

## Prüfungsteil A

Prüfling (private Anschrift):	Ausbildungsbetrieb:
-------------------------------	---------------------

### Bestätigung über durchgeführte Projektarbeit

diese Bestätigung ist mit der Projektdokumentation einzureichen

Ausbildungsberuf (bitte unbedingt angeben):
---

Projektbezeichnung:
---------------------

Projektbeginn: _____	Projektfertigstellung: _____	Zeitaufwand in Std.: _____
----------------------	------------------------------	----------------------------

### Bestätigung der Ausbildungsfirma:

Wir bestätigen, dass der/die Auszubildende das oben bezeichnete Projekt einschließlich der Dokumentation im Zeitraum

vom: \_\_\_\_\_ bis: \_\_\_\_\_ selbständig ausgeführt hat.

Projektverantwortliche(r) in der Firma:

Vorname	Name	Telefon	Unterschrift
---------	------	---------	--------------

Ausbildungsverantwortliche(r) in der Firma:

Vorname	Name	Telefon	Unterschrift
---------	------	---------	--------------

### Eidesstattliche Erklärung:

Ich versichere, dass ich das Projekt und die dazugehörige Dokumentation selbständig erstellt habe.

Ort und Datum: \_\_\_\_\_ Unterschrift des Prüflings: \_\_\_\_\_



Abschlussprüfung Sommer 2017

Fachinformatiker für Anwendungsentwicklung  
Dokumentation zur betrieblichen Projektarbeit

# Globales Aktualisieren von Dokumenten

Computergestützte Betriebsprüfung - Abschluss & Dokumente

Abgabetermin: Berlin, den 02.06.2017

**Prüfungsbewerber:**

Guido Eckelt  
Boddinstraße 30  
12053 Berlin



**Deutsche  
Rentenversicherung  
Bund**

**Ausbildungsbetrieb:**

DEUTSCHE RENTENVERSICHERUNG Bund  
Ruhrstraße 2  
10704 Berlin

Dieses Werk einschließlich seiner Teile ist **urheberrechtlich geschützt**. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Autors unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Einspeicherung und Verarbeitung in elektronischen Systemen.



## Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>III</b>
<b>Tabellenverzeichnis</b>	<b>IV</b>
<b>Abkürzungsverzeichnis</b>	<b>V</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Projektumfeld . . . . .	1
1.2 Projektziel . . . . .	1
1.3 Projektbegründung . . . . .	2
1.4 Projektschnittstellen . . . . .	2
1.5 Projektabgrenzung . . . . .	2
<b>2 Projektplanung</b>	<b>3</b>
2.1 Projektphasen . . . . .	3
2.2 Ressourcenplanung . . . . .	3
2.3 Entwicklungsprozess . . . . .	4
<b>3 Analysephase</b>	<b>6</b>
3.1 Ist-Analyse . . . . .	6
3.2 Auszug aus dem Fachkonzept . . . . .	6
3.3 Wirtschaftlichkeitsanalyse . . . . .	6
3.3.1 „Make or Buy“-Entscheidung . . . . .	6
3.3.2 Projektkosten . . . . .	6
3.3.3 Amortisationsdauer . . . . .	7
3.4 Zwischenstand . . . . .	7
<b>4 Entwurfsphase</b>	<b>8</b>
4.1 Zielplattform . . . . .	8
4.2 Benutzeroberfläche . . . . .	8
4.3 Datenmodell . . . . .	8
4.4 Geschäftslogik . . . . .	9
4.5 Maßnahmen zur Qualitätssicherung . . . . .	10
4.6 Zwischenstand . . . . .	10
<b>5 Implementierungsphase</b>	<b>11</b>
5.1 Implementierung der Geschäftslogik . . . . .	11
5.2 Verwendete Entwurfsmuster . . . . .	11
5.3 Zwischenstand . . . . .	11
<b>6 Abnahmephase</b>	<b>12</b>



*Inhaltsverzeichnis*

---

6.1	Komponententest . . . . .	12
6.2	Abnahmetests . . . . .	12
6.3	Zwischenstand . . . . .	12
<b>7</b>	<b>Dokumentation</b>	<b>12</b>
7.1	Zwischenstand . . . . .	12
<b>8</b>	<b>Fazit</b>	<b>13</b>
8.1	Soll-/Ist-Vergleich . . . . .	13
8.2	Lessons Learned . . . . .	13
8.3	Ausblick . . . . .	13
	<b>Literaturverzeichnis</b>	<b>14</b>
	<b>Eidesstattliche Erklärung</b>	<b>15</b>
<b>A</b>	<b>Anhang</b>	<b>i</b>
A.1	Detaillierte Zeitplanung . . . . .	i
A.2	Klassendiagramm . . . . .	ii
A.3	Sequenzdiagramm . . . . .	ii
A.4	Screenshots der Anwendung . . . . .	iii
A.5	Testfälle . . . . .	v



## **Abbildungsverzeichnis**

1	Ausschnitt des Klassendiagramms . . . . .	9
2	Klassendiagramm zur Verwendung des Farbig-Entwurfsmusters . . . . .	11
3	Vollständiges Klassendiagramm . . . . .	ii
4	Sequenzdiagramm . . . . .	ii
5	Menüeintrag zum Anstoß des Globalen Aktualisierens . . . . .	iii
6	Dokumentenbaum mit den verschiedenen Dokumenttypen sortiert nach Prüfgebieten .	iv



