

Dokumentation „Demonstrator“

Einleitung

Die folgende Dokumentation zeigt wichtige Lösungsansätze aus dem Erstellprozess des Demonstrators auf. Da der dortige Programmcode als Prototyp ausgelegt wurde und der Fokus auf ein rasches Abbilden der GUI-Funktionalitäten lag, können bestehende Codeteile nur bedingt direkt übernommen werden. Sie können aber als Vorlage dienen. Aus dem gleichen Grund werden auch bekannte Probleme aufgezeigt.

Entwicklungstools

Für die Entwicklung des Demonstrators wurde Microsoft Visual Studio 2008 und Microsoft Expression Blend verwendet. Im Visual Studio wurde die Codelogik generiert. Für das Designen hat sich Blend besser bewährt als das Visual Studio. Es ist keine spezielle Version nötig zum Bearbeiten der Dateien, jedoch muss beides installiert sein.

Das Programm basiert auf dem Microsoft .NET Framework 3.5. Speziell verwendet es die Windows Presentation Foundation (WPF) zur Formuldarstellung. Diese wurde aufgrund der vielfältigen Möglichkeiten zur Gestaltung der GUI verwendet und hat sich gut bewährt.

Projektstruktur

Das Programm ist auf 4 Hauptprojekte aufgegliedert:

- Burkolter_Control und Custom Controls: In diesen beiden Projekten befinden sich alle Steuerelemente. In Custom Controls sind alle für den Demonstrator erstellten Benutzersteuerelemente eingefügt worden. Zudem ist auch die Datenlogik hier angesiedelt.
- Help Editor: Dieses Projekt beinhaltet einen Editor für die Hilfedatei.
- Demonstrator: Dies ist das eigentliche Hauptprojekt. Es beinhaltet das Hauptformular.

Die Programmlogik zur Datenmanipulation (Hauptsächlich Anlegen, Löschen und Sortieren) befindet sich im Benutzerdefinierten DataSet (dsDemonstrator). Die restliche Programmlogik zur Datenmanipulation wird über die WPF bereitgestellt.

Die restlichen Dateien beinhalten hauptsächlich Formulare für die verschiedenen GUI – Elemente. Diese werden vom Hauptformular wieder zusammengefasst.

Datenmodell

Die Daten werden alle in einem XML-File abgelegt. Das dazugehörige Datenschema befindet sich in der Datei dsDemonstrator.xsd.

Die eigentliche Prüfung beinhaltet 3 Kernelemente:

- Prüfung: Hier werden die verschiedenen Prüfungen abgelegt.
- Prüfungsteil: Dieser ist immer einer Prüfung zugeordnet. Hier kann die Prüfung in verschiedene Fachgebiete aufgeteilt werden.

- Aufgabenblock: Ein Aufgabenblock ist immer einem Prüfungsteil zugeordnet. Innerhalb eines Prüfungsteils können die Aufgaben wiederum frei gruppiert werden.
- Aufgabe: Die Aufgabe ist immer einem Aufgabenblock zugeordnet. Die Aufgabe schlussendlich beinhaltet alle Angaben zur Prüfungsfrage inklusive Musterlösung.

Alle weiteren Tabellen sind als Nachschlagelisten zu sehen.

Der Aufgabentext wird als Flowdokument in der Datenbank abgelegt. Die Datenhaltung in einer grossen Datenbank sollte für die entsprechenden Textfelder nochmals überprüft werden.

Grundsätzlich hat sich diese Aufteilung für die Datenspeicherung bewährt. Auch kann so jederzeit die ganze Prüfung wieder abgerufen werden.

Eine zukünftige Anforderung an das Tool ein Aufgabenpool. Dieser sollte zwingend von den Daten in den Prüfungen getrennt werden. Insbesondere, da dies andernfalls verschiedene Schwierigkeiten mit der Vermeidung von Seiteneffekten mit sich bringt (Bsp: Aufgabe 1 in Prüfung A wird editiert, diese ist jedoch auch Aufgabe 4 in Prüfung B).

Es gilt sich auch zu überlegen, ob eine Sperrung der Prüfung nötig ist. Beispielsweise nachdem die Prüfung freigegeben wurde.

Daneben existiert ein simples Datenschema für das Hilfesystem (Eine Tabelle mit dem Link auf die Datendatei und Hierarchierungsoptionen, eine Tabelle für die Schlagworte mit Gewichtung). Weitere Abklärungen haben ergeben, dass die erstellten HTML-Dokumente in die von der Entwicklungsumgebung angebotenen Möglichkeiten integriert werden können. Dies sollte aufgrund von besseren Möglichkeiten in der Beschlagwortung und Suche durchgeführt werden.

