

Fachhochschule Köln

University of Applied Science Cologne

Campus Gummersbach

Fakultät für Informatik und Ingenieurwesen

Fachhochschule Dortmund

Fachbereich Informatik

Verbundstudiengang Wirtschaftsinformatik

Diplomarbeit

Konzeption und Erstellung einer webbasierten Datenbank zur technischen Dokumentation von Prototypen

Erstprüfer:	Prof. Dr. Holger Günther
Zweitprüfer:	Prof. Dr. rer. nat., Dipl.-Inform. Frank Viktor
vorgelegt von cand.	André Grahl Hellerstraße 56 44229 Dortmund
vorgelegt am	02. Januar 2008
eMail:	Studium@Andre-Grahl.de
Matr.-Nr.:	704 2991

Inhaltsverzeichnis

Inhaltsverzeichnis.....	3
1 Einleitung	6
1.1 Thematik der Diplomarbeit	6
1.2 Zielsetzung der Arbeit	8
1.3 Vorgehensweise und Gliederung.....	10
2 Das Vorgehensmodell	14
3 Teilsystem 1: Das Sicherheitskonzept	18
3.1 Authentifizierung für die Webanwendung	18
3.1.1 Die Benutzer – Authentifizierung für die Webanwendung	18
3.1.1.1 Windows – Standardauthentifizierung	18
3.1.1.2 Windows – Digestauthentifizierung	19
3.1.1.3 Integrierte Windows – Authentifizierung	19
3.1.1.4 Abgrenzung der Authentifizierungsmethoden	20
3.1.2 Auswahl der Authentifizierungsmethode für die Anwendung.....	21
3.2 Authentifizierung für die Datenbank.....	22
3.2.1 Windows - Authentifizierungsmethode für den Datenbankserver	22
3.2.2 Gemischter Modus als Authentifizierungsmethode	23
3.2.3 Auswahl der Authentifizierungsmethode für den SQL-Server	24
3.3 Umsetzung der Authentifizierungen.....	25
3.3.1 Umsetzung der Authentifizierungen für den Datenbankzugriff	25
3.3.2 Umsetzung der Authentifizierungen für die Webanwendung.....	26
3.4 Datensicherheit innerhalb der Anwendung	27
3.4.1 SQL-Injection.....	27
3.4.2 Direkte Validierung der Benutzereingaben.....	28
3.4.2.1 Überprüfung anhand eines regulären Ausdrucks	28
3.4.2.2 Überprüfung auf Eingabe eines Pflichtfeldes	29
3.4.2.3 Überprüfung eines korrekten Datums	30
3.4.2.4 benutzerdefinierte Validierung	31
3.4.2.5 Vorteile der direkten Validierung	33
3.4.3 Datenintegrität innerhalb der Datenbank	33
3.5 Fazit des Sicherheitskonzeptes.....	34
4 Teilsystem 2: Datenerfassung und -pflege	35
4.1 Die Datenerfassung.....	36
4.1.1 Projektdaten erfassen.....	36
4.1.2 Musterstandsdaten erfassen.....	37

4.1.3	Prototypendaten erfassen.....	39
4.1.3.1	Einen neuen Prototypen erfassen.....	39
4.1.3.2	Daten eines existierenden Prototypen kopieren.....	41
4.1.4	Fazit der Dateneingabe	43
4.2	Die Datenpflege	44
4.2.1	Navigation innerhalb der Daten	44
4.2.2	Pflege der Projektdaten	50
4.2.2.1	Pflege der Projekt-Stammdaten	51
4.2.2.2	Pflege der Projekt-Berechtigungen	53
4.2.2.3	Hinzufügen von weiteren Aufträgen	53
4.2.2.4	Hinzufügen von Projekt-Informationen	54
4.2.2.5	Löschen von Daten innerhalb eines Projektes	56
4.2.3	Musterstände bearbeiten	56
4.2.3.1	Hinzufügen von weiteren Musterständen	56
4.2.3.2	Bearbeiten von Musterstandsdaten	57
4.2.3.3	Freigabe eines Musterstandes.....	58
4.2.4	Prototypen bearbeiten	60
4.2.4.1	Pflege der Prototypendaten	60
4.2.4.1.1	Prototypen-Stammdatenpflege.....	60
4.2.4.1.2	Prototypen-Kommentare erfassen	62
4.2.4.1.2.1	Prototypen-Prüfungen erfassen.....	63
4.2.4.1.2.2	Manuelle Erfassung anhand der PTDB	63
4.2.4.1.3	Automatische Erfassung via Interface	64
4.2.4.2	Freigabe eines Prototypen.....	65
5	Teilsystem 3: Suchfunktionen der PTDB	68
5.1	Suchen nach Projekten	68
5.1.1	Optimierung der Projektsuche	71
5.2	Suchen nach Prototypen	73
6	Teilsystem 4: Reportingfunktionen zur Qualitätssicherung	77
6.1	Allgemeine Reporte.....	79
6.1.1	Allgemeine Reporte auf Projektebene	80
6.1.2	Allgemeine Reporte auf Musterstandsebene	82
6.1.3	Allgemeine Reporte auf Prototypebene	83
6.2	Interne Reporte	84
6.3	Externe Reporte	84
6.4	Sonderreport als XML	84

6.4.1	Was ist XML	85
6.4.2	Welche Daten enthält die XML-Datei	86
7	Abschluss / Resümee der Arbeit	88
7.1	Zusammenfassung	88
7.2	Fazit	88
7.2.1	Erfolge	89
7.3	Ausblick	90
7.4	Persönliches Schlusswort	91
	Eidesstattliche Erklärung	92
8	Abbildungsverzeichnis	93
9	Tabellenverzeichnis	95
10	Abkürzungsverzeichnis / Glossar	95
11	Literaturverzeichnis	99
12	Anhang	102
12.1	Funktion „MakeDBText“	102
12.2	Anhang: TreeView füllen inklusive Prototypen	103
12.3	Anhang: Section-Handler „connectionStrings“ für den Datenbankzugriff	108
12.3.1	Einstellung für die Production-Environment	109
12.3.2	Einstellung für die Development-Environment	109
12.3.3	Einstellung für die UAT-Environment	109
12.4	Anhang: View zur Projektsuche	110
12.5	Anhang: Beispielberichte	111
12.5.1	Musterstände eines Projektes	111
12.5.2	Musterstände eines Projektes inkl. Prototypen	112
12.5.3	Softwarestände eines Projektes	113
12.5.4	Hardwarestände eines Projektes	114
12.5.5	Konstruktionsstände eines Projektes	115
12.5.6	Prototypen-Übersicht auf Projektebene	116
12.5.7	Hardwareteile eine Musterstandes	117
12.5.8	Software-Funktionen eines Musterstandes	118
12.5.9	Software-Funktionen eines Prototypen	119
12.5.10	interner Prototypen-Lebenslauf	120
12.5.11	externer Prototypen-Lebenslauf	121
12.5.12	XML-Report zu einem Prototyp	122
12.6	internes Auftragsformular	126

1 Einleitung

1.1 *Thematik der Diplomarbeit*

Das Unternehmen Behr Hella Thermocontrol (kurz BHTC) ist ein Joint Venture der Hella KG Hueck & Co. und der Behr GmbH & Co. zu gleichen Teilen.

Die BHTC entwickelt und fertigt Bedien- und Steuergeräte für die Fahrzeugklimatisierung an. Darüber hinaus werden lineare und getaktete Gebläseregler sowie elektrische Zuheizer auf Seiten der BHTC entwickelt und produziert.

Im Bereich der Klimasteuergeräteentwicklung ist es die Regel, dass die Kunden der BHTC zu den einzelnen Stadien der Entwicklung eines Steuergerätes so genannte Prototypen anfordern. Ein Prototyp (griechisch „Das Urbild“) bezeichnet ein Vorab-Exemplar einer späteren Serienfertigung, das zur Erprobung von Eigenschaften dient¹. Im Rahmen der Entwicklung unterscheidet die BHTC hier verschiedene Stadien eines Prototypen, die sich durch die implementierten Softwarefunktionen, die unterschiedlich verbauten Hardwareteile und die unterschiedlichen Designs des Gehäuses unterscheiden.

Seit der Gründung im Jahre 1999 spielt die BHTC GmbH eine wichtige Rolle im Segment der Automobilzulieferer. In vielen Ländern nehmen die Spezialisten in Sachen Bedien- und Steuergeräte für die Fahrzeugklimatisierung inzwischen die Position des Marktführers ein. Maßgeblichen Anteil am Erfolg hat das zügige Engagement in den USA, China und Indien, da die Automobilindustrie als global agierende Branche von ihren Zulieferern erwartet, stets präsent vor Ort zu sein. Bereits im Januar 2003 folgte BHTC dem Beispiel deutscher Kfz-Hersteller und siedelte sich mit einem Entwicklungs- und Vertriebszentrum in Plymouth, Michigan an. In einem Werk in Flora, Illinois wird seitdem für amerikanische Kunden produziert. Von dort aus beliefert BHTC Inc. die gesamte NAFTA-Region.

Die Standorte Shanghai (China) und Pune (Indien) bedienen die dortigen lokalen Märkte sowie den gesamten asiatischen Raum.

¹ [wiki1]

Alle relevanten Daten im Zusammenhang mit den entwickelten Bedien- und Steuergeräten werden bei der BHTC elektronisch erfasst. Zu Beginn der Diplomarbeit setzte die BHTC jedoch eine Insellösung ein, die nur innerhalb ihrer eigenen Grenzen wirksam war und nicht mit ähnlichen oder verwandten Systemen der Umgebung zusammenwirken konnte bzw. kompatibel war. Diese Ursprungslösung der „MuLi DB“ basierte auf Microsoft Access und wurde in der Abteilung „Technisches Prüflabor“ entwickelt, um alle geprüften Prototypen zu dokumentieren.

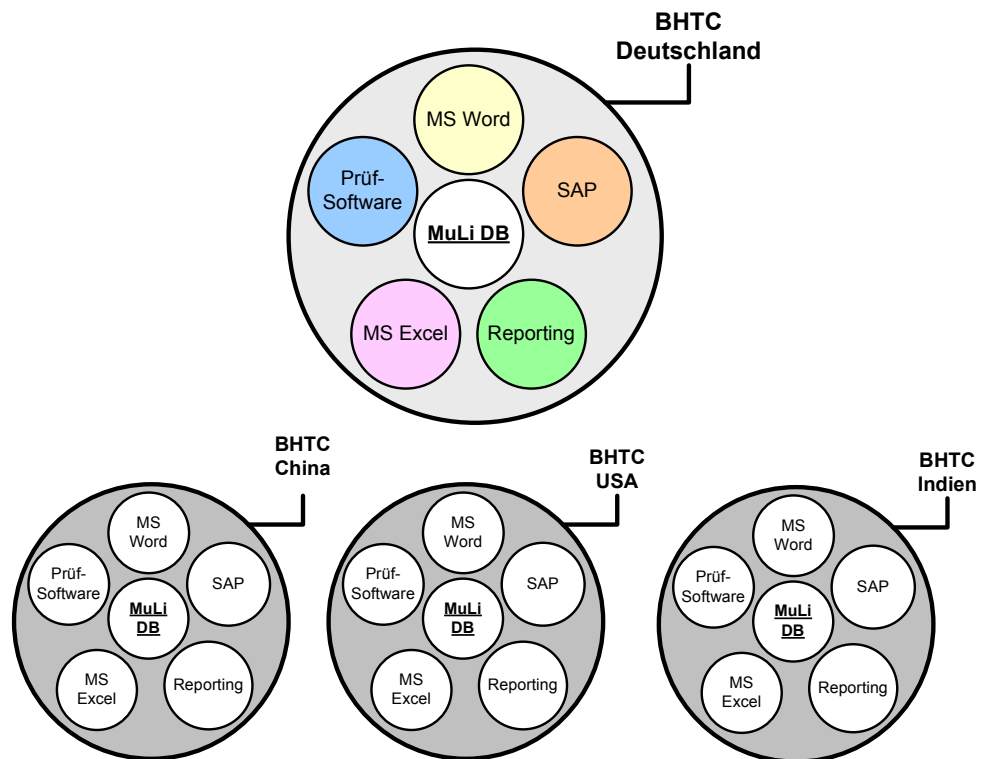


Abbildung 1: Momentane Insellösung "MuLi DB"

Auf Grund dieser bisherigen Lösung wurden viele Daten doppelt erfasst, was zusätzlich zu einem unnötig hohen Arbeitsaufwand, auch eine hohe Fehleranfälligkeit bedeutete. Eine system- und standortübergreifende Qualitätssicherung war auf Grund der bisherigen Systemarchitektur unmöglich und jeder Standort musste eine eigene Qualitätskontrolle implementieren. Dieses führte auf Seiten des Unternehmens zu dem Wunsch, eine einheitliche Datenbank zu entwickeln, die einen optimierten Arbeitsablauf sicherstellen sollte.

1.2 Zielsetzung der Arbeit

Im Rahmen der Diplomarbeit wurde für die BHTC ein Konzept für eine webbasierte Datenbank zur technischen Dokumentation von Prototypen entwickelt und umgesetzt. Die entwickelte Datenbankanwendung mit dem Namen „PTDB“² bildet das Bindeglied zwischen den einzelnen Unternehmensstandorten und den jeweilig vor Ort eingesetzten Systemen.

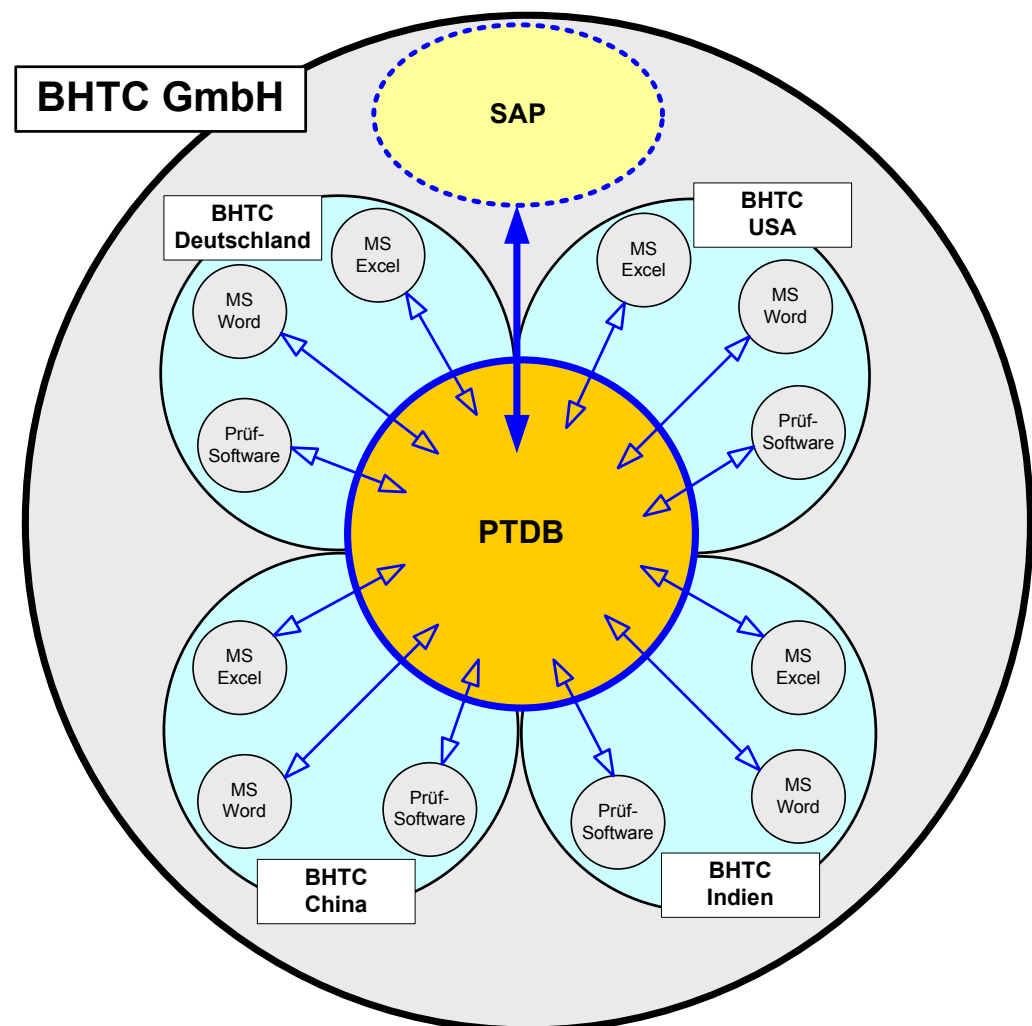


Abbildung 2: Bindeglied PTDB

Erstmalig ist es nun möglich, standortübergreifende Reports zu erstellen, mit deren Hilfe die Qualität der erfassten und entwickelten Prototypen kontrolliert und stetig verbessert werden kann.

² PTDB = Prototypen-Datenbank

Da die Behr-Hella Thermocontrol GmbH als Automobilzulieferer nach DIN EN ISO 9001³, ISO TS 16949⁴ und QS-9000⁵ zertifiziert ist, war es eine maßgebliche Vorgabe für die Entwicklung, dass sich die Anwendung an die bestehenden Prozesse hält und diese weitestgehend unterstützt.

Die DIN EN ISO 9001 beschreibt modellhaft das gesamte Qualitätsmanagementsystem und ist die Basis für ein umfassendes Qualitätsmanagementsystem. Jedes Produkt unterliegt anderen spezifischen Anforderungen und ist demnach nur unter individuellen Qualitätssicherungsmaßnahmen zu erzeugen. Qualitätsmanagementsysteme hingegen sind nicht produktorientiert und können daher unabhängig von der Branche und den spezifischen Produkten einen ähnlichen Aufbau festlegen. Unter dem Begriff QS-9000 ist das Regelwerk der nordamerikanischen Automobilindustrie bezüglich der Qualitätssicherung zusammengefasst, wobei die ISO TS 16949 eine weltweit anerkannte ISO Technische Spezifikation ist und die Anforderungen der internationalen Automobilhersteller zusammen fasst.

Auch die Konzeption der entwickelten Anwendung unterliegt einer Qualitätsanforderung, die sich wie folgt katalogisieren lässt:

- Funktionalität: eine korrekte Realisierung genau der gewünschten Funktionen
- Sicherheit: Verhinderung unberechtigter Zugriffe
- Robustheit: keine „Systemzusammenbrüche“; Aufrechterhaltung der Funktionalität auch bei unplanmäßigen Einwirkungen, z.B. durch Bedienungsfehler
- Ergonomie: leichte Erlernbarkeit und Bedienbarkeit
- Effizienz: sparsamer Umgang mit Betriebsmitteln und/oder Rechenzeit
- Änderbarkeit/Erweiterbarkeit: leichte Behebbarkeit von Fehlern und einfache Anpassung an geänderte Anforderungen
- Wiederverwendbarkeit: einfache Übertragbarkeit des Systems oder seiner Teile in eine andere organisatorische Umgebung oder in ein anderes IT-System

³ [wiki11]

⁴ [wiki12]

⁵ [wiki11]

Es liegt in der Natur dieser Qualitätsmerkmale, dass sie eine unterschiedliche Gewichtung innerhalb der BHTC haben. So besitzt etwa die Wiederverwendbarkeit nur eine geringe Priorität, da das entwickelte Konzept darauf ausgelegt ist unternehmensweit eingesetzt zu werden. Die Funktionalität wiederum hat innerhalb der PTDB eine hohe Priorität, da zu jedem Zeitpunkt sichergestellt sein muss, dass die Anwendung sich an die definierten und zertifizierten Prozesse hält und eine korrekte Realisierung dieser Prozesse gewährleistet ist.

1.3 Vorgehensweise und Gliederung

Um die Anwendung qualitätsorientiert zu entwickeln, wurden im Rahmen der Projektarbeit („Analyse zur Entwicklung einer webbasierten Datenbank zur technischen Dokumentation von Prototypen“) Grobziele erfasst, die dann im weiteren Verlauf, in Zusammenarbeit mit den spezifischen Fachabteilungen der BHTC, genauer definiert und analysiert wurden. Die einzelnen Schritte der detaillierten Feinziele der Projektarbeit (Anforderungen an die Anwendung) wurden anhand von Workflows niedergeschrieben und durch die BHTC abgenommen.

Als Workflow oder auch Arbeitsablauf wird der automatisierte Ablauf von Aktivitäten in einer Organisation verstanden, der durch ein IT-System gesteuert, überwacht und/oder koordiniert wird.⁶

Als weiteres Mittel der Dokumentation und Erfassung der genauen Anforderungen wurden sogenannte „Case Studies“ (auch Fallstudien genannt) erstellt. Die Case Studies wurden mit der BHTC entwickelt und konnten durch anschauliche Beispiele leicht geprüft und genehmigt werden.

Die eigentliche Umsetzung des Konzeptes wurde anhand des „Inkrementellen Vorgehens“ realisiert. Das inkrementelle Vorgehensmodell beschreibt einen Prozess der kontinuierlichen Verbesserung, der in kleinen oder sogar kleinsten Schritten vollzogen wird. Die Anforderungen wurden zu Projektbeginn möglichst exakt definiert und sauber gegliedert. Durch die folgende Gliederung war es möglich auch innerhalb der Anwendung Teilsysteme zu entwickeln, die vom Anwender getestet werden konnten.

⁶ [it1]

Teil 0: Anforderungsanalyse

- Teil 1: Voraussetzungen an die Sicherheit
 1. bezüglich der Datenbank
 2. bezüglich der Benutzeranmeldung
- Teil 2: Datenerfassung
 1. Erfassen der Projektdaten
 2. Erfassen der Musterstandsdaten
 3. Erfassen der Prototypdaten
- Teil 3: Suchfunktionen
 1. Suche nach Projekten
 2. Suche nach Musterständen
 3. Suche nach Prototypen
- Teil 4: Reportfunktionen
 1. Report auf Projektebene
 2. Report auf Musterstandsebene
 3. Report auf Prototypenebene

Jeder dieser Teile beschreibt ein Teilsystem innerhalb der Anwendung, das z. T. auch einzeln verwendbar ist. Das Zusammenspiel der einzelnen Teilsysteme in Zusammenhang mit dem Anwender ist in Abbildung 3 „Zusammenspiel der Teilsysteme“ schematisch dargestellt.

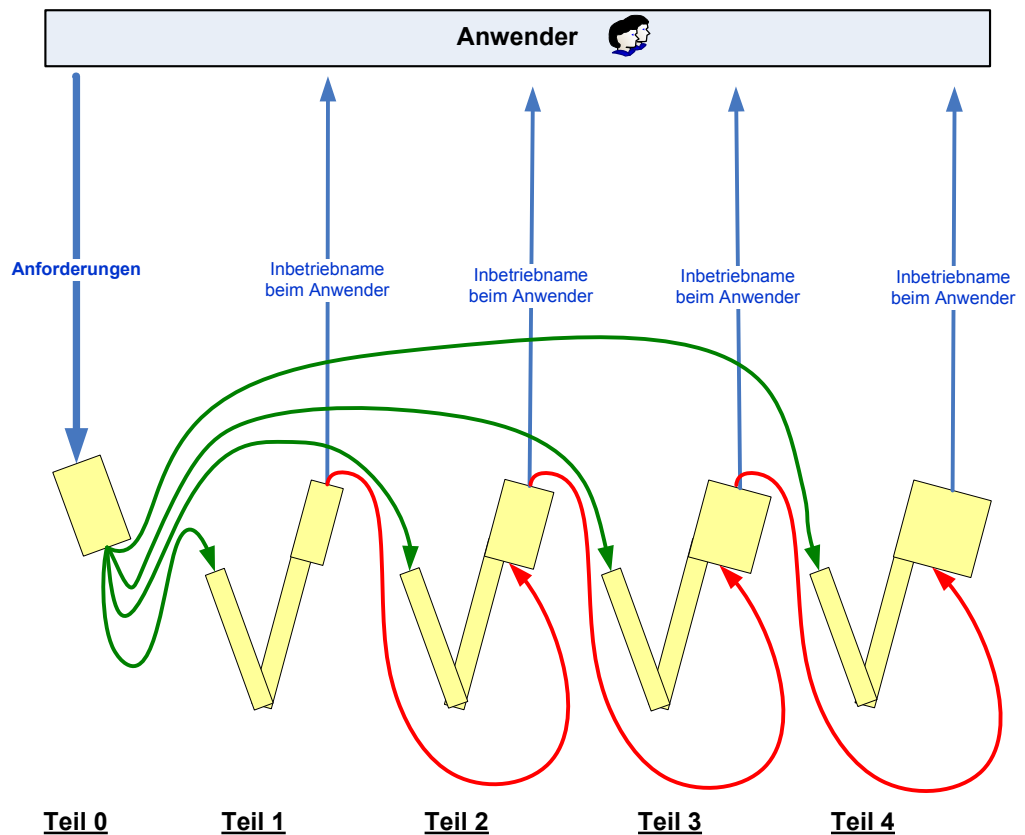


Abbildung 3: Zusammenspiel der Teilsysteme

In der ersten Phase (Teil 0) wurden, wie bereits beschrieben, die Anforderungen erfasst, die als Eintrittspunkt für jedes einzelne Teilsystem (grüner Weg) genutzt wurden. Nach erfolgreicher Entwicklung eines Teilsystems wurde dieses beim Anwender in Betrieb genommen, der dieses testete und evtl. aufgetretene Fehler dokumentierte und/oder Änderungswünsche mitteilen konnte.

Parallel zur Testphase beim Anwender wurde das nächste Teilsystem (Teil 2, Teil 3 usw.) entwickelt. Abschließend wurden die Fehler des getesteten Teilsystems bereinigt und durch das neu entwickelte Teilsystem ergänzt (roter Weg). Das neue Teilsystem wurde nun erneut beim Anwender in Betrieb genommen und getestet.

Dieser Vorgang wiederholte sich bis schließlich das Gesamtsystem beim Anwender vorlag.

Der Vorteil dieses inkrementellen Ansatzes lag in der guten Kalkulierbarkeit auf Grund der geringen Gefahr des „Abdriftens“, da jedes Teilsystem eine genaue Vorlage anhand der Anforderungsanalyse erhielt. Weiterhin konnte bei diesem Ansatz ein zeitlicher Gewinn verbucht werden, da der Anwender auch während der

Entwicklung bereits erstellte Teilsysteme abnehmen konnte, was zusätzlich die Qualität der Anwendung förderte.

In den folgenden Kapiteln sollen nun die einzelnen Teilsysteme 1 bis 4 anhand von Workflows und Case Studies inklusive ihrer Lösungen näher dargestellt werden. Die Phase 0, Anforderungsanalyse, kann in der Projektarbeit „Analyse zur Entwicklung einer webbasierten Datenbank zur technischen Dokumentation von Prototypen“ nachgeschlagen werden.

2 Das Vorgehensmodell

Wie kann man die Entwicklung und Konzeption eines Programms logisch gliedern, das nicht auf einem herkömmlichen Modell von Klassen und Interfaces beruht?

Diese grundlegende Frage stellte sich mir direkt zu Beginn der Diplomarbeit. Üblicherweise bestehen objektorientierte Anwendungen aus einer Vielzahl von Klassen, die über Interfaces (Schnittstellen, über die die Kommunikation gesteuert wird) miteinander Daten austauschen, um diese dann zu verarbeiten. Auf Seiten der Webanwendungsentwicklung erreicht man dieses Konzept durch den Einsatz so genannter Framesets. Mit Hilfe von Framesets kann man den Anzeigebereich des Browsers in verschiedene, frei definierbare Segmente aufteilen. Jedes Segment kann eigene Inhalte haben, die sowohl statisch als auch wechselhaft sein können.

Eine signifikante Anforderung an die PTDB, die durch die BHTC definiert wurde, war es, dass die Anwendung unter ASP.NET in Verbindung mit VB.NET zu entwickeln sei. Da man jedoch HTML-Framesets nicht mit ASP.NET steuern kann, stand diese Maßgabe im Konflikt mit dem üblichen Modell der objektorientierten Web-Programmierung.

HTML-Framesets sind eine clientseitige Technologie und der Webserver (und damit auch ASP.NET) sieht nur einzelne Programmcodeblöcke der Frameelemente der gesamten Webseite. Daher kann ASP.NET auf die Anordnung oder den Inhalt des gesamten Framesets keinen Einfluss nehmen, sondern immer nur Einfluss auf den gerade angeforderten Inhalt des einzelnen Frameelements, was wiederum zur Folge hatte, dass die Anwendung ein Programm-Koloss wurde und eine Gliederung hier nur schwer möglich war.

Da eine sinnvolle Gliederung auf Seiten der Programmlogik ausschied, musste eine andere Ebene der Gliederung gefunden werden. Da es sich bei der PTDB-Anwendung um ein Programm zur Erfassung und Pflege von Daten handelt, erschien hier eine chronologische Reihenfolge sinnvoll zu sein. Denn bevor man Daten pflegen kann, müssen diese eingegeben werden und bevor man Daten eingeben

kann, muss die Sicherheit geprüft werden. Betrachtet man also den Ablauf anhand eines Zeitstrahls, so könnte sich dieser wie folgt darstellen:



Abbildung 4: einfacher Zeitstrahl

Durch diese chronologische Abfolge lassen sich nun die beschriebenen Phasen aus Abbildung 3 „Zusammenspiel der Teilsysteme“ einordnen und das inkrementelle Vorgehensmodell kann auf die gesamte Entwicklung angewendet werden:

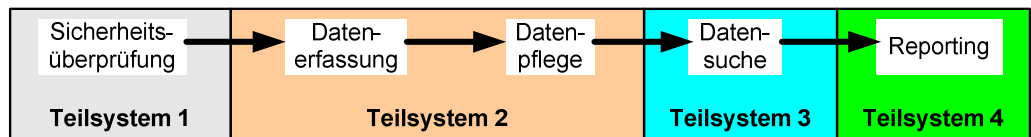


Abbildung 5: Teilsysteme im Zeitstrahl

Normalerweise sind innerhalb des Zeitstrahls auch Sprünge zu abgeschlossenen Stationen möglich, da hier nicht nur die Daten eines Projektes oder eines Prototypen betrachten werden müssen. Jedoch können diese Sprünge bei der Entwicklung der Konzeption und des Programmcodes vorerst unberücksichtigt bleiben, da sie keinen direkten Einfluss auf die Entwicklung haben.

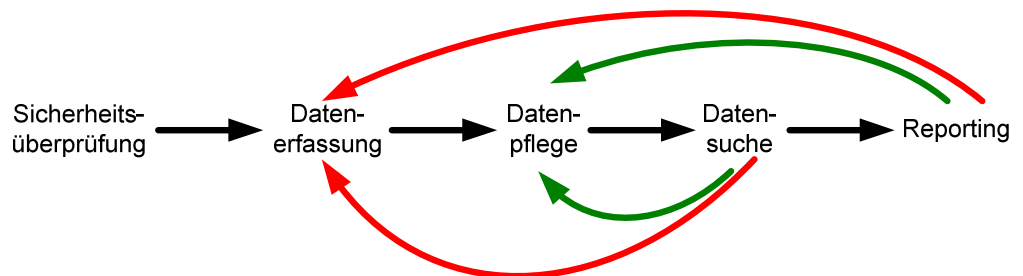


Abbildung 6: Sprünge im Zeitstrahl

Diese Sprünge werden im späteren Programmablauf berücksichtigt und der Anwender ist nicht an eine strikte zeitliche Abfolge gebunden.

Im gesamten Prozess der Prototypenentwicklung gibt es einige Teilbereiche, die hier nicht näher im Detail dargestellt werden, da sie nur eine indirekte Rolle im Rahmen der PTDB-Anwendung haben.

Als einfaches Beispiel hierfür kann die Auftragserfassung innerhalb von SAP genannt werden. Sie spielt für die Prototypenentwicklung eine sehr wichtige Rolle, hat aber innerhalb der PTDB-Anwendung nur eine „Zulieferer“-Funktion.