## **Tabellenverzeichnis**

1	Zeitplanung . . . . .	3
2	Kostenaufstellung . . . . .	7
3	Zwischenstand nach der Analysephase . . . . .	7
4	Zwischenstand nach der Entwurfsphase . . . . .	10
5	Zwischenstand nach der Implementierungsphase . . . . .	11
6	Zwischenstand nach der Abnahmephase . . . . .	12
7	Zwischenstand nach der Dokumentation . . . . .	13
8	Soll-/Ist-Vergleich . . . . .	13



## **Abkürzungsverzeichnis**

<b>DRV Bund</b>	DEUTSCHE RENTENVERSICHERUNG Bund
<b>CBP-AD</b>	Computergestützte Betriebsprüfung - Abschluss & Dokumente
<b>CBP-NB</b>	Computergestützte Betriebsprüfung - Nachberechnung
<b>VB</b>	Visual Basic
<b>SDK</b>	Software Development Kit
<b>HTTP</b>	Hypertext Transport Protocol
<b>API</b>	Application Programming Interface
<b>ERM</b>	Entity-Relationship-Modell
<b>UML</b>	Unified Modeling Language
<b>GUI</b>	Grafische Benutzeroberfläche
<b>CI</b>	Corporate Identity



## 1 Einleitung

### 1.1 Projektumfeld

Die DEUTSCHE RENTENVERSICHERUNG Bund ([DRV Bund](#)) ist ein bundesweit tätiger Träger der gesetzlichen Rentenversicherung in der Bundesrepublik Deutschland mit ca. 17000 Mitarbeitern.

Zum Aufgabenfeld zählen:

- Bearbeitung von Rentenanträgen
- Überprüfung von Sozialabgaben auf Richtigkeit
- Beratung zu gesetzlichen Pflichten und Rechten

Für die Überprüfung von Sozialabgaben entwickelt die IT-Abteilung der [DRV Bund](#) verschiedene Anwendung, um diesen Prozess elektronisch zu vereinfachen.

Die Computergestützte Betriebsprüfung - Abschluss & Dokumente ([CBP-AD](#)) ist eine Desktopanwendung mit der jeweilige benötigte Dokumente, Anlagen und Schreiben für Betriebsprüfungen erzeugt werden können. Sie ist in Visual Basic ([VB](#)).NET mit einer Windows-Forms-Oberfläche implementiert. Die entsprechenden Daten zum Einbinden in die Dokumente werden über die Desktopanwendung Computergestützte Betriebsprüfung - Nachberechnung ([CBP-NB](#)), durch die Berechnungen zu den Sozialabgaben von Betrieben berechnet werden können, angefordert.

### 1.2 Projektziel

Die Betriebsprüfer erstellen und bearbeiten Dokumente und Anlagen, die mit Daten aus Berechnungen der Anwendung [CBP-NB](#) befüllt werden. Wenn in dieser Daten verändert werden, die schon in erzeugten Dokumenten vorkommen, sind diese Dokumente in einem „asynchronen“ Zustand und müssen vor Weiterverwendung aktualisiert werden.

Im Hauptmenü der [CBP-AD](#) soll ein neuer Menüeintrag bereitgestellt werden, dessen Kommando einen Aktualisierungsprozess anstößt, der alle Dokumente auf Asynchronität prüft und anschließend veraltete aktualisiert.

Für die verschiedenen Dokumenttypen gibt es zurzeit auch noch unterschiedliche Strukturen, wie die jeweiligen Dokumente neu erzeugt werden. Für die Funktionalität „Globales Aktualisieren“ sollen nun alle Dokumenttypen auf eine Struktur umgebaut werden.

Erzeugungsstrukturen für einige Dokumenttypen berechnen ihren Fortschritt eigenständig und geben diesen in eigenen Fenstern aus. Für diese soll eine Möglichkeit der Unterdrückung dieser Fortschrittsausgabe implementiert werden, damit der Aktualisierungsprozess „Globales Aktualisieren“ dies einheitlich für alle Dokumenttypen ausgeben kann.





### 1.3 Projektbegründung

Durch diese Erweiterung wird eine Vereinheitlichung der Dokumentenaktualisierung erreicht, die zugleich eine erhebliche Vereinfachung für den Anwender mit sich bringt.

### 1.4 Projektschnittstellen

Daten aus den Berechnungen der Sozialabgaben von Betrieben und ihren Mitarbeitern werden über eine [HTTP-API](#) der [CBP-NB](#) angefordert.

Die Benutzer der Anwendung sind die Betriebsprüfer der [DRV Bund](#).

### 1.5 Projektabgrenzung

Dieses Projekt zur Erweiterung der [CBP-AD](#) ist unabhängig von der Entwicklung der [CBP-NB](#), da nur bereits festgelegte Schnittstellen zum Datenaustausch benutzt werden und keine Änderung dieser notwendig sind.



## 2 Projektplanung

### 2.1 Projektphasen

Die Projektphase begann am 13.03.2017 und endete am 02.06.2016. Die tägliche Arbeitszeit betrug 7 Stunden 48 Minuten mit 30 Minuten Mittagspause. Die Projektarbeit fand nicht durchgängig statt, da Betriebsinterne Aufgaben und Ereignisse berücksichtigt werden müssen.