Erstellen der Aufgabentexte / Musterlösungen

Für die Erstellung der Aufgabentexte wurde der sogenannte Fliesstext aus dem WPF-Framework verwendet. Dabei wurde ein einfacher Editor aus dem Internet (RichTextEditor) als Steuerelement integriert. Dieser erlaubt die grundlegendsten Bearbeitungsmöglichkeiten. Das Einfügen von weiteren Editiermöglichkeiten wird jedoch nötig sein.

Für ein einfacheres Weiterverarbeiten sollten Ergänzungen überlegt werden. Aufgrund von Abklärungen sollten diese alle mit diesem Fliesstextformat realisierbar sein:

- Einfügemöglichkeit für Tabellen. Diese werden in Flowdokumenten unterstützt. Dies scheint jedoch eine komplexere Angelegenheit zu sein, wenn man diese beliebig manipulieren will (Zeilen anfügen, Spalten löschen, etc.). Ein fixes Einfügen von X Spalten und Y Zeilen ist einfach lösbar.
- Editor für Formeln einbinden. Dies geschieht momentan über Grafiken.
- Bessere Möglichkeiten zum Einfügen von Bildern. Grössenänderungen sind noch nicht möglich.
- Erstellen von Formatvorlagen ähnlich wie in Word für verschiedene Texttypen (Titel, Kommentar, Aufgabentext, etc.). Dies würde zudem die weitere Bearbeitung des Aufgabentexts in der Prüfungsausgabe vereinfachen.

Allenfalls sollte das Internet nochmals nach einem besseren Editor (Allenfalls für ein anderes Format) durchsucht werden. Das Flowtext-Format ist noch relativ neu und daher werden erst wenige gute freie Editoren, welche für den Demonstrator gesucht wurden, verfügbar sein.

Abläufe / Funktionen

Sortierfunktionen

Die eigentlich wichtigste Funktion des Demonstrators beinhaltet die Sortierfunktionalität. Diese ist im Demonstrator etwas unglücklich gelöst, da mehrmals nach Aufgaben, Prüfungsteilen und Prüfung gesucht werden muss. Hinzu kommt, dass die Sortierung auf der gesamten Datenbank (sprich XML-File) suchen muss. Zur Optimierung der Performance sollte die Datenbasis zuerst auf eine einzelne Prüfung reduziert werden. Eine Vereinfachung der Programmlogik ist möglich, indem die Daten als einzelne grosse Tabelle geladen werden, dies war in der aktuellen Fassung nicht möglich.

Die Sortierlogik soll anhand der Aufgaben aufgezeigt werden (für den Aufgabenblock und die Prüfungsteile gelten diese analog). Die Aufgabe wird zuerst innerhalb des Aufgabenblocks an die neue Stelle verschoben. Ist sie am Anfang, beziehungsweise am Ende, wird die Aufgabe in den Vorgänger-, bzw. Nachfolgeraufgabenblock verschoben. Dies gilt auch für den Fall, dass dieser Aufgabenblock in einem anderen Prüfungsteil liegt. Das eigentliche verschieben erfolgt durch das Austauschen der Werte in der Sortierspalte und dem Anpassen des Verweises auf den Aufgabenblock aus der Aufgabe hinaus.

Datenanzeige und Update der Datenbank

Im Demonstrator gibt es grössere Probleme mit der Datenanzeige nachdem etwas bearbeitet wurde. Die Ursache liegt vermutlich in der nicht-einheitlichen Bearbeitung (Bearbeitung über die integrierte Datenbindung von .NET und Bearbeitung mit Programmcode über das Dataset mit den Prüfungsdaten). Dies sollte Optimiert werden. Zudem muss in der zukünftigen Fassung nur eine einzelne Prüfung im Speicher geladen werden.

GUI

Design

Die Softwareoberfläche ist auf Microsofts WPF.NET – Modell aufgebaut. Für die Benutzeroberfläche wurden nur Standardkomponenten verwendet. In der Ansicht „Aufgabenübersicht“ wurden zwar Benutzersteuerkomponenten hinzugefügt, welche jedoch wiederum nur Standardkomponenten verwenden, also nur eine Kombination von diesen sind.

In einer zukünftigen Version sollten die einzelnen