Betrachten wir einmal die Abfolge der Auftragserfassung in SAP als UML-Diagramm, so würde sich der Vorgang grob skizziert wie folgt darstellen:

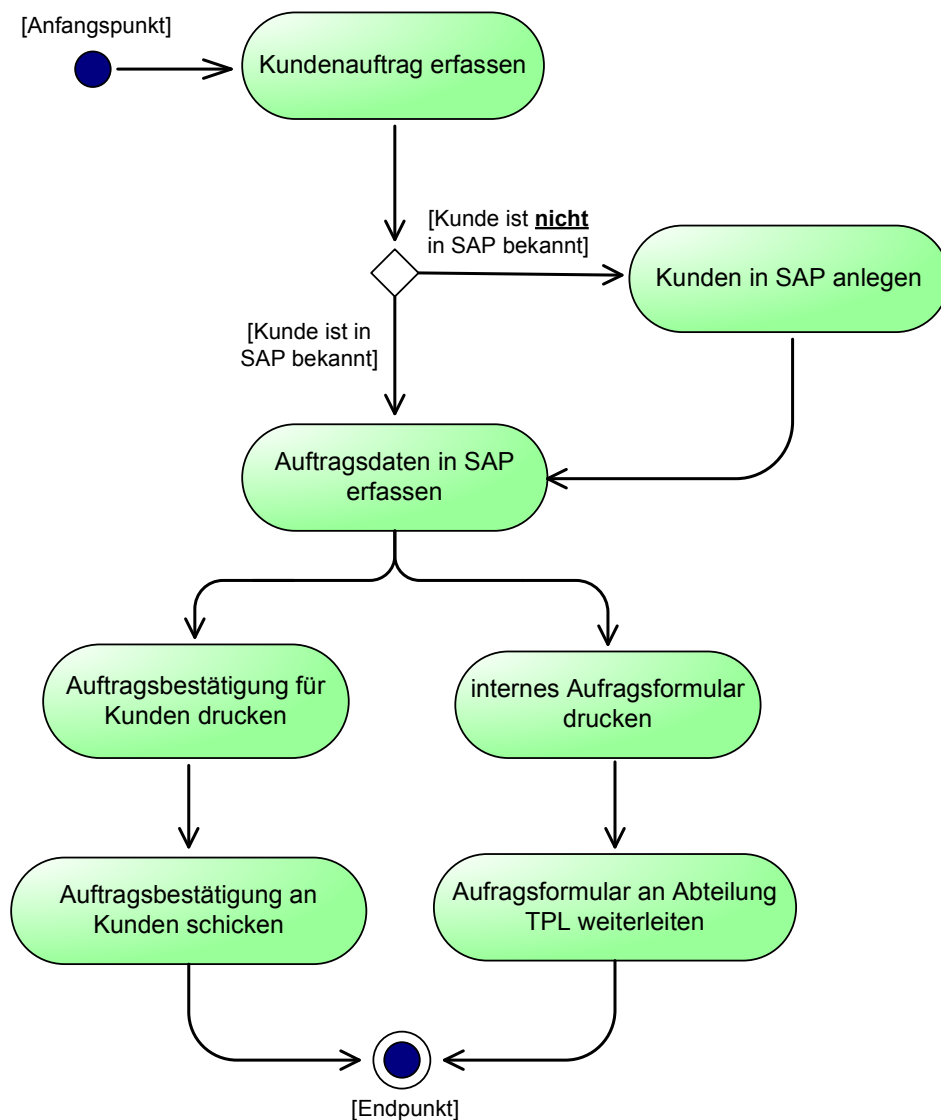


Abbildung 7: UML-Grobskizze SAP-Auftragserfassung

Wie man anhand des Diagramms sehen kann, ist erst der Endpunkt für die eigentliche PTDB-Anwendung wichtig, da eines der Ergebnisse (das interne Auftragsformular) als „Zulieferer“ für die PTDB gesehen werden kann.

Beschränken wir uns also nur auf die Prozessabläufe, die sich primär innerhalb der Anwendung widerspiegeln, so müssen genau die vier Teilsysteme „Sicherheitsprüfung“, „Datenerfassung und -pflege“, „Datensuche“ und „Reporting“ betrachtet werden. Eine genaue Darstellung dieser Teilsysteme finden Sie in den folgenden Kapiteln.

3 Teilsystem 1: Das Sicherheitskonzept

Die Sicherheitsstrategie der Anwendung stützt sich weitestgehend auf die Windows angebotenen Features.

Zu unterscheiden ist hier zwischen dem Zugriff auf die eigentliche Webanwendung, dem Zugriff auf die Daten der Datenbank und die Datensicherheit innerhalb der Anwendung. Im Folgenden sollen nun die einzelnen Bereiche näher erläutert werden, um abschließend zu den einzelnen Punkten die gewählte und eingesetzte Methode darzustellen.

3.1 Authentifizierung für die Webanwendung

3.1.1 Die Benutzer – Authentifizierung⁷ für die Webanwendung

Am Anfang steht die Authentifizierung. Hiermit versteht man die Überprüfung / Verifikation einer behaupteten Echtheit einer Person oder eines Computersystems.

Sich bei der Authentifizierung auf Windows zu stützen bedeutet, dass jeder Anwender über ein Windows-Zugangskonto verfügen muss. Der Anwender kann sich dann über den Browser einloggen. Für die Windows-Authentifizierung gibt es drei Varianten:

- die Windows-Standardauthentifizierung,
- die Windows-Digestauthentifizierung und
- die integrierte Windows-Authentifizierung.

3.1.1.1 Windows – Standardauthentifizierung

Die einfachste Variante der Authentifizierung nennt sich Standardauthentifizierung. Bei dieser Variante werden Anwendername und Passwort unverschlüsselt über das Netzwerk übertragen, so dass sich nur geringe Sicherheitsanforderungen erfüllen lassen. Der Vorteil dieser Methode besteht darin, dass sie nur geringe Anforderungen an den Browser stellt. So gut wie jeder Browser kommt mit dieser Methode klar. Um die Sicherheit zu erhöhen, kann außerdem die Kommunikation SSL-verschlüsselt werden (Secure Sockets Layer (SSL) ist ein hybrides Ver-

⁷ [asp1]

schlüsselungsprotokoll für Datenübertragungen, siehe hierzu auch [http://de.wikipedia.org/wiki/ Transport_Layer_Security](http://de.wikipedia.org/wiki/Transport_Layer_Security)).

3.1.1.2 Windows – Digestauthentifizierung

Bei der Methode der Digestauthentifizierung wird das Passwort verschlüsselt, bevor es über das Netzwerk übertragen wird. Diese Methode erfordert zahlreiche Voraussetzungen sowohl auf der Seite des Browsers als auch auf der Seite des Servers.

Die Digestauthentifizierung steht nur zur Verfügung, wenn man an eine Domäne angeschlossen ist. Die Anwender müssen über ein Benutzerkonto in einem Active Directory (der Verzeichnisdienst von Microsoft Windows 2000/Windows Server 2003, siehe hierzu auch http://de.wikipedia.org/wiki/Active_Directory) verfügen. Das Passwort der Anwender muss außerdem mit Hilfe einer umkehrbaren Verschlüsselung gespeichert werden, dies kann bei den Optionen des WindowsBenutzerkontos eingestellt werden kann.

Der Anwender muss für die Digestauthentifizierung außerdem auf jeden Fall einen Internet Explorer ab Version 5.x verwenden. Die Digestauthentifizierung steht auf der Seite des Servers erst ab Windows 2000 zur Verfügung.

3.1.1.3 Integrierte Windows – Authentifizierung

Im Unterschied zu den beiden ersten Methoden der Windows - Authentifizierung erscheint bei der integrierten Windows - Authentifizierung kein Dialogfeld zur Eingabe von Anwendername und Passwort im Browser. Stattdessen kommunizieren Browser und Server in verschlüsselter Form miteinander, wobei das Passwort selbst nicht übertragen wird. Dabei werden der Anwendername und das Passwort verwendet, mit dem sich der Anwender ohnehin bereits am PC angemeldet hat. Für den Anwender ist das also ein sehr bequemer Weg. Die Tatsache, dass im Hintergrund eine Authentifizierung stattfindet, bleibt ihm dabei völlig verborgen. Diese Methode funktioniert nur dann, wenn der Anwender einen Internet Explorer unter einem Windows-Betriebssystem verwendet.

3.1.1.4 Abgrenzung der Authentifizierungsmethoden

Die Entscheidung für oder gegen eine bestimmte Authentifizierungsmethode hängt von den Voraussetzungen ab, die die Anwendung erfüllen muss und vom Komfort für den Benutzer, der mit der Anwendung arbeiten muss. Die Tabelle 1: Abgrenzung der Authentifizierungsmethoden listet die diversen Kriterien tabellarisch auf.

Authentifizierungsmethode	Arbeitsweise	Voraussetzung für den Browser	Voraussetzung auf dem Server	Höhe der Sicherheit
Windows-Standardauthentifizierung	der Anwendername und das Passwort werden in einem Formular abgefragt und im Klartext an den Server übertragen	mit fast allen Browsern kompatibel	der Anwender benötigt ein Benutzerkonto auf dem Domainserver	Gering. Bei zusätzlicher Verwendung von HTTPS (SSL) jedoch hoch
Windows-Digestauthentifizierung	der Anwendername und das Passwort werden vom Anwender abgefragt und vor der Übertragung vom Browser verschlüsselt	nur der Internet Explorer ab Version 5.x beherrscht diese Art der Authentifizierung	erst ab Windows 2000 verfügbar. Es ist ein Benutzerkonto im Active Directory erforderlich. Das Passwort des Anwenders muss mit der Option „umkehrbare Verschlüsselung“ gespeichert werden	Mittel

Authentifizierungsmethode	Arbeitsweise	Voraussetzung für den Browser	Voraussetzung auf dem Server	Höhe der Sicherheit
integrierte Windows-Authentifizierung	der Anwendername und das Passwort, mit denen der Anwender sich bei Windows angemeldet hat, werden automatisch vom Server überprüft. Es ist keine erneute Eingabe erforderlich	ab Internet Explorer 3.01 verfügbar	es ist ein Benutzerkonto auf dem Domainserver notwendig	Hoch

Tabelle 1: Abgrenzung der Authentifizierungsmethoden

3.1.2 Auswahl der Authentifizierungsmethode für die Anwendung

In der Anforderung „3.4.1 PTDB-V01-R005: Login-Verwaltung“⁸ wurde festgelegt, dass es wünschenswert sei, wenn der Benutzer anhand seiner Windows-Anmelde-daten identifiziert werden könnte. Diese Anforderung kann nur durch die Authentifizierungsmethode „integrierte Windows-Authentifizierung“ erreicht werden. Diese Methode bietet dem Anwender den größten Komfort und dennoch gewährt sie eine hohe Sicherheit.

Ein weiterer Vorteil dieser Methode ist der relativ einfache Wartungsaufwand bei der Anwendung, da der Systemadministrator keine Benutzerdaten-Pflege auf Seiten der Anwendung tätigen muss.

Da innerhalb der BHTC der Internet Explorer 5.0 als Standardbrowser und Windows XP als Standard-Betriebssystem vorgegeben sind, sind die notwendigen Voraussetzungen für diese Methode vorhanden.

⁸ [ang1]

3.2 Authentifizierung für die Datenbank⁹

Der eingesetzte Datenbankserver Microsoft SQL Server 2005 stellt bei der Authentifizierung zwei Methoden für einen sicheren Zugriff auf den Server und seine Daten bereit:

- der Windows - Authentifizierungsmodus
- und der gemischte Modus

3.2.1 Windows - Authentifizierungsmethode für den Datenbankserver

Der Windows - Authentifizierungsmodus ist der standardmäßige Authentifizierungsmodus im Microsoft SQL Server 2005. Im Windows-Authentifizierungsmodus wird für SQL Server 2005 nur die Windows - Authentifizierung des Benutzers verwendet, vergleichbar mit der integrierten Windows-Authentifizierung der Web-Anwendung.

Windows-Benutzern oder -Gruppen kann dann der Zugriff auf den Server gewährt werden. Verbindungen, die in diesem Modus mit dem Server hergestellt werden, werden auch vertraute Verbindungen genannt, da der Server der Identität des Anwenders vertraut.

Der Datenbankadministrator kann den Zugriff auf den SQL Server durch Erteilen von Anmelderechten gewähren.

Intern wird der Anmeldezugriff innerhalb des Datenbankservers über sogenannte Security Identifier (SID, deutsch Identifikationsnummer) geregelt. Security Identifier sind Teil des Security Descriptors (SD) und wird ab Microsoft Windows NT schon während der Installation vergeben. Da es in einem Netzwerk gleichnamige Benutzer geben kann, verwendet Windows zur eindeutigen Kennzeichnung von Benutzern die SIDs. Der Security Identifier wird für einen Benutzer automatisch angelegt, wenn er lokal oder in einer Domäne erstellt wird. Die SID eines Benutzers ändert sich, wenn er in eine andere Domäne verschoben wird, da in der SID auch die betreffende Domäne des Benutzers hinterlegt wird.

⁹ [ms1]

Da Windows-SIDs verwendet werden, kann der Datenbankadministrator Windows-Benutzern oder -Gruppen direkt Anmeldezugriff gewähren.

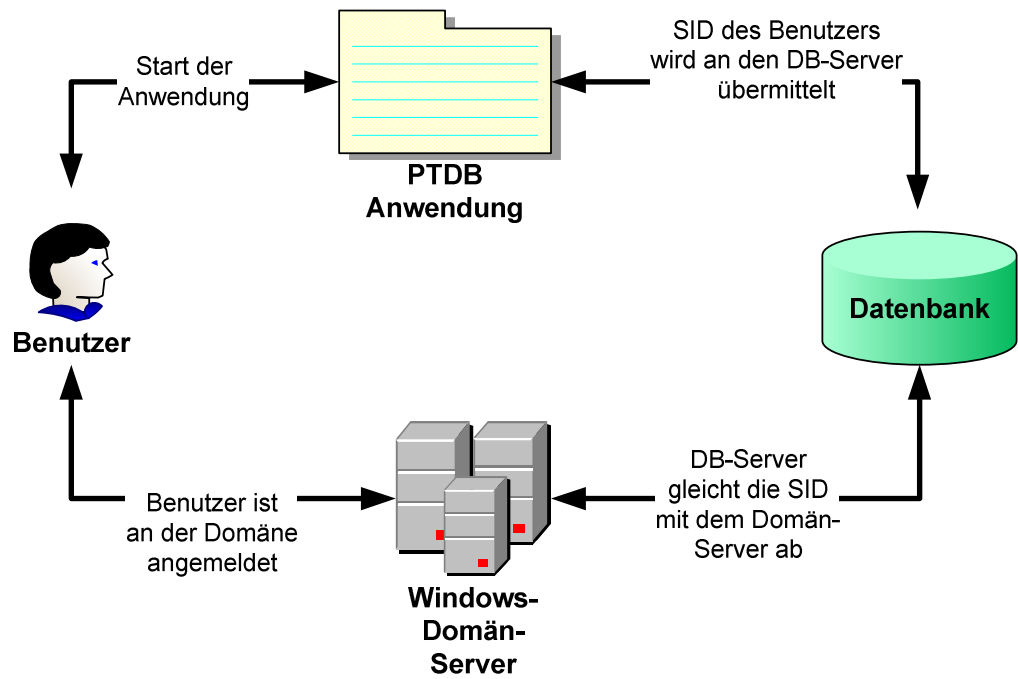


Abbildung 8: Windows - Authentifizierungsmethode des Datenbankservers

3.2.2 Gemischter Modus als Authentifizierungsmethode

Im gemischten Modus können Benutzer durch die Windows - Authentifizierung oder durch die SQL Server - Authentifizierung authentifiziert werden. Die Benutzernamen und Kennwörter von Benutzern, die vom SQL Server authentifiziert werden, werden direkt im SQL Server verwaltet und sind nicht direkt an ein Windows-Login gebunden.

Ebenso wie im Windows - Authentifizierungsmodus wird für den Server mit SQL Server auch bei einer im gemischten Modus hergestellten Verbindung die Authentifizierung der Benutzer durch Windows verwendet, wenn auf dem Client und dem Server NTLM oder Kerberos - Anmeldeauthentifizierungsprotokolle verwendet werden können. Wenn auf dem Client keine Windows-Standardanmeldung verwendet werden kann, benötigt SQL Server einen Benutzernamen und ein Kennwort. Diese werden dann mit den gespeicherten Angaben in den Systemtabellen verglichen. Verbindungen, die auf Benutzernamen und Kennwort beruhen, werden nicht vertraute Verbindungen oder SQL-Verbindungen genannt.

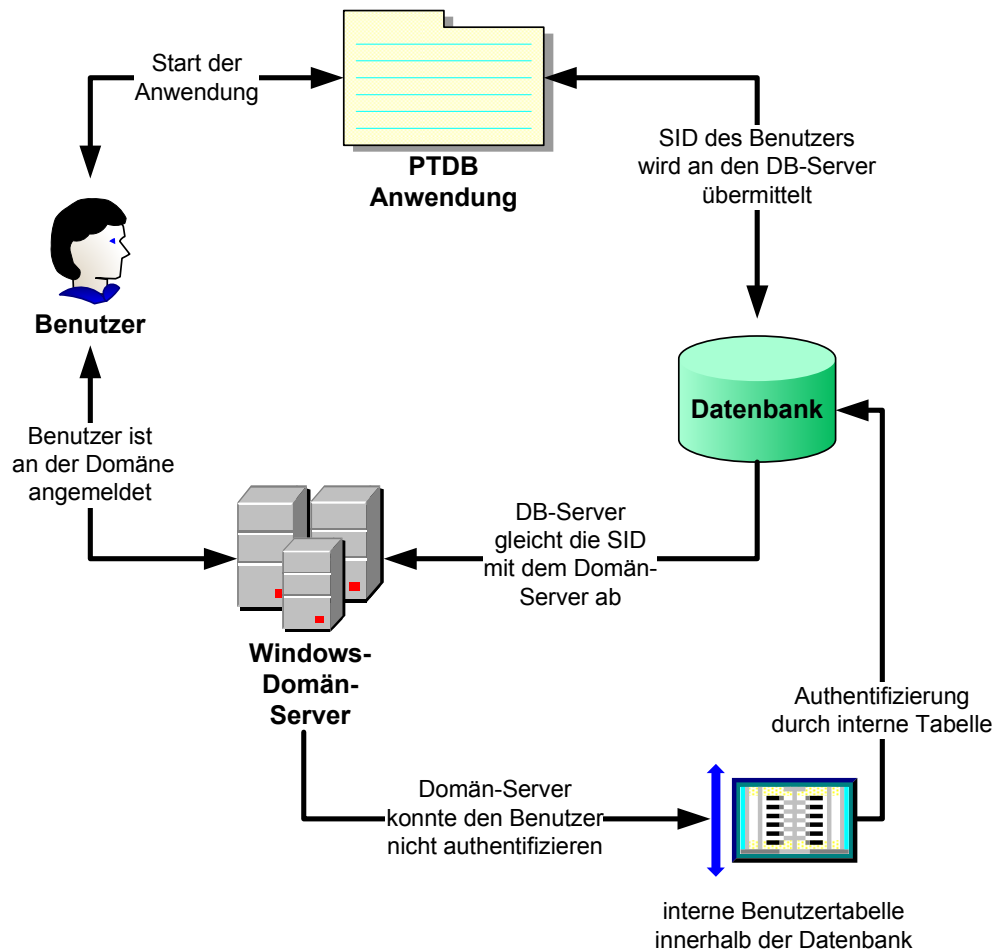


Abbildung 9: Gemischter Modus als Authentifizierungsmethode des Datenbank-servers

3.2.3 Auswahl der Authentifizierungsmethode für den SQL-Server

Parallel zu der Authentifizierungsmethode für die Anwendung wird auch der SQL-Server die Authentifizierung über die Windows – Authentifizierung vornehmen. Bei dieser Methode ist der spätere Verwaltungsaufwand überschaubar und zusätzliche Anwender können ohne zusätzliche administrative Arbeiten direkt mit der Webanwendung arbeiten.

3.3 Umsetzung der Authentifizierungen

3.3.1 Umsetzung der Authentifizierungen für den Datenbankzugriff

Der Zugriff auf die Datenbank wird über die globale Web-Konfigurationsdatei `web.config` gesteuert. Die `web.config` Datei ist eine auf XML (Extensible Markup Language (engl. für „erweiterbare Auszeichnungssprache“)) basierende, hierarchisch aufgebaute Textdatei, die eine Vielzahl von Einstellungsmöglichkeiten für die Applikationen erlaubt. Angefangen von Applikationseinstellungen, über Data Source Name bis hin zu Optionen für Web Services. Der hierarchische Aufbau der Datei erlaubt eine übersichtliche Gliederung der einzelnen Elemente.

Die hierarchische Ableitung erfolgt von der ASP.NET Hauptkonfigurationsdatei (`machine.config`) bis in die einzelnen virtuellen Verzeichnisse jeder einzelnen am .NET Webserver befindlichen Applikation. Bildlich gesprochen sieht dies für einen Beispiel-.NET Webserver wie folgt aus:

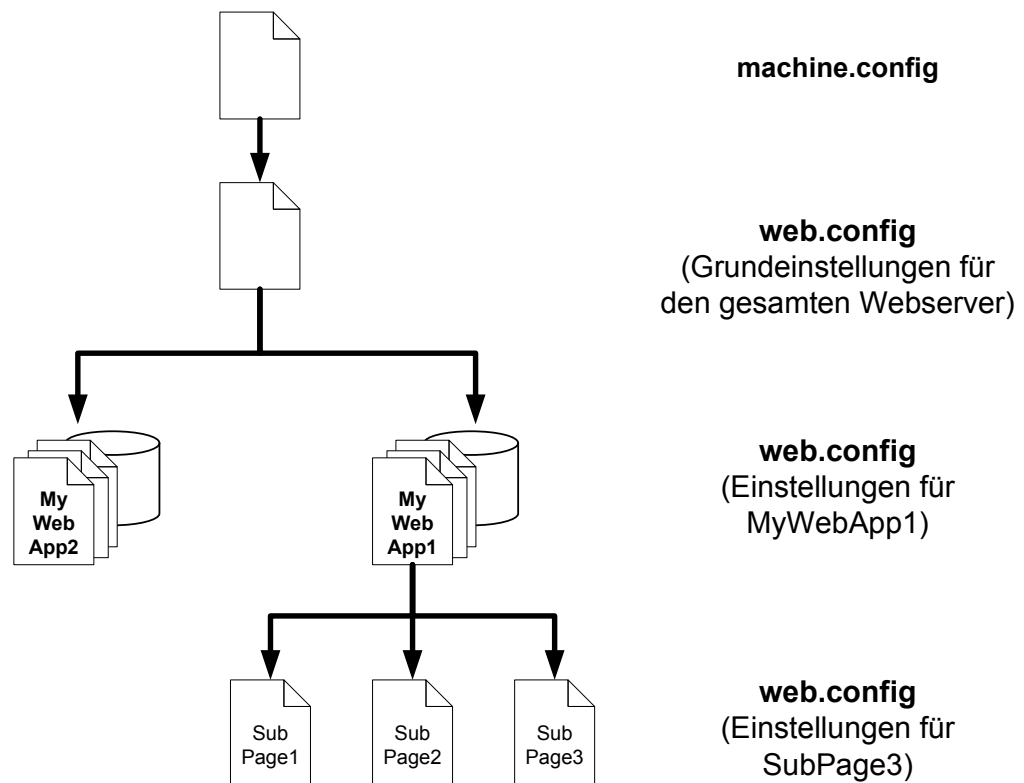


Abbildung 10: Konzept der Web.config

Die web.config-Datei ist in einzelne Section Handler unterteilt. Die Section Handler sind einzelne Klassen innerhalb der Konfigurationsdatei, die die unterschiedlichen Einstellungen beinhalten.

Im Section Handler „connectionStrings“ wird der Data Source Name der ASP.NET Anwendung zur Verfügung gestellt. In der Einstellung zum Data Source Name kann unter anderem die Identifizierung eines Benutzers (Benutzername und Passwort) hinterlegt werden. Der genaue Aufbau kann im Anhang 12.3 Anhang: Section-Handler „connectionStrings“ für den Datenbankzugriff nachgeschlagen werden.

Diese globale Einstellung der Anwendung wird innerhalb der Applikation im Programmcode ausgelesen und in einer internen, öffentlichen Variablen (ConnStr) gespeichert:

```
Public ConnStr As String = ┐  
WebConfigurationManager.ConnectionStrings("PTDBCONN"). ┐  
ToString()10
```

Über diese Methode ist es möglich, im weiteren Programmverlauf über die Variable *ConnStr* jeden Datenbankzugriff herzustellen. Weiterhin ist über diese Methode gewährleistet, dass eine einfache Umstellung auf andere Datenbanken (z. B. auf die Entwicklungsumgebung) realisiert werden kann. Hierzu muss nur der ConnectionString innerhalb der web.config verändert werden.

3.3.2 Umsetzung der Authentifizierungen für die Webanwendung

Der Zugriff auf die Prototypenanwendung wird, wie in 3.1.1.3 Integrierte Windows – Authentifizierung beschrieben, durch die „integrierte Windows-Authentifizierung“ realisiert. Hierzu wird innerhalb der Anwendung eine interne, öffentliche Variable mit dem Benutzernamen gefüllt, den der Anwender zur Anmeldung an der Workstation benutzt hat. Innerhalb der Visual Basic Systemklasse *System.Environment* steht in der Variablen *UserName* der Anwendername bereit. Die Definition der öffentlichen Programmvariablen sieht innerhalb des Programmcodes wie folgt aus:

```
Public PTDBUser As String = ┐  
UCase(System.Environment.UserName)
```

¹⁰ ┐ = Dieses Zeichen gibt an, dass die Zeile hier nicht abgeschlossen ist

Diese Variable wird im weiteren Verlauf der Anwendung dazu benutzt, den Anwender zu identifizieren und ggf. Logbucheinträge eindeutig einem Benutzer zuzuweisen und die Rechte des Anwenders innerhalb eines Projektes zu überprüfen.

Durch die interne Visual Basic-Funktion *Ucase* wird der Anmeldename immer in Großbuchstaben umgewandelt, um spätere Inkonsistenzen zu vermeiden.

Als Voraussetzung für die „integrierte Windows-Authentifizierung“ muss innerhalb der Web-Konfigurationsdatei `web.config` folgender Bereich hinterlegt werden:

```
<authentication mode="Windows"/>
<identity impersonate="true"/>
```

3.4 Datensicherheit innerhalb der Anwendung

3.4.1 SQL-Injection¹¹

Mit SQL-Injektion bezeichnet man das Ausnutzen einer Sicherheitslücke in Zusammenhang mit SQL-Datenbanken, die durch mangelnde Maskierung oder Überprüfung von Metazeichen in Benutzereingaben entsteht. Der Angreifer versucht dabei über die Anwendung, die den Zugriff auf die Datenbank bereitstellt, eigene Datenbankbefehle einzuschleusen. Sein Ziel ist es, dabei Daten in seinem Sinne zu verändern oder Kontrolle über den Server zu erhalten.

Für SQL-Injektionen nutzbare Fehler treten auf, wenn eine Anwendung Benutzereingaben als Teil von SQL-Abfragen an den Server weiterreicht, ohne die vom Benutzer eingegebenen Werte ausreichend zu prüfen und etwaige enthaltene Metazeichen zu maskieren, um ihnen so die Sonderfunktion zu nehmen. Metazeichen in SQL sind zum Beispiel der umgekehrte Schrägstrich (engl. Backslash (\)), das Anführungszeichen (“), der Apostroph (') und das Semikolon (;). Diesen Zeichen kann durch Voranstellen des Maskierungszeichens, einem umgekehrten Schrägstrich (\), die Funktion als Metazeichen entzogen werden, sodass es als normales Textzeichen gewertet wird. Dieser Vorgang wird auch „Escapen“ genannt.

Da es sich bei der Prototypendatenbank um eine interne BHTC-Anwendung handelt, die nicht von externen Anwendern benutzt wird, kann davon ausgegangen

¹¹ [wiki3]

werden, dass keine mutwillige Einschleusung von SQL-Injections auftreten wird. Nichts desto weniger, wird innerhalb des Programmcodes eine Überprüfung auf evtl. auftretende Fehleingaben hinterlegt, um versehentliche „gefährliche“ Benutzereingaben abzufangen und zu korrigieren. Die hierzu implementierte Funktion kann dem Anhang 12.1 Funktion „MakeDBText“ entnommen werden.

3.4.2 Direkte Validierung der Benutzereingaben

Zusätzlich zum bereits erläuterten Abfangen von SQL-Injections, werden Großteile der eingegebenen Daten direkt nach erfolgter Eingabe validiert, um dem Anwender auf direktem Wege mitzuteilen, dass eine Korrektur erforderlich ist.

Auch bei der Validierung der Daten wird das Konzept befolgt, sich möglichst weitgehend auf die angebotenen Windows-Features zu stützen. Hierzu werden die Validator-Controls, die Bestandteil der ASP.NET-Entwicklungsumgebung sind, genutzt. Grundsätzlich erfolgt die Prüfung der eingegebenen Daten auf dem Server.

Innerhalb der Anwendung werden vier Methoden der Validierung benutzt:

- Überprüfung anhand eines regulären Ausdrucks
- Überprüfung auf Eingabe eines Pflichtfeldes
- Überprüfung eines korrekten Datums
- benutzerdefinierte Validierung

Alle Validatoren sind zur Laufzeit der Anwendung zunächst unsichtbar. Wird jedoch versucht einen fehlerhaften Eingabewert an den Server zu übermitteln, so erscheint die gewünschte Fehlermeldung im Validator-Control und der Anwender muss seine Eingabe korrigieren (Beispiele hierzu später im Bereich der einzelnen Überprüfungsbeschreibungen).

3.4.2.1 Überprüfung anhand eines regulären Ausdrucks

Ein sehr mächtiges Werkzeug im Bereich der Gültigkeitsprüfung ist das Validieren anhand so genannter regulärer Ausdrücke.

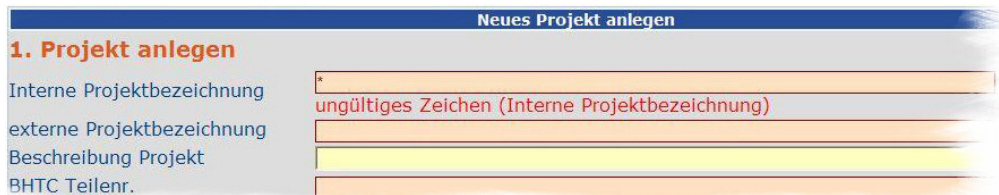
Ein regulärer Ausdruck ist in der Informatik eine Zeichenkette, die der Beschreibung von Mengen beziehungsweise Untermengen von Zeichenketten mit Hilfe bestimmter syntaktischer Regeln dient.

Reguläre Ausdrücke stellen eine Art Filterkriterium für Texte dar, indem der jeweilige reguläre Ausdruck in Form eines Musters mit dem Text abgeglichen wird. So ist es beispielsweise möglich, alle Wörter, die mit S beginnen und mit D enden, zu finden, ohne die zwischenliegenden Buchstaben explizit vorgeben zu müssen.

Innerhalb der Prototypen-Anwendung werden anhand von regulären Ausdrücken mögliche SQL-Injections in Textfeldern abgefangen. Hierzu wird folgender Validator benutzt:

```
<asp:RegularExpressionValidator ID="regex_sql"           ↵
  runat="server"                                       ↵
  ErrorMessage=""                                     ↵
  ControlToValidate="txt1_desc"                       ↵
  ValidationExpression="^[^'\*\%\\{\}\^\^]*$"         ↵
  Display="Dynamic">                                  ↵
</asp:RegularExpressionValidator>
```

In diesem Beispiel wird überprüft, ob im Eingabefeld *txt1_desc* eines der möglicherweise „gefährlichen“ Zeichen eingegeben wurde (Zirkumflex (^), Hochkomma (’), Sternchen (*), Prozent (%)) oder Backslash (\)). Sollte die Überprüfung eines der Zeichen ermitteln, so wird eine Fehlermeldung angezeigt:



The screenshot shows a web form titled "Neues Projekt anlegen". Under the heading "1. Projekt anlegen", there are four input fields. The first field, "Interne Projektbezeichnung", contains an asterisk (*) and has a red error message "ungültiges Zeichen (Interne Projektbezeichnung)" displayed below it. The other fields, "externe Projektbezeichnung", "Beschreibung Projekt", and "BHTC Teilnr.", are empty.