Projektphase	Teilzeit	Gesamtzeit
1. Analyse		5 h
1.1 Ist-Zustand	1 h	
1.2 Pflichtenheft	2 h	
1.3 Wirtschaftlichkeitsanalyse	2 h	
2. Entwurf		5 h
2.1 Klassendiagramm zur Architektur	3 h	
2.2 Sequenzdiagramm zur Abfolge	2 h	
3. Implementierungsphase		45 h
3.1 Aktualisierungsprozess „Globales Aktualisieren“	25 h	
3.2 Umbau der Dokumentenerzeugung	10 h	
3.3 Fortschrittsausgabe vereinheitlichen	10 h	
4. Qualitätssicherung		5 h
4.1 Unit-Tests	3 h	
4.2 Abnahme	2 h	
5. Dokumentation		10 h
5.1 Projektdokumentation	6 h	
5.2 Programmdokumentation	4 h	
<b>Gesamt</b>		<b>70 h</b>

Tabelle 1: Zeitplanung

Eine detailliertere Zeitplanung findet sich im Anhang [A.1: Detaillierte Zeitplanung](#) auf Seite [i](#).

### 2.2 Ressourcenplanung

#### Hardware

- Büroarbeitsplatz mit Tisch, Stuhl, Stromanschlüsse
- Arbeitsmaschine 1 mit Windows7
- Arbeitsmaschine 2 mit Kartenleser und Zugang zum Entwicklungsnetz der [DRV Bund](#)

#### Software



- Visual Studio Professional 2013 + .NET-Framework (mindestens v2.0)
- Software Development Kit ([SDK](#))s für Visual Studio
  - **CrytalReports for VisualStudio**  
Framework zum Erzeugung von PDFs aus Dokumenten
  - **GhostScript**  
Framework zum Drucken von Dokumenten
  - **TX-Spell-dotNet-0300**  
Framework zur Rechtschreibprüfung von Dokumenten
  - **TX-Text-Control-dotNet-1900**  
Framework zur Darstellung von Dokumenten
- MiKTeX - Distribution des Textsatzsystems TEX
- TeXStudio - Entwicklungsumgebung für L<sup>A</sup>T<sub>E</sub>X
- microTool inStep - Projektverwaltungstool für Arbeitsmaschine 2

#### Personal

- Projektbetreuer zur Unterstützung

## 2.3 Entwicklungsprozess

Die ausgewählte Vorgehensweise ist das Wasserfall-Modell<sup>1</sup>. Es ist konventionell vorgesehen, dass alle Schritte im Wasserfall-Modell sequentiell zu bearbeiten sind, d. h. Kein Schritt darf übersprungen werden und Ein Neustart oder Abbruch des Projektes bei nicht erfolgreichen Abschluss eines Schrittes. In der IT-Branche wird jedoch meist statt einem Projektneustart eher jeweilige Schritte zurückgegangen, da ein Neustart wirtschaftlich gesehen ein vieler größer Aufwand wäre.

### 1. Systemanforderungen:

Alle Anforderungen, die selbst nicht direkt das Software-Produkt betreffen, werden zunächst festgelegt. Dazu zählen:

- Preis
- Verfügbarkeit
- Sicherheitsaspekte
- Dokumentation

---

<sup>1</sup>Wasserfall-Modell nach [DR. WINSTON W. ROYCE](#)



**2. Softwareanforderungen:**

Alle Anforderung an die Software selber werden definiert. Jegliche Funktionen, Interaktionen und Besonderheiten werden konkretisiert, so dass sich aus den Systemanforderungen und Softwareanforderungen das Lastenheft ergibt.

**3. Analyse:**

Anforderungen aus Lastenheft und Ist-Zustand der Situation werden analysiert, so dass diese in ein Pflichtenheft umformuliert werden können. Die Wirtschaftlichkeit eines Projektes wird ebenfalls hier geprüft.

**4. Entwurf:**

Es wird das Datenmodell, die Architektur und die Schnittstellen zu anderen Anwendungen herausgearbeitet. Zwischenergebnisse können sein:

- Entity-Relationship-Modell (ERM)
- UML-Diagramme (Klassendiagramm, Sequenzdiagramm, Anwendungsfalldiagramm usw.)
- Mockups zur GUI
- Schnittstellen-Verzeichnis

**5. Implementierung:**

Umsetzung der Funktionalitäten nach Pflichtenheft und Entwurf in eine lauffähige Anwendung.

**6. Test/Qualitätssicherung:**

Es wird nach der Implementierungsphase die Software auf Fehler, Schwachstellen und Unstimmigkeiten überprüft. weitverbreitete Testmethoden:

- Komponententests (Unit-Test)
- Modultests
- Systemtests
- Integrationstests

**7. Inbetriebnahme:**

letzter Schritt in der Softwareentwicklung. Bei fehlerloser Überprüfung kann die Anwendung abgenommen werden und in Produktion gehen.



## 3 Analysephase

### 3.1 Ist-Analyse

Die Betriebsprüfer müssen die Dokumente einzeln über den Dokumenttypenbaum aktualisieren. Dies kann sehr aufwendig sein, da manche Berechnungen in mehreren Dokumenten aufgelistet werden und sind diese erst ausfindig zu machen, um dann aktualisiert werden zu können.

Diese Funktionalität ist von den Betriebsprüfern dringend erwünscht, da es eine enorme Zeitersparnis für sie ergeben würde.

### 3.2 Auszug aus dem Fachkonzept

„Hier wird ein Auszug aus dem Fachkonzept stehen“

### 3.3 Wirtschaftlichkeitsanalyse

#### 3.3.1 „Make or Buy“-Entscheidung

Da die Entwicklung der **CBP-AD** ein internes Projekt der **DRV Bund** ist und nur eine Funktionserweiterung ist, lässt sich kein fertiges Produkt finden, dass alle Anforderungen, vor allem fachliche, erfüllt. Daher wird dieses Projekt in Eigenentwicklung von der IT-Abteilung der **DRV Bund** umgesetzt.

