Inhalt

[Einführung 2](#_Toc530384453)

[Auftrag 2](#_Toc530384454)

[Einsatzfeld des Produktes 2](#_Toc530384455)

[Ist-Zustand 2](#_Toc530384456)

[Soll-Zustand 2](#_Toc530384457)

[Analyse 2](#_Toc530384458)

[Analyse des Problems 2](#_Toc530384459)

[Abgrenzung des zu entwickelnden Systems 2](#_Toc530384460)

[Beschreibung der Schnittstellen 2](#_Toc530384461)

[Gewünschtes Verhalten/ Effekte 2](#_Toc530384462)

[Grafik über die Erstellung eines Prozesses 2](#_Toc530384463)

[Grafik über die Nutzung eines Prozesses 2](#_Toc530384464)

[Produkt-/Funktionsübersicht 2](#_Toc530384465)

[Muss-Funktionen 2](#_Toc530384466)

[Soll-Funktionen 2](#_Toc530384467)

[Kann-Funktionen 2](#_Toc530384468)

[Praktische Umsetzung der Produkt-/Funktionsübersicht 3](#_Toc530384469)

[Muss-Funktionen 3](#_Toc530384470)

[Soll-Funktionen 3](#_Toc530384471)

[Kann-Funktionen 3](#_Toc530384472)

[Beurteilung der Machtbarkeit 3](#_Toc530384473)

[Projektstrukturplan 3](#_Toc530384474)

[Zeitplanung 3](#_Toc530384475)

[Phasenplan 3](#_Toc530384476)

[Ablaufkontrolle und Meilensteine 3](#_Toc530384477)

[Anforderungen an die Qualität 3](#_Toc530384478)

[Einsatz von Techniken und Tools 3](#_Toc530384479)

[Tests 3](#_Toc530384480)

[Ergänzungen/ Sonstiges 3](#_Toc530384481)

# Einführung

Das Projekt des Antrags- und Formularverwaltungstool wurde uns am 13.11.2018 vorgeschlagen. Es handelt sich um ein Tool, was die internen Prozesse in der Berufsschule ATIW in Paderborn vereinfacht.

# Auftrag

## Einsatzfeld des Produktes

Das Produkt soll lediglich intern im ATIW von den Lehrkräften und Schülern genutzt werden.

## Ist-Zustand

Am ATIW müssen Dokumente händisch über Word- oder PDF-Vorlagen ausgefüllt, gedruckt und unterzeichnet werden. Danach folgt die Weitergabe an die nächst höhere Distanz, die oftmals eine weitere Unterschrift bzw. gegebenenfalls Änderungen am Dokument vornehmen muss. Dadurch entsteht oftmals ein erhöhter Aufwand an Druckerpapier, Laufwege und Zeit.

## Soll-Zustand

Bestmöglich sollte ein Tool erstellt werden, welches das händische Ausfüllen der Vorlagen ersetzt und die zu prüfenden Daten in einer Datenbank abbilden kann. Dies soll den Zeit und Papier bzw. Laufaufwand deutlich senken.

# Analyse

## Analyse des Problems

Das Hauptproblem liegt in dem persönlichen überbringen der Dokumente zur nächsten Instanz. Sowie keinem einheitlichen Bearbeitungstool oder Ablagesystem für die Vorlagen.

## Abgrenzung des zu entwickelnden Systems

Aufgrund strenger zeitlicher Begrenzungen soll das Projekt nicht dem Anspruch einer vollfunktionsfähigen Verwaltungslösung genügen. Vielmehr soll es als gedanklicher Ansatz betrachtet werden, der die Grundlage für eine Erweiterung der hier bereitgestellten Grundfunktionen bietet.

In diesem Sinne werden Schnittstellen bereitgestellt, sowie Programmfunktionen gekapselt um leichte Anpassung und Erweiterung möglich zu machen.

## Zusammenfassung der Systemarchitektur

Das System ist grob in drei Komponenten unterteilt. Einen Datenbankserver auf MySQL Basis, einen Webserver zur Kommunikation mit dem HTTP-Protokoll, der in .NET entwickelt wird und alle grundlegenden Funktionalitäten bereitstellt, sowie einem Frontend für die User selbst, ebenfalls in .NET, das auf diese Funktionen zurückgreift.

## Beschreibung der Schnittstellen

Die Kommunikation zwischen MySQL-Server und http-Server wird über die Standard .NET Bibliotheken realisiert und geschieht in Reaktion auf Interaktionen auf der Clientseite, die mit dem Webserver über eine REST-Schnittstelle kommuniziert (auch diese werden über .NET Bibliotheken umgesetzt). REST (für Representational State Transfer) bietet eine standardisierte Umsetzung einer Webschnittstelle an, die stateless ist, also ohne Sessions auskommt und stattdessen alle für eine Aktion benötigten Informationen wie Nutzernamen, Passwort und weiteres mitüberträgt. Als Vorteil ergibt sich zum Beispiel die Möglichkeit, auf die komplexe Verwaltung aktiver Usersessions zu verzichten, damit die Server zu entlasten und die Codebasis aufgeräumter zu halten. Zudem müssen weniger Ausnahmefälle berücksichtigt werden. Die .NET Standardbibliotheken reduzieren den Dokumentations- und Wartungsaufwand den externe Bibliotheken mit sich bringen würden.