Abbildung 11: Validierung auf ungültige Zeichen

3.4.2.2 Überprüfung auf Eingabe eines Pflichtfeldes

Innerhalb der ASP-Entwicklungsumgebung gibt es ein Validator-Control das überprüft, ob Felder ausgefüllt wurden. Diese Gültigkeitsprüfung wird auch innerhalb der Datenbank vollzogen. Da der Benutzer jedoch einen Programmabbruch angezeigt bekommen würde, wenn ein Pflichtfeld innerhalb der Datenbank nicht korrekt gefüllt wurde, ist es sinnvoll eine Überprüfung schon vor der Übermittlung an die Datenbank vorzunehmen. Der Aufbau dieses Controls sieht innerhalb der Anwendung wie folgt aus:

```

<asp:RequiredFieldValidator ID="regex_mandatory"           ↵
  runat="server"                                         ↵
  ErrorMessage=""                                       ↵
  ControlToValidate="txt1_no"                           ↵
  Display="Dynamic">                                    ↵
</asp:RequiredFieldValidator>

```

In diesem Beispiel wird überprüft, ob das Eingabefeld "txt1_no" gefüllt wurde und bei fehlender Eingabe erscheint folgende Fehlermeldung:

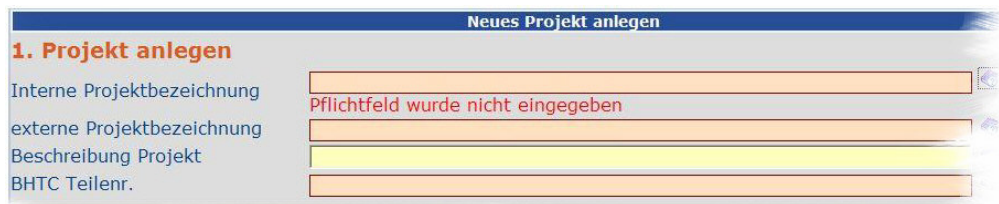


Abbildung 12: Validierung eines Pflichtfeldes

3.4.2.3 Überprüfung eines korrekten Datums

Mit Hilfe des sogenannten RangeValidator wird innerhalb der Anwendung auf die Einhaltung eines bestimmten Wertebereiches geprüft. Ein gültiger Wertebereich für die Eingabe eines Datums ist, nach Absprache mit den betreuenden Personen der BHTC, der Bereich zwischen dem 01.01.2000 und dem 31.12.2107.

Der Aufbau des implementierten RangeValidators sieht innerhalb der Applikation wie folgt aus:

```

<asp:RangeValidator ID="regex_date_5"                   ↵
  runat="server"                                       ↵
  ControlToValidate="txt2_caddate"                     ↵
  ErrorMessage="-"                                     ↵
  MaximumValue="31.12.2107"                            ↵
  MinimumValue="01.01.2000"                            ↵
  Type="Date"                                           ↵
  Display="Dynamic">                                   ↵
</asp:RangeValidator>

```

In diesem Beispiel wird überprüft, ob das im Feld "txt2_caddate" eingegebene Datum zwischen dem 01.01.2000 und dem 31.12.2107 liegt. Bei erkannter Fehlein-gabe erscheint folgende Fehlermeldung:

CAD-Zeichnungsnr.:

CAD-Zeichnungsdatum: ungültiges Datum (CAD-Zeichnungsdatum)

CAD-Zeichner:

Abbildung 13: Validierung eines Datums

3.4.2.4 benutzerdefinierte Validierung

Einige Überprüfungen erfordern den Einsatz von Programmcode zur Validierung. Hierzu wird der CustomValidator benutzt, der direkt mit einem Event innerhalb des Programmcodes verbunden wird. Innerhalb des aspx-Seite sieht der Validator wie folgt aus:

```
<asp:CustomValidator ID="CustVal_4"           ↵
    runat="server"                            ↵
    ControlToValidate="txt1_no"               ↵
    ErrorMessage=""                           ↵
    Display="Dynamic">                       ↵
</asp:CustomValidator>
```

Dieser Validator ist mit dem Eingabefeld "txt1_no" verbunden und nach erfolgter Eingabe wird ein Event innerhalb des Programmcodes ausgelöst. Der entsprechende Programmcode hierzu sieht wie folgt aus:

```
Protected Sub
    CustVal_4_ServerValidate                   ↵
        (ByVal source As Object,              ↵
         ByVal args As ServerValidateEventArgs) ↵
        Handles CustVal_4.ServerValidate

    Dim MKind As String = ddl_mtyp.SelectedValue & args.Value
    CustVal_4.ErrorMessage = PTDBF.Get_Lang_from_DB(MyLang, ↵
    "CustVal_4_1") & MKind & PTDBF.Get_Lang_from_DB(MyLang, ↵
    "CustVal_4_2")
    Dim Status As Boolean
    Dim SQLCmd As New SqlCommand("SELECT COUNT(*) FROM ↵
    PTDB_modelrelease WHERE PROJECT_ID=" & txt_SELPJID.Text & _
    " AND modelclass='" & MKind & "'", DBC1)
```

```
SQLCmd.Connection.Open()  
If (SQLCmd.ExecuteScalar < 1) Then  
    Status = True  
Else  
    Status = False  
End If  
  
SQLCmd.Connection.Close()  
args.IsValid = Status  
  
End Sub
```

Dieser Programmcode überprüft, ob bereits ein Musterstand mit der eingegebenen Musterstandsbezeichnung unterhalb des Projektes existiert und liefert als Rückgabewert an das ValidatorControl „True“ oder „False“. Die Fehlermeldung innerhalb der Anwendung wird wie folgt dargestellt:



Abbildung 14: benutzerdefinierte Validierung

Weitere Beispiele für eine benutzerdefinierte Überprüfung von Werten innerhalb der Anwendung sind:

- Überprüfung, ob ein Projekt mit dem angegebenen Namen bereits innerhalb der Datenbank existiert
- Überprüfung, ob die angegebene Auftragsnummer bereits innerhalb der Datenbank existiert

3.4.2.5 Vorteile der direkten Validierung

Der Vorteil der direkten Datenvalidierung seitens der Anwendung liegt darin, dass der Anwender direkt dazu aufgefordert wird, die eingegebenen Werte zu korrigieren. Hierdurch wird vermieden, dass falsche Werte in der Datenbank hinterlegt werden können oder Schadcode ausgeführt wird.

Ein weiterer Vorteil dieser Überprüfung, inklusive der direkten Rückmeldung, liegt darin, dass der Anwender „indirekt“ trainiert wird, korrekte Angaben zu tätigen.

3.4.3 Datenintegrität innerhalb der Datenbank

Unter Datenintegrität oder auch Konsistenz innerhalb einer Datenbank spricht man, wenn die Widerspruchsfreiheit von Daten gewährleistet ist. Im Regelfall wird die Datenintegrität seitens des Datenbank-Servers geprüft, was durch die Definition von Integritätsbedingungen festgelegt werden kann.

Beispiele für mögliche Integritätsbedingungen beim Einfügen, Löschen oder Ändern innerhalb eines Datenbankmanagementsystems sind:

- es dürfen keine doppelten Primärschlüssel entstehen
- Datensätze müssen dem Relationsschema entsprechen (Einhaltung von Fremdschlüsselbedingungen (Referenzielle Integrität))
- eingegebene Werte müssen im Gültigkeitsbereich liegen (nicht NULL)

Sollte das Datenbankmanagementsystem einen Fehler feststellen, so wird eine Exception (ein Fehler) ausgelöst und der Zugriffsvorgang (Einfügen, Löschen, Ändern) wird zurückabgewickelt („Rollback“) so dass der vorige Zustand wiederhergestellt ist.

Innerhalb der PTDB-Anwendung werden sowohl Integritätsbedingungen innerhalb der Datenbank eingesetzt, als auch eine vorherige Überprüfung von Werten innerhalb des Programmcodes. Hierdurch wird eine höchstmögliche Sicherheit der Datenintegrität gewährleistet.

3.5 Fazit des Sicherheitskonzeptes

Mit Hilfe der eingesetzten Authentifizierungsmethode innerhalb der Webanwendung, der Authentifizierungsmethode innerhalb des Datenbankmanagementsystems, den diversen Überprüfungen von Werten innerhalb der Anwendung und der Befolgung der Datenintegrität wurde die Anwendung auf stabile und sichere Beine gestellt.

Alle eingesetzten Konzepte stellen sicher, dass die Anwendung nahezu ohne Fehler arbeitet und versehentliche Falscheingaben keinen Schaden verursachen können. Dennoch bieten die gewählten Konzepte einen hohen Komfort auf Seiten des Systemadministrators, da die Wartbarkeit einfach zu bewerkstelligen ist und man zu keiner Zeit direkten Eingriff in den Programmcode vornehmen muss.

4 Teilsystem 2: Datenerfassung und -pflege

Die Grundlage einer Anwendung zur Datenverwaltung bilden naturgemäß die Daten. Innerhalb der PTDB-Anwendung werden die Daten, wie bereits vorgestellt, innerhalb einer SQL-Datenbank vorgehalten. Im Wesentlichen werden innerhalb der Datenbank drei Hauptfamilien von Daten bezüglich der Prototypen erfasst, die untereinander „verwandt“ sind. Betrachtet man die Daten innerhalb der Datenbank als Stammbaum, so würde sich die Struktur wie folgt darstellen:

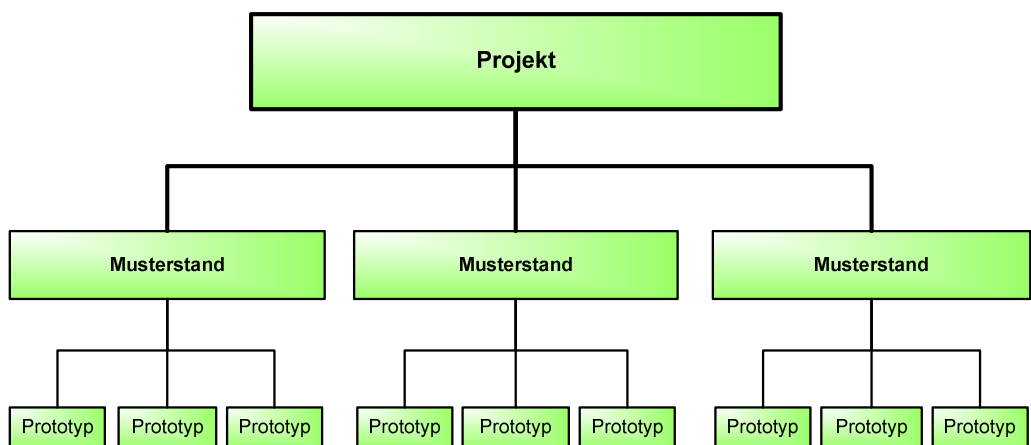


Abbildung 15: Die drei Datenfamilien

Wie man anhand der Abbildung erkennen kann, muss innerhalb der Anwendung als erstes ein Projekt existieren, da die Daten der Musterstände immer einem Projekt zugeordnet sind. Gleiches gilt für die Beziehung zwischen den Prototypen und den Musterständen, denn jeder Prototyp ist eindeutig einem Musterstand zugeordnet. Im Folgenden wird nun dargestellt, wie die einzelnen Datenfamilien innerhalb der Anwendung erfasst und gepflegt werden.

Sobald alle notwendigen Daten zur Dokumentation der Prototypen im System erfasst wurden und somit in der Datenbank vorrätig sind, können diese Daten durch die unterschiedlichen Anwender anhand von Masken innerhalb der Anwendung bearbeitet und ergänzt werden.

4.1 Die Datenerfassung

4.1.1 Projektdaten erfassen

Die Projektdaten sind die essentiellen Grundlagen der gesamten Datenhaltung. Auf dieser Ebene werden die Informationen über die Kunden, die Aufträge und den allgemeinen Rahmen des Projektes gehalten. Alle notwendigen Informationen zu einem Projekt werden anhand des internen Auftragsformulars (siehe auch Anhang 12.6 internes Auftragsformular) übermittelt. Die wesentlichen Bestandteile eines Projektes sind die allgemeinen Projektdaten, die Auftragsdaten, die Kundendaten und die dem Projekt zugewiesenen Anwender. Betrachtet man den Ablauf der Projekterstellung innerhalb der Anwendung, so stellt sich der Prozess wie folgt dar:

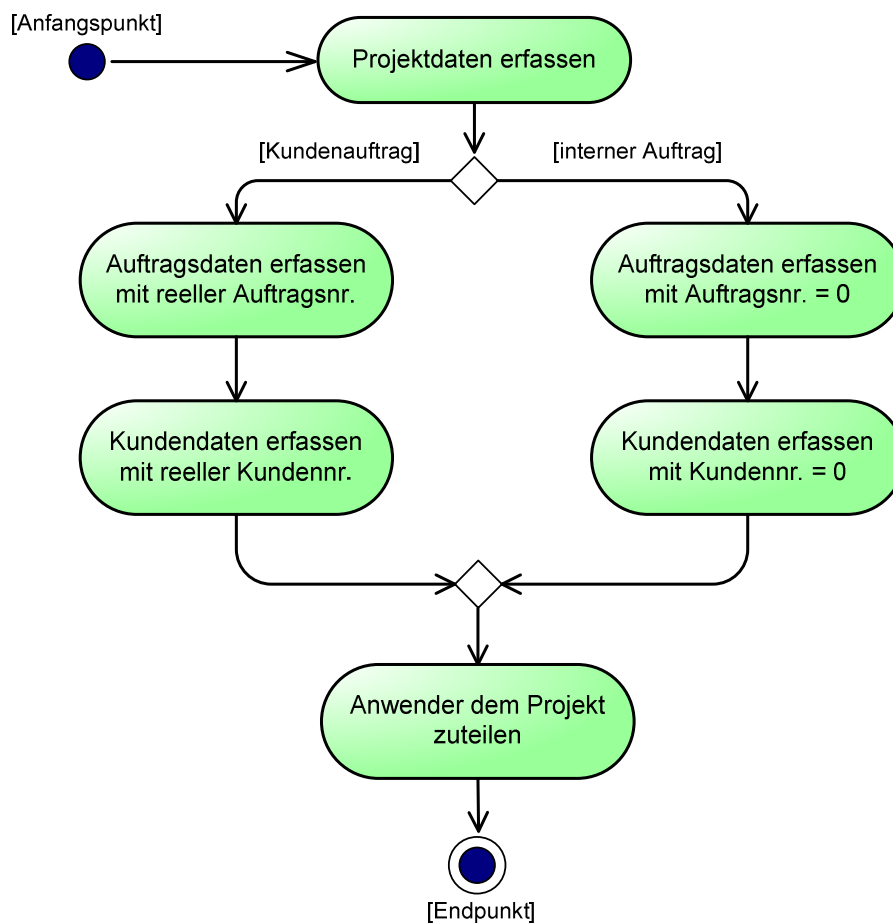


Abbildung 16: UML-Diagramm Projekterstellung

Alle benötigten Daten zur Erstellung eines Projektes werden anhand eines internen Auftragsformulars weitergegeben.

Eine Besonderheit stellt hier die Unterscheidung zwischen „echten“ Kundenaufträgen und internen Aufträgen dar. Innerhalb der Konzeptionsphase wurde mit der BHTC ein Konzept zur Prüfung von Auftrags- und Kundennummern entwickelt, die auf den Daten von SAP aufsetzt. Da jedoch interne Aufträge (z.B. eine interne Bestellung des Prüflabors zur Qualifizierung von Musterständen) zurzeit noch nicht in SAP erfasst werden, musste hier eine separate Lösung für interne Aufträge entwickelt werden. Da sowohl die Kundennummer als auch die Auftragsnummer mit Hilfe einer benutzerdefinierten Validierung geprüft werden (siehe hierzu auch 3.4.2.4 benutzerdefinierte Validierung), wurde definiert, dass interne Aufträge immer durch die Auftragsnummer 0 und interne Kunden parallel hierzu immer mit der Kundennummer 0 eingegeben werden. Durch diesen Softschalter ist es möglich auch interne Aufträge zu erfassen ohne die Validierung zu verletzen.

4.1.2 Musterstandsdaten erfassen

Wie in der Projektarbeit („Analyse zur Entwicklung einer webbasierten Datenbank zur technischen Dokumentation von Prototypen“) bereits erläutert, stellt der Musterstand den jeweiligen Generationsstand eines Prototyps dar und gilt für alle Prototypen gleich, die unter ihm angelegt werden. Jeder Musterstand muss direkt mit einem Projekt verbunden sein und kann nicht separat betrachtet werden, da gleiche Musterstände innerhalb unterschiedlicher Projekte, unterschiedlichen Spezifikationen unterliegen können. Der erste Schritt zum Anlegen eines neuen Musterstandes ist es daher, das man das zugehörige Projekt aus der Baumansicht der verfügbaren Projekte auswählt.

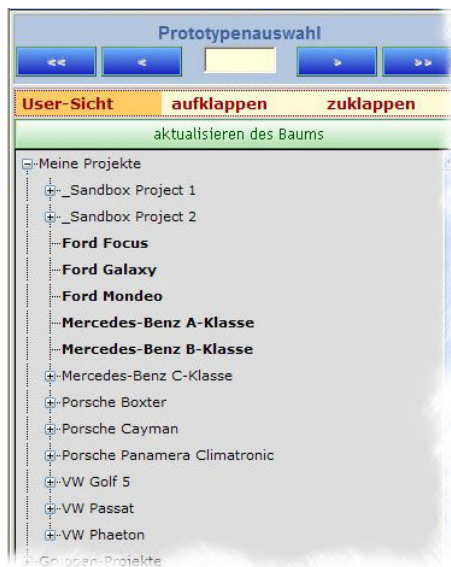


Abbildung 17: Projektnavigation

Über diese Baumansicht kann man leicht ein Projekt wählen, um diesem einen Musterstand hinzuzufügen. Ein Musterstand setzt sich, wie in der vorangegangenen Projektarbeit beschrieben, aus allgemeinen Informationen, einem Konstruktionsstand, einem Softwarestand und einem Hardwarestand zusammen. Jeder dieser Unterbereiche wird innerhalb der Webanwendung vom Benutzer abgefragt. Als UML-Diagramm stellt sich der Ablauf wie folgt dar:

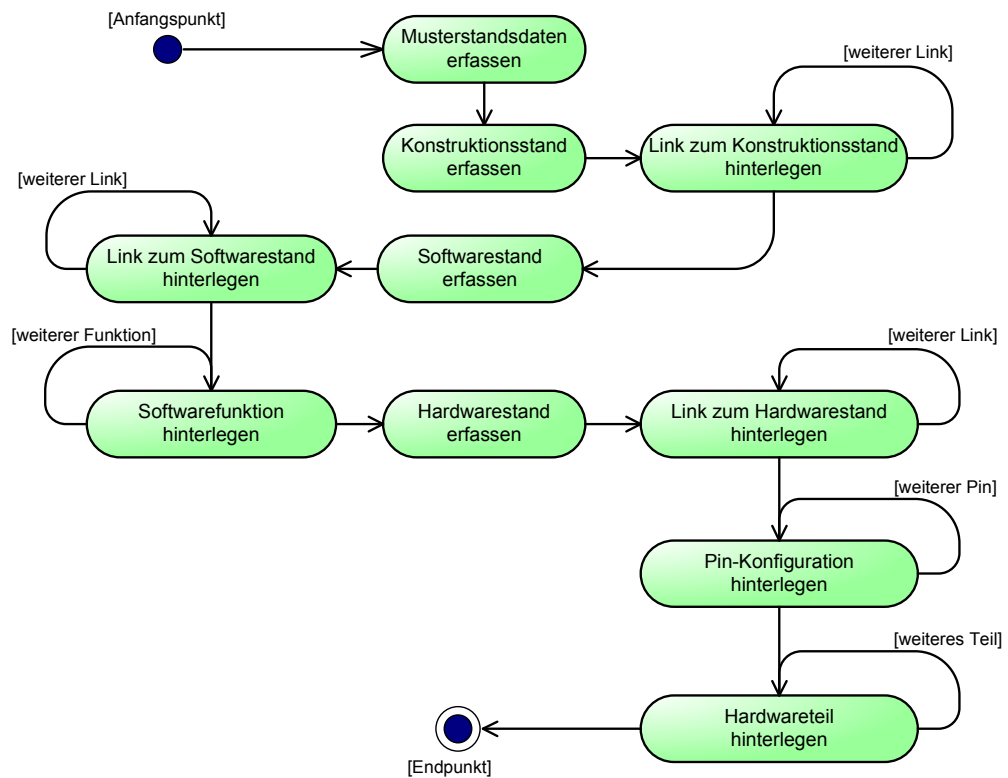


Abbildung 18: UML-Diagramm Musterstanderstellung

Die einzelnen Unterpunkte wie z. B. Links zu externen Dokumenten eines Softwarestandes können selbstverständlich auch später noch innerhalb der Anwendung ergänzt werden. Nähere Information hierzu sind im Abschnitt 4.2.3.2 Bearbeiten von Musterstandsdaten zu finden.

4.1.3 Prototypendaten erfassen

4.1.3.1 Einen neuen Prototypen erfassen

Die eigentliche Aufgabe der Anwendung ist es, laut Definition in der Projektarbeit (PTDB-V01-R001¹²), Prototypen zu dokumentieren. An dieser Stelle kommen wir nun zur Datenerfassung der Prototypen.

Da jeder Prototyp direkt mit einem Projekt, respektive mit einem Musterstand eines Projektes, verbunden ist, muss man im ersten Schritt über die bekannte Projektnavigation das entsprechende Projekt auswählen. Hier kann man nun direkt den be-

¹² [angl]

troffenen Musterstand auswählen oder aber den Prototyp auf Projektebene erfassen und bei der Eingabe den zugehörigen Musterstand auswählen:



Abbildung 19: Projektnavigation inkl. zugehöriger Musterstände

Da ein Projekt üblicherweise mehrere Auftragsnummern umfasst, jeder Prototyp aber eindeutig einem Auftrag zugewiesen werden muss, muss bei der Datenerfassung des Prototyps auch die zugehörige Auftragsnummer hinterlegt werden, was über eine Auswahlliste innerhalb der Maske geschieht. Ist ein Auftrag dem System noch nicht bekannt, so kann er ohne Umstände dem ausgewählten Projekt über eine eigene Maske hinzugefügt werden.

Ein Sonderfall im Bereich der Prototypen ist es, dass ein Prototyp auf einen anderen Prototypen des gleichen oder anderer Projekte verweisen kann. Dieser Verweis (Link) ist dann sinnvoll, wenn ein Klimasteuergerät aus mehreren Teilen besteht. (In der Praxis ist dies der Fall beim Klimasteuergerät des aktuellen Porsche Panamera Projektes). Hierzu ist in der Anwendung eine Suchfunktion hinterlegt, mit derer Hilfe man bereits angelegte Prototypen suchen und finden kann, um diese als Verweis zu hinterlegen. Der gesamte Ablauf der Prototypenerfassung stellt sich somit als UML-Diagramm wie folgt dar:

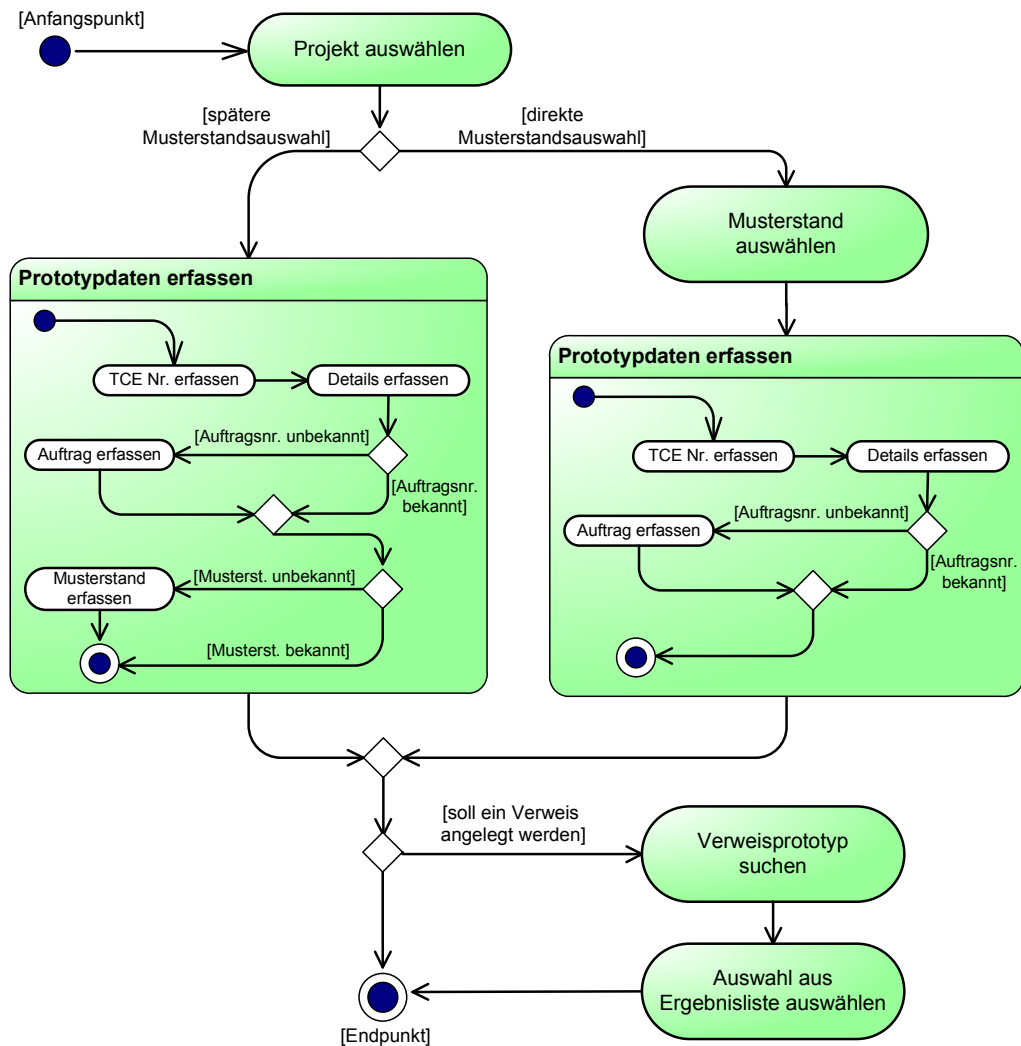


Abbildung 20: UML-Diagramm Prototyperstellung

4.1.3.2 Daten eines existierenden Prototypen kopieren

In der Realität ist es eine gängige Praxis, dass mit einem Auftrag nicht nur ein Prototyp bestellt wird, sondern eine größere Spanne von Prototypen. Im Regelfall umfasst eine Bestellung des Kunden mindestens 15 oder mehr Prototypen. Um hier das Anlegen mehrerer Prototypen in der Datenbank zu vereinfachen, wurde eine Kopierfunktion auf Prototypenebene implementiert.

Über diese Kopierfunktion ist es dem Bearbeiter möglich, einen Ur-Prototyp in der Datenbank zu hinterlegen, der dann vervielfältigt werden kann. Ein besonderes Augenmerk bei der Umsetzung musste hier auf die zu kopierenden Informationen gelegt werden, da nicht alle Informationen auf Prototypenebene allgemeingültig sind und auf Ebene der Prototypen variieren können; andere Informationen wiederum

gelten für alle Prototypen eines Auftrages und können somit kopiert werden. Zu den universellen Daten eines Prototyps gehören zum Beispiel die Referenznr. des Kunden, die Kundenteile-Nr. und die Netzplan-Nr., wobei das Lieferdatum, das Produktionsdatum und die Freigabe-Stati je Prototyp individuell sind. Das Kopieren stellt sich als UML-Diagramm wie folgt dar:

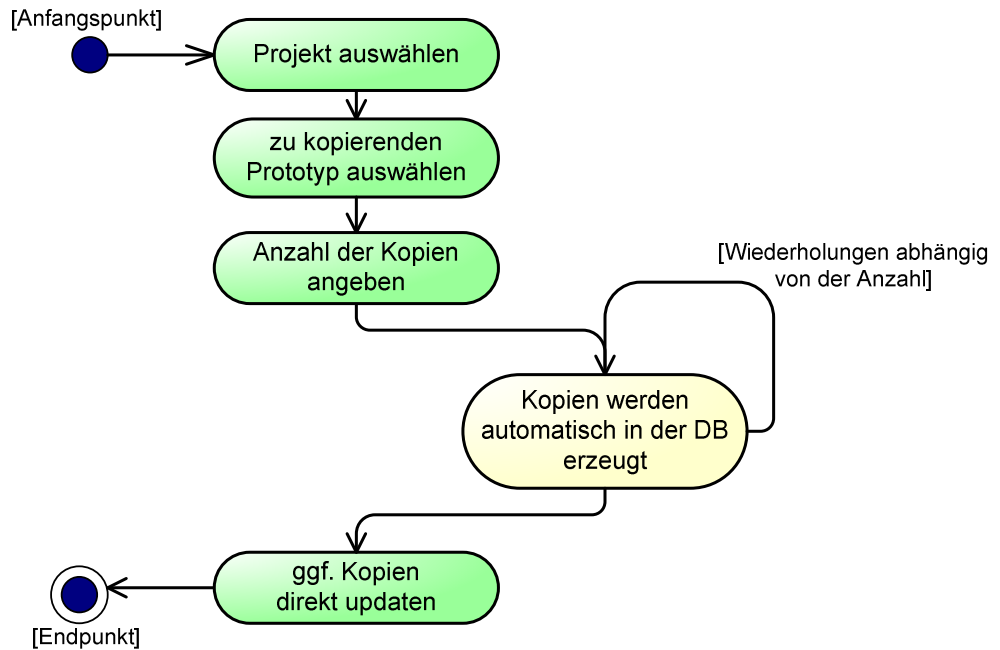


Abbildung 21: UML-Diagramm Prototypen kopieren

4.1.4 Fazit der Dateneingabe

Der gesamte Ablauf der Datenerfassung stellt sich somit wie folgt dar:

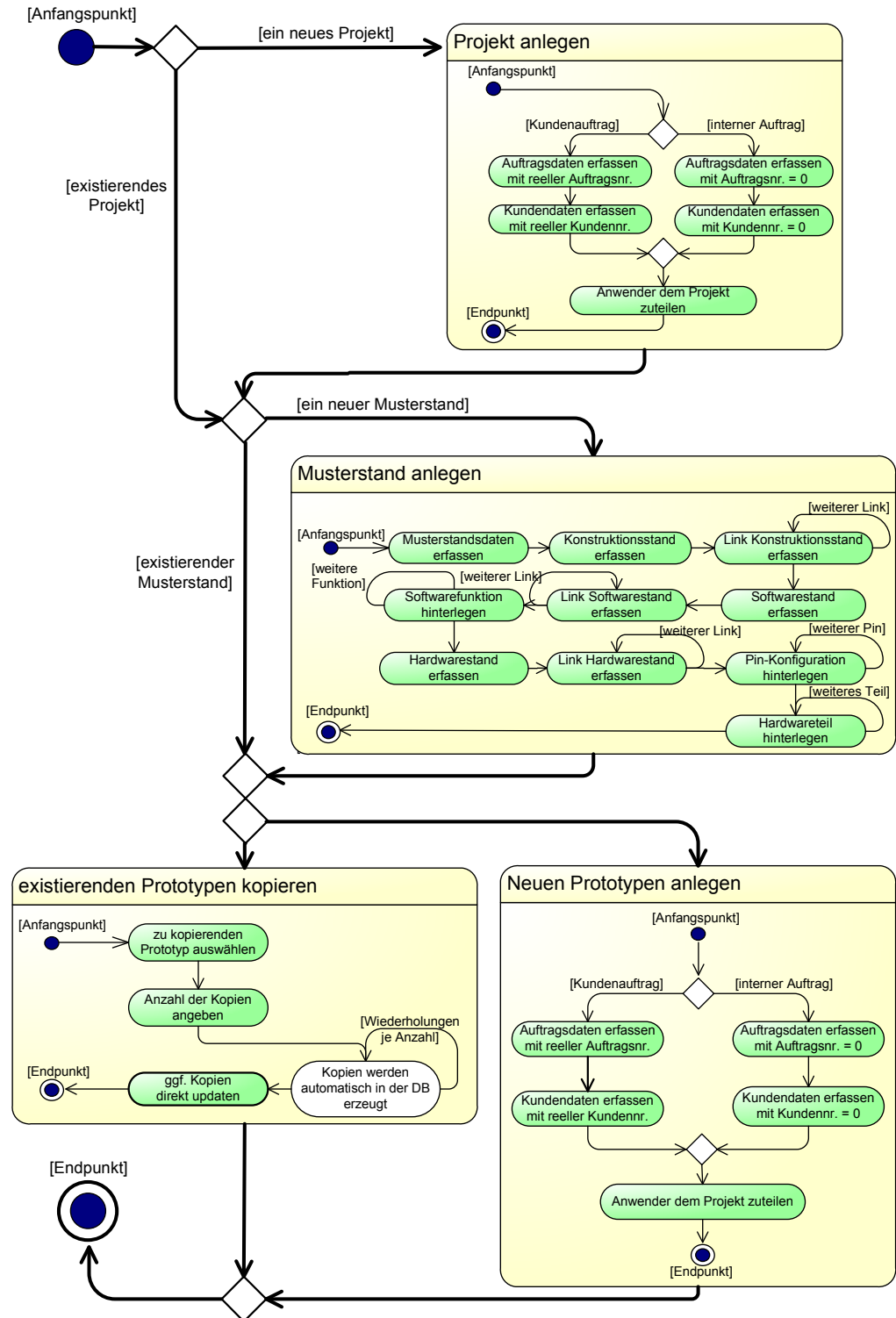


Abbildung 22: UML-Diagramm Ablauf Datenerfassung

Verfolgt man die chronologische Abfolge eines Projektes, von der Erstellung bis zur Erfassung und der Prototypdaten im Einzelnen, so erschließt sich das entwickelte Konzept dem Anwender automatisch durch die gewohnte und bekannte Abfolge. Durch diese Vorgehensweise ist es dem Benutzer der Anwendung ein Leichtes sich innerhalb dieser zurechtzufinden und die korrekten Abläufe intuitiv zu erschließen.