#### 3.3.2 Projektkosten

Die Kosten für die Durchführung des Projekts setzen sich sowohl aus Personal-, als auch aus Ressourcenkosten zusammen. Laut Tarifvertrag<sup>2</sup> verdient ein Auszubildender im dritten Lehrjahr pro Monat 949,02 € Brutto.

$$8 \text{ h/Tag} \cdot 220 \text{ Tage/Jahr} = 1760 \text{ h/Jahr} \quad (1)$$

$$949.02 \text{ €/Monat} \cdot 12,9 \text{ Monate/Jahr} \approx 12242,36 \text{ €/Jahr} \quad (2)$$

$$\frac{12242,36 \text{ €/Jahr}}{1760 \text{ h/Jahr}} \approx 6,96 \text{ €/h} \quad (3)$$

Es ergibt sich also ein Stundenlohn von 6,96 €. Die Durchführungszeit des Projekts beträgt 70 Stunden. Für die Nutzung von Ressourcen<sup>3</sup> wird ein pauschaler Stundensatz von 15 € angenommen. Für

---

<sup>2</sup>Tarifvertrag des öffentlichen Dienstes

<sup>3</sup>Räumlichkeiten, Arbeitsplatzrechner etc.



### 3 Analysephase

die anderen Mitarbeiter wird pauschal ein Stundenlohn von 25 € angenommen. Eine Aufstellung der Kosten befindet sich in Tabelle 2 und sie betragen insgesamt 1697,20 €.

Vorgang	Zeit	Kosten pro Stunde	Kosten
Entwicklungskosten	70 h	6,96 € + 15 € = 21,96 €	1537,20 €
Fachgespräch	3 h	25 € + 15 € = 40 €	120,00 €
Abnahmetest	1 h	25 € + 15 € = 40 €	40,00 €
			<b>1697,20 €</b>

Tabelle 2: Kostenaufstellung

#### 3.3.3 Amortisationsdauer

Bei einer Zeiteinsparung von 10 Minuten am Tag für jeden der 4000 Betriebsprüfer und 220 Arbeitstagen im Jahr ergibt sich eine gesamte Zeiteinsparung von

$$4000 \cdot 220 \text{ Tage/Jahr} \cdot 10 \text{ min/Tag} = 8800000 \text{ min/Jahr} \approx 146667 \text{ h/Jahr} \quad (4)$$

Dadurch ergibt sich eine jährliche Einsparung von

$$146667 \text{ h} \cdot (25 + 15) \text{ €/h} = 5866680 \text{ €} \quad (5)$$

Die Amortisationszeit beträgt also  $\frac{1697,20 \text{ €}}{5866680 \text{ €/Jahr}} \approx 0,0002 \text{ Jahre} \approx 2 \text{ Stunden}$ .

#### 3.4 Zwischenstand

Tabelle 3 zeigt den Zwischenstand nach der Analysephase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Analyse des Ist-Zustands	1 h	1 h	
2. „Make or buy“-Entscheidung und Wirtschaftlichkeitsanalyse	2 h	2 h	

Tabelle 3: Zwischenstand nach der Analysephase



## 4 Entwurfsphase

### 4.1 Zielplattform

CBP-AD ist als Desktopanwendung in VB.NET, wie in 1.1: [Projektumfeld](#) erwähnt, auf Basis des .NET-Framework Version 2.0 implementiert. Sie wird als x86-Anwendung für zurzeit Windows-XP und Windows-7 entwickelt.

### 4.2 Benutzeroberfläche

Die GUI der CBP-AD ist, wie in 1.1: [Projektumfeld](#) erwähnt, als Windows-Forms-Oberfläche realisiert. Das Aussehen der Steuerelemente ist durch Corporate Identity (CI) vordefiniert. Für diese Funktionalität sind keine Änderung an der GUI nötig, da die Oberfläche zur Fortschrittsausgabe schon in bereits bestehenden Klassen implementiert wurde.

### 4.3 Datenmodell

Die Funktionalität „Globales Aktualisieren von Dokumenten“ beinhaltet keine Speicherung von Entitäten, daher ist kein neues Datenmodell erforderlich.



## 4.4 Geschäftslogik

Das Kommando stellt die Schnittstelle zur Oberfläche zum Aufrufen der Funktionalität dar. Dieses instantiiert dann seinen Prozess, der dann wiederum seinen Ueberarbeiter instantiiert. Der Ueberarbeiter ist das zentrale Logikgerüst dieser Funktionalität. Er delegiert Prüfungs- und Aktualisierungsaufgaben an seine internen Bearbeiter, die dann je nach Aufgabe diese selbst implementieren oder an entsprechende Klasse weitergeben. Die Ausführung und Fortschrittsausgabe dieses Prozess ist in den Klassen JobgruppeAllgemein und JobAllgemein gekapselt.