## Gewünschtes Verhalten/ Effekte

Die Software soll die vielen verschiedenen Prozesse in einer einheitlichen Oberfläche zusammenfassen die leicht zu überblicken und Nutzen ist und es obsolet macht sich den Prozessablauf für jedes Dokument einzuprägen. Darüber hinaus reduziert die Verlagerung des Fokus auf digitale Verarbeitung den Papieraufwand und erhöht damit die Übersichtlichkeit.

### Grafik über die Erstellung eines Prozesses

### Grafik über die Nutzung eines Prozesses

## Produkt-/Funktionsübersicht

### Muss-Funktionen

* XML-Prozessdateien überprüfen
  + Auf Struktur
  + Auf Syntax
  + Allgemeine Korrektheit
* XML-Prozessdateien interpretieren
  + Daraus eine DB-Tabelle erzeugen
  + Daraus Objekte erzeugen
* UI dynamisch aus Objekten erzeugen
  + Einträge aus der UI in die erzeugte DB-Tabelle füllen
* Ermöglichung allgemeiner Prozessgestaltung
  + Genehmigung
  + Zielperson eines Prozessschrittes festlegen
  + Belege erzeugen
    - Automatische Befüllung von Word-Vorlagen nach der XML-Prozessdatei
  + E-Mail Bestätigung

### Soll-Funktionen

* Prozessfortschritt in DB hinterlegen
* Benachrichtigungssystem (Programmintern)
* Rollensystem
* Standardrollen (Sekretärin, Schulleiter etc.)
* Erstellung, Bearbeitung und Aktualisierung von XML-Prozessdateien von Serverseite

### Kann-Funktionen

* Einfache Rechenoperationen
* Eingabebedingungen

## Praktische Umsetzung der Produkt-/Funktionsübersicht

### Muss-Funktionen

#### XML-Prozessdateien überprüfen

Um sicherzustellen, dass im weiteren Programmablauf keine Fehler geschehen muss sichergestellt sein, dass Prozessdateien in sich korrekt und schlüssig sind. Zum Beispiel sollten Benutzeroberflächen-Elemente immer nur in ihren zugehörigen „<Dialog>“ Tags gefasst werden, statt frei im Dokument zu stehen.

* + Auf Struktur

Es muss sichergestellt sein, dass die allgemeine Struktur eines Dokuments eingehalten wird. Die allgemeine Struktur ist wie folgt:

***<process …>***

***<process-step …>***

***<dialog …>***

***<[inputelements] …/>***

***</dialog>***

***<[processelements] …/>***

***</process-step>***

***</process>***

* + Auf Syntax

Die Syntax muss eingehalten werden. Ein Beispiel wäre die Syntax für die Verwendung von Platzhaltern. Platzhalter werden so formuliert: „[Placeholder]“ und rühren her von technischen Namen in anderen Teilen der XML-Datei, die aber ohne Klammern formuliert werden.

Ein Beispiel wäre ein Eingabetag im ersten Prozessschritt, der später weiterverwendet wird:

***<process …>***

***<process-step …>***

***<dialog …>***

***<teacher-dropdown name=”teacher”…/>***

***</dialog>***

***</processstep>***

***<process-step …>***

***<mail-notify target=”[teacher]” …/>***

***</process-step>***

***</process>***

* + Allgemeine Korrektheit

An dieser Stelle wird die Einhaltung des allgemeinen Konzeptes geprüft. Das heißt, dass zum Beispiel kein Prozessschritt ohne Interaktion bestehen darf. Gibt es zum Beispiel weder ein Eingabefeld (Dialog -> Teacherdropdown) noch eine Genehmigungsaufforderung (Validate) sondern nur eine E-Mailbenachrichtigung (Mail-Notify) ist das kein gültiger Prozessschritt, da er obsolet wäre.

#### XML-Prozessdateien zu Objekten interpretieren

Zu diesem Zweck wird die XML-Datei Tag für Tag ausgelesen. Zu oberst liegt ein Process Objekt, dem dann eine Liste aus Process-Step Objekten zugeordnet wird. Jeder Process-Step hält dann eine Liste aus Dialog-Objekten, denen dann zuletzt Input-Elemente zugeordnet werden. Ebenfalls den Process-Steps untergeordnet, ist eine Auflistung an Prozessinteraktionen (Validate, etc.) und an Automatisierungen (Mail-Notify etc.). Diese Struktur ermöglicht die komplette Abbildung des Prozesses in .NET Objekten. Somit muss die XML-Datei nur einmal eingelesen werden und danach kann der komplette Prozess davon unabhängig abgearbeitet werden. Dadurch ist der Quelltext besser separiert und lesbar.

#### UI dynamisch aus Objekten erzeugen

Die Benutzeroberfläche wird in Dialogverarbeitung gestaltet sein. Der Dialog Tag in einer Prozessdatei beschreibt einen eigenen Block mit Benutzeroberflächenelementen. Es wird immer nur ein Block auf einmal angezeigt.