4.2 Die Datenpflege

Da die Datenpflege nicht von einem Anwender sondern von den unterschiedlichsten Anwendern innerhalb der Anwendung vollzogen wird, zum Beispiel dürfen Freigaben von Musterständen nur durch autorisierte Anwender gesetzt werden, muss der Datenpflegebereich separat betrachtet werden.

Weiterhin soll hier ein Augenmerk auf die Navigation innerhalb von Projekten gelegt werden, um darzustellen auf welche Arten man die zu bearbeitenden Daten aufrufen kann.

4.2.1 Navigation innerhalb der Daten

Die Navigation innerhalb der Anwendung zum Abrufen der Daten ist ein zentraler Punkt im Rahmen der Datenpflege. Es sollte für den Benutzer leicht verständlich sein, Daten aufzurufen und zu bearbeiten. Um den Benutzer hier ein intuitives Verständnis der Navigation zu verschaffen, wurde innerhalb der PTDB auf die, aus der Microsoft Windows-Welt bekannten, Navigationselemente gesetzt.

Da die Daten logisch wie ein Baum strukturiert sind (vergleiche auch Abbildung 5 „Die drei Datenfamilien“), wurde die Treeview, wie sie aus dem Windows-Dateiexplorer bekannt ist, als darstellendes Navigationselement ausgewählt. In der ursprünglichen Überlegung sollten alle Daten (also Projekte, Musterstände und angelegte Prototypen) innerhalb der Baumansicht dargestellt werden. Eine Analyse der Laufzeit hat jedoch ergeben, dass diese Darstellungsform nicht praktikabel ist, da sie im Konflikt mit der Benutzbarkeit steht. Testläufe während der Entwicklung haben für unterschiedliche Prototypenanzahlen folgendes Bild ergeben:

Anzahl Prototypen	10	50	100	500	1.000	2.500	5.000
Aufbauzeit der Seite in Millisekunden	15	15	16	28	2468	4103	21670

Tabelle 2: Aufbauzeit bei dargestellten Prototypen*

* Testsystem: AMD Athlon 64bit – 3500+ mit 2,21 Ghz und 2 GB RAM
im SingleUser-Betrieb mit lokaler Anbindung an den Web- und
Datenbanken-Server

Die Tabelle zeigt eindrucksvoll, dass knapp 22 Sekunden benötigt werden, um 5.000 Prototypen innerhalb der Baumansicht einzuordnen und den Baum zu generieren. Es ist weiterhin klar zu erkennen, dass die benötigte Zeit mit weiteren Prototypen steigen wird, was auch im folgenden Diagramm abzulesen ist:

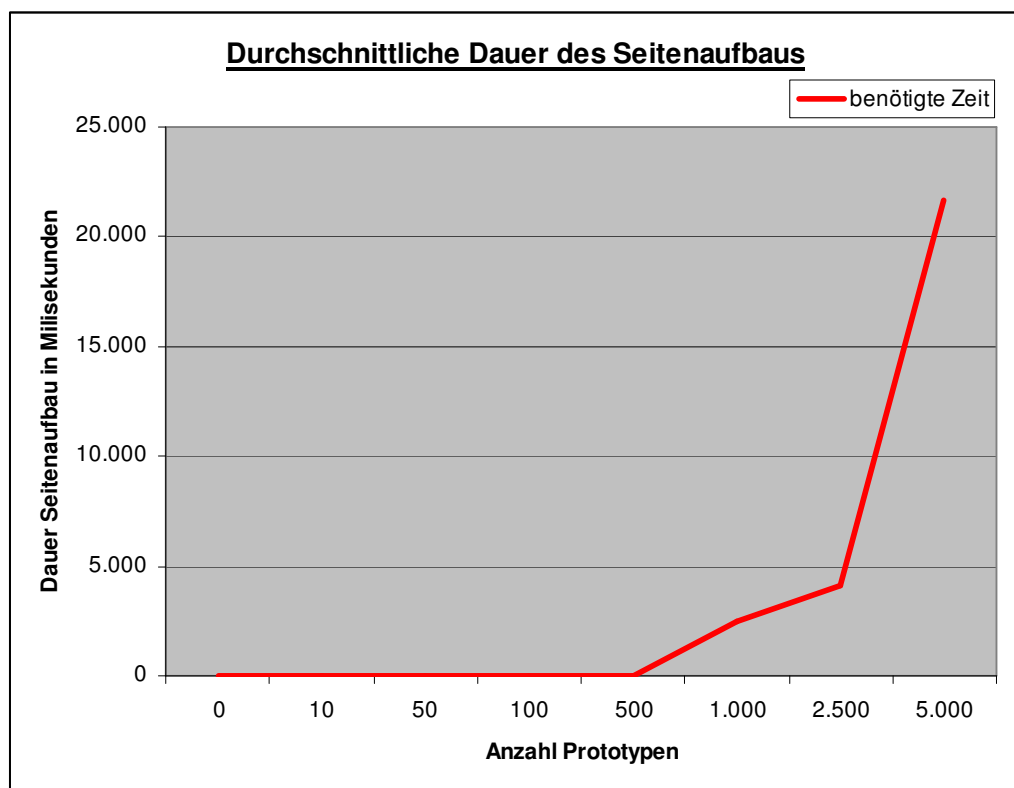


Abbildung 23: Durchschnittliche Dauer des Seitenaufbaus vor der Optimierung

Eine weitergehende Untersuchung der Laufzeit hat ergeben, dass die benötigte Zeit für diese Darstellungsform nicht weiter optimierbar ist, da sie nicht anhand der Datenbankzugriffe oder anhand des Quellcodes optimiert werden kann.

Die Ursache der Laufzeit lässt sich mit der Dateigröße der generierten HTML-Seite erklären.

Anzahl Prototypen	10	50	100	500	1.000	2.500	5.000
Dateigröße der Seite in Kilobyte	94	108	224	758	1396	3367	6630

Tabelle 3: Dateigröße bei dargestellten Prototypen

Bereits bei 5.000 Prototypen beträgt die Dateigröße über 6 Megabyte. Auch hier ist das stetige Wachstum der Dateigröße bereits an der Tabelle abzulesen, was sich im Diagramm noch deutlicher darstellt:

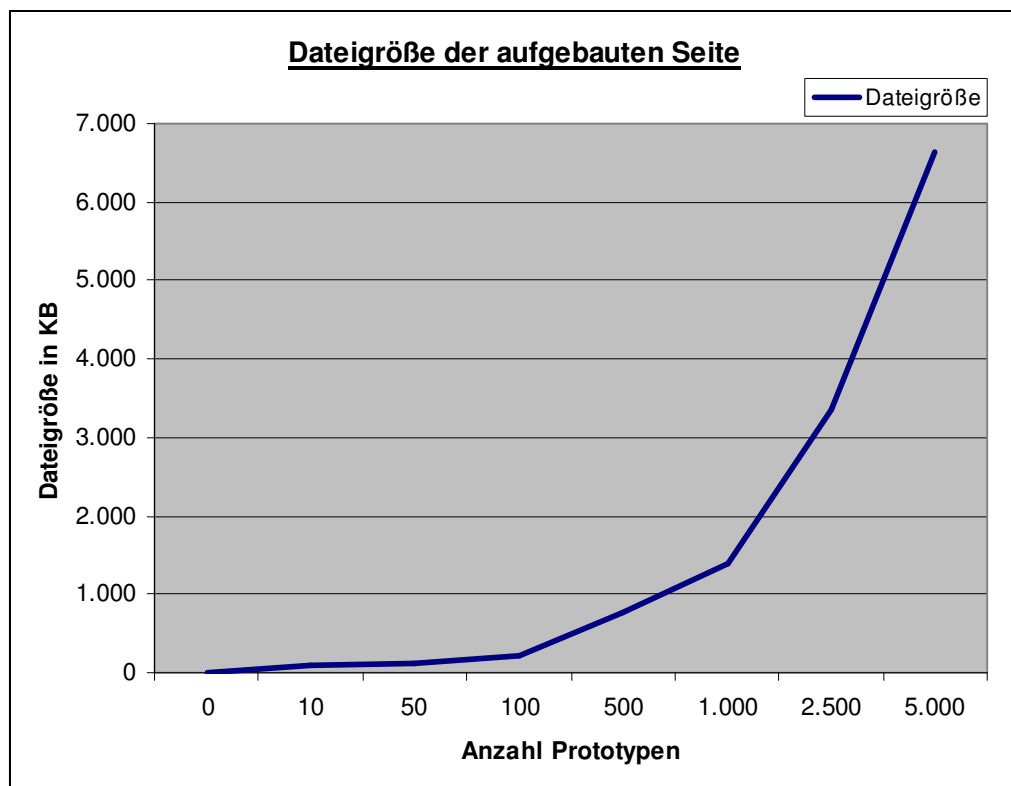


Abbildung 24: Dateigröße der aufgebauten Seite vor der Optimierung

Da eine 6 MB große Datei sowohl auf Seiten des Servers Systemressourcen beansprucht, als auch die Übermittlung der generierten Daten Ressourcen im Netzwerk beanspruchen, musste hier eine Lösung gefunden werden, die Dateigröße zu minimieren. Eine Auswertung der historischen Projektdaten auf Seiten der BHTC hat ergeben, dass ein Projekt im Regelfall zwischen 4 und maximal 20 Musterständen enthält, wobei die Prototypenanzahl auf Projektebene zwischen 250 und 8.000 Prototypen variieren kann. Die einzige Optimierung, die also seitens der PTDB möglich war, war ein Ausschließen der Prototypen aus der Baumansicht. Schließt man die Prototypen aus der Treeview aus, so stellt sich die Analyse wie folgt dar:

Anzahl Prototypen	10	50	100	500	1.000	2.500	5.000
Aufbauzeit der Seite in Millisekunde	15	15	15	15	15	15	15

Tabelle 4: Aufbauzeit bei ausgeschlossenen Prototypen

Die Dateigröße der generierten HTML-Seite wächst bei dieser Lösung auch trotz zusätzlicher Prototypen nur unwesentlich:

Anzahl Prototypen	10	50	100	500	1.000	2.500	5.000
Dateigröße der Seite in Kilobyte	93	94	94	94	95	95	96

Tabelle 5: Dateigröße bei ausgeschlossenen Prototypen

Sowohl das Diagramm der Aufbauzeit

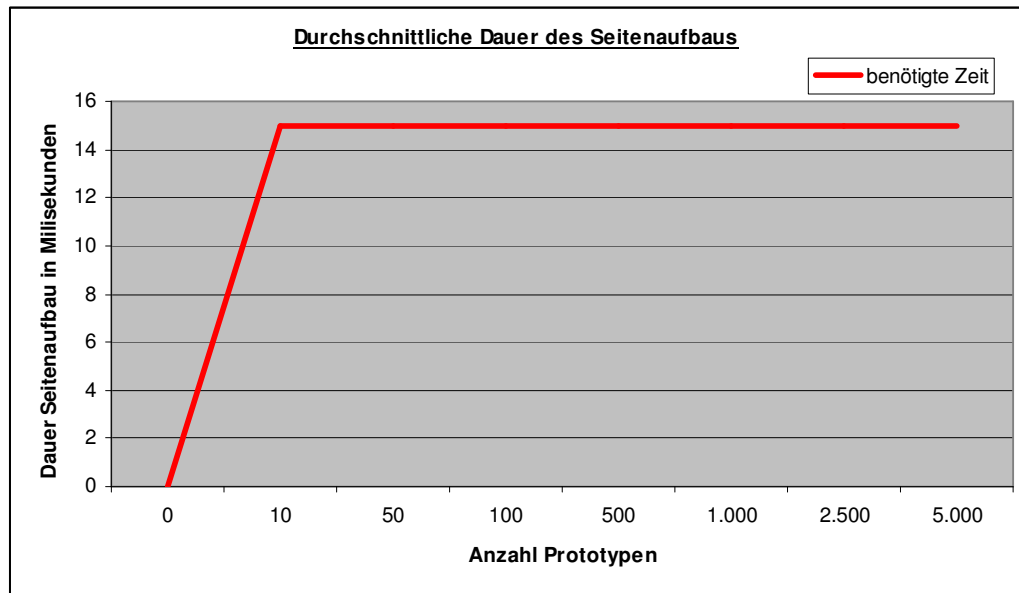


Abbildung 25: Durchschnittliche Dauer des Seitenaufbaus nach der Optimierung

als auch das Diagramm der Dateigröße

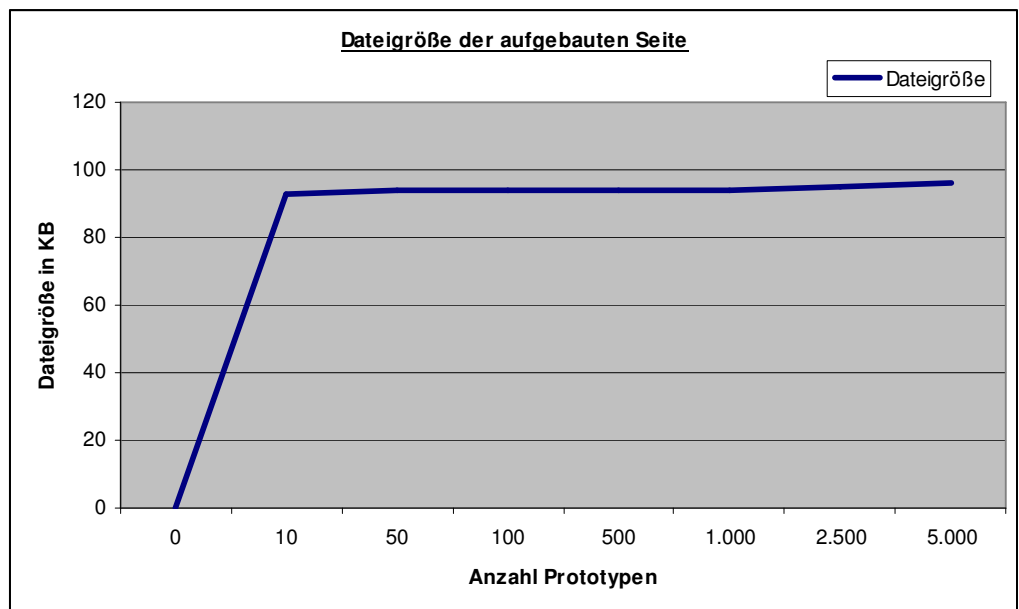


Abbildung 26: Dateigröße der aufgebauten Seite nach der Optimierung

zeigt, dass es nur ein unwesentliches Wachstum gibt, was eine Optimierung der Anwendung zu hundert Prozent darstellt.

Da nun das Aufrufen der angelegten Prototypen nicht mehr durch die Baumansicht erfolgen kann, wurde hier ein weiteres Element zur Navigation eingefügt. Auch das Navigationselement für die Prototypen lässt sich vom Anwender intuitiv benutzen, da es einem Bedienelement aus dem Alltag entliehen ist.



Abbildung 27: Navigationselement für Prototypen

Das Steuerelement entspricht in seiner Funktion und in seinem Design dem eines CD- oder MP3-Abspielgerätes. Es gibt jeweils einen Button um an den Anfang oder an das Ende aller Prototypen eines Projektes zu springen und jeweils einen Button, um einen Prototypen weiter oder zurück zu springen. Weiterhin ist es für den Anwender möglich, direkt die aufzurufende Prototypen-Nummer einzugeben.

Innerhalb eines Projektes stellt sich somit die Navigation mit Hilfe der Navigations-Knöpfe wie folgt dar:

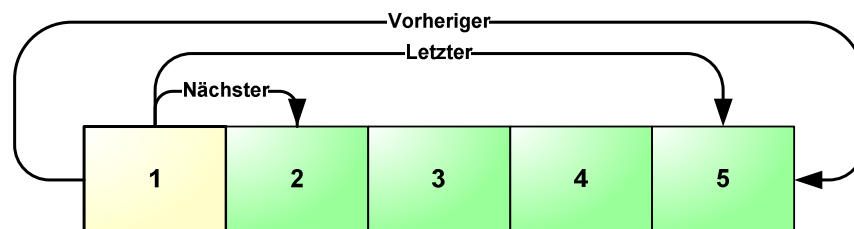


Abbildung 28: Prototypenwahl via Navigations-Buttons am Anfang

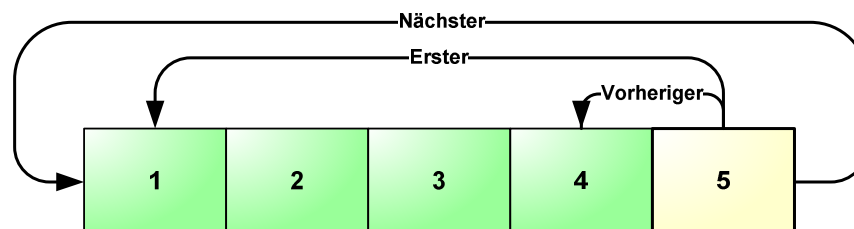


Abbildung 29: Prototypenwahl via Navigations-Buttons am Ende

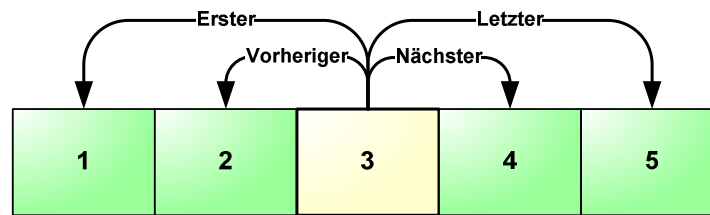


Abbildung 30: Prototypenwahl via Navigations-Buttons in der Mitte

Somit wurde dem Anwender ein Steuerelement an die Hand gegeben, das er leicht verstehen und ebenso leicht einsetzen kann.

Um einen definierten „Urzustand“ innerhalb der Navigation zu haben, wurde im Programmcode festgelegt, dass bei der Wahl „Erster Prototyp“ oder „Vorheriger Prototyp“ immer der Prototyp 1 des Projektes angezeigt wird, wenn vorher kein Prototyp explizit ausgewählt wurde. Der Urzustand für „Letzter Prototyp“ und „Nächster Prototyp“ innerhalb eines Projektes ist der letzte Prototyp (mit der höchsten Prototypen-Nummer).

Nachdem nun das Laufzeitproblem gelöst wurde, ist dem Anwender durch die zwei bekannten Navigationselemente (Baumansicht für Projekte und Musterstände sowie CD-Abspielknöpfe für die Prototypenauswahl) eine einfache Handhabung an die Hand gegeben, um die zu bearbeitenden Daten aufzurufen.

4.2.2 Pflege der Projektdaten

Im Regelfall sollten sich die Stammdaten der Projekte im Verlauf der Bearbeitung eigentlich nicht ändern. Da es aber in der Realität durchaus vorkommen kann, dass eine interne oder externe Projektbezeichnung geändert werden muss oder sich die zuständigen Mitarbeiter im Bereich der Konstruktion, Entwicklung oder Überprüfung ändern, wurden dem Anwender diverse Masken zum Aktualisieren dieser Daten bereitgestellt. Weiterhin muss es eine Möglichkeit geben zusätzliche Aufträge zu erfassen oder Bearbeiter einem Projekt hinzuzufügen. Grob kann man also die Datenpflege auf Projektebene wie folgt untergliedern:

- I. Pflege der Projekt-Stammdaten
- II. Pflege der Projekt-Berechtigungen
- III. Hinzufügen von weiteren Aufträgen
- IV. Hinzufügen von Informationen zum Projekt
- V. Hinzufügen von Musterständen
- VI. Hinzufügen von Prototypen

Die Abschnitte V. und VI. sind zwar der Projektdatenpflege zuzuordnen, jedoch kann man sie auch als eigenständige Prozessabläufe betrachten. Aus Übersichtlichkeitsgründen werden somit die Punkte V. und VI. in separaten Abschnitten erläutert (siehe hierzu auch Abschnitt 4.2.3 Musterstände bearbeiten und 4.2.4 Prototypen bearbeiten).

4.2.2.1 Pflege der Projekt-Stammdaten

Man kann innerhalb der PTDB die Projekt-Stammdaten aktualisieren, in dem man aus der Baumansicht das betroffene Projekt auswählt, um anschließend die angezeigten Daten zu bearbeiten.

Projekt: _ Sandbox Project 1			
Musterstand: -			
Prototypennr.: -			
Projekt	Musterklasse	Prototyp	Information zum Projekt
Interne Projektbezeichnung	_ Sandbox Project 1		
externe Projektbezeichnung	Synonym Project 4		
Beschreibung Projekt	Testprojekt 4 - Development Environment		
BHTC Teilernr.	5HB009461- 4		
Ausführender Konstruktion	Carl Constructor		
Ausführender Entwicklung	Den Developer		
Ausführender Tester	Ethan Engineer, Trevor Tester		
Teilebenennung	Teilebenennung - TPDEV4		
<input type="button" value="Bearbeiten"/>			

Abbildung 31: Änderungen am Projekt im Read-Only-Modus

Damit für den Bearbeiter klar zu erkennen ist, in welchem Modus er sich befindet (Lese-Modus oder Update-Modus), sind die unterschiedlichen Masken farblich von einander abgehoben. Diese farbliche Unterscheidung ist in der gesamten Anwendung umgesetzt, damit der Anwender an jeder Stelle klar erkennen kann, in welchem Modus er sich befindet.

Projekt: _ Sandbox Project 1			
Musterstand: -			
Prototypennr.: -			
Projekt	Musterklasse	Prototyp	Information zum Projekt
Interne Projektbezeichnung	_Sandbox Project 1		
externe Projektbezeichnung	Synonym Project 4		
Beschreibung Projekt	Testprojekt 4 - Development Environment		
BHTC Teilernr.	5HB009461- 4		
Ausführender Konstruktion	Carl Constructor		
Ausführender Entwicklung	Den Developer		
Ausführender Tester	Ethan Engineer, Trevor Tester		
Teilebenennung	Teilebenennung - TPDEV4		
Aktualisieren			
Abbrechen			

Abbildung 32: Änderungen am Projekt im Update-Modus

Weiterhin wird dem Bearbeiter im Kopfbereich der Maske das gewählte Projekt, der ausgewählte Musterstand und ggf. der ausgewählte Prototyp angezeigt. Nachdem der Anwender die betroffenen Daten in der Maske aktualisiert hat, kann er die Daten via „Aktualisieren“ permanent in der Datenbank speichern oder via „Abbrechen“ die Änderungen verwerfen. Als UML-Diagramm würde sich der Änderungsprozess somit wie folgt darstellen:

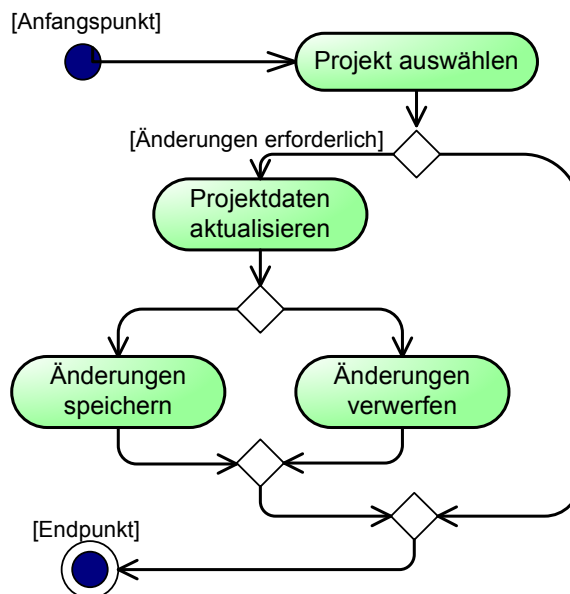


Abbildung 33: UML-Diagramm Projektänderungen

4.2.2.2 Pflege der Projekt-Berechtigungen

Zu Beginn eines Projektes, also beim Erstellen eines Projektes, wird festgelegt, welche Mitarbeiter an diesem Projekt beteiligt sind und welche Berechtigungen sie innerhalb des Projektes besitzen. Nun kann es in der Praxis durchaus vorkommen, dass Mitarbeiter aus dem Projekt abgezogen werden oder andere Mitarbeiter dem Projekt zugewiesen werden. Hierzu werden dem Supervisor Masken zur Verfügung gestellt, um die Projektbenutzer zu bearbeiten. Der Prozessablauf hierfür stellt sich wie folgt dar:

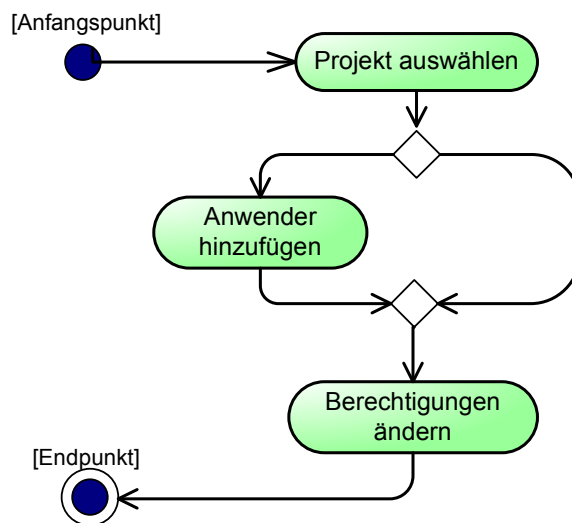


Abbildung 34: Anwender hinzufügen

Es ist also auf Projektebene möglich, Anwender hinzuzufügen oder zu entfernen und den dem Projekte zugewiesenen Anwendern Berechtigungen zu erteilen oder wieder zu entziehen.

4.2.2.3 Hinzufügen von weiteren Aufträgen

Da zum Zeitpunkt der initialen Projekterstellung naturgemäß nicht alle zukünftigen Aufträge bekannt sind, kann man zu jedem Zeitpunkt weitere Aufträge erfassen. In der Auftragserfassungsmaske werden nur die rudimentären Daten abgefragt und gespeichert, da eine ausführliche Auftragsaufstellung in SAP vorgehalten wird.

Abbildung 35: zusätzlichen Auftrag erfassen

Sobald ein Auftrag im System hinterlegt wurde, steht er dem Anwender in der Maske zur Prototypenerfassung zur Verfügung.

Abbildung 36: Auftragsauswahl bei der Prototypenerfassung

4.2.2.4 Hinzufügen von Projekt-Informationen

Welche Projekt-Informationen, die nicht zu den technischen oder kundenspezifischen Informationen gehören, sollten dennoch in der Datenbank hinterlegt werden?

Um diese Frage zu klären, müssen wir als erstes klären, was man unter „Information“ versteht. Für die Informationswissenschaft sind die Begriffe „Wissen“ und „Information“ von zentraler Bedeutung. Information ist dabei der Wissenstransfer, beziehungsweise „Wissen in Aktion“. Information entsteht in diesem Sinne immer nur punktuell, wenn ein Mensch zur Problemlösung Wissen (eine bestimmte Wissenseinheit) benötigt. Diese Wissenseinheit geht als Information aus einem Wissensvorrat in einen anderen über, beispielsweise aus einer Datenbank in den Wissensvorrat eines Menschen. Wissen wird intern repräsentiert, Information wird - zum besseren Verständnis für den Informationssuchenden - präsentiert.¹³

¹³ [wiki2]

Unter diesem Ansatz kann man also als Projekt-Informationen Daten und Texte bezeichnen, die während einer Problemlösung innerhalb eines Projektes angefallen sind. Sie können dazu dienen, zukünftige Probleme innerhalb des Projektes schneller zu lösen oder auf eventuelle Gefahren innerhalb eines Projektes hinzuweisen, um diese frühzeitig zu vermeiden.

Um das Gesamtkonzept der Anwendung, das auf die intuitive Erschließung der Programmfunktionen und das Verbinden der Datendarstellung mit Alltagssituationen setzt, beizubehalten, wurden die zusätzlichen Projektinformationen für den Anwender logisch unterteilt. So kann er die Informationen analog zu einer Ampel mit den Farben grün, gelb und rot markieren, wobei eine grüne Kennzeichnung Unbedenklichkeit bedeutet und eine rote Kennzeichnung ein Warnsignal kennzeichnet. Eine Projektinformation kann weiterhin den Status „offen“, „in Bearbeitung“ oder „erledigt“ haben, was innerhalb des Anzeigebereiches durch ein analoges Voltmeter dargestellt wird.

Die Art der Projekt-Zusatzinformation wird unterschieden in „Informationen“, „Kommentar“, „Hinweis“, „Aufgabe“, „Risiko“ und „Warnung“, was in der Anzeige der bereits erfassten Informationen jeweils durch ein eigenes Symbol dargestellt wird.

Kennzeichnung:

Informationsstatus:

Art der Information:

Informationstext:







	geändert am	geändert durch	Informationstext	Art	Flag	Status
<input type="button" value="Bearbeiten"/> <input type="button" value="Löschen"/>	24.11.2007 16:54:09	ANDRE.GRAHL	Bitte den Kunden kontaktieren, um die Details zu Auftrag 0815 zu klären und Missverständnisse zu vermeiden!			
<input type="button" value="Bearbeiten"/> <input type="button" value="Löschen"/>	24.11.2007 16:54:52	ANDRE.GRAHL	Durch die Rückversicherung der Farbauswahl mit dem Kunden, konnten Fehlproduktionen vermieden werden.			

Abbildung 37: Zusätzliche Projektinformationen

4.2.2.5 Löschen von Daten innerhalb eines Projektes

Da die Prototypendatenbank sowohl zur internen Dokumentation als auch zu Revisionszwecken eingesetzt wird, ist es nicht vorgesehen, dass der Anwender Daten innerhalb der Anwendung löschen kann. Zu einem späteren Zeitpunkt kann man eventuell via Change-Process ein Archivierungsmodul entwickeln, um bereits abgeschlossene Projekte zu archivieren und innerhalb einer getrennten Ebene darzustellen.

4.2.3 Musterstände bearbeiten

Die unterschiedlichen Musterstände innerhalb eines Projektes spiegeln die unterschiedlichen Generationen eines Prototyps wieder, wobei ein „A-Musterstand“ den ersten Entwicklungszustand angibt und ein „Erstmusterstand“ die Beschreibung des finalen Produktionsstandes wiedergibt. Jeder Musterstand wiederum setzt sich aus einer allgemeinen Beschreibung und den Beschreibungen der Konstruktions-, der Software- und der Hardwarestände zusammen.

Es muss also innerhalb der PTDB möglich sein, weitere / zusätzliche Musterstände zu erfassen sowie bestehende Musterstände zu aktualisieren.

4.2.3.1 Hinzufügen von weiteren Musterständen

Wie man erkennen kann setzt sich ein Musterstand logisch aus mehreren Unterständen zusammen und als Modul-Schaubild stellt sich der Musterstand somit wie folgt dar:

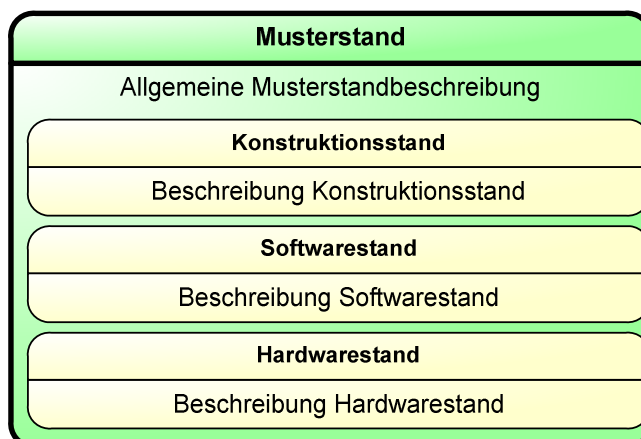


Abbildung 38: Musterstand-Schaubild

Die einzelnen logischen Unterstände werden auch einzeln durch die Anwendung abgefragt, was dem Anwender bereits durch das Anlegen eines neuen Projektes (vergleiche auch Abbildung 18: UML-Diagramm Musterstanderstellung) bekannt ist.