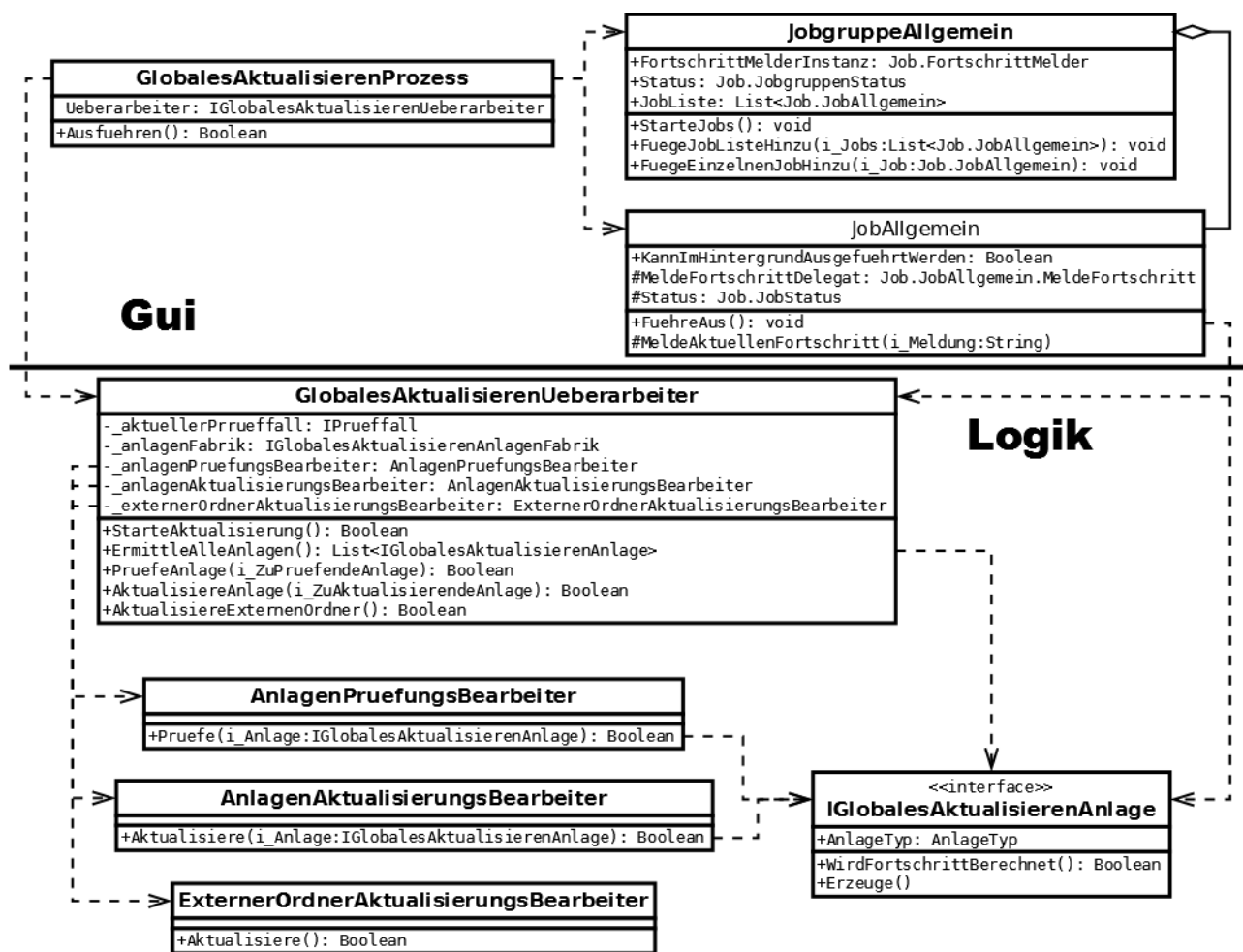


Abbildung 1: Ausschnitt des Klassendiagramms

Das vollständige Klassendiagramm kann im Anhang [A.2: Klassendiagramm](#) auf Seite [ii](#) eingesehen werden. Der Ablauf des Prozesses kann im Anhang [A.3: Sequenzdiagramm](#) auf Seite [ii](#) eingesehen werden.





## 4.5 Maßnahmen zur Qualitätssicherung

Die Funktionalität "Globales Aktualisieren" wird durch die Komponententests auf korrekte Ausführung mit der Entwicklungsumgebung geprüft. Für die Einführung in die nächste Release-Version der Computergestützte Betriebsprüfung - Abschluss & Dokumente werden nochmal Verbundtests, bei denen alle Funktionalitäten in Verbindung mit den anderen Anwendung der Computergestützte Betriebsprüfung geprüft werden, durch die Abteilung durchgeführt.

## 4.6 Zwischenstand

Tabelle 4 zeigt den Zwischenstand nach der Entwurfsphase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Erstellen eines UML-Klassendiagramms	3 h	3 h	
2. Erstellen eines UML-Sequenzdiagramms	2 h	2 h	

