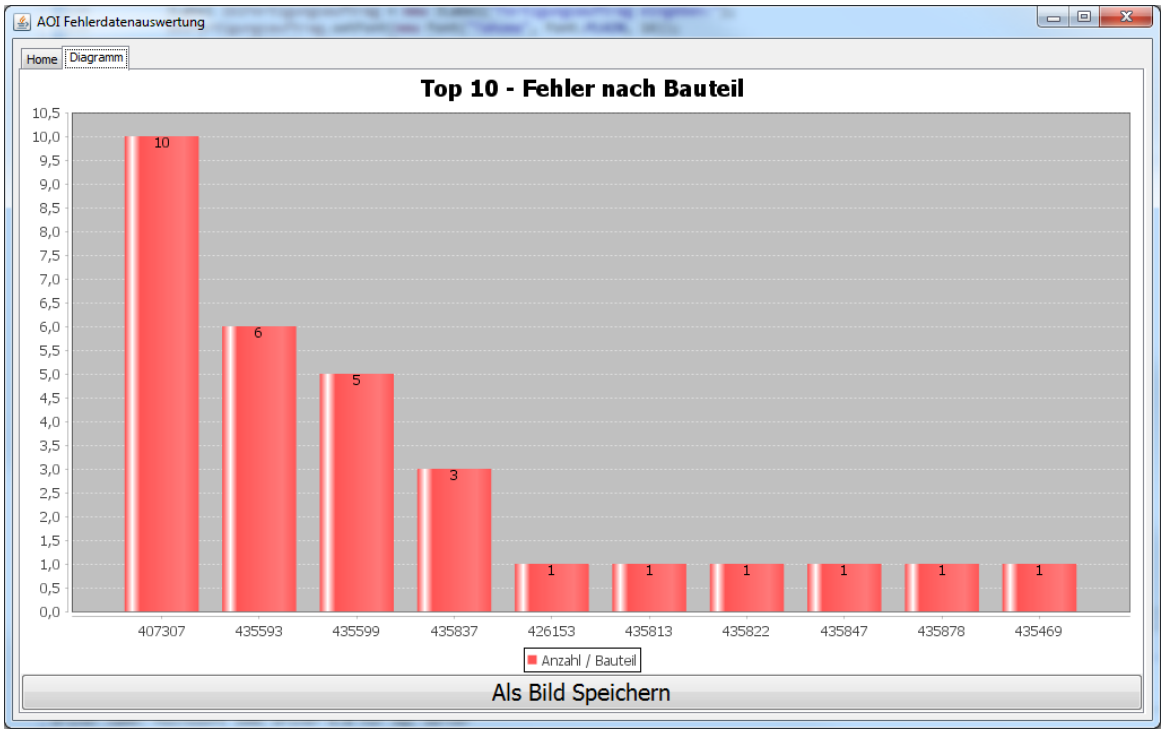
Screenshot 5: FPY Ergebnistabelle



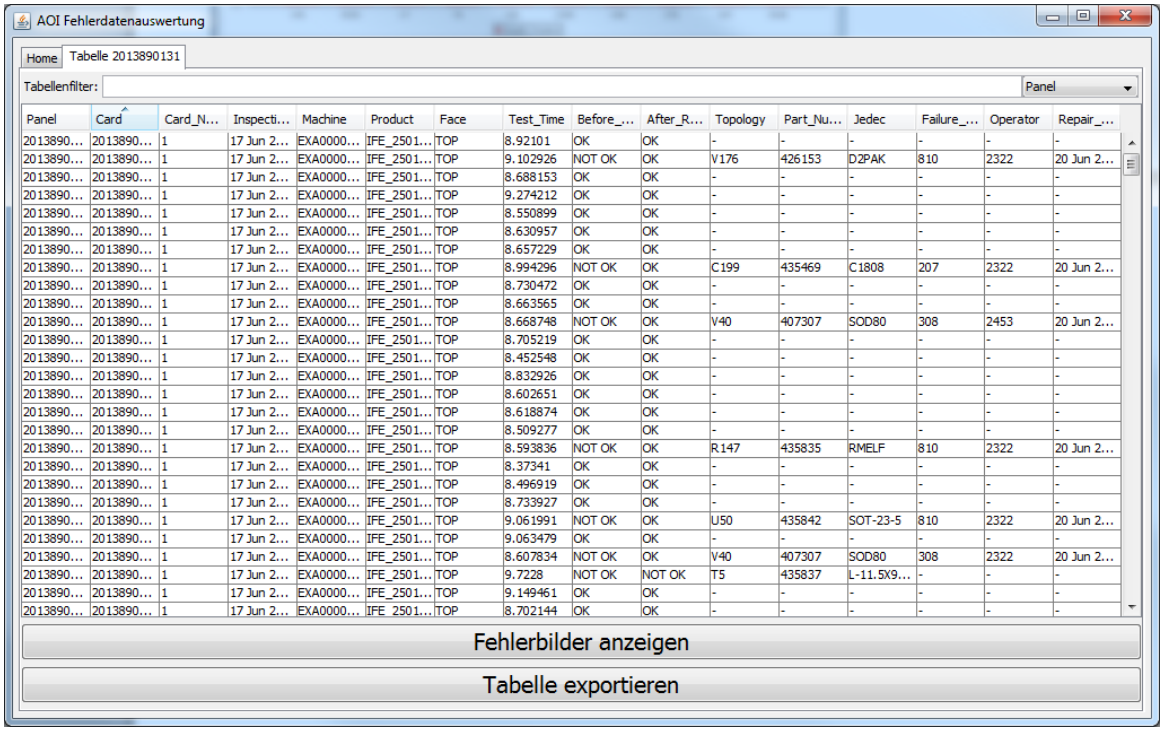
Screenshot 6: Dialogfeld für Diagrammfilter