4.2.3.2 Bearbeiten von Musterstandsdaten

Existiert ein Musterstand im System, so muss es für den Bearbeiter möglich sein, diesen zu bearbeiten und Informationen auf jeder Ebene zu ändern oder hinzuzufügen. Voraussetzung für die Bearbeitung ist jedoch, dass der Musterstand noch nicht von den zuständigen Abteilungen geprüft und freigegeben wurde. Das UML-Diagramm der Musterstandsbearbeitung stellt sich wie folgt dar:

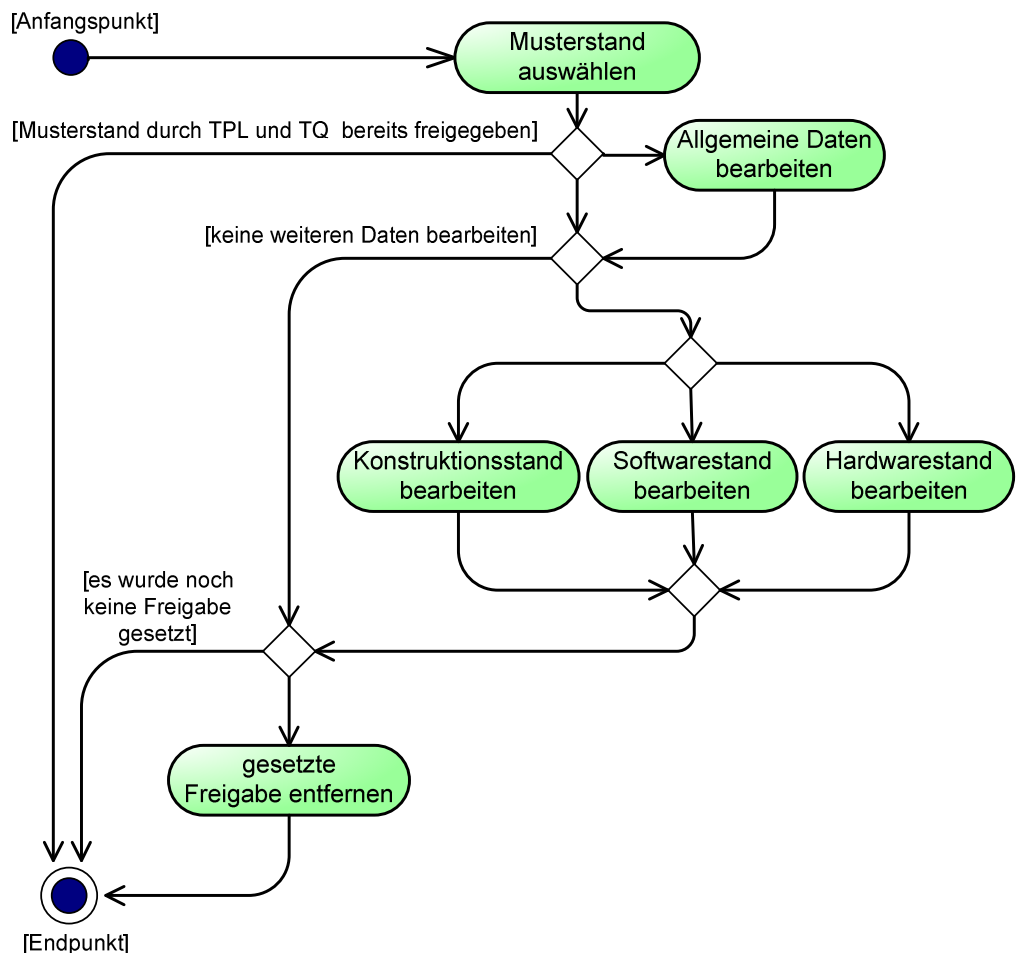


Abbildung 39: Musterstand bearbeiten

Sollte der Musterstand zum Zeitpunkt der Bearbeitung nur von einer Abteilung freigegeben sein (TPL oder TQ), so ist es möglich diesen zu bearbeiten. Eine

Bearbeitung verursacht jedoch das Entfernen der bereits erteilten Freigabe, da die ursprüngliche Freigabe auf einen anderen Stand beruht hat.

4.2.3.3 Freigabe eines Musterstandes

Jeder Musterstand muss durch einen technischen Qualitätsmanager und durch einen technischen Projektleiter auf korrekte Definition und korrekte Umsetzung geprüft werden. Hierzu muss innerhalb des Projektes einem Anwender die Berechtigung „TQ“ (technischer Qualitätsmanager) und einem Anwender die Berechtigung „TPL“ (technisches Prüflabor) erteilt werden (siehe hierzu auch 4.2.2.2 Pflege der Projekt-Berechtigungen). Hat ein Benutzer die entsprechende Berechtigung, so kann er einen Musterstand freigeben. Sobald ein Musterstand durch beide Bereiche freigegeben wurde, kann er nicht mehr verändert werden, da sonst die Freigabe auf einem anderen Stand basiert, als er im System erfasst wurde.

Um einen Musterstand auf Fehlerfreiheit überprüfen zu können, muss ein Prototyp zu diesem Musterstand existieren. Dieser erste produzierte Prototyp wird auch Referenztyp genannt, da er als Referenz für den Musterstand dient. Existiert im System noch kein Referenztyp, so muss er bei der entsprechenden Abteilung angefordert werden. Das Fehlen des Referenzprototypen ist programmtechnisch nicht abgefangen, da die BHTC Sonderfälle dargestellt hat, bei denen eine Musterstands freigabe auch ohne existierenden Referenztypen möglich sein muss. Die gesamte Musterstands freigabe als UML-Diagramm ist auf der folgenden Seite dargestellt.

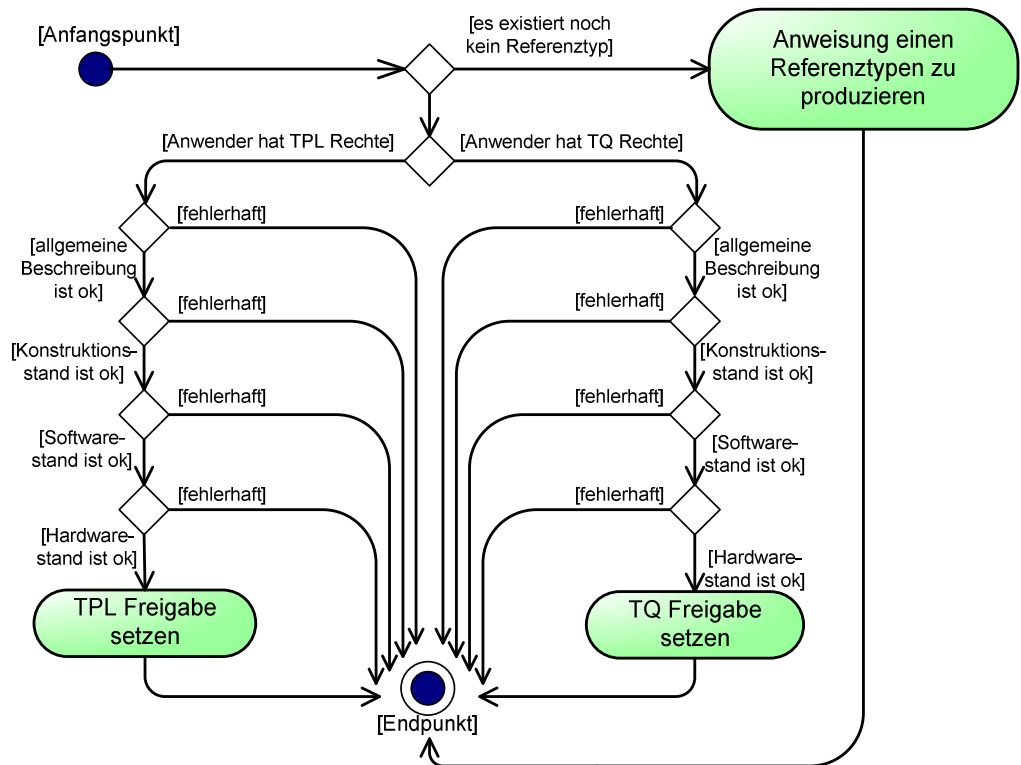


Abbildung 40: Musterstand-Freigabe

Nachdem ein Musterstand durch beide Prüfungsabteilungen freigegeben wurde, wird es dem Benutzer durch ein besonderes Symbol innerhalb der Baumansicht angezeigt (🟢). Ein ungeprüfter Musterstand wird durch ein eigenes Symbol erweitert (🟡). Durch diese Art der Darstellung kann der Anwender bereits bei der Projekt- / Musterstandauswahl erkennen, ob eine Freigabe erteilt wurde oder nicht. Somit kann sich die Treeview für den Anwender wie folgt darstellen:



Abbildung 41: Treeview mit geprüften Musterständen

Im Beispiel wurden die Musterstände A-1 und B-1 bereits durch die Abteilungen TPL und TQ freigegeben, wobei die Musterstände A-2 und B-2 noch nicht oder nur teilweise freigegeben wurden.

4.2.4 Prototypen bearbeiten

Nachdem die Prototypen wie unter „4.1.3 Prototypendaten erfassen“ innerhalb der Datenbank angelegt wurden, werden im Verlauf des Projektes nach und nach weitere Informationen zu den einzelnen Prototypen erfasst. Es kann sich hierbei sowohl um genauere Details zu den einzelnen Prototypen handeln, als auch um Prüfungen oder Kommentaren zu den Prototypen. Hat ein Prototyp alle Abteilungen durchlaufen und wurden alle notwendigen Informationen erfasst, so kann er vergleichbar mit der Musterstands freigabe (siehe auch 4.2.3.3 Freigabe eines Musterstandes) als geprüft gekennzeichnet werden.

4.2.4.1 Pflege der Prototypendaten

4.2.4.1.1 Prototypen-Stammdatenpflege

Unter der Prototypen-Stammdatenpflege wird die Pflege der rudimentären Daten verstanden. Hierzu wird dem Anwender in der Darstellungsmaske ein „Bearbeiten“-Button angezeigt solange der Prototyp noch nicht freigegeben wurde (siehe auch 4.2.4.2 Freigabe eines Prototypen).

TZ Freigabe gesetzt durch	
TZ freigegeben	<input type="checkbox"/>
TZ Freigabe gesetzt am	
TZ Freigabe gesetzt durch	
Prototypenprüfung	Prototypen:
Kommentar zum Prototyp	Kommentar

Bearbeiten

TPL-Freigabe setzen TZ-Freigabe setzen

Abbildung 42: Bearbeiten-Knopf der Prototypendatenpflege

Ist der Prototyp bereits freigegeben, so darf er nicht mehr verändert werden, da sich sonst die Prüfung auf veränderten Daten berufen würde, daher wird der Bearbeiten-Knopf bei freigegebenen Prototypen nicht angezeigt. Wurde bisher nur eine unvollständige Freigabe getätigt (d.h. es wurde noch nicht von allen notwendigen Abteilungen eine Freigabe erteilt), so werden die bereits gesetzten Freigaben wieder

aus dem System entfernt, sobald eine Änderung an den Daten vorgenommen wurde.

Eine Besonderheit bei der Prototypenstammdatenpflege stellt der Verweis auf andere Prototypen dar. Hier muss innerhalb der Datenbank der Index des verlinkten Prototypen gespeichert werden, der jedoch dem Anwender nicht bekannt und nicht intuitiv erschließbar ist. Um dieses Dilemma elegant zu lösen, wurde an der Stelle des Prototypenverweises eine Suchmaske implementiert, über die der Anwender den zu verlinkenden Prototypen suchen kann.

Prototypen-Suche

Filter: SAP Kundennummer: ohne Einschränkung
 Projekt: ohne Einschränkung
 Musterstand: ohne Einschränkung

Art der Suche: exakter Treffer
 Teil des Feldinhaltes

Suche in: lfd. Prototyp-Nr. Teileanhänger Kommentar
 TCE-Nr. Kundenteile-Nr.
 Referenznummer (Kunde) Netzplan Nr.
 Verwendungszweck

suchen nach: 12

	PROTOTYPE_ID	prototype_no	name	tce_no
Auswählen	109	10	VW Passat	tce 123 tce 123 tce 123 tce 12
Auswählen	111	12	VW Passat	TCE (30.07.) 11111111111111111111
Auswählen	1159	12	Sandbox Project 2	1
Auswählen	37	12	Porsche Panamera Climatron	TCE2
Auswählen	52	12	VW Phaeton	TCE TEST
Auswählen	96	12	VW Golf 5	cded

Prototyp übernehmen

Abbildung 43: Suche Prototyp-Link

Über diese Suchmaske ist es dem Anwender nun möglich auf einfache Weise den zu verlinkenden Prototypen auszuwählen und zu verknüpfen ohne dass er die interne Struktur (oder den internen Datenbank-Index) des Prototypen kennen muss.

4.2.4.1.2 Prototypen-Kommentare erfassen

Unter einem Kommentar versteht man im Allgemeinen die schriftlich oder mündlich kund getane Meinung zu einem vorgegebenen Sachverhalt.¹⁴

Bezieht man diese Beschreibung nun auf die Prototypendatenbank, so ist ein Kommentar jegliche Anmerkung zu einem Prototypen, die ein Anwender dokumentiert. Hierbei sind die Kommentare zu unterscheiden in Anmerkungen interner und externer Natur, d.h. Anmerkungen, die nur für die BHTC relevant sind und Anmerkungen, die auch der Kunde einsehen darf.

Zur Erfassung dieser Kommentare steht dem Benutzer innerhalb der Anwendung eine eigene Eingabemaske zur Verfügung.

ID	Anwendername	Datum	Art der Information	Kommentar
70	ANDRE.GRAHL	08.12.2007 17:34:23	E	Prototype wurde umlackiert
69	ANDRE.GRAHL	08.12.2007 17:34:02	I	der Prototyp musste nachjustiert werden
68	ANDRE.GRAHL	08.12.2007 17:03:52	A	TPL Freigabe gesetzt durch ANDRE.GRAHL

Abbildung 44: Kommentar zum Prototyp erfassen

Wie man in Abbildung 44 erkennen kann, gibt es einen zusätzlichen Kommentar-Modus, den der Anwender nicht setzen kann. Hierbei handelt es sich um den Modus „A“ wie Anwendungskommentar. Dieser kennzeichnet automatisch gesetzte Kommentare der Anwendung, wie z. B. erfolgte/gesetzte Freigaben.

¹⁴ [wiki4]

4.2.4.1.2.1 Prototypen-Prüfungen erfassen

Jeder produzierte Prototyp durchläuft innerhalb der BHTC mehrere Prüfungen, die innerhalb der Datenbank erfasst und dokumentiert werden. Hierbei gibt es zwei Methoden, wie die erfolgten Prüfungen erfasst werden können. Die manuelle Erfassung erfolgt durch den Anwender via PTDB-Anwendung und bei der automatischen Erfassung werden die Prüfprotokolle automatisch durch ein Interface in die Datenbank geschrieben.

4.2.4.1.2.2 Manuelle Erfassung anhand der PTDB

Bei der manuellen Erfassung der Prüfung kann der Mitarbeiter, der die Prüfung durchgeführt hat, innerhalb der Datenbank dokumentieren, dass eine Prüfung stattgefunden hat. Hierzu wird ihm eine Maske zur Verfügung gestellt, in der er die folgenden Daten erfassen kann:

- Grund der Prüfung
- Datum der Prüfung
- M-Protokoll
- Test-Protokoll
- Lap-Nr.
- Status der Prüfung

Diese Informationen sollen nur einen groben Überblick über die erfolgte Prüfung geben. Nähere Informationen zur Prüfungen können dann anhand der BHTC-internen Dokumente „M-Protokoll“ und „Test-Protokoll“ nachgeschlagen werden. Innerhalb der Anwendung ist es nicht vorgesehen, bereits eingetragene / erfasste Prüfungen zu ändern oder zu entfernen, was aus Gründen der Auditierbarkeit zwingend erforderlich ist.

ID	Prüfungsgrund	Prüfungsdatum	M-Protokoll	Test-Protokoll	LAP Nr.	Status	eintragen
27	regular prototype audit	23.07.2007 22:01:52	M-protocoll 123	T-P 123abc654	1	n.i.o.	AND
26	regular prototype audit	23.07.2007 17:48:37	M-protocoll 123	T-P 123abc654	22	b.i.o.	AND
25	regular prototype audit	23.07.2007 17:48:28	M-protocoll 123	T-P 123abc654	555444333	i.o.	AND
24	regular prototype audit	23.07.2007 17:48:01	M-protocoll 123	T-P 123abc654	555444333	i.o.	AND

Abbildung 45: Prototypprüfung Eingabemaske

4.2.4.1.3 Automatische Erfassung via Interface

Im Rahmen der Anforderungsanalyse wurde festgelegt, dass keine Schnittstellen oder Interfaces in andere Systeme entwickelt werden. Dieser Punkt nun weicht die ursprüngliche Anforderung geringfügig auf, denn es wurde innerhalb der Entwicklungsphase, unter Zusammenarbeit mit dem zuständigen Mitarbeiter im Prüflabor, eine Schnittstelle in das System entwickelt. Diese Schnittstelle ist nur im weitesten Sinne eine Schnittstelle, denn per Definition ist eine Softwareschnittstelle ein logischer Berührungspunkt in einem Softwaresystem. Sie definiert, wie Kommandos und Daten zwischen verschiedenen Prozessen und Komponenten ausgetauscht werden. Dabei unterscheidet man Schnittstellen zum Zugriff auf Systemroutinen, zur Kommunikation mit anderen Prozessen und zum Verbinden einzelner Softwarekomponenten (Module) eines Programms bzw. programmübergreifende Schnittstellen.¹⁵

Bei der entwickelten Schnittstelle werden jedoch weder Systemroutinen noch Softwarekomponenten benutzt, sondern es werden lediglich Daten von einer externen Anwendung direkt in die entsprechenden Tabellen der Datenbank hinterlegt bzw. ausgelesen.

Bei der externen Anwendung handelt es sich um die Software, die auf den so genannten Prüfschränken der BHTC betrieben wird. Ein Prüfschrank ist vergleichbar mit einem Rack oder einem Server-Schrank aus der IT-Technologie, wobei er diverse elektronische und mechanische Bauteile beinhaltet, um Prototypen zu

¹⁵ [wiki5]

testen. Im ersten Schritt liest die Prüfschrank-Software die Daten des Projektes und des Musterstandes aus der Datenbank aus, führt im zweiten Schritt die Prüfung des Prototypen durch, um im dritten Schritt die Ergebnisse der Prüfung in der Datenbank zu hinterlegen. Als UML-Diagramm stellt sich dieser Vorgang wie folgt dar:

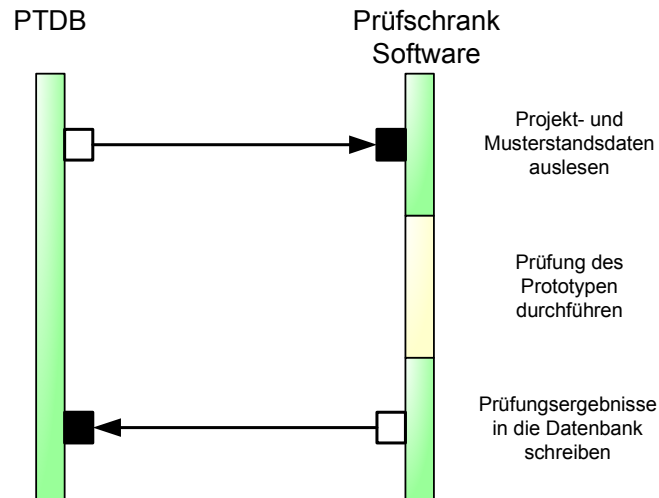


Abbildung 46: Prototypprüfung im Prüfschrank

4.2.4.2 Freigabe eines Prototypen

Vergleichbar mit der Musterstandsfreigabe (siehe auch 4.2.3.3 Freigabe eines Musterstandes), muss auch jeder entwickelte Prototyp freigegeben, d.h. als geprüft gekennzeichnet werden. Im Unterscheid zur Musterstandsfreigabe muss ein Prototyp jedoch nicht nur von zwei Abteilungen freigegeben werden, sondern die erforderlichen Freigaben sind abhängig von der Art des Prototypen. Zu unterscheiden sind hier die folgenden Prototypenarten inkl. ihrer erforderlichen Freigaben:

a) **Referenz-Prototyp**

Der Referenz-Typ ist der erste angelegte und produzierte Prototyp innerhalb eines Musterstandes. Er dient dazu, den Musterstand zu überprüfen und wird durch die Abteilungen TPL und T-ZL freigegeben.

b) **Regulärer-Prototyp**

Der reguläre Prototyp ist jeder Prototyp, der weder in die Kategorie Referenz-Typ noch in die Kategorie Sonder-Typ fällt. Er muss durch die Abteilungen TPL und T-ZL freigegeben werden.

c) Sonder-Prototyp

Bei einem Sonder-Prototypen handelt es sich zum Beispiel um ein Vorstandsmuster, das der Vorstand des Kunden angefordert hat. Bei dieser Art des Prototypen muss eine zusätzliche Kontrolle durchgeführt werden und daher sind hier zusätzlich zur Freigabe der Abteilungen TPL und T-ZL auch die Freigabe der Abteilung TQ und OFK erforderlich.

Jede Freigabe wird innerhalb der Datenbank gespeichert und analog zur Musterstandsfreigabe bewirkt eine vollständige Freigabe, d.h. eine Freigabe durch alle notwendigen Abteilungen, dass der Prototyp „frozen“ wird und somit nicht mehr veränderbar ist. Das UML-Schaubild stellt sich somit für die Prototypenfreigabe wie folgt dar:

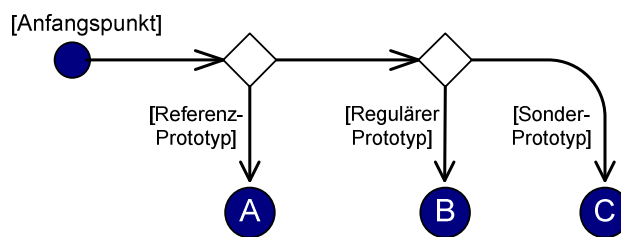


Abbildung 47: UML-Diagramm Prototypen-Freigabe

Die einzelnen Freigaben der unterschiedlichen Prototypen-Arten stellen sich in UML wie folgt dar:

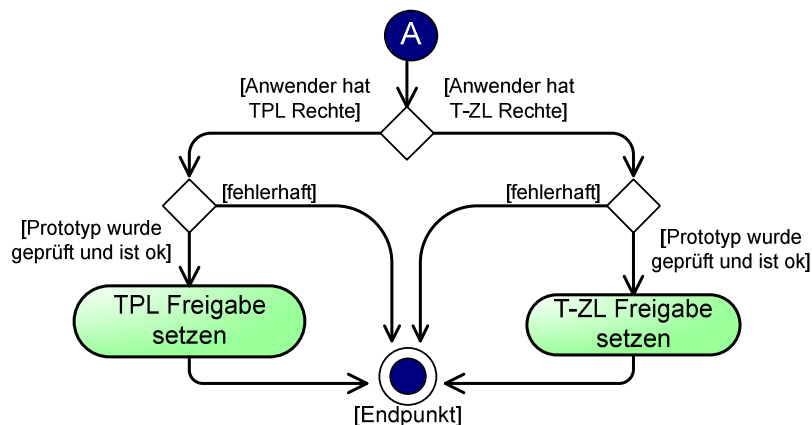


Abbildung 48: UML-Diagramm Prototyp-Freigabe eines Referenztypen

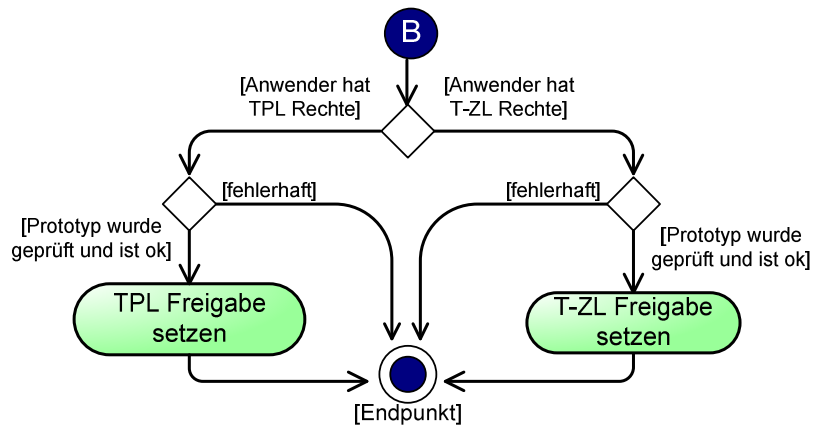


Abbildung 49: UML-Diagramm Prototyp-Freigabe eines regulärer Prototypen

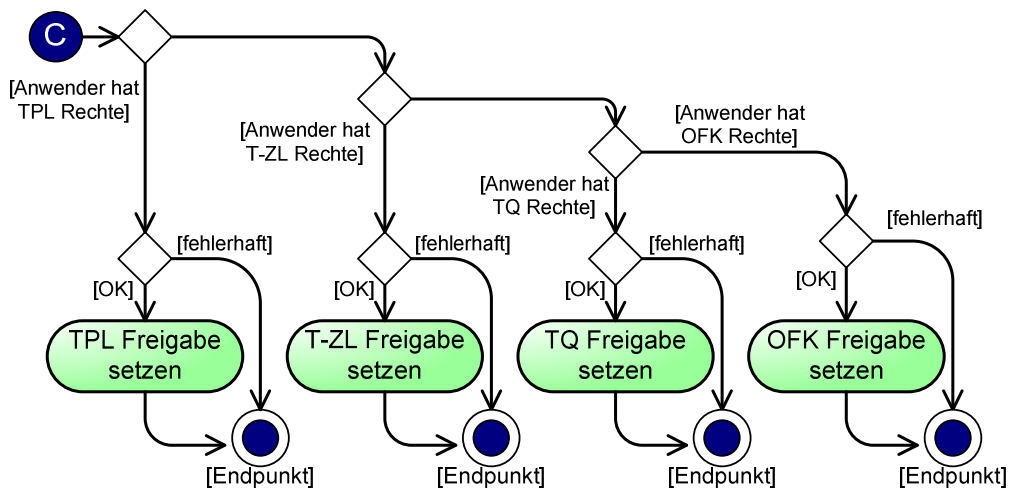


Abbildung 50: UML-Diagramm Prototyp-Freigabe eines Sonder-Typen

5 Teilsystem 3: Suchfunktionen der PTDB

In Software-Programmen sind sehr oft komfortable Suchfunktionen verfügbar, die ein schnelles (Dauer wenige Sekunden) Durchsuchen riesiger Datenbestände (Milliarden Datensätze) ermöglichen. Wesentliche Voraussetzung für die schnelle Suche ist die Sortierung bzw. Indizierung der Datenbestände. Die Entwicklung und Realisierung von Suchalgorithmen ist ein Teilgebiet der Informatik.¹⁶

Was genau bedeutet das nun für die Prototypen-Datenbank? Innerhalb der Anforderungsanalyse wurden auch diesem Gebiet mehrere Meetings gewidmet, um die Bedürfnisse der BHTC bezüglich der Suchfunktionen zu analysieren. Der erste, initiale Wunsch der BHTC, eine Suche über sämtliche Daten zu ermöglichen, also eine Volltextsuche innerhalb der gesamten Datenbank, hätte den Rahmen der ursprünglichen Diplomarbeit bei weitem gesprengt, da es sich hierbei um eine eigenständige Diplomarbeit handeln würde. Zu beachten wäre bei solch einer Suche nicht nur das Ermitteln der Datensätze, sondern auch das Bewerten der ermittelten Daten, um Treffer entsprechend darstellen zu können.

Ergebnis der Meetings war es letztendlich, Suchfunktionen auf zwei wichtigen Ebenen zu ermöglichen. Es sollten folgende Suchfunktionen implementiert werden:

- eine Suche nach Projekten
- und eine Suche nach Prototypen

Wie man anhand der Definition für Suchfunktionen erkennen kann, ist eine effiziente Suche nur dann möglich, wenn man den Einsatz von Datenbankschlüsseln (Indices), sorgfältig plant und nutzt. Eine genaue Übersicht über die Datenbank-Struktur ist hier also unumgänglich, was in den einzelnen Teilbereichen später auch deutlich dargestellt wird.

5.1 Suchen nach Projekten

Wie bereits mehrfach aufgeführt ist das Projekt die oberste Ebene der Datenstruktur. Um eine ebenso effiziente wie komfortable Suche für den Anwender

¹⁶ [wiki13]

zu entwickeln, musste im ersten Schritt definiert werden, anhand welcher Informationen eine Suche möglich sein sollte. Die Analyse hat ergeben, dass die Projekte im Regelfall anhand folgender Daten ermittelt werden können:

- Interne Projektbezeichnung
- externe Projektbezeichnung
- Beschreibung Projekt
- Teilebenennung
- BHTC Teilernr.
- Ausführender „Konstruktion“
- Ausführender „Entwicklung“
- Ausführender „Tester“
- Auftragsnummer
- Referenznummer (Kunde)
- Besteller / Verbleib
- Musterstandbeschreibung
- Musterstandinformation
- SAP Kundennummer
- externe Referenznummer

Nicht jedes Feld muss immer in die Suche einbezogen werden, daher wurde innerhalb der PTDB eine Auswahlmöglichkeit implementiert, sodass der Anwender angeben kann, in welchen Feldern der Text gesucht werden soll. Nachdem nun die zu durchsuchenden Daten definiert wurden, können wir uns das Datenmodell näher betrachten, um zu ermitteln, welche Tabellen jeweils in die Suche einbezogen werden müssen.

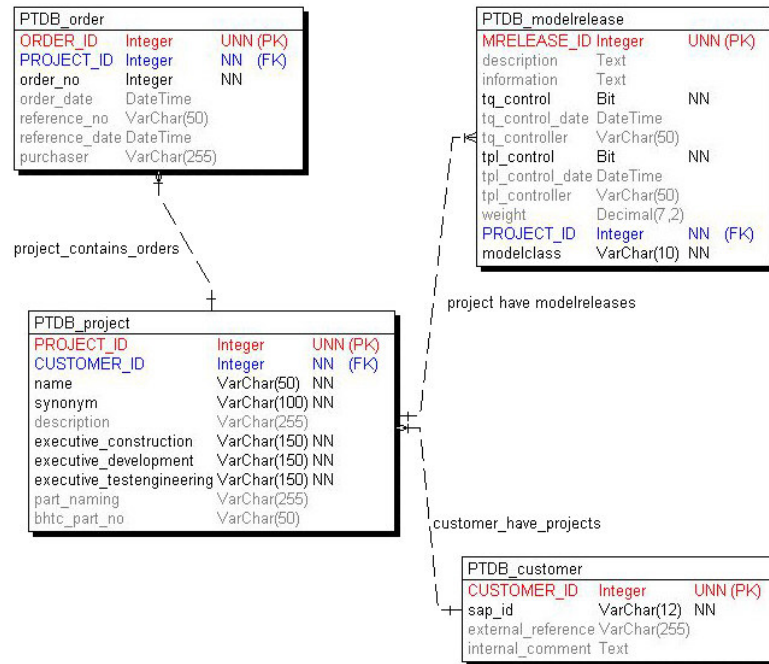


Abbildung 51: ER-Modell Projektsuche

Betroffen von der Suche sind also im größten Fall vier Tabellen der Datenbank und im kleinsten Fall nur eine Tabelle. Ähnlich variabel kann auch der zu suchende Text gehandhabt werden. Es kann entweder nach einem exakten Treffer, d.h. der gesamte Text muss identisch sein, oder nach einem Teil des Feldinhaltes gesucht werden. Um mindestens ein Suchkriterium zu erhalten, wurde in der Bedarfsanalyse festgehalten, dass das Feld „Interne Projektbezeichnung“ immer in die Suche einbezogen werden soll. Die Suchmaske stellt sich somit dem Anwender wie folgt dar:

Projekt-Suche

Art der Suche: exakter Treffer
 Teil des Feldinhaltes

Suche in:

<input checked="" type="checkbox"/> Interne Projektbezeichnung	<input type="checkbox"/> Ausführender Konstruktion	<input type="checkbox"/> Besteller / Verbleib
<input type="checkbox"/> externe Projektbezeichnung	<input type="checkbox"/> Ausführender Entwicklung	<input type="checkbox"/> Musterstandbeschreibung
<input type="checkbox"/> Beschreibung Projekt	<input type="checkbox"/> Ausführender Tester	<input type="checkbox"/> Musterstandinformation
<input type="checkbox"/> Teilebenennung	<input type="checkbox"/> Auftragsnummer	<input type="checkbox"/> SAP Kundennummer
<input type="checkbox"/> BHTC Teilern.	<input type="checkbox"/> Referenznummer (Kunde)	<input type="checkbox"/> externe Referenznummer

suchen nach: Suche Projekt

Abbildung 52: Maske Projektsuche

Die Ergebnisse der Suche werden anschließend in einer übersichtlichen Tabellenform dargestellt:

	PROJECT_ID	projectname	
<input type="button" value="Auswählen"/>	4	_Sandbox Project 1	Teilebenennung - TPDE
<input type="button" value="Auswählen"/>	5	_Sandbox Project 2	Teilebenennung - TPDE

Abbildung 53: Projektsuche - Ergebnistabelle

Um den Anwender nicht mit Informationen zu „überfluten“, werden in der Ergebnistabelle nur die Felder angezeigt, die auch Bestandteil der Suche waren.