Tabelle 4: Zwischenstand nach der Entwurfsphase



## 5 Implementierungsphase

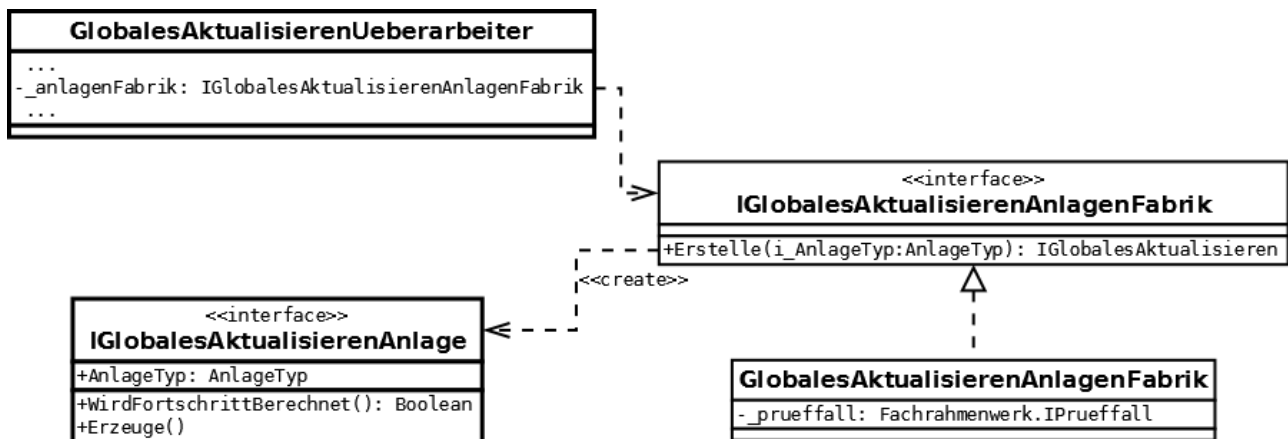


Abbildung 2: Klassendiagramm zur Verwendung des Farbik-Entwurfsmusters

## 5 Implementierungsphase

### 5.1 Implementierung der Geschäftslogik

Der Prüfungsalgorithmus zur Bestimmung der Asynchronität der Anlage ist im AnlagenPruefungsBe-  
 arbeiter. Jede Aktualisierungslogik ist in der Methode „Erzeuge“ der jeweiligen konkreten IGlobalesAktualisierenAn-  
 lage Klasse implementiert.

### 5.2 Verwendete Entwurfsmuster

Für die Instanziierung der einzelnen konkreten Objekte der Schnittstelle „IGlobalesAktualisierenAn-  
 lage“ verwendete ich das Fabrik-Entwurfsmuster nach folgenden Schema.

### 5.3 Zwischenstand

Tabelle 5 zeigt den Zwischenstand nach der Implementierungsphase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Aktualisierungsprozess „Globales Aktualisieren“	25 h	25 h	
2. Umbau der Dokumentenerzeugung	10 h	10 h	
3. Fortschrittsausgabe vereinheitlichen	10 h	10 h	

Tabelle 5: Zwischenstand nach der Implementierungsphase



## 6 Abnahmephase

### 6.1 Komponententest

Für die in 4.5 beschriebenen Komponententests befinden sich Screenshots von Auszügen im Anhang A.5: Testfälle auf Seite v.

### 6.2 Abnahmetests

Bei erfolgreichen Komponententests prüft ein Facharbeiter mittels Code-Review gegen, ob sich auffällige Stelle, Flüchtigkeitsfehler oder ähnliches vorliegen. Bei vorläufiger Abnahme eines Facharbeiters ist die Erweiterung soweit zum nächsten Verbundtest<sup>4</sup> in die Anwendung integriert zu werden, sodass bestätigt werden kann, ob sich alle Funktionen der Anwendung weiterhin fehlerfrei ausführen lassen und sich an die Richtlinien zur Barrierefreiheit und Usability gehalten wird.

### 6.3 Zwischenstand

Tabelle 6 zeigt den Zwischenstand nach der Abnahmephase.

Vorgang	Geplant	Tatsächlich	Differenz
1. Abnahmetest der Fachabteilung	2 h	2 h	

Tabelle 6: Zwischenstand nach der Abnahmephase

## 7 Dokumentation

Die Entwicklerdokumentation zum Quellcode ist mittels C#Doc automatisch generiert worden.

Die Benutzer bekommen bei Neuerungen an der Anwendung einen Hinweis auf ein durch die Fachabteilung erstelltes Dokument, auf dem alle geänderten und neuen Funktionen verzeichnet und erläutert sind.

### 7.1 Zwischenstand

Tabelle 7 zeigt den Zwischenstand nach der Dokumentation.

<sup>4</sup>System- und Benutzeroberflächentests aller Programme der Fachabteilungen



Vorgang	Geplant	Tatsächlich	Differenz
1. Erstellen der Programmdokumentation	4 h	4 h	
2. Erstellen der Projektdokumentation	6 h	8 h	+2 h

Tabelle 7: Zwischenstand nach der Dokumentation

## 8 Fazit

### 8.1 Soll-/Ist-Vergleich

Wie in Tabelle 8 zu erkennen ist, konnte die Zeitplanung bis auf wenige Ausnahmen eingehalten werden.

Phase	Geplant	Tatsächlich	Differenz
Analysephase	5 h	6 h	+1 h
Entwurfsphase	5 h	5 h	
Implementierungsphase	45 h	45 h	
Komponententests	2 h	2 h	
Abnahmetest der Fachabteilung	2 h	2 h	
Erstellen der Dokumentation	10 h	12 h	+2 h
Gesamt	70 h	73 h	

Tabelle 8: Soll-/Ist-Vergleich

### 8.2 Lessons Learned

Durch Projekte wie dieses wird einem erst bewusst, wie wichtig Anforderungen und Spezifikationen sind, da sie zum Teil die einzigen Anhaltspunkte zur Feststellung des Entwicklungsfortschritts sind

### 8.3 Ausblick

Es sind keine Erweiterungen der Funktionalität geplant.