Es gibt verschiedene Elemente zur Benutzereingabe, doch alle gehen auf die gleichen Basisklassen zurück. Das ermöglicht eine einheitliche Listenführung. In diesem Sinne haben alle Benutzereingabe-Objekte mehrere Eigenschaften. Die genaue Struktur ist dem Klassendiagramm zu entnehmen. Gesagt sei aber, dass jedem Objekt ein WPF Control hinterlegt ist, das in der Benutzeroberfläche erscheinen soll. Das Eingabe-Objekt dient als Container, der dieses Control erzeugt und mit Daten befüllt. Das kann zum Beispiel durch Anbindung an eine Datenbank geschehen.

Beispielsweise ist „teacher-dropdown“ eine normale Combobox zugeordnet. Diese wird von der Containerklasse mit einer Lehrerliste aus der Datenbank befüllt.

##### Ermöglichung allgemeiner Prozessgestaltung

Die Orientierung dieser Softwarelösung am Prozessdenken macht es notwendig, für jeden Prozess-Schritt festlegen zu können, wer diesen durchführen soll. Dazu muss die Möglichkeit bestehen einen Prozess in der Mitte abzubrechen wenn Situationen eintreten in denen er nicht mehr genehmigt werden kann. Tritt ein solcher Fall ein, dann muss der Nutzer informiert werden können, nicht nur in der Software selbst, die die notwendigen Informationen aus der Datenbank beziehen kann, sondern auch per E-Mail, sollte er derzeit keinen Zugriff haben.

Gesetzliche Vorgaben können es zuweilen nötig machen, zur Papierform zurückzukehren. Deshalb muss die Software eine Möglichkeit bieten, Belege in Form von Word Vorlagen vorzubereiten, diese um Platzhalter anzureichern (Name, Adresse, etc.), wobei diese auch dem bereits bekannten Platzhalterformat „[name]“ folgen und diese auf den Server hochzuladen. In der XML Datei kann dann der Name der Vorlage angegeben und diese automatisch befüllt werden. Platzhalter für die es keine Äquivalente gibt werden durch Leerzeichen ersetzt.

### Soll-Funktionen

#### Prozessfortschritt verfolgen

Die Datenbank muss Tabellen beinhalten, mit denen es möglich ist den Fortschritt eines angestoßenen Prozesses zu überprüfen. Die Software zeigt das dann im unteren Teil des Fensters als Fortschrittsanzeige an. In der XML-Prozessdatei erhalten die Prozess-Schritte auch Beschreibungstexte, diese werden dann für den aktuellen Schritt als Text in der UI angezeigt.

#### Benachrichtigungssystem

Die Client-Software stellt in regelmäßigen Abständen Anfragen an den Server, ob neue Prozessinstanzen bereitstehen die ein Mitarbeiter abarbeiten muss, sowie ob es für durch einen Mitarbeiter erstellte Prozesse Aktualisierungen gibt, z.B. diese abgelehnt wurden.

Der Webserver übermittelt den Status der aktuell für den User hinterlegten Prozesse und nach Abgleich der Fortschrittsdaten erfragt der Client alle weiteren benötigten Informationen.

#### Rollensystem

Dem bisher vorgestellten Konzept muss ein Rollensystem zu Grunde liegen um datenschutzrechtlichen Anforderungen zu genügen. In diesem Sinne wird jedoch darauf verzichtet für jeden Lehrer eigene Rollen zuzuweisen. Stattdessen erhalten nur die Teilnehmer eines Prozesses die übermittelten Informationen. Zuletzt gibt es Standardrollen, die in der Datenbank leicht austauschbar sind. Zu diesen gehören zum Beispiel Schulleiter und Sekretariat. Mitglieder dieser Rolle können in den XML Dateien über diese Bezeichnungen angesprochen werden.

Beispiel:

***<process-step target=“Secretary“ …>***

Es ist also keine Platzhalternotation von Nöten. Findet sich in einem target Attribut ein Text ohne Klammern wird davon ausgegangen, dass es sich um eine Standardrolle handelt und diese über die Datenbank abgeglichen.

#### Verwaltung und Erreichbarkeit von Prozessdateien

Um die verschiedenen Prozesse allgemein Verfügbar zu machen, werden diese über einen Teil des Clienttools auf den HTTP-Server hochgeladen und werden von dort bei Anfragen bezogen. So können die Prozesse zentral verwaltet und aktualisiert werden, was Verwirrung vermeiden und Standardisierung erleichtern kann.

### Kann-Funktionen

Bei Zeiten kann es ebenfalls nötig sein, Eingaben auf Bedingungen zu prüfen oder einfache Berechnungen durch Formeln in der XML-Datei durchzuführen. Diese würden .NET seitig als Strings aufgeschlüsselt, in Formeln umgewandelt, Platzhalter aufgelöst und dann berechnet werden.

## Beurteilung der Machtbarkeit

### Projektstrukturplan

### Zeitplanung

### Phasenplan

### Ablaufkontrolle und Meilensteine

## Anforderungen an die Qualität

## Einsatz von Techniken und Tools

# Tests

# Ergänzungen/ Sonstiges