Worin liegt der Nutzen der Suche und worin liegt der Vorteil für den Anwender?

Mit Hilfe dieser Suchmaske kann der Benutzer der PTDB auf einfache Weise Daten innerhalb der Datenbank suchen und direkt zu ihnen navigieren. Er hat die Möglichkeit, innerhalb der Ergebnistabelle das gewünschte Projekt auszuwählen und direkt zu diesem zu navigieren, ohne dass er vorher die Abhängigkeiten kennen muss.

5.1.1 Optimierung der Projektsuche

Wie die Definition der Suchfunktion bereits erwähnte, ist es für eine effiziente Suche erforderlich, dass die Daten sortiert und indiziert sind. Eine einfache Möglichkeit die Suche zu optimieren ist das Erstellen von sogenannten Views innerhalb der Datenbank.

Eine View (deutsch: Sicht) ist eine logische Relation (auch virtuelle Relation oder virtuelle Tabelle) in einem Datenbanksystem. Diese logische Relation wird über eine im Datenbankmanagementsystem (DBMS) gespeicherte Abfrage definiert. Der Datenbankbenutzer kann eine View wie eine normale Tabelle abfragen. Wann immer eine Abfrage diese View benutzt, wird diese zuvor durch das Datenbankmanagementsystem berechnet. Eine View stellt im Wesentlichen einen Alias für eine Abfrage dar.¹⁷

¹⁷ [wiki6]

Die Aufgabe einer View ist es, den Zugriff auf das Datenbankschema zu vereinfachen. Normalisierte Datenbankschemata verteilen Daten auf zahlreiche Tabellen mit komplexen Abhängigkeiten. Dieses führt zu aufwändigen SQL-Abfragen. Außerdem wird ein hohes Maß an Wissen über das Schema vorausgesetzt, um solche Abfragen zu erstellen. Das Bereitstellen geeigneter Views erlaubt einen einfachen Zugriff, ohne Kenntnis des darunter liegenden Schemas und ohne Aufweichung der Normalisierung.

Ein weiterer Vorteil von Views ist, dass das DBMS keinen zusätzlichen Aufwand zur Vorbereitung der Abfrage benötigt. Die View-Abfrage ist vom Parser bereits bei der Erstellung syntaktisch zerlegt worden und wurde bereits beim Anlegen vom Abfrageoptimierer vereinfacht. Ein Parser (engl.: to parse = „analysieren“) ist ein Programm, das in der Computertechnik für die Zerlegung und Umwandlung einer beliebigen Eingabe in ein für die Weiterverarbeitung brauchbares Format zuständig ist.¹⁸

Die View, die zur Projektsuche entwickelt wurde kann dem Anhang 12.4 „Anhang: View zur Projektsuche“ entnommen werden.

Weiterhin wurden logische Indices auf die betroffenen Tabellen gelegt, um die Suche zu beschleunigen. Folgende Indices wurden angelegt:

- **auf die Tabelle PTDB_project**

```
CREATE INDEX idx_c_NAME ON PTDB_project
(PROJECT_ID, name)
```

- **auf die Tabelle PTDB_modelrelease**

```
CREATE INDEX idx_c_MCLASS ON PTDB_modelrelease
(MRELEASE_ID, modelclass)
```

- **auf die Tabelle PTDB_order**

```
CREATE INDEX idx_c_ONO ON PTDB_order
(ORDER_ID, order_no)
```

- **auf die Tabelle PTDB_customer**

```
CREATE INDEX idx_c_SAPID ON PTDB_customer
(sap_id, CUSTOMER_ID)
```

¹⁸ [wiki7]

Analysen der Laufzeit haben gezeigt, dass diese zusätzlichen Indices in der ersten Version noch keinen Zeitgewinn bewirken, jedoch im Laufe der Zeit, abhängig von den eingetragenen Daten, an Bedeutung gewinnen, da sie erst bei einer großen Datenmenge vom DBMS angewendet werden.

5.2 Suchen nach Prototypen

Die für den Anwender wichtigste Suchfunktion ist die direkte Suche nach einem Prototyp. Bei dieser Art der Suche liegen dem Benutzer im Regelfall mindestens eine der folgenden Informationen vor:

- lfd. Prototyp-Nr.
- TCE-Nr.
- Referenznummer (Kunde)
- Verwendungszweck
- Teileanhänger Kommentar
- Kundenteile-Nr.
- Netzplan-Nr.

Die am häufigsten benutzten Suchkriterien sind die Felder „lfd. Prototyp-Nr.“ und „TCE-Nr.“, da diese auf einem Aufkleber auf dem produzierten Prototypen vermerkt sind. Da jedoch die laufende Prototyp-Nr. nur eindeutig auf Projektebene ist, besteht für den Anwender die Möglichkeit, die Suche anhand von drei Filtern einzuschränken, die die folgenden Daten vorfiltern:

- SAP Kundennummer
- Projekt
- Musterstand

Die Filter sind untereinander abhängig, d.h. wählt man bei den Filtern eine SAP Kundennummer aus, so stehen nur noch die Projekte des betroffenen Kunden zur Verfügung und wählt man ein Projekt aus, so stehen nur noch die unter diesem Projekt angelegten Musterstände zur Verfügung, wobei es auch möglich ist, innerhalb der Maske nur einen Filter zu setzen.

Auch bei dieser Art der Suche gilt, dass nicht jedes Feld zwangsläufig in die Suche einbezogen werden muss, da der Anwender die zu durchsuchenden Felder innerhalb der Maske auswählen kann. Auch hier ist es dem Anwender möglich zu unterscheiden zwischen einer „exakten Suche“ und einer „Teilsuche“, wie es bereits aus der Projektsuche bekannt ist.

Basierend auf den ausgewählten Suchfeldern und den gesetzten Filtern, muss die Suche entweder nur die Tabelle für die Prototypen-Daten einbeziehen (ohne gesetzte Filter) oder auch bis zu drei weitere Tabellen:

PTDB_prototype		
PROTOTYPE_ID	Integer	UNN (PK)
prototype_no	Integer	NN
production_date	DateTime	
delivery_date	DateTime	
reference_no	VarChar(50)	
reference_date	DateTime	
purpose	VarChar(255)	
part_tag_comment	VarChar(255)	
prototype_mode	Char(1)	NN
customer_part_no	VarChar(50)	
tce_no	VarChar(30)	NN
network_plan	Integer	
variety_description	Text	
variety_index_no	Integer	
PROTOTYPE_LINK	Integer	
tq_control	Bit	NN
tq_control_date	DateTime	
tq_controller	VarChar(50)	
tpl_control	Bit	NN
tpl_control_date	DateTime	
tpl_controller	VarChar(50)	
tzl_control	Bit	NN
tzl_control_date	DateTime	
tzl_controller	VarChar(50)	
ofk_control	Bit	NN
ofk_control_date	DateTime	
ofk_controller	VarChar(50)	

Abbildung 54: ER-Modell Prototypsuche ohne aktivierte Filter

Da die zusätzlich eingebundenen Tabellen nur einen geringen Prozentsatz der Daten innerhalb der Datenbank beanspruchen, war es hier nicht erforderlich eine zusätzliche View zu definieren. Weiterhin sind die drei zusätzlichen Tabellen direkt mit der Prototypen-Tabelle über Foreign-Keys verbunden. Ein Foreign-Key (deutsch: Fremdschlüssel) ist ein Sekundärschlüssel einer Relation, der in einer anderen Relation Primärschlüssel oder Schlüsselkandidat ist. Er dient als Verweis zwischen zwei Relationen, d.h. er zeigt an, welche Tupel der Relationen inhaltlich miteinander in Verbindung stehen.¹⁹

¹⁹ [wiki8]

Im folgenden ER-Modell wird die Beziehung zwischen den einzelnen Relationen dargestellt:

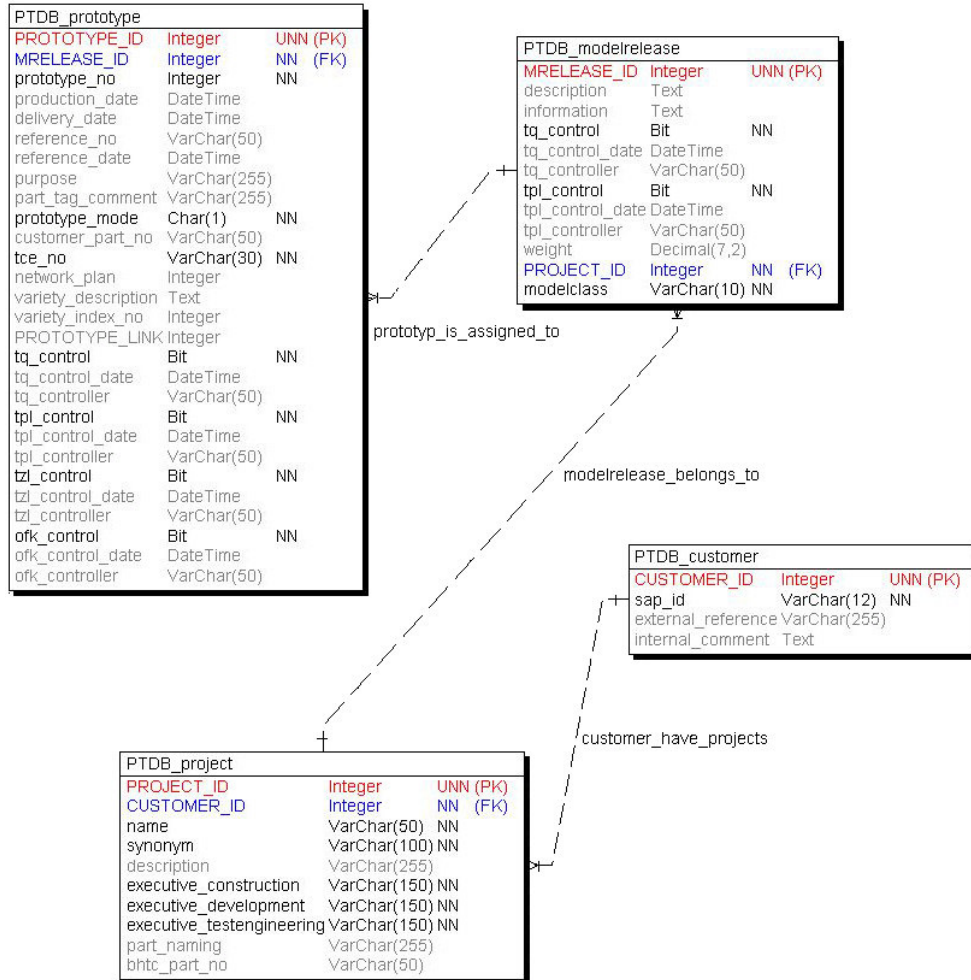


Abbildung 55: ER-Modell Prototypsuche mit aktiviertem Filter

Wählt der Benutzer also innerhalb des Filters eine SAP-Kundennummer aus, so wird ihm in der Projektauswahl nur noch eine Teilmenge (basierend auf der SAP Kundennr.) angezeigt und wählt er weiter ein Projekt aus, so wird ihm auf Ebene des Musterstandes wiederum nur eine Teilmenge (basierend auf der Projektauswahl) angezeigt. Durch diese Vorgehensweise ist es möglich, die Suche schnell und effizient zu gestalten, da sie jeweils nur einen limitierten Datenbestand durchsuchen muss.

Sollten dem Anwender keine Informationen über die SAP Kundenummer, das Projekt oder dem Musterstand vorliegen, so hat er ebenfalls die Möglichkeit, die Suche ohne Einschränkung der Filter durchzuführen. Die Suchmaske der Prototypensuche stellt sich somit wie folgt dar:

Abbildung 56: Prototypen-Suchmaske

Auch in dieser Suchmaske werden nur die Informationen in der Ergebnis-Tabelle gelistet, die der Anwender in seine Suche einbezogen hat (vgl. auch 5.1 Suchen nach Projekten):

	PROTOTYPE_ID	prototype_no	name	prototype_mode	modelclass
<input type="button" value="Auswählen"/>	5	1	Testproject 1 - DEV	0	A-1
<input type="button" value="Auswählen"/>	16	1	_Sandbox Project 1	0	A-1
<input type="button" value="Auswählen"/>	98	1	VW Passat	1	A-1
<input type="button" value="Auswählen"/>	103	1	Testprojekt AS2	0	A-1
<input type="button" value="Auswählen"/>	1148	1	_Sandbox Project 2	0	A-1

Abbildung 57: Prototypen-Suche Ergebnistabelle

Auch an dieser Stelle ist es dem Benutzer möglich, einen Prototyp aus der Ergebnismenge auszuwählen und direkt zu diesem zu springen; somit ist es jedem Anwender möglich, auf einfache und komfortable Weise Prototypen aufzurufen ohne das zugehörige Projekt oder die zugehörigen Musterstände zu kennen.

6 Teilsystem 4: Reportingfunktionen zur Qualitätssicherung

Unter dem Begriff Reporting versteht man die Einrichtungen eines Berichtes mit der übergeordneten Zielsetzung eines Unterrichtszweckes. Ein Report oder auch Bericht stellt zusammengefasste Informationen dar.

Berichte unterstützen auf allen Ebenen die strategische und operative Unternehmensführung: Vom Top-Management über die Leitung der Unternehmensbereiche bis hin zu den einzelnen operativen Einheiten. Sie unterstützen die Zusammenarbeit mit Kunden und Geschäftspartnern, indem sie die Grundlagen der Zusammenarbeit transparent machen. Sie dienen als Information der Kunden, indem sie das bisher Erreichte und den dokumentierten Status darstellen.²⁰

Zu unterscheiden ist hier auf Seiten der PTDB nach internen und externen Reporten. Die Unterscheidung ist erforderlich, da innerhalb der Anwendung eine Vielzahl von Informationen hinterlegt werden können, die rein interner Natur sind und nicht an die Kunden weitergegeben werden sollen. Bisher existieren in der Anwendung nur ein interner und ein externer Prototypen-Lebenslauf, weitere interne und externe Berichte werden ggf. in einer späteren Programmversion implementiert.

Der in der vorangegangenen Projektarbeit entwickelte Lösungsansatz, die Daten der Berichte als XML zur Verfügung zu stellen, wurde aus Kostengründen wieder verworfen. Im ursprünglichen Ansatz sollte das Werkzeug „Adobe Designer“ eingesetzt werden, um das Layout der Berichte zu entwickeln, in das die generierten XML-Daten geladen werden sollten. Im Laufe der Umsetzung hat sich jedoch herausgestellt, dass die entwickelten PDF-Vorlagen nur in einer Vollversion des Adobe Designers ordnungsgemäß eingesetzt werden können, nicht jedoch in der frei erhältlichen Version des Adobe Readers. Das wiederum hätte zur Folge gehabt, dass auf jedem Einzelplatzsystem, das zur Berichtserstellung genutzt werden soll, eine Installation des Adobe Designers erforderlich wäre. Bei einem momentanen Preis von über 350,00 Euro pro Installation hätte dieser Ansatz schnell Kosten verursacht, die nicht eingeplant waren.

²⁰ [wiki9]

Hintergrund für den gewählten Lösungsansatz war es, die Mehrsprachigkeit implementieren zu können, ohne für jede Sprache ein eigenständiges Berichtslayout entwerfen zu müssen, um hier den Entwicklungsaufwand möglichst gering und die Wartbarkeit möglichst hoch zu gestalten.

Die Lösung: Reporting Services

Die eingesetzte Entwicklungsplattform „Microsoft Visual Studio 2005“ stellt mit den Reporting Services eine Report-Technologie zur Verfügung, die anhand einer abgespeckten Version des Report Designers, leicht und unkompliziert, Berichte entwickeln kann. Die folgende Aufzählung umfasst die wichtigsten Eigenschaften des Report Designers:

- Effektive Datenverarbeitung bei Filtern, Sortieren und Gruppieren
- Viele Möglichkeiten der Datenpräsentation wie Listen, Tabellen, Charts, Matrizen und Kreuztabellen
- Variables visuelles Erscheinungsbild mit Fonts, Farben, Umrandungen oder Hintergrundbildern
- Interaktives Verhalten bei Dokumentenmappe, Sortieren, Lesezeichen, zusammenklappbare Selektion
- Unterstützung bedingter Formatierung, wie z.B. Ausdrücke in den Report einfügen, um das Aussehen dynamisch in Abhängigkeit von den Daten zu verändern
- Druck- und Druckvorschau-Funktion
- Datenexport nach Microsoft Excel und in ein PDF-Format

Da der Report Designer mit einem hinterlegten Berichtslayout arbeitet, musste eine Lösung gefunden werden, um die Berichte in den unterschiedlichen Sprachen abbilden zu können. Die PTDBF-Anwendung nutzt hier die Berichtsparemeter, die innerhalb des Programmcodes gefüllt werden. So können z.B. die Parameter wie folgt gefüllt werden:

```
Dim RepParameter() As ReportParameter = { _  
    New ReportParameter( "str_prototype_no", PTDBF.Get_Lang_from_DB(MyLang, "REP_iptvita_ptno")), _
```

```
New ReportParameter( "str_HEAD",           ↵
                    PTDBF.Get_Lang_from_DB(MyLang, ↵
                    "REP_eptvita_head")), _
New ReportParameter( "str_nt",           ↵
                    PTDBF.Get_Lang_from_DB(MyLang, ↵
                    "REP_iptvita_nt")), _
New ReportParameter("str_na",           ↵
                    PTDBF.Get_Lang_from_DB(MyLang, ↵
                    "REP_iptvita_na")), _
New ReportParameter("str_ni",           ↵
                    PTDBF.Get_Lang_from_DB(MyLang, ↵
                    "REP_iptvita_ni")), _
New ReportParameter("str_ok",           ↵
                    PTDBF.Get_Lang_from_DB(MyLang, ↵
                    "REP_iptvita_ok")) _
                    }
```

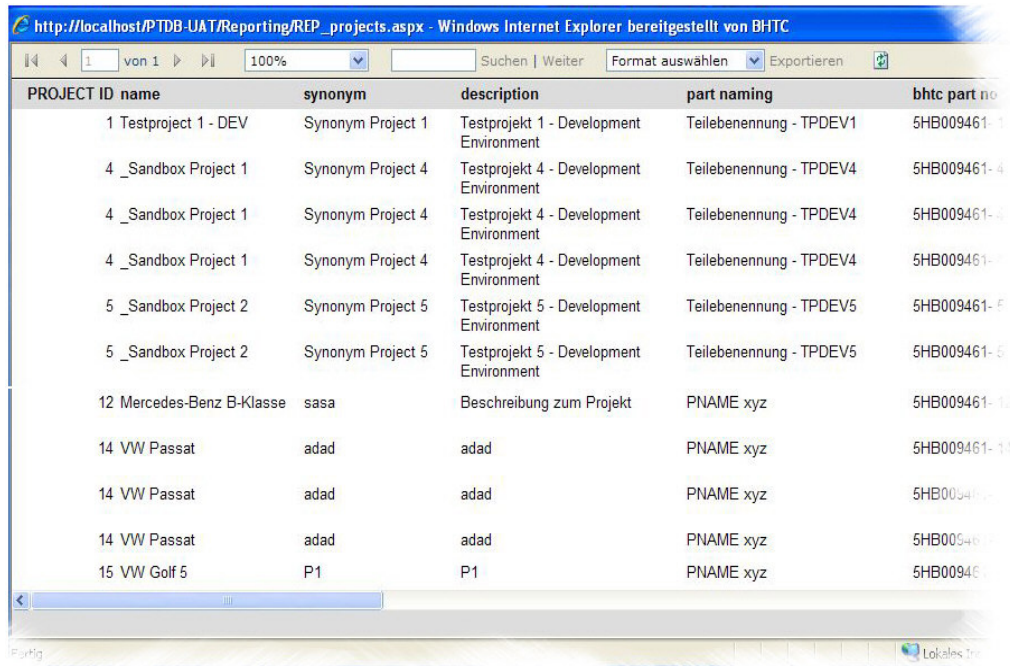
```
ReportViewer1.LocalReport.SetParameters(RepParameter)
```

Durch diese Vorgehensweise ist es möglich, die im Datenbankmanagementsystem hinterlegten und je Sprache unterschiedlichen Texte vor Generierung des Berichtes zu laden und entsprechend zu benutzen.

Nach Rücksprache mit der BHTC wurde diese Lösung umgesetzt und ein Set an Standardreporten implementiert. Weitere Berichte können später in der Anwendung, je nach Bedürfnis der BHTC, umgesetzt werden.

6.1 Allgemeine Reporte

Auf Grund der Übersicht gibt es nur einen Report, der übergreifend auf alle Daten der Datenbank zugreift und nicht an eine Ebene der Datenfamilie (vergleiche auch Abbildung 5 „Die drei Datenfamilien“) gebunden ist. Dieser Report listet alle existierenden Projekte der Datenbank auf, inkl. Teile der zugehörigen Informationen. Das folgende Bild stellt den Report dar, den die Anwendung generiert:



PROJECT ID	name	synonym	description	part naming	bhct part no
1	Testproject 1 - DEV	Synonym Project 1	Testprojekt 1 - Development Environment	Teilebenennung - TPDEV1	5HB009461-1
4	_Sandbox Project 1	Synonym Project 4	Testprojekt 4 - Development Environment	Teilebenennung - TPDEV4	5HB009461-4
4	_Sandbox Project 1	Synonym Project 4	Testprojekt 4 - Development Environment	Teilebenennung - TPDEV4	5HB009461-4
4	_Sandbox Project 1	Synonym Project 4	Testprojekt 4 - Development Environment	Teilebenennung - TPDEV4	5HB009461-4
5	_Sandbox Project 2	Synonym Project 5	Testprojekt 5 - Development Environment	Teilebenennung - TPDEV5	5HB009461-5
5	_Sandbox Project 2	Synonym Project 5	Testprojekt 5 - Development Environment	Teilebenennung - TPDEV5	5HB009461-5
12	Mercedes-Benz B-Klasse	sasa	Beschreibung zum Projekt	PNAME xyz	5HB009461-12
14	VW Passat	adad	adad	PNAME xyz	5HB009461-14
14	VW Passat	adad	adad	PNAME xyz	5HB009461-14
14	VW Passat	adad	adad	PNAME xyz	5HB009461-14
15	VW Golf 5	P1	P1	PNAME xyz	5HB009461-15

Abbildung 58: Report: Alle Projekte der Datenbank

Dieser Report ist nicht dazu gedacht, direkt gedruckt zu werden, da sein Layout keiner druckbaren Vorlage entspricht. Er soll vielmehr dazu genutzt werden, eine Projektübersicht bereitzustellen, die zur weiteren Verarbeitung nach Microsoft Excel exportiert werden kann.

6.1.1 Allgemeine Reporte auf Projektebene

Die weiteren Reporte sind innerhalb der Anwendung logisch gruppiert. Eine Gruppe von Reporten sind die Berichte, die auf Projektebene erstellt werden. Um einen Bericht auf dieser Ebene erstellen zu können, muss der Anwender im ersten Schritt das gewünschte Projekt auswählen:



Report: Musterstände eines Projektes

Auswahl Projekt: Testproject 1 - DEV

zeige den Report an

Abbildung 59: Projektauswahl für Reporte

Die generierten Reporte entsprechen in ihrem Layout den z. Z. genutzten Berichten der BHTC und sind somit als PDF nutzbar. Zu unterscheiden sind hier die folgenden Reporte:

- **Musterstände eines Projektes**

In diesem Bericht werden die einzelnen Musterstände auf Projektebene dargestellt. Er enthält sowohl die Beschreibung und Informationen zum Musterstand als auch die Freigaben des Musterstandes. Ein Beispiel dieses Berichtes kann im Anhang 12.5.1 nachgeschlagen werden.

- **Musterstände eines Projektes inklusive der erfassten Prototypen-Anzahl**

Dieser Bericht liefert eine Aufstellung der angelegten Musterstände auf Projektebene inklusive der Anzahl der erfassten Prototypen. Ein Beispiel zu diesem Report kann dem Anhang 1.1.1 entnommen werden.

- **Softwarestände eines Projektes**

Eine Übersicht der hinterlegten Softwarestände ist der Zweck dieses Berichtes. Er enthält sowohl die allgemeinen Informationen zu den jeweiligen Softwareständen als auch eine Aufstellung der implementierten Softwarefunktionen und ihrer Prüfungen. Ein Beispiel kann im Anhang 1.1.1 betrachtet werden.

- **Hardwarestände eines Projektes**

Vergleichbar mit dem Bericht der Softwarestände, liefert dieser Report eine Aufstellung der hinterlegten Hardwarestände eines Projektes inklusive der verbauten Hardwareteile und derer Prüfungen. Im Anhang 12.5.4 kann ein Beispiel-Bericht nachgeschlagen werden.

- **Konstruktionsstände eines Projektes**

Dieser Bericht entspricht in groben Zügen den beiden vorherigen Berichten, jedoch liefert er im Unterschied zu den Software- bzw. Hardwarestand-Berichten keine weiteren Details über etwaige Prüfungen, da der Konstruktionsstand keine separaten Bestandteile enthält, die geprüft werden müssen. Eine Übersicht des Berichtes kann dem Anhang 12.5.5 entnommen werden.

- **Prototypen-Übersicht auf Projektebene**

Eine Aufstellung aller produzierten und in der Datenbank erfassten Prototypen kann diesem Report entnommen werden. Er enthält zu jedem Prototyp den Software-, Hardware- und Konstruktionsstand sowie weitere Informationen auf Prototyp-Ebene. Ein Beispiel kann im Anhang 12.5.6 gefunden werden.

6.1.2 Allgemeine Reporte auf Musterstandsebene

Eine weitere Gruppe von Reporten bilden die Berichte auf Musterstandsebene. Insgesamt gibt es zwei Berichte auf dieser Ebene, die in ihrem Layout einer druckbaren Version entsprechen und bereits bei der BHTC eingesetzt wurden. Da die Berichte auf dieser Ebene auf einem gewählten Musterstand basieren, muss bei der Auswahl des Berichtes zusätzlich zum Projekt auch der gewünschte Musterstand angegeben werden. Die Startmaske stellt sich somit wie folgt dar:



Abbildung 60: Projektauswahl inkl. Musterstandwahl für Reporte

Nachdem das gewünschte Projekt inklusive des gewünschten Musterstandes gewählt wurde, werden die Informationen der entsprechenden Berichte generiert.

- **Hardwareteile eines Musterstandes**

Dieser Bericht bezieht sich auf den Hardwarestand, der zum gewählten Musterstand hinterlegt wurde. Es wird jedes verbaute Hardwareteil in einer Listenform ausgegeben, inklusive des Prüfzustandes. Ein Beispiel für diesen Report kann dem Anhang 12.5.7 entnommen werden.

- **Software-Funktionen eines Musterstandes**

Dieser Bericht liefert eine Aufstellung der implementierten Software-funktionen, die zum gewählten Musterstand in der Datenbank hinterlegt sind. Vergleichbar mit dem Bericht der Hardwareteile wird hier auch der jeweilige Prüfungsstatus der Softwarefunktion gelistet. Im Anhang 12.5.8 ist ein Beispiel dieses Berichtes zu finden.

6.1.3 Allgemeine Reporte auf Prototypenebene

Die letzte Ebene der allgemeinen Berichte bildet ein Report auf Prototypenebene. Auch auf dieser Ebene weicht die Startmaske von den bisher gezeigten ab, da nun nicht nur das Projekt sondern auch der gewünschte Prototyp angegeben werden muss.



Abbildung 61: Projektauswahl inkl. Prototypenwahl für Reporte

Ein weiterer Unterschied zu den bisherigen Startmasken der Berichte ist es, dass die Prototypen-Nummer nicht via DropDown-Feld ausgewählt werden kann, sondern manuell eingegeben werden muss. Diese Vorgehensweise musste gewählt werden, damit die Laufzeit nicht negativ beeinflusst wird (vergleiche auch 4.2.1 Navigation innerhalb der Daten). Da also für diesen Report die Prototypen-Nummer bekannt sein muss, kann man den Bericht aus 6.1.1 Prototypen-Übersicht auf Projektebene nutzen um die hinterlegten Prototypen zu erfahren.

Nachdem eine existierende Prototypen-Nummer des Projektes gewählt wurde, zeigt der Bericht eine Aufstellung der implementierten Software-Funktionen in einer druckbaren Form an. Ein Beispiel dieses Berichtes kann man im Anhang 12.5.9 nachschlagen.

6.2 Interne Reporte

Die Unterscheidung nach internen und externen Berichten ist z. Z. nur bei zwei Berichten notwendig. Da die bisher vorgestellten Berichte keine internen Kommentare enthalten, sondern nur allgemeine Informationen liefern, kann jeder der gezeigten Berichte auch an Kunden verschickt werden. Einen Unterschied bildet hier der nun gezeigte Bericht.

Der interne Prototypen-Lebenslauf

Die Generierung dieses Berichtes erfordert die Auswahl des Projektes und die Eingabe der Prototypen-Nummer, wie wir es schon aus 6.1.3 Allgemeine Reporte auf Prototypebene kennen. Nachdem diese Daten eingegeben wurden, liefert die Anwendung einen Bericht zu dem gewählten Prototypen. Da es sich hier um einen internen Bericht handelt, werden u. a. interne Kommentare aufgelistet, die durch die Anwender eingegeben wurden. Aus diesem Grund darf der Report nicht an Kunden verschickt werden. Ein Beispiel kann dem Anhang 12.5.10 entnommen werden.

6.3 Externe Reporte

Innerhalb der Anwendung gibt es z. Z. nur einen Report, der explizit als externer Report gekennzeichnet/ausgewiesen wurde.

Der externe Prototypen-Lebenslauf

Bei diesem Bericht handelt es sich um das Gegenstück zum internen Prototypen-Lebenslauf mit dem Unterschied, dass hier nur die externen Kommentare aus der Datenbank ausgedruckt werden. Auch bei diesem Report muss der Anwender im ersten Schritt das Projekt und die gewünschte Prototypen-Nummer angeben, um den Bericht generieren zu lassen. Der Report wurde zum Zeitpunkt der Entwicklung bereits durch die BHTC eingesetzt und an Kunden zu Dokumentationszwecken verschickt. Im Anhang 12.5.11 kann ein Beispiel-Report betrachtet werden.

6.4 Sonderreport als XML

Da die BHTC auf Grund der fest codierten Berichte die Flexibilität eigene Reporte zu entwickeln verloren hat, wurden der BHTC im Rahmen der Gespräche bezüglich der Berichte zwei mögliche Lösungen vorgeschlagen.

1. Zugriff auf die Datenbank mit Microsoft Access via ODBC-Schnittstelle
2. Ein XML-Report mit allen Prototypendaten

Auf die erste Zugriffsmethode via ODBC und Microsoft Access wird hier nicht näher eingegangen, da sie ggf. erst in einem späteren Stadium durch die BHTC genutzt wird, weil der Nachteil der Zusatzkosten (eine lizenzierte Version von Microsoft Access ist Voraussetzung) hier nicht zu vernachlässigen ist.

Die zweite Variante wurde nach einigen Internet-Recherchen in der Anwendung hinterlegt, da es freie Werkzeuge gibt die mit der Basis einer XML-Datei einen Bericht erstellen können (z.B. XML-Publisher oder Eclipse BIRT), einige kostengünstige Programme XML-Dateien verarbeiten können (z.B. Crystal Reports) und zusätzlich die generierten XML-Dateien mit Microsoft Excel problemlos eingelesen und weiterverarbeitet werden können.

6.4.1 Was ist XML

Die Extensible Markup Language (kurz XML, engl. für „erweiterbare Auszeichnungssprache“), ist ein Format zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien. XML wird bevorzugt für den Austausch von Daten zwischen unterschiedlichen IT-Systemen eingesetzt, speziell über das Internet.²¹

Ein XML-Element kann ganz unterschiedliche Daten enthalten und beschreiben. Ein Grundgedanke hinter XML ist es, Daten und ihre Repräsentation zu trennen, um Daten beispielsweise einmal als Tabelle und einmal als Grafik auszugeben, aber für beide Arten der Auswertung die gleiche Datenbasis im XML-Format zu nutzen.