## **Literaturverzeichnis**

### **Dr. Winston W. Royce**

DR. WINSTON W. ROYCE: *Managing the developement of large software systems*. <https://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf>, Abruf: 17.01.2017



## **Eidesstattliche Erklärung**

Ich, Guido Eckelt, versichere hiermit, dass ich meine **Dokumentation zur betrieblichen Projektarbeit** mit dem Thema

*Globales Aktualisieren von Dokumenten – Computergestützte Betriebsprüfung - Abschluss  
& Dokumente*

selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Berlin, den 02.06.2017

---

GUIDO ECKELT



## A Anhang

### A.1 Detaillierte Zeitplanung

<b>Analysephase</b>	<b>5 h</b>
1. Analyse des Ist-Zustands	1 h
1.1. Fachgespräch mit der EDV-Abteilung	1 h
2. Erstellen eines Pflichtenheftes	2 h
2. „Make or buy“-Entscheidung und Wirtschaftlichkeitsanalyse	2 h
<b>Entwurfsphase</b>	<b>5 h</b>
1. Erstellen eines UML-Klassendiagramms der Anwendung	3 h
2. Erstellen eines UML-Sequenzdiagramms des Hauptprozesses	2 h
<b>Implementierungsphase</b>	<b>45 h</b>
1. Aktualisierungsprozess „Globales Aktualisieren“	25 h
1.1. Verallgemeinertes Interface für Anlagen,Dokumenten etc.	5 h
1.2. Aufrufen der jeweiligen Dokumentenerzeugungsprozesse	10 h
1.3. Aufrufen der jeweiligen Dokumentenerzeugungsprozesse	10 h
2. Umbau der Dokumentenerzeugung	10 h
3. Fortschrittsausgabe vereinheitlichen	10 h
<b>Abnahmetest der Fachabteilung</b>	<b>5 h</b>
1. Abnahmetest der Fachabteilung	2 h
<b>Erstellen der Dokumentation</b>	<b>10 h</b>
1. Erstellen der Projektdokumentation	6 h
2. Programmdokumentation	4 h
<b>Gesamt</b>	<b>70 h</b>

```

classDiagram
    class IGlobalesAktualisierenUebarerbeiter {
        <<interface>>
        +StarteAktualisierung(): Boolean
        +ErmittleAlleAnlagen(): List<IGlobalesAktualisierenAnlage>
        +PruefeAnlage(I_Anlagen: IGlobalesAktualisierenAnlage): Boolean
        +AktualisiereAnlage(i_Anlagen: IGlobalesAktualisierenAnlage): Boolean
        +AktualisiereExternenOrdner(): Boolean
    }
    class GlobalesAktualisierenUebarerbeiter {
        -aktuellerPrueffall: IPrueffall
        -anlagenFabrik: IGlobalesAktualisierenAnlagenFabrik
        -anlagenPruefungsBearbeiter: AnlagenPruefungsBearbeiter
        -anlagenAktualisierungsBearbeiter: AnlagenAktualisierungsBearbeiter
        -externerOrdnerAktualisierungsBearbeiter: ExternerOrdnerAktualisierungsBearbeiter
    }
    class ExternerOrdnerAktualisierungsBearbeiter {
        +Aktualisiere(): Boolean
    }
    class AnlagenAktualisierungsBearbeiter {
        +Aktualisiere(i_Anlage: IGlobalesAktualisierenAnlage): Boolean
    }
    class AnlagenPruefungsBearbeiter {
        +Pruefe(i_Anlage: IGlobalesAktualisierenAnlage): Boolean
    }
    class IGlobalesAktualisierenAnlage {
        <<interface>>
        +AnlageTyp: AnlageTyp
        +MeldeFortschrittBerechnet(): Boolean
        +Erzeuge()
    }
    class GlobalesAktualisierenAnlagenErzeugungsJob {
        #Status: Job.JobStatus
        #MeldeFortschrittDelegate: Job.JobAllgemein.MeldeFortschritt
        FuehreAus(): void
    }
    class GlobalesAktualisierenAnlagenPruefungsJob {
        #Status: Job.JobStatus
        #MeldeFortschrittDelegate: Job.JobAllgemein.MeldeFortschritt
        FuehreAus(): void
    }
    class JobgruppeAllgemein {
        +FortschrittMelderInstanz: Job.FortschrittMelder
        +Status: Job.JobgruppenStatus
        +JobsListe: List<Job.JobAllgemein>
        +StarteJobs(): void
        +FuegeJobIstehInzu(i_Jobs: List<Job.JobAllgemein>): void
        +FuegeEinzelnenJobHinzuz(i_Job: Job.JobAllgemein): void
    }
    class JobAllgemein {
        +KannImHintergrundAusgefuehrtWerden: Boolean
        #MeldeFortschrittDelegat: Job.JobAllgemein.MeldeFortschritt
        #Status: Job.JobStatus
        +FuehreAus(): void
        #MeldeAktuellenFortschritt(i_Meldung:String)
    }
    class FortschrittMelder {
        +Melde(i_Meldung:String): void
        +New(i_MelDefortschrittDelegat: Job.FortschrittMelder.MeldeFortschritt)
    }
    class JobgruppenStatus {
        +ErfolgreichAusgefuehrt: Boolean
        +GesammelteFehlerBeiJobAusfuehrung: List<System.Exception>
    }
    class JobStatus {
        #ErfolgreichAusgefuehrt: Boolean
        #FehlerLiesesTabbruchAus: Boolean
        +GeworfeneException: System.Exception
    }
    class KommandoMitGesiichertemBearbeitungszustand {
        +Name: String
        +Kommandokategorie: ApplikationsRahmenwerk.Kommandokategorie
        +Schluessel: String
        +Icon: System.Drawing.Icon
        +Tastenkuerzel: System.Windows.Forms.Shortcut
        +Aktiviert: Boolean
        +Sichtbar: Boolean
        +Standard: Boolean
        +Angekreuzt: Boolean
        +Ausfuehren(io_Fehler:FehlerRahmenwerk.IFehler): Boolean
        +KommandoSpezifischesAusfuehren(io_Fehler:FehlerRamenwerk.IFehler): Boolean
        +PruefeBearbeitungszustand(): Boolean
    }
    class GlobalesAktualisierenKommando {
        +Ausfuehren(io_Fehler:FehlerRahmenwerk.IFehler): Boolean
        +KommandoSpezifischesAusfuehren(io_Fehler:FehlerRahmenwerk.IFehler): Boolean
    }

    IGlobalesAktualisierenUebarerbeiter <|-- GlobalesAktualisierenUebarerbeiter
    IGlobalesAktualisierenUebarerbeiter <|-- ExternerOrdnerAktualisierungsBearbeiter
    IGlobalesAktualisierenUebarerbeiter <|-- AnlagenAktualisierungsBearbeiter
    IGlobalesAktualisierenUebarerbeiter <|-- AnlagenPruefungsBearbeiter
    IGlobalesAktualisierenUebarerbeiter <|-- IGlobalesAktualisierenAnlage
    IGlobalesAktualisierenAnlage <|-- GlobalesAktualisierenAnlagenErzeugungsJob
    IGlobalesAktualisierenAnlage <|-- GlobalesAktualisierenAnlagenPruefungsJob
    JobgruppeAllgemein *-- JobAllgemein
    JobgruppeAllgemein *-- FortschrittMelder
    JobgruppeAllgemein *-- JobgruppenStatus
    JobgruppeAllgemein *-- JobStatus
    GlobalesAktualisierenKommando --> GlobalesAktualisierenProzess
    GlobalesAktualisierenKommando --> KommandoMitGesiichertemBearbeitungszustand
    
```