Screenshot 7: Säulendiagramm (mit Positionsfilter)

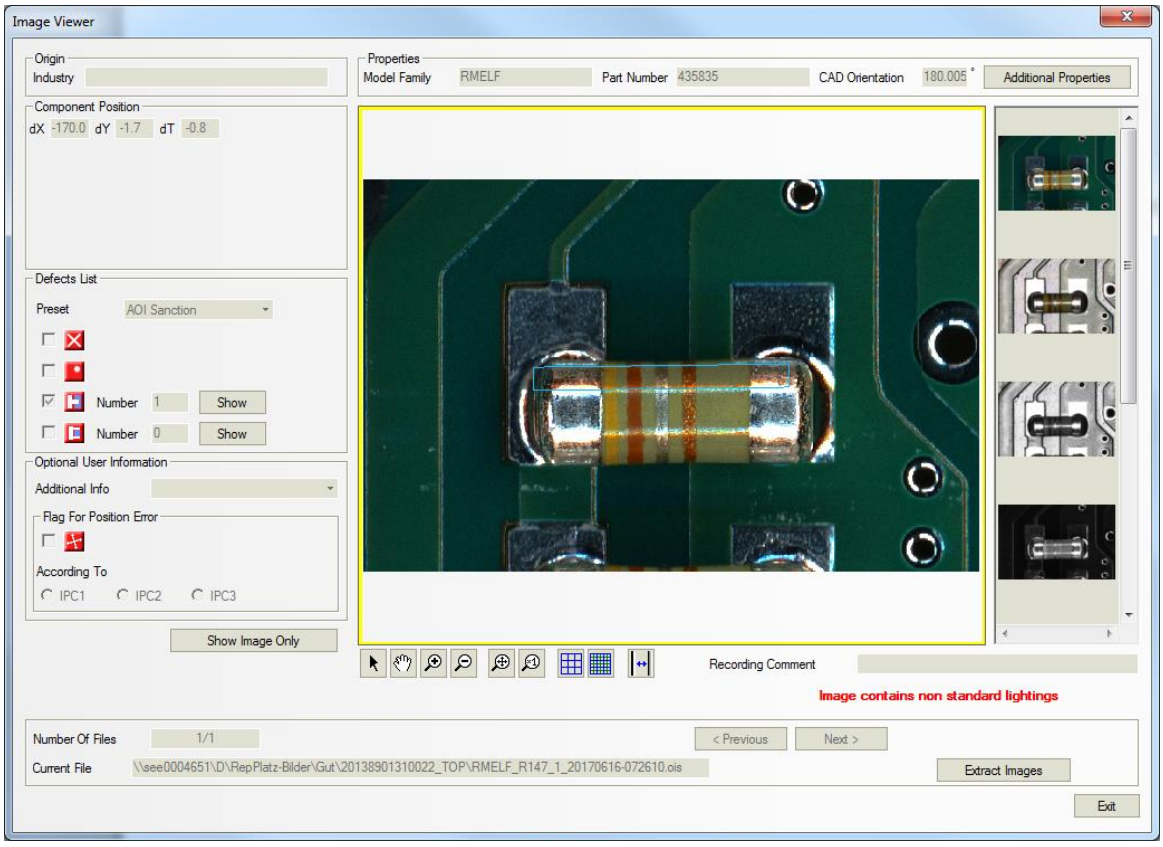


Screenshot 8: Säulendiagramm (mit Bauelementfilter)



Panel	Card	Card_N...	Inspecti...	Machine	Product	Face	Test_Time	Before_...	After_R...	Topology	Part_Nu...	Jedec	Failure_...	Operator	Repair_...
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.92101	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	9.102926	NOT OK	OK	V176	426153	D2PAK	810	2322	20 Jun 2...
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.688153	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	9.274212	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.550899	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.630957	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.657229	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.994296	NOT OK	OK	C199	435469	C1808	207	2322	20 Jun 2...
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.730472	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.663565	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.668748	NOT OK	OK	V40	407307	SOD80	308	2453	20 Jun 2...
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.705219	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.452548	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.832926	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.602651	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.618874	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.509277	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.593836	NOT OK	OK	R147	435835	RMELF	810	2322	20 Jun 2...
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.37341	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.496919	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.733927	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	9.061991	NOT OK	OK	U50	435842	SOT-23-5	810	2322	20 Jun 2...
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	9.063479	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.607834	NOT OK	OK	V40	407307	SOD80	308	2322	20 Jun 2...
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	9.7228	NOT OK	NOT OK	T5	435837	L-11.5X9...	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	9.149461	OK	OK	-	-	-	-	-	-
2013890...	2013890...	1	17 Jun 2...	EXA0000...	IFE_2501...	TOP	8.702144	OK	OK	-	-	-	-	-	-

Screenshot 9: Detailtabelle im Einzelselektions-Modus



Screenshot 10: Fehlerbild-Anzeige mit OIS-Viewer

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Obergreith, am 20.09.2017 *Markus Strohmaier*