²¹ [wiki10]

Der logische Aufbau einer XML-Datei ist eine Baumstruktur und damit hierarchisch strukturiert. Als Baumknoten gibt es:

- **Elemente**, die physikalisch durch ein Paar aus Start-Tag (<Tag-Name>) und End-Tag (</Tag-Name>) oder einem Empty-Element-Tag (<Tag-Name />) dargestellt werden können,
- **Attribute** als bei einem Start-Tag oder Empty-Element-Tag beschriebene Schlüsselwort-Werte-Paare (Attribut-Name="Attribut-Wert") für Zusatzinformationen über Elemente (eine Art Meta-Information),
- **Kommentare** (<!-- Kommentar-Text -->)
- **Text**, der als normaler Text auftreten kann.

Ein XML-Dokument muss genau ein Element auf der obersten Ebene enthalten. Unterhalb von diesem Element können weitere Elemente verschachtelt werden.

6.4.2 Welche Daten enthält die XML-Datei

Auch bei diesem Bericht muss der Anwender im ersten Schritt das gewünschte Projekt und die gewünschte Prototypen-Nummer angeben. Ist dies erfolgt, generiert die Anwendung eine XML-Datei mit allen Informationen zum gewählten Prototyp. Die Informationen werden sowohl auf Prototypenebene analysiert, als auch auf einigen höher liegenden Ebenen (z.B. auf der Projektebene).

Der Aufbau der generierten XML-Datei kann dem Schaubild auf der folgenden Seite entnommen werden.

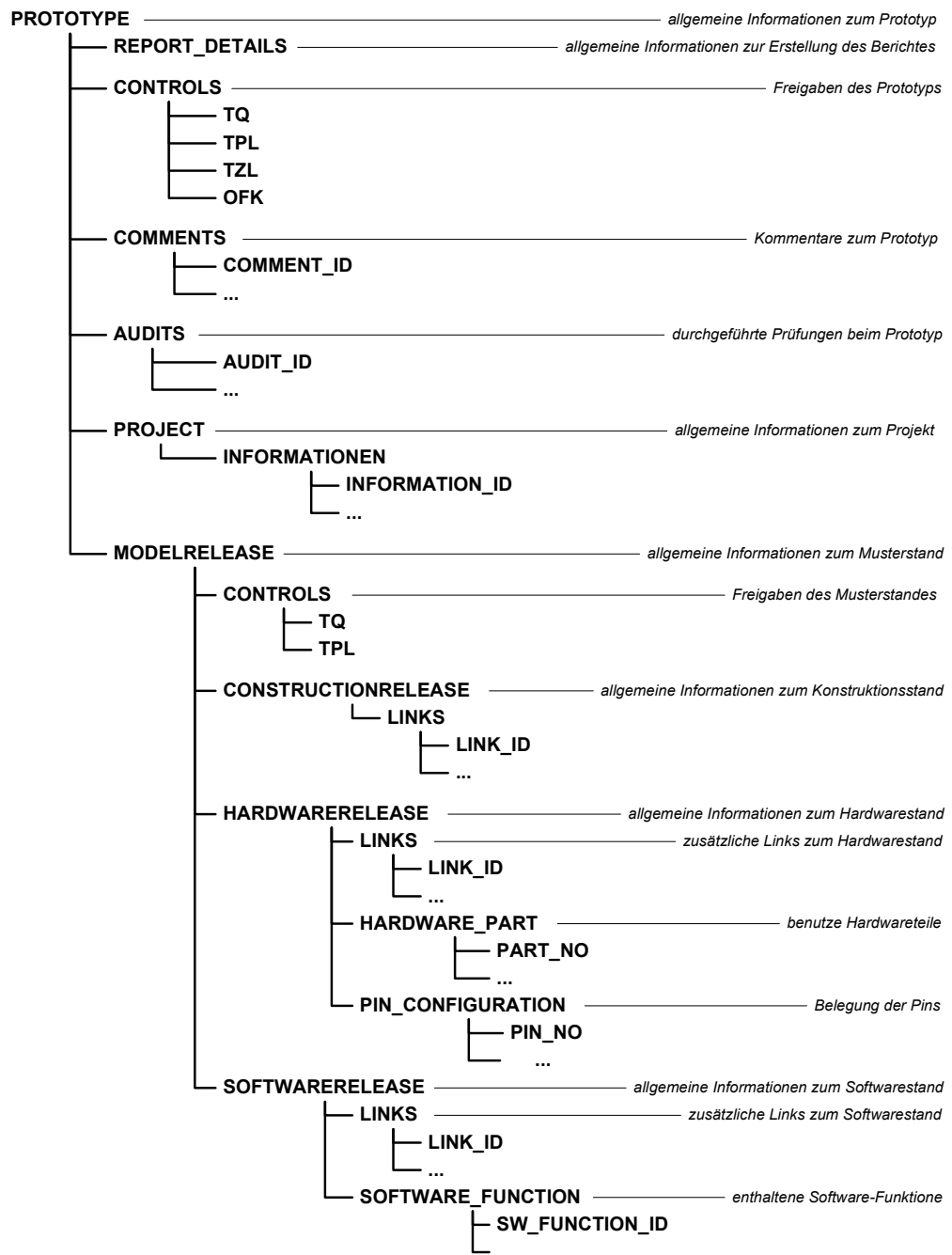


Abbildung 62: XML-Schema der XML-Datei

Die im Schema aufgeführten Punkte „...“ deuten Wiederholungen an, da es keinen, einen oder mehrere Einträge auf diesen Ebenen geben kann. Ein Beispiel kann dem Anhang 12.5.12 XML-Report zu einem Prototypen entnommen werden.

7 Abschluss / Resümee der Arbeit

7.1 Zusammenfassung

Ziel der vorliegenden Diplomarbeit war es, der Behr Hella Thermocontrol GmbH eine Konzeption für eine webbasierte Datenbank zur technischen Dokumentation von Prototypen zu erstellen und diese im Rahmen der Arbeit umzusetzen. Die Anfertigung des Konzepts erforderte vom Verfasser dieser Arbeit anfangs eine intensive Einarbeitung in die Thematik der Prototypenerstellung, um Missverständnisse durch mehrfach benutzte Begrifflichkeiten auszuschließen. Die vorausgegangene Projektarbeit „Analyse zur Entwicklung einer webbasierten Datenbank zur technischen Dokumentation von Prototypen“ lieferte eine umfangreiche Sammlung von Anforderungen, die im Rahmen dieser Arbeit umgesetzt wurden.

Das Kapitel „4 Teilsystem 2: Datenerfassung und -pflege“ stellt den Kern dieser Arbeit dar. Es befasst sich mit der allgemeinen Erfassung der Daten innerhalb der Anwendung, um die Dokumentation von Prototypen zu gewährleisten.

Ein weiterer Schwerpunkt dieser Arbeit wurde auf das Sicherheitskonzept der Anwendung gelegt, das im Kapitel „3 Teilsystem 1: Das Sicherheitskonzept“ erläutert wurde. Das Sicherheitskonzept spielt für die BHTC eine wichtige Rolle, da es sich bei der beauftragenden Firma um einen Automobilzulieferer handelt und die Geheimhaltung der Daten eine zentrale Rolle im Unternehmen spielt.

7.2 Fazit

Im Laufe der Umsetzung hat sich herausgestellt, dass die geforderte Programmiersprache und Entwicklungsumgebung (ASP.NET und VB.NET) für solche komplexen Aufgaben nur bedingt eingesetzt werden kann. Trotz dieser Umstände war es dem Verfasser möglich eine voll funktionsfähige Anwendung zu erstellen, die alle Ansprüche der BHTC umgesetzt hat.

Durch die intensive Einarbeitung in den Fachterminus der BHTC und deren Prozessdefinitionen wurden im Laufe der Umsetzung einige Schwachstellen an den definierten Prozessen aufgezeigt und an die zuständigen Fachabteilungen der BHTC gemeldet. So ist es zum Beispiel risikobehaftet, dass Musterstände auch ohne einen existierenden Referenz-Prototypen (der zur Prüfung des Musterstandes benötigt

wird) freigegeben werden können, da hier die Gefahr besteht, dass Freigaben auch ohne eine korrekte Prüfung erfolgen können.

Ein weiterer kritischer Programmablauf ist es, dass man Prototypen auch zu einem Musterstand anlegen kann auch wenn dieser noch nicht freigegeben wurde. Sollten später Änderungen am Musterstand notwendig werden, um entdeckte Fehler zu bereinigen, so sind die bereits erfassten Prototypen unkorrekt.

Weiterhin sind im Verlauf der Umsetzung ungenaue Definitionen von Abläufen aufgefallen, die nicht innerhalb der Entwicklungsphase final spezifiziert werden konnten. So konnte man sich zum Beispiel nicht final darauf einigen, ob und wie die jeweils andere Abteilung (TPL bzw. TQ) über entdeckte Fehler innerhalb eines Musterstandes informiert werden soll.

Die BHTC wurde über alle entdeckten Prozess- und Programmablaufschwachstellen informiert und man kann diese eventuell in einer späteren Programmversion beheben.

7.2.1 Erfolge

Die durch die BHTC definierten Anforderungen konnten im vollen Rahmen erfüllt werden und die Anwendung zeichnet sich durch seine intuitive Handhabung aus. Das gewählte Vorgehensmodell (siehe auch Abbildung 3 „Zusammenspiel der Teilsysteme“) konnte mit einem großen Erfolg eingesetzt und umgesetzt werden, da bereits nach den ersten Entwicklungen Änderungswünsche durch die Fachabteilungen angemerkt werden konnten.

Der Verfasser hat sich in kürzester Zeit mit der Sprache der BHTC vertraut gemacht und konnte sich zusätzlich in die Umgebung ASP.NET und VB.NET einarbeiten. Durch kontinuierliches Nachfragen, konnten Missverständnisse bereits in ihrer Entstehung vermieden werden.

7.3 Ausblick

Mit der bereitgestellten Anwendung wurde ein solides Fundament für die Dokumentation von Prototypen entwickelt, auf dem nun weiter aufgebaut werden kann. Bereits kurz nach der Einführung des neuen Systems in der deutschen Niederlassung des Unternehmens, zeichneten sich die ersten Erweiterungswünsche ab, die anbei kurz zusammengefasst werden sollen:

Aktuell

- Einsatz der PTDB in der deutschen Niederlassung
- Überführen der alten Projektdaten aus der MuLi DB in die PTDB
- Implementierung weiterer Berichte zur Qualitätssicherung

kurzfristig (ab Januar 2008)

- Schulung weiterer Abteilungen und Unternehmensbereiche auf die PTDB
- Erfassen weiterer Anforderungen / Änderungswünsche (z. B. Schnittstellenentwicklung in andere Systeme des Unternehmens)
- Einsatz der PTDB auch in den internationalen Niederlassungen
- Implementierung einer eMail-Funktion, um Anwender über kritische Ereignisse (z. B. das Entfernen einer bereits erteilten Freigabe) automatisch zu informieren

langfristig (Planung 2008)

- Anpassung des Prototypenentstehungsprozesses, um während der Entwicklung des Konzeptes aufgefallene Schwachstellen auszubessern
- Implementierung weiterer Kontrollmechanismen, um den Ablauf des PEP innerhalb der PTDB weiter zu kontrollieren und zu reglementieren
- Implementierung eines Archivierungsmodules, um abgeschlossene Projekte separat zu sichern

7.4 Persönliches Resümee

Die gesamte Entwicklung der Anwendung „PTDB“, angefangen von der Ausarbeitung des Konzeptes bis hin zur Umsetzung in ASP.NET, hat mir sehr viel Freude bereitet.

Erstmalig war es mir möglich, die unterschiedlichen Themengebiete des im Studium erlernten Stoffes praktisch umzusetzen und die einzelnen Fachbereiche der Informatik miteinander zu „verweben“. Da ich bei diesem Projekt gleichzeitig mehrere Rollen übernehmen musste, war es mir ein Anliegen bereits zu Beginn des Projektes als selbstsicherer Entwickler und Projektmanager aufzutreten, um bei der BHTC einen kompetenten Eindruck zu hinterlassen.

Die Zusammenarbeit mit den zuständigen Ansprechpartnern der BHTC war immer überdurchschnittlich gut. So ist besonders die Zusammenarbeit mit der Abteilung „T-ZL“ hervorzuheben. Herr Thomas Lütteken hat auf meine Fragen immer umgehend reagiert und mich bei Verständnisschwierigkeiten tatkräftig unterstützt indem er mir komplexe Sachverhalte ausführlich und verständlich erläutert hat. Weiterhin war Herr Lütteken maßgeblich am Erfolg der Anwendung beteiligt, da er alle nur erdenklichen Vorgehensweisen innerhalb der Anwendung getestet hat und es somit möglich war, bereits in der Entwicklungsphase potentielle Fehler zu vermeiden. Die enge Zusammenarbeit mit Herrn Lütteken hat mir gezeigt, dass eine erfolgreiche Umsetzung mit der Unterstützung des Auftraggebers steht und fällt, denn ohne die erbrachte Zusammenarbeit wäre diese Entwicklung nicht möglich gewesen.

Weiterhin hat mir diese Diplomarbeit gezeigt, wie umfangreich Projekte innerhalb der Informationstechnologie sind und dass es nur schwer möglich ist, Projekte dieser Größenordnung zusätzlich zur „normalen Arbeit“ zu bewältigen. Ohne die Unterstützung meiner Frau, Anja Grahl, hätte ich diese Herausforderung wohl nicht schaffen können. Sie hat mir den Rücken während der gesamten Zeit freigehalten und stand mehr stets mit ihrem Rat zur Seite. Danke dafür!

Eidesstattliche Erklärung

Gemäß § 26 (1) der DPO erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbständig angefertigt habe. Ich habe mich keiner fremden Hilfe bedient und keine anderen, als die angegebenen Quellen und Hilfsmittel benutzt. Alle Stellen, die wörtlich oder sinngemäß veröffentlichten oder nicht veröffentlichten Schriften und anderen Quellen entnommen sind, habe ich als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Ort, Datum

(Vorname Nachname)

8 Abbildungsverzeichnis

Abbildung 1: Momentane Insellösung "MuLi DB"	7
Abbildung 2: Bindeglied PTDB.....	8
Abbildung 3: Zusammenspiel der Teilsysteme.....	12
Abbildung 4: einfacher Zeitstrahl.....	15
Abbildung 5: Teilsysteme im Zeitstrahl.....	15
Abbildung 6: Sprünge im Zeitstrahl.....	15
Abbildung 7: UML-Grobskizze SAP-Auftragserfassung	16
Abbildung 8: Windows - Authentifizierungsmethode des Datenbankservers	23
Abbildung 9: Gemischter Modus als Authentifizierungsmethode des Datenbankservers	24
Abbildung 10: Konzept der Web.config.....	25
Abbildung 11: Validierung auf ungültige Zeichen	29
Abbildung 12: Validierung eines Pflichtfeldes.....	30
Abbildung 13: Validierung eines Datums	31
Abbildung 14: benutzerdefinierte Validierung.....	32
Abbildung 15: Die drei Datenfamilien.....	35
Abbildung 16: UML-Diagramm Projekterstellung	36
Abbildung 17: Projektnavigation.....	38
Abbildung 18: UML-Diagramm Musterstanderstellung	39
Abbildung 19: Projektnavigation inkl. zugehöriger Musterstände	40
Abbildung 20: UML-Diagramm Prototyperstellung	41
Abbildung 21: UML-Diagramm Prototypen kopieren	42
Abbildung 22: UML-Diagramm Ablauf Datenerfassung.....	43
Abbildung 23: Durchschnittliche Dauer des Seitenaufbaus vor der Optimierung.....	45
Abbildung 24: Dateigröße der aufgebauten Seite vor der Optimierung	46
Abbildung 25: Durchschnittliche Dauer des Seitenaufbaus nach der Optimierung ..	48
Abbildung 26: Dateigröße der aufgebauten Seite nach der Optimierung.....	48
Abbildung 27: Navigationselement für Prototypen	49
Abbildung 28: Prototypenwahl via Navigations-Buttons am Anfang	49
Abbildung 29: Prototypenwahl via Navigations-Buttons am Ende	49
Abbildung 30: Prototypenwahl via Navigations-Buttons in der Mitte.....	50
Abbildung 31: Änderungen am Projekt im Read-Only-Modus	51
Abbildung 32: Änderungen am Projekt im Update-Modus.....	52

Abbildung 33: UML-Diagramm Projektänderungen	52
Abbildung 34: Anwender hinzufügen	53
Abbildung 35: zusätzlichen Auftrag erfassen	54
Abbildung 36: Auftragsauswahl bei der Prototypenerfassung	54
Abbildung 37: Zusätzliche Projektinformationen	55
Abbildung 38: Musterstand-Schaubild	56
Abbildung 39: Musterstand bearbeiten	57
Abbildung 40: Musterstand-Freigabe	59
Abbildung 41: Treeview mit geprüften Musterständen	59
Abbildung 42: Bearbeiten-Knopf der Prototypendatenpflege.....	60
Abbildung 43: Suche Prototyp-Link.....	61
Abbildung 44: Kommentar zum Prototyp erfassen	62
Abbildung 45: Prototypprüfung Eingabemaske	64
Abbildung 46: Prototypprüfung im Prüfschrank.....	65
Abbildung 47: UML-Diagramm Prototypen-Freigabe.....	66
Abbildung 48: UML-Diagramm Prototyp-Freigabe eines Referenztypen	66
Abbildung 49: UML-Diagramm Prototyp-Freigabe eines regulärer Prototypen	67
Abbildung 50: UML-Diagramm Prototyp-Freigabe eines Sonder-Typen.....	67
Abbildung 51: ER-Modell Projektsuche.....	70
Abbildung 52: Maske Projektsuche.....	70
Abbildung 53: Projektsuche - Ergebnistabelle.....	71
Abbildung 54: ER-Modell Prototypsuche ohne aktivierte Filter	74
Abbildung 55: ER-Modell Prototypsuche mit aktiviertem Filter	75
Abbildung 56: Prototypen-Suchmaske	76
Abbildung 57: Prototypen-Suche Ergebnistabelle	76
Abbildung 58: Report: Alle Projekte der Datenbank	80
Abbildung 59: Projektauswahl für Reporte	80
Abbildung 60: Projektauswahl inkl. Musterstandswahl für Reporte	82
Abbildung 61: Projektauswahl inkl. Prototypenwahl für Reporte.....	83
Abbildung 62: XML-Schema der XML-Datei	87

9 Tabellenverzeichnis

Tabelle 1: Abgrenzung der Authentifizierungsmethoden.....	21
Tabelle 2: Aufbauzeit bei dargestellten Prototypen*	45
Tabelle 3: Dateigröße bei dargestellten Prototypen	46
Tabelle 4: Aufbauzeit bei ausgeschlossenen Prototypen	47
Tabelle 5: Dateigröße bei ausgeschlossenen Prototypen	47

10 Abkürzungsverzeichnis / Glossar

Abkürzung	Beschreibung
ASP.NET	Active Server Pages .NET, ist eine serverseitige Technologie von Microsoft zum Erstellen von Webanwendungen auf Basis des Microsoft-.NET-Frameworks.
BHTC	Behr - Hella Thermocontrol GmbH, der Name des Unternehmens, für das die Prototypendatenbank erstellt wurde
CD	Compact Disc, ein scheibenförmiges Speichermedium
d.h.	das heißt
DB	Datenbank
DBMS	Datenbankmanagementsystem
DIN	Deutsches Institut für Normung e. V. ist die nationale Normungsorganisation der Bundesrepublik Deutschland mit Sitz in Berlin.
engl.	englisch
GB	Gigabyte
ggf.	gegebenenfalls
Ghz	Gigahertz
GmbH	Gesellschaft mit bestimmter Haftung
HTML	Hypertext Markup Language ist eine textbasierte Auszeichnungssprache zur Strukturierung von Inhalten wie Texten, Bildern und Hyperlinks in Dokumenten.

Abkürzung	Beschreibung
HTTPS	HyperText Transfer Protocol Secure (deutsch sicheres Hypertext-Übertragungsprotokoll) ist ein Schema, das eine zusätzliche, sichere Schicht zwischen HTTP und TCP definiert.
ID	Identifizier; deutsch Bezeichner
inkl.	inklusive
ISO	Internationale Organisation für Normung, internationale Vereinigung der Standardisierungsgremien
IT-System	Informationstechnisches System
lfd.	laufende
MB	Megabyte
MP3	MPEG-1 Audio Layer 3 (MP3) ist ein Dateiformat zur verlustbehafteten Audiodatenkompression
M-Protokoll	Musterprotokoll
MS	Microsoft
MuLi DB	Musterlisten-Datenbank; die ursprünglich eingesetzte Microsoft Access Datenbank zur Dokumentation von Prototypen
NAFTA	engl. North American Free Trade Agreement; das Nordamerikanische Freihandelsabkommen ist ein ausgedehnter Wirtschaftsverbund zwischen Kanada, den USA und Mexiko und bildet eine Freihandelszone im nordamerikanischen Kontinent
Nr.	Nummer
NTLM	NT LAN Manager ist ein Computer-Authentifizierungsschema
ODBC	Open Database Connectivity (dt. etwa: „Offene Datenbank-Verbindungsfähigkeit“) ist eine standardisierte Datenbank-schnittstelle, die SQL als Datenbanksprache verwendet.
OFK	Oberer Führungskreis (eine Rolle innerhalb des Musterentstehungsprozesses und gleichzeitig die Bezeichnung einer Abteilung innerhalb der BHTC)

Abkürzung	Beschreibung
PC	Personal Computer, ein stationärer Einzelplatzrechner
PDF	Portable Document Format (deutsch: übertragbares Dokumentenformat) ist ein plattformübergreifendes Dateiformat für Dokumente
PEP	Prototypenentstehungsprozess
PTDB	Prototypen-Datenbank
QS	Qualitätssicherung
RAM	engl. Random Access Memory, Halbleiterspeicher
SD	Security Descriptors
SID	Security Identifier (deutsch Identifikationsnummer)
SQL	Structured Query Language (deutsch Strukturierte Abfragesprache) ist eine Datenbanksprache zur Definition, Abfrage und Manipulation von Daten in relationalen Datenbanken.
SSL	Secure Sockets Layer ist ein Netzwerkprotokoll zur sicheren Übertragung von Daten u. a. von Internetseiten
Teilenr.	Teilenummer
TPL	Technischer Projektleiter (eine Rolle innerhalb des Musterentstehungsprozesses und gleichzeitig die Bezeichnung einer Abteilung innerhalb der BHTC)
TQ	Qualitätsplaner (eine Rolle innerhalb des Musterentstehungsprozesses und gleichzeitig die Bezeichnung einer Abteilung innerhalb der BHTC)
TS	Technische Spezifikation
T-ZL	Zentrale Labore (eine Rolle innerhalb des Musterentstehungsprozesses und gleichzeitig die Bezeichnung einer Abteilung innerhalb der BHTC)
u.a.	unter anderem

Abkürzung	Beschreibung
UML	Unified Modeling Language (deutsch Vereinheitlichte Modellierungs-Sprache) ist eine von der Object Management Group (OMG) entwickelte und standardisierte Sprache für die Modellierung von Software und anderen Systemen.
USA	United States of America
VB.NET	Visual Basic .NET ist vollständig objektorientiert. Der Quellcode wird bei der Kompilierung in Code der Common Intermediate Language, dem .NET-Pendant des Java-Bytecodes übersetzt. Dieser Code wird zur Laufzeit in Maschinencode umgesetzt und kann an die aktuelle Plattform angepasst werden.
XML	Extensible Markup Language (deutsch erweiterbare Auszeichnungssprache)
z.B.	zum Beispiel
z.T.	zum Teil
z.Z.	zur Zeit

11 Literaturverzeichnis

[ang1]

André Grahl, 03. Mai 2007, „Analyse zur Entwicklung einer web-basierten Datenbank zur technischen Dokumentation von Prototypen“

[asp1]

Lohrer, Matthias, 2003, „Einstieg in ASP.NET“, ISBN 978-3-89842-302-1

[it1]

Prof. Dr. Frank Viktor, 2004, „IT Consulting – Prozessbezogene IT-Konzepte“, Lerneinheit des Fachs IT-Consulting, 001357-001098

[ms1]

Girish Chander, James Hamilton, Willis Johnson, Richard Waymire, o. Datum, „Microsoft SQL Server 2003 SP – Sicherheitsfunktionen und bewährte Methoden“,
< <http://www.microsoft.com/germany/technet/datenbank/articles/600213.aspx#EID> > (10.12.2007)

[wiki1]

o. V., o. Datum, „Prototyp Wikipedia“,
< <http://de.wikipedia.org/wiki/Prototyp> > (14.12.2007)

[wiki2]

o. V., o. Datum, "Information Wikipedia",
< <http://de.wikipedia.org/wiki/Information#Informationswissenschaft> >
(03.12.2007)

[wiki3]

o. V., o. Datum, "SQL-Injection Wikipedia",
< <http://de.wikipedia.org/wiki/SQL-Injection> > (08.12.2007)

[wiki4]

o. V., o. Datum, "Kommentar Wikipedia",
< <http://de.wikipedia.org/wiki/Kommentar> > (14.12.2007)

[wiki5]

o. V., o. Datum, "Schnittstelle Wikipedia",
< <http://de.wikipedia.org/wiki/Schnittstelle> > (14.12.2007)

[wiki6]

o. V., o. Datum, "Sicht (Datenbank) Wikipedia",
< http://de.wikipedia.org/wiki/Sicht_%28Datenbank%29>
(14.12.2007)

[wiki7]

o. V., o. Datum, "Parser Wikipedia",
< <http://de.wikipedia.org/wiki/Parser> > (14.12.2007)

[wiki8]

o. V., o. Datum, "Fremdschlüssel Wikipedia",
< http://de.wikipedia.org/wiki/Foreign_Key#Fremdschl.C3.BCssel >
(14.12.2007)

[wiki9]

o. V., o. Datum, "Berichtswesen Wikipedia",
< <http://de.wikipedia.org/wiki/Reporting> > (14.12.2007)

[wiki10]

o. V., o. Datum, " Extensible Markup Language Wikipedia",
< <http://de.wikipedia.org/wiki/XML> > (14.12.2007)

[wiki11]

o. V., o. Datum, "Qualitätsmanagementnorm Wikipedia",
< <http://de.wikipedia.org/wiki/Qualit%C3%A4tsmanagementnorm> >
(14.12.2007)

[wiki12]

o. V., o. Datum, " ISO/TS 16949 Wikipedia",
< http://de.wikipedia.org/wiki/ISO/TS_16949 > (14.12.2007)

[wiki13]

o. V., o. Datum, "Suchfunktion Wikipedia",
< <http://de.wikipedia.org/wiki/Suchfunktion> > (14.12.2007)

12 Anhang

12.1 Funktion „MakeDBText“

Diese Funktion wird innerhalb des Programmcodes dazu benutzt, evtl. auftretende SQL-Injections zu bereinigen.

```
Public Function MakeDBText(ByVal strvalue As String) As String

    Dim ToCorrect As String = strvalue
    Dim CorrectedTxt As String = Replace(ToCorrect, "'", "'")
    CorrectedTxt = CorrectTxt.Replace("\", "\\")
    CorrectedTxt = CorrectTxt.Replace("%", "\%")
    CorrectedTxt = CorrectTxt.Replace("; ", "\; ")

    Return CorrectedTxt

End Function
```

Der zu bereinigende Text wird der Funktion als Übergabeparameter mitgegeben und der bereinigte Text wird als Rückgabewert geliefert.

Beispielaufruf:

```
MakeDBText( eingabe.Text )
```

12.2 Anhang: TreeView füllen inklusive Prototypen

```
''' <summary>
''' generate the treeview for all projects incl. prototypes
''' group projects and misc projects are not in scope for time measuring
''' </summary>
''' <remarks>test sub for measuring the time,
''' original sub is renamed to Fill_Project_Treeview_ORIG</remarks>
Public Sub Fill_Project_Treeview_W()
    Dim STARTTIME As DateTime = DateTime.Now           ' remember start time when function called
    tv_PEdit.Nodes.Clear()                            ' clear all existing entries in treeview

    ' first of all set entries for "my project",
    ' "group projects" and "ALL projects" and build
    ' treenodes for these groups
    ' =====
    Dim tnMAINmy As TreeNode = New TreeNode()
    tnMAINmy.Text = PTDBF.Get_Lang_from_DB(MyLang, "tv_MYProject")
    tnMAINmy.Value = "MY"
    tv_PEdit.Nodes.Add(tnMAINmy)
    tv_PEdit.Nodes(0).SelectAction = TreeNodeSelectAction.None
    Dim tnMAINgroup As TreeNode = New TreeNode()
    tnMAINgroup.Text = PTDBF.Get_Lang_from_DB(MyLang, "tv_GrpProject")
    tnMAINgroup.Value = "GROUP"
    tv_PEdit.Nodes.Add(tnMAINgroup)
    tv_PEdit.Nodes(1).SelectAction = TreeNodeSelectAction.None
    Dim tnMAINall As TreeNode = New TreeNode()
    tnMAINall.Text = PTDBF.Get_Lang_from_DB(MyLang, "tv_AllProject")
    tnMAINall.Value = "PUBLIC"
    tv_PEdit.Nodes.Add(tnMAINall)
    tv_PEdit.Nodes(2).SelectAction = TreeNodeSelectAction.None
```

```
' now read the projects from database
' which belongs to "my projects"
' =====
Dim SQLCmd_MyProjects As New SqlCommand("SELECT PERM.USERNAME, PERM.PROJECT_ID AS PID, " & _
    "PRO.name As PNAME FROM PTDB_permission AS PERM " & _
    "INNER JOIN PTDB_project AS PRO ON PERM.PROJECT_ID = " & _
    "PRO.PROJECT_ID WHERE (PERM.USERNAME = '" & _
    PTDBF.PTDBUser & "') AND (PERM.roll1 = 1) " & _
    "ORDER BY PRO.name", DBC1)

SQLCmd_MyProjects.Connection.Open()
Dim MY_Projects As SqlDataReader = SQLCmd_MyProjects.ExecuteReader()
While MY_Projects.Read()
    Dim tnMYPRJ As TreeNode = New TreeNode()
    tnMYPRJ.Text = MY_Projects("PNAME").ToString()
    tnMYPRJ.Value = MY_Projects("PID").ToString()
    tv_PEdit.Nodes(0).ChildNodes.Add(tnMYPRJ)
    Dim MYPID As Integer = tv_PEdit.Nodes(0).ChildNodes.IndexOf(tnMYPRJ)

    ' -----
    ' read the modelreleases of each project
    ' -----
    Dim SQLCmd_Modelrelease As New SqlCommand("SELECT modelclass,MRELEASE_ID FROM " & _
        "PTDB_modelrelease WHERE " & _
        "(PROJECT_ID = " & MY_Projects("PID") & ") " & _
        "ORDER BY modelclass", DBC2)

    SQLCmd_Modelrelease.Connection.Open()
    Dim MY_Projects_Modelrelease As SqlDataReader = SQLCmd_Modelrelease.ExecuteReader()
    While MY_Projects_Modelrelease.Read()

        ' count the prototypes of modelrelease
        ' *****
        Dim SQLCmd_PTCount As New SqlCommand("SELECT COUNT(*) FROM PTDB_prototype WHERE " & _
            "MRELEASE_ID=" & _
            MY_Projects_Modelrelease("MRELEASE_ID"), DBC3)
```



```
SQLCmd_PTCount.Connection.Open()
Dim PTCounter As Integer = SQLCmd_PTCount.ExecuteScalar()
SQLCmd_PTCount.Connection.Close()
Dim tnMYPRJMREL As TreeNode = New TreeNode()
tnMYPRJMREL.Text = MY_Projects_Modelrelease("modelclass") & ": " & _
    PTCounter.ToString()
tnMYPRJMREL.Value = MY_Projects_Modelrelease("MRELEASE_ID")

' use different treenode picture depending
' if a modelrelease is already approved or not
' *****
Dim DBC4 As New SqlConnection(PTDBF.ConnStr)
Dim SQLCmd_Approved As New SqlCommand("SELECT tq_control, tpl_control FROM " & _
    "PTDB_modelrelease WHERE MRELEASE_ID=" & _
    MY_Projects_Modelrelease("MRELEASE_ID"), DBC4)

SQLCmd_Approved.Connection.Open()
Dim ReaderApproval As SqlDataReader = SQLCmd_Approved.ExecuteReader()
While ReaderApproval.Read()
    If (ReaderApproval("tq_control")) And (ReaderApproval("tpl_control")) Then
        tnMYPRJMREL.ImageUrl = "images/APPROVED.gif"
    Else
        tnMYPRJMREL.ImageUrl = "images/OPEN.gif"
    End If
End While
ReaderApproval.Close()
SQLCmd_Approved.Connection.Close()

tv_PEdit.Nodes(0).ChildNodes(MYPID).ChildNodes.Add(tnMYPRJMREL)
Dim MYPJMRID As Integer = tv_PEdit.Nodes(0).ChildNodes(MYPID).ChildNodes.IndexOf(tnMYPRJMREL)

' -----
' read the prototypes of each modelrelease
' if any prototypes exist
' -----
```

```
    If PTCOUNTER > 0 Then
        Dim DBC5 As New SqlConnection(PTDBF.ConnStr)
        Dim SQLCmd_Prototype As New SqlCommand("SELECT tce_no, PROTOTYPE_ID FROM " & _
            "PTDB_prototype WHERE MRELEASE_ID=" & _
            MY_Projects_Modelrelease("MRELEASE_ID") & _
            "ORDER BY PROTOTYPE_ID", DBC5)

        SQLCmd_Prototype.Connection.Open()
        Dim ReaderPrototype As SqlDataReader = SQLCmd_Prototype.ExecuteReader()
        While ReaderPrototype.Read()
            Dim tnMYPRJMRELPT As TreeNode = New TreeNode()
            tnMYPRJMRELPT.Text = ReaderPrototype("tce_no") & " [PTID:" & _
                ReaderPrototype("PROTOTYPE_ID") & "]"
            tnMYPRJMRELPT.Value = ReaderPrototype("PROTOTYPE_ID")
            tv_PEdit.Nodes(0).ChildNodes(MYPID).ChildNodes(MYPJMRID).ChildNodes.Add(tnMYPRJMRELPT)
        End While
        ReaderPrototype.Close()
        SQLCmd_Prototype.Connection.Close()
    End If

    End While
    MY_Projects_Modelrelease.Close()
    SQLCmd_Modelrelease.Connection.Close()
End While
MY_Projects.Close()
SQLCmd_MyProjects.Connection.Close()

tv_PEdit.ExpandAll()
```

```
' before leaving the SUB
' calculate and print time span
' *****
Dim ENDTIME As DateTime = DateTime.Now           ' get the end time after treeview was builded
Dim TIMESPAN As TimeSpan = ENDTIME - STARTTIME   ' calculate the time for building the treeview
lbl_Name.Text = "TimeSpan=" & TIMESPAN.Milliseconds & " milisec." ' show time span in the GUI
End Sub
```

12.3 Anhang: Section-Handler „connectionStrings“ für den Datenbankzugriff

Im Folgenden ist der Section-Handler „appSettings“ für den DSN dargestellt:

```
<connectionStrings>
  <add name="Verbindungsname"
        connectionString="Data Source=Servername;
        Initial Catalog=Datenbankname;
        Integrated Security=True"
        providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Die einzelnen Parameter teilen sich wie folgt auf:

- **name**
Der Name der des Connectionsstrings ist unabhängig von der Art der Anwendung (Development, UAT oder Production) immer gleich. Durch diesen Namen wird der Zugriff im Code-Behind realisiert.
- **connectionString**
Über diesen Parameter wird der Servername angegeben, auf dem die Anwendung den SQL Server findet.
- **Initial Catalog**
Hier wird angegeben auf welche Datenbank die Anwendung innerhalb des SQL Server verweisen soll.
- **Integrated Security**
Über diesen Parameter wird die integrierte Sicherheit eingeschaltet.
- **providerName**
Hier wird der Anwendung mitgeteilt, die Verbindung zur Datenbank realisiert werden soll.

Je nach Art der Anwendung (Development, UAT oder Production) lässt sich auf einfache Weise der Zugriff auf eine andere Datenbank herstellen, da der jeweilige Name der Connection identisch ist.

12.3.1 Einstellung für die Production-Environment

```
<connectionStrings>
  <add name="PTDBCONN"
        connectionString="Data Source=Servername;
        Initial Catalog=PTDB;
        Integrated Security=True"
        providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Durch den Parameter "Initial Catalog" wird die Anwendung auf die Produktionsumgebung eingestellt.