The diagram illustrates the separation of logic and GUI components. The left side contains logic classes, including interfaces like `IGlobalesAktualisierenUebarerbeiter` and `IGlobalesAktualisierenAnlage`, concrete worker classes like `GlobalesAktualisierenUebarerbeiter`, `AnlagenAktualisierungsBearbeiter`, and `AnlagenPruefungsBearbeiter`, and job classes like `GlobalesAktualisierenAnlagenErzeugungsJob` and `GlobalesAktualisierenAnlagenPruefungsJob`. The right side contains GUI classes, including `GlobalesAktualisierenKommando`, `KommandoMitGesiichertemBearbeitungszustand`, `FortschrittMelder`, `JobgruppenStatus`, and `JobStatus`. Dashed arrows indicate dependencies from GUI components to logic components.

### A.3 Sequenzdiagramm







#### A.4 Screenshots der Anwendung

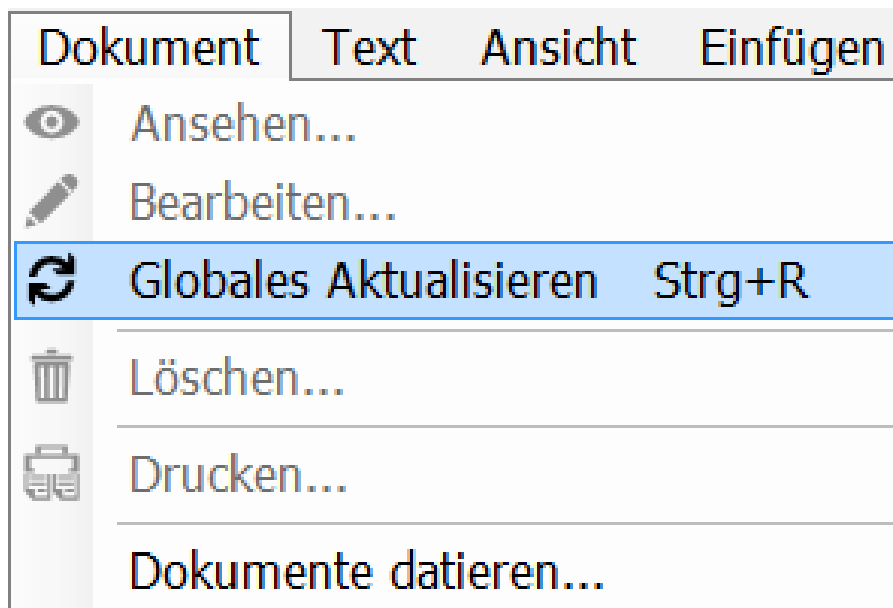


Abbildung 5: Menüeintrag zum Anstoß des Globalen Aktualisierens

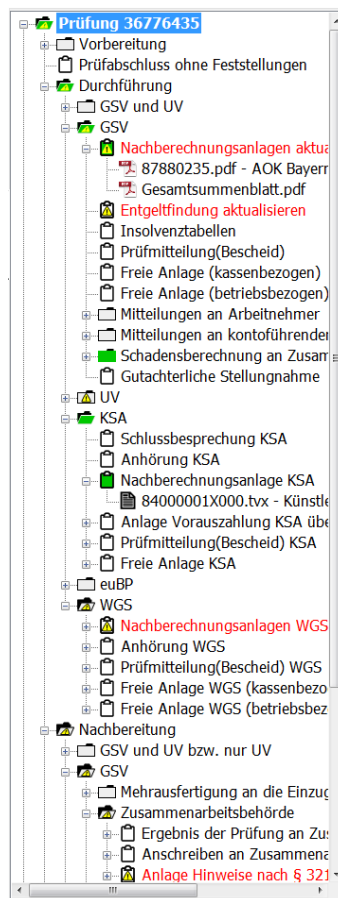


Abbildung 6: Dokumentenbaum mit den verschiedenen Dokumenttypen sortiert nach Prüfgebieten



## **A.5 Testfälle**

„hier könnte ihre Testfälle stehen“