12.3.2 Einstellung für die Development-Environment

```
<connectionStrings>
  <add name="PTDBCONN"
        connectionString="Data Source=Servername;
        Initial Catalog=PTDB_DEV;
        Integrated Security=True"
        providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Durch den Parameter "Initial Catalog" wird die Anwendung auf die Entwicklungsumgebung eingestellt.

12.3.3 Einstellung für die UAT-Environment

```
<connectionStrings>
  <add name="PTDBCONN"
        connectionString="Data Source=Servername;
        Initial Catalog=PTDB_UAT;
        Integrated Security=True"
        providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Durch den Parameter "Initial Catalog" wird die Anwendung auf die Testumgebung eingestellt.

12.4 Anhang: View zur Projektsuche


Die folgende View (deutsch Sicht; ist eine logische Relation) wurde in der Datenbank hinterlegt, um die Suche auf Projektebene zu vereinfachen:

```
CREATE VIEW v_PROJECT_ORDER_MODELREL_CUSTOMER
AS
    SELECT      P.PROJECT_ID,
               M.MRELEASE_ID,
               C.CUSTOMER_ID,
               O.ORDER_ID,
               P.name As projectname,
               P.synonym As projectsynonym,
               P.description As projectdescription,
               P.executive_construction As constructor,
               P.executive_development As developer,
               P.executive_testengineering As testengineer,
               P.part_naming As part_naming,
               P.bhtc_part_no As bhtc_part_no,
               O.order_no As order_no,
               O.reference_no As order_reference_no,
               O.purchaser As order_purchaser,
               M.description As modelreleasedescription,
               M.information As modelreleaseinformation,
               C.sap_id As customer_sap_id,
               C.external_reference As customer_external_ref
    FROM        PTDB_project As P,
               PTDB_order As O,
               PTDB_modelrelease As M,
               PTDB_customer As C
    WHERE
               (P.PROJECT_ID = O.PROJECT_ID)
    AND         (P.CUSTOMER_ID = C.CUSTOMER_ID)
    AND         (M.PROJECT_ID = P.PROJECT_ID)
```

12.5 Anhang: Beispielberichte

12.5.1 Musterstände eines Projektes

Musterstände
PTDB v.1.0
[P-J6-MR]



Interne Projektbezeichnung: Porsche Panamera Climatronik
 Beschreibung Projekt: Porsche Panamera - Beschreibung
 externe Projektbezeichnung: Porsche Panamera
 BHTC Teilnr.: 5HB000481- 0

M.release	description	Information	weight			
A-1	Teststand 1	Information 1	150,00			
				tq control	tq control date	tq controller
				-		
				tpi control	tpi control date	tpi controller
				-		
A-2	Test Musterstand 7-30-21-13 bla	keine zusätzl. Informationen zum Musterstand				
				tq control	tq control date	tq controller
				X	31.08.2007 20:22:55	ANDRE GRAHL
				tpi control	tpi control date	tpi controller
				X	31.08.2007 20:22:58	ANDRE GRAHL
B-1	M.Stand B1	M.Stand B1	1500,00			
				tq control	tq control date	tq controller
				-		
				tpi control	tpi control date	tpi controller

Behr-Hella Thermocontrol GmbH, Lippstadt den, 13.12.2007

Seite: 1/2

12.5.2 Musterstände eines Projektes inkl. Prototypen


Musterstände inkl. Prototypenanzahl PTDB v.1.0




Interne Projektbezeichnung: Porsche Panamera Climatronic
 Beschreibung Projekt: Porsche Panamera - Beschreibung
 externe Projektbezeichnung: Porsche Panamera
 BHTC Teilnr.: 5HB009461- 8
 Anzahl Prototypen: 49

Musterstand	Anzahl Prototypen
A-1	20
A-2	1
B-1	0
B-2	0
C-1	8
Erst-1	20
	<u>49</u>

12.5.3 Softwarestände eines Projektes

Softwarestände								
PTDB v.1.0								
[PJ6-SWR]								
Interne Projektbezeichnung:	Porsche Panamera Climatronic							
Beschreibung Projekt:	Porsche Panamera - Beschreibung							
externe Projektbezeichnung:	Porsche Panamera							
BHTC Teilnr.:	5HB009481- 8							
Beschreibung	Musterstand	Int. Kommentar	ext. Kommentar	Software-Funktion	n.I.	n.a.	n.I.	ok
Desc SWRel 10	A-1	internal comment on SWREL-Level	external comment on SWREL-Level	Function 1	-	X	-	X
				SWFUNC2	-	X	X	-
				SWFUNC3	-	-	-	X
				SWFUNC4	X	-	X	-
				swfunc 5	-	-	-	X
Softwarestand 21	A-2	internal comment on SWREL-Level	external comment on SWREL-Level	SW-Function 10-21	-	-	-	-
				SW-Function 10-22	X	-	-	-
				SW-Function 10-23	-	-	-	X
				SW-Function 10-24	X	X	X	-
				SW-Function 10-25	-	-	-	X
				SW-Function 10-26	X	-	-	X
SW36 - MStand B1	B-1	MStand B1	MStand B1	SW1	-	-	X	-
Behr-Hella Thermocontrol GmbH, Lippstadt den, 13.12.2007				Seite: 1/2				

12.5.4 Hardwarestände eines Projektes

Hardwarestände									
PTDB v.1.0									
[PJ6-HWR]									
Interne Projektbezeichnung:		Porsche Panamera Climatronic							
Beschreibung Projekt:		Porsche Panamera - Beschreibung							
externe Projektbezeichnung:		Porsche Panamera							
BHTC Teilnr.:		5HB009481-8							

Beschreibung	Musterstand	Board-Version	Int. Kommentar	ext. Kommentar	Teil-Nr.	n.I.	n.L.	n.A.	OK
H001 (Hardwarestand ID 8)	A-1	Board-Version 8	Interner Kommentar zu HWStand 8	Externer Kommentar zum HWStand 8	1	-	-	-	X
					ABD3	X	-	-	-
					ABD-Sipo	-	-	-	X
					DL	-	-	-	X
					DSTN-LC Display	-	-	-	X
					DTEL1	-	-	-	X
					LP Front	X	-	-	-
					LP Heck	X	-	-	-
					P3	X	-	-	-
Beschreibung Hardware-Stand 13	A-2	Board-Version 13	Interner Kommentar zu HWStand 13	Externer Kommentar zum HWStand 13	P1	-	-	-	X
					P2	-	X	-	-
Behr-Hella Thermocontrol GmbH, Lippstadt den, 13.12.2007					Seite: 1/2				

12.5.5 Konstruktionsstände eines Projektes

Konstruktionsstände PTDB v.1.0

[PJ6-CR]



Interne Projektbezeichnung: Porsche Panamera Climatronio
 Beschreibung Projekt: Porsche Panamera - Beschreibung
 externe Projektbezeichnung: Porsche Panamera
 BHTC Teilnr.: 5HB009481- 8

Beschreibung	Musterstand	int. Kommentar	ext. Kommentar	CAD-Zeichnungsnr.	CAD-Datum	CAD-Zeichner
Konstruktionsstand 19	A-1	ic ID=19	externer Kommentar zu CREL 19	CAD007_18_19	02.07.2007	Bert Baum -19
Description CREL 30	A-2	Text ist Text!	Text ist Text ist Text ist Text!	CAD-30a	30.07.2007	Carl Cadder [30]
Musterstand B1	B-1	MStand B1	MStand B1	CAD007_18_48	10.06.2007	CAD Zeichner CREL 48
Musetstand B2	B-2	MStand B2	MStand B2	CAD007_18_49	10.06.2007	CAD Zeichner CREL 49
C1-PO-CR	C-1	C1-PO-CR	C1-PO-CR	CAD-153	26.08.2007	A. Grahl
E1-P-CR	Erst-1	E1-P-CR	E1-P-CR	CAD-334213	26.08.2007	A. Grahl

12.5.6 Prototypen-Übersicht auf Projektebene

Prototypen-Übersicht

PTDB v.1.0

[J16-PT0]



Interne Projektbezeichnung: Porsche Panamera Climatronico
 Beschreibung Projekt: Porsche Panamera - Beschreibung
 externe Projektbezeichnung: Porsche Panamera
 BHTC Teilnr.: 5HB009461- 6
 Anzahl Prototypen: 49

PT Nr.	Matr.Nr.	Prod.datum	Softwarezustand	Konstruktionszustand	Hardwarezustand	Netzplan Nr.	Kd. Teile Nr.	Auftr. Nr.	TCE-Nr.
1	A-1		Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	1526	kundeteile nr	1955	TCE-1
2	A-1	21.07.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	221	MY kundt2a	1955	MY TCE2
3	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	kundt2	1955	TCE2
4	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	kundt2	1955	TCE2
5	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	kundt2	1955	TCE2
6	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	kundt2	1955	TCE2
7	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	kundt2	1955	TCE2
8	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	kundt2	1955	TCE2
9	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	1KD 820 047	1955	TCE2
10	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	1KD 820 047	1955	TCE2
11	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	1KD 820 047	1955	TCE2
12	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	1KD 820 047	1955	TCE2
13	A-1	13.05.2007	Desc SWRel 10	Konstruktionszustand 19	HD01 (Hardwarezustand ID 8)	2	1KD 820 047	1955	TCE2

12.5.7 Hardwareteile eine Musterstandes

Hardware-Teile PTDB v.1.0

[PJ6M4-HWP]



Interne Projektbezeichnung: Porsche Panamera Climatronik
 Beschreibung Projekt: Porsche Panamera - Beschreibung
 externe Projektbezeichnung: Porsche Panamera
 BHTC Teilernr.: 5HB009481- 6

Musterstand: A-1
 Hardwarestand: H001 (Hardwarestand ID 8)

Teile-Nr.	Int. Kommentar	ext. Kommentar	n.t.	n.a.	n.l.	ok	Link
1	Display 1	externer Kommentar zum Hardwareteil 1-8	-	-	-	X	
ABD3	Abdeckung	Abdeckung	X	-	-	-	
ABD-Slpo	Abdeckung-Gr mit Slpo ohne Silbe	Abdeckung-Gr mit Slpo ohne Silbe	-	-	-	X	
DL	Dekoreiste	Dekoreiste	-	-	-	X	
DSTN-LC Display	DSTN-LC Display	DSTN-LC Display	-	-	-	X	
DTEL1	Drehbedienelement	Drehbedienelement	-	-	-	X	
LP Front	Leiteplatte Front	Leiteplatte Front	X	-	-	-	
LP Heck	Leiteplatte Heck	Leiteplatte Heck	X	-	-	-	
P3	Display 1 - P3-8	externer Kommentar zum Hardwareteil P3-8	X	-	-	-	

12.5.8 Software-Funktionen eines Musterstandes

Softwarefunktionen PTDB v.1.0

[PJ6M4-SWFM]




Interne Projektbezeichnung: Porsche Panamera Climatronic
 Beschreibung Projekt: Porsche Panamera - Beschreibung
 externe Projektbezeichnung: Porsche Panamera
 BHTC Teilnr.: 6HB009461- 6


Musterstand: A-1
 Softwarestand: Desc SWRel 10

Software-Funktion	int. Kommentar	ext. Kommentar	n.t.	n.a.	n.l.	ok
Function 1	just an internal comment	Funktion umgesetzt	-	X	-	X
SWFUNC2	internal comment SWFUNC2	external comment SWFUNC2	-	X	X	-
SWFUNC3	int SWFUNC3	ext SWFUNC3	-	-	-	X
SWFUNC4	int SWFUNC4	ext SWFUNC4	X	-	X	-
swfunc 5	int swfunc 5	external swfunc 5	-	-	-	X

12.5.9 Software-Funktionen eines Prototypen

		Prototyp Funktionen PTDB v.1.0		
-Porsche Panamera- <small>[PJ6-PT20-MR4-OID17-PTF]</small>		Softwarestand: Desc SWRel 10 Hardwarestand: HD01 (Hardwarestand ID 8) Konstruktionsstand: Konstruktionsstand 19 ID-M-Protokoll:		
Teilebenennung: Teile - 1	Kundenteile-Nr. kundeteile nr	BHTC Teilnr. SHB009461- 6	CAD Nr.: CAD007.18_19	Prototyp-Nr. 1
Besteller / Verbleib: Manfred Mustermann		Status	CAD-Datum: 02.07.2007	Musterstand: A-1
Lieferdatum: 13.06.2007	Referenz-Nr.: refcust	Referenz-Datum: 13.06.2007	Auftragsnummer: 1955	TCE-Nr. TCE-1
Software-Funktion	n.t.	n.a.	n.i.	ok
Function 1	-	X	-	X
SWFUNC2	-	X	X	-
SWFUNC3	-	-	-	X
SWFUNC4	X	-	X	-
swfunc 5	-	-	-	X
Kommentar: external comment on SWREL-Level				

12.5.10 interner Prototypen-Lebenslauf

INTERNER Prototypenlebenslauf PTDB v.1.0 <small>[PJ6-PT20-MR4-01D17-I]</small>		 BHTC <small>COMFORT IN MOTION</small>				
Prototyp-Nr.	Kundenteile-Nr.	BHTC Teilernr.	TCE-Nr.			
1	kundeteile nr	5HB009461- 8	TCE-1			
Besteller / Verbleib:		Verw 1				
Lieferdatum:		13.06.2007				
Musterstand:		A-1				
Referenzdatum (Kunde):		13.06.2007				
Referenznummer (Kunde):		refcust				
Produktionsdatum:						
Konstruktionsstand:		Konstruktionsstand 19				
Hardwarestand:		HD01 (Hardwarestand ID 8)				
Softwarestand:		Desc SWRel 10				
Ausführender Konstruktion:		Carl Constructor				
Ausführender Entwicklung:		T. Lütkeken				
Ausführender Tester:		Ethan Engineer, Trevor Tester				
Bemerkungen:						
Datum	Bemerkungen					
19.07.2007 21:32:17	aaa					
21.07.2007 13:48:59	ACHTUNG ! Rückläufer !					
21.07.2007 15:50:56	sa					
13.12.2007 11:20:40	removed approval as new audit entered					
Hardware-Teile:						
Telle-Nr.	Int. Kommentar	ext. Kommentar	n.t.	n.a.	n.l.	ok
1	Display 1	externer Kommentar zum Hardwareteil 1-8	-	-	-	X
ABD3	Abdeckung	Abdeckung	X	-	-	-
ABD-Sipo	Abdeckung-Gr mit Sipo ohne Sibe	Abdeckung-Gr mit Sipo ohne Sibe	-	-	-	X
DL	Dekorielste	Dekorielste	-	-	-	X
DSTN-LC Display	DSTN-LC Display	DSTN-LC Display	-	-	-	X
DTEL1	Drehbedenelement	Drehbedenelement	-	-	-	X
LP Front	Lelleplatte Front	Lelleplatte Front	X	-	-	-
LP Heck	Lelleplatte Heck	Lelleplatte Heck	X	-	-	-
P3	Display 1 - P3-8	externer Kommentar zum Hardwareteil P3-8	X	-	-	-
Behr-Hella Thermocontrol GmbH, Lippstadt den, 13.12.2007				Seite: 1/1		

12.5.11 externer Prototypen-Lebenslauf

Prototyp-Nr.		Kundenteile-Nr.	BHTC Teilernr.	TCE-Nr.
1		kundeteile nr	5HB009461- 8	TCE-1

Prototypenlebenslauf
PTDB v.1.0
 [PJ6-PT20-MR4-OID17-E]

BHTC
 COMFORT IN MOTION

Besteller / Verbleib: Verw 1
Lieferdatum: 13.06.2007
Musterstand: A-1

Referenzdatum (Kunde): 13.06.2007
Referenznummer (Kunde): refcust
Produktionsdatum:

Konstruktionsstand: Konstruktionsstand 19
Hardwarestand: H001 (Hardwarestand ID 8)
Softwarestand: Desc SWRel 10

Ausführender Konstruktion: Carl Constructor
Ausführender Entwicklung: T. Lütkecken
Ausführender Tester: Ethan Engineer, Trevor Tester

Bemerkungen:

Datum	Bemerkungen
21.07.2007 15:50:56	sa

Hardware-Teile:

Telle-Nr.	Bemerkungen	n.I.	n.a.	n.L.	OK
1	externer Kommentar zum Hardwareteil 1-8	-	-	-	X
ABD3	Abdeckung	X	-	-	-
ABD-Sipo	Abdeckung-Gr mit Sipo ohne Sibe	-	-	-	X
DL	Dekorleiste	-	-	-	X
DSTN-LC Display	DSTN-LC Display	-	-	-	X
DTEL1	Drehbedienelement	-	-	-	X
LP Front	Lelleplatte Front	X	-	-	-
LP Heck	Lelleplatte Heck	X	-	-	-
P3	externer Kommentar zum Hardwareteil P3-8	X	-	-	-

Behr-Hella Thermocontrol GmbH, Lippstadt den, 13.12.2007 Seite: 1/1

12.5.12 XML-Report zu einem Prototyp

```
- <PROTOTYPE>
- <REPORT_DETAILS>
  <GENERATED_BY>ANDRE.GRAHL</GENERATED_BY>
  <DATE>13.12.2007 16:18:52</DATE>
  <APPLICATION_ID>PTDB v.1.0</APPLICATION_ID>
</REPORT_DETAILS>
<PROTOTYPE_ID>20</PROTOTYPE_ID>
<PROTOTYPE_NO>1</PROTOTYPE_NO>
<DELIVERY_DATE>13.06.2007</DELIVERY_DATE>
<PRODUCTION_DATE />
<REFERENCE_NO>Porsche Pana 1_18</REFERENCE_NO>
<REFERENCE_DATE>13.06.2007</REFERENCE_DATE>
<PURPOSE>Muster zur Verbauung Cockpit</PURPOSE>
<PART_TAG_COMMENT>Teileanhänger Comment ABC</PART_TAG_COMMENT>
<CUSTOMER_PART_NO>Kundeteile-Nr. 76</CUSTOMER_PART_NO>
<NETPLAN_NO>1528332</NETPLAN_NO>
<TCE_NO>TCE-12-15-23</TCE_NO>
<VARIETY_INDEX_NO>1</VARIETY_INDEX_NO>
<VARIETY_DESCRIPTION>Abart S.Misch-Vorlage</VARIETY_DESCRIPTION>
<PROTOTYPE_LINK />
- <CONTROLS>
- <TQ>
  <TQ_CONTROL_STATUS>False</TQ_CONTROL_STATUS>
  <TQ_CONTROLLER />
  <TQ_CONTROL_DATE />
</TQ>
- <TPL>
  <TPL_CONTROL_STATUS>False</TPL_CONTROL_STATUS>
  <TPL_CONTROLLER />
  <TPL_CONTROL_DATE />
</TPL>
- <TZL>
  <TZL_CONTROL_STATUS>False</TZL_CONTROL_STATUS>
  <TZL_CONTROLLER />
  <TZL_CONTROL_DATE />
</TZL>
- <OFK>
  <OFK_CONTROL_STATUS>False</OFK_CONTROL_STATUS>
  <OFK_CONTROLLER />
  <OFK_CONTROL_DATE />
</OFK>
</CONTROLS>
- <COMMENTS>
- <COMMENT_ID>
  5
  <ENTERED_BY>ANJA.GRAHL</ENTERED_BY>
  <COMMENT_DATE>19.07.2007 21:32:17</COMMENT_DATE>
  <COMMENT_MODE>I</COMMENT_MODE>
  <COMMENT>take care and be allegiant</COMMENT>
</COMMENT_ID>
</COMMENTS>
- <AUDITS>
```

```
- <AUDIT_ID>
65
<REASON>1</REASON>
<AUDIT_DATE>13.12.2007 11:20:35</AUDIT_DATE>
<M_PROTOCOLL>1</M_PROTOCOLL>
<TEST_PROTOCOLL>1</TEST_PROTOCOLL>
<LAP_NO>1</LAP_NO>
<STATUS>0</STATUS>
<ENTERED_BY>PETER.SIEKMANN</ENTERED_BY>
</AUDIT_ID>
</AUDITS>
- <PROJECT>
<NAME>Porsche Panamera Climatronic</NAME>
<SYNONYM>Porsche Panamera</SYNONYM>
<DESCRIPTION>Porsche Panamera - Beschreibung</DESCRIPTION>
<EXECUTIVE_CONSTRUCTION>Sebastian Misch</EXECUTIVE_CONSTRUCTION>
<EXECUTIVE_DEVELOPMENT>T. Lüttecken</EXECUTIVE_DEVELOPMENT>
<EXECUTIVE_TESTENGINEERING>Lisa Benneker, Anja
Grahl</EXECUTIVE_TESTENGINEERING>
<PART_NAMING>Teile-Naming - X1</PART_NAMING>
<BHTC_PART_NO>5HB009461- 6</BHTC_PART_NO>
- <INFORMATION>
- <INFORMATION-ID>
2
<ENTERED_BY>LISA.BENNEKER</ENTERED_BY>
<KIND>WARNING</KIND>
<DATE>23.07.2007 20:44:30</DATE>
<TEXT>potential RISK - FLAG RED - OFFEN</TEXT>
</INFORMATION-ID>
</INFORMATION>
</PROJECT>
- <MODELRELEASE>
<MODELRELEASE_ID>4</MODELRELEASE_ID>
<MODELCLASS>A-1</MODELCLASS>
<DESCRIPTION>Teststand 1</DESCRIPTION>
<INFORMATION>Information 1</INFORMATION>
<WEIGHT>150,00</WEIGHT>
- <CONTROLS>
- <TQ>
<TQ_CONTROL_STATUS>False</TQ_CONTROL_STATUS>
<TQ_CONTROLLER />
<TQ_CONTROL_DATE />
</TQ>
- <TPL>
<TPL_CONTROL_STATUS>False</TPL_CONTROL_STATUS>
<TPL_CONTROLLER />
<TPL_CONTROL_DATE />
</TPL>
</CONTROLS>
- <CONSTRUCTIONRELEASE>
<CONSTRUCTIONRELEASE_ID>19</CONSTRUCTIONRELEASE_ID>
<DESCRIPTION>Konstruktionsstand 19</DESCRIPTION>
<INTERNAL_COMMENT>ic ID=19</INTERNAL_COMMENT>
<EXTERNAL_COMMENT>externer Kommentar zu CREL 19</EXTERNAL_COMMENT>
<CAD_NO>CAD007.18_19</CAD_NO>
<CAD_DATE>02.07.2007</CAD_DATE>
<CAD_DRAWER>Bert Baum -19</CAD_DRAWER>
```

```

- <LINKS>
- <LINK_ID>
  42
  <DESCRIPTION>testlink</DESCRIPTION>
  <LINK>t:/test.pdf</LINK>
  </LINK_ID>
</LINKS>
</CONSTRUCTIONRELEASE>
- <HARDWARERELEASE>
  <HARDWARERELEASE_ID>8</HARDWARERELEASE_ID>
  <DESCRIPTION>H001 (Hardwarestand ID 8)</DESCRIPTION>
  <INTERNAL_COMMENT>Interner Kommentar zu HWStand
8</INTERNAL_COMMENT>
  <EXTERNAL_COMMENT>Externer Kommentar zum HWStand
8</EXTERNAL_COMMENT>
  <BOARD_VERSION>Board-Version 8</BOARD_VERSION>
- <HARDWAREPART>
- <PART_NO>
  1
  <INTERNAL_COMMENT>Display 1</INTERNAL_COMMENT>
  <EXTERNAL_COMMENT>externer Kommentar zum Hardwareteil 1-
8</EXTERNAL_COMMENT>
  <NOT_TESTED>False</NOT_TESTED>
  <NOT_APPLICABLE>False</NOT_APPLICABLE>
  <NOT_IMPLEMENTED>False</NOT_IMPLEMENTED>
  <OK>True</OK>
</PART_NO>
- <PART_NO>
  DSTN-LC Display
  <INTERNAL_COMMENT>DSTN-LC Display</INTERNAL_COMMENT>
  <EXTERNAL_COMMENT>DSTN-LC Display</EXTERNAL_COMMENT>
  <NOT_TESTED>False</NOT_TESTED>
  <NOT_APPLICABLE>False</NOT_APPLICABLE>
  <NOT_IMPLEMENTED>False</NOT_IMPLEMENTED>
  <OK>True</OK>
</PART_NO>
</HARDWAREPART>
- <LINKS>
- <LINK_ID>
  4
  <DESCRIPTION>Bestückungsliste 4-8</DESCRIPTION>
  <LINK>f:/best.pdf</LINK>
  </LINK_ID>
</LINKS>
- <PIN_CONFIGURATION>
- <PIN_NO>
  1
  <CONFIGURATION>Klemme 30</CONFIGURATION>
</PIN_NO>
- <PIN_NO>
  2
  <CONFIGURATION>5<y8</CONFIGURATION>
</PIN_NO>
</PIN_CONFIGURATION>
</HARDWARERELEASE>

```

```
- <SOFTWARERELEASE>
  <SOFTWARERELEASE_ID>10</SOFTWARERELEASE_ID>
  <DESCRIPTION>Desc SWRel 10</DESCRIPTION>
  <INTERNAL_COMMENT>internal comment on SWREL-
Level</INTERNAL_COMMENT>
  <EXTERNAL_COMMENT>external comment on SWREL-
Level</EXTERNAL_COMMENT>
- <LINKS>
- <LINK_ID>
  6
  <DESCRIPTION>SWLink Description 6-10</DESCRIPTION>
  <LINK>I:\BHTCInc\Testdokument-6-10.doc</LINK>
  </LINK_ID>
</LINKS>
- <SOFTWARE_FUNCTIONS>
- <FUNCTION_ID>
  5
  <FUNCTION>Function 1</FUNCTION>
  <INTERNAL_COMMENT>just an internal comment</INTERNAL_COMMENT>
  <EXTERNAL_COMMENT>Funktion umgesetzt</EXTERNAL_COMMENT>
  <NOT_TESTED>False</NOT_TESTED>
  <NOT_APPLICABLE>True</NOT_APPLICABLE>
  <NOT_IMPLEMENTED>False</NOT_IMPLEMENTED>
  <OK>True</OK>
</FUNCTION_ID>
</SOFTWARE_FUNCTIONS>
</SOFTWARERELEASE>
</MODELRELEASE>
</PROTOTYPE>
```

12.6 internes Auftragsformular

zuständiger Mitarbeiter <hr/> Dipl.-Ing. Dirk Petersen	Mustererstellung						
Lieferanschrift Kunde Firma Dr. Ing. h.c.F. Porsche AG Zentraler Wareneingang Porschestrasse 71287 Weissach	Nummer/Datum 7980 /23.03.2007 zust. Mitarbeiter bei K-K PETERSEN Bestellnummer /Datum 4500424612 /16.03.2007						
<hr/> <table border="1"> <thead> <tr> <th data-bbox="505 1062 540 1083">Pos.</th> <th data-bbox="591 1062 764 1104">Material / Bezeichnung Menge</th> </tr> </thead> <tbody> <tr> <td data-bbox="505 1146 581 1167">000010</td> <td data-bbox="591 1146 1117 1329"> Blende Kundenartikelnummer 970.653.557.00 FFF EM10 15 Stück Woche 16.2007 15 Stück Woche 18.2007 7 Stück Woche 20.2007 Baustufe LOS1 PT PSP-Element: 94013638 37 ST Liefertermin: Tag 16.04.2007 </td> </tr> <tr> <td data-bbox="505 1350 581 1371">000020</td> <td data-bbox="591 1350 1117 1533"> Z KLIEMABEDIENEINHEITHIGH-R/STZ Kundenartikelnummer 970.653.305.00 FFF EM10 15 Stück Woche 16.2007 20 Stück Woche 18.2007 9 Stück Woche 22.2007 Baustufe LOS1 PT PSP-Element: 94013638 44 ST Liefertermin: Tag 10.04.2007 </td> </tr> </tbody> </table>		Pos.	Material / Bezeichnung Menge	000010	Blende Kundenartikelnummer 970.653.557.00 FFF EM10 15 Stück Woche 16.2007 15 Stück Woche 18.2007 7 Stück Woche 20.2007 Baustufe LOS1 PT PSP-Element: 94013638 37 ST Liefertermin: Tag 16.04.2007	000020	Z KLIEMABEDIENEINHEITHIGH-R/STZ Kundenartikelnummer 970.653.305.00 FFF EM10 15 Stück Woche 16.2007 20 Stück Woche 18.2007 9 Stück Woche 22.2007 Baustufe LOS1 PT PSP-Element: 94013638 44 ST Liefertermin: Tag 10.04.2007
Pos.	Material / Bezeichnung Menge						
000010	Blende Kundenartikelnummer 970.653.557.00 FFF EM10 15 Stück Woche 16.2007 15 Stück Woche 18.2007 7 Stück Woche 20.2007 Baustufe LOS1 PT PSP-Element: 94013638 37 ST Liefertermin: Tag 16.04.2007						
000020	Z KLIEMABEDIENEINHEITHIGH-R/STZ Kundenartikelnummer 970.653.305.00 FFF EM10 15 Stück Woche 16.2007 20 Stück Woche 18.2007 9 Stück Woche 22.2007 Baustufe LOS1 PT PSP-Element: 94013638 44 ST Liefertermin: Tag 10.04.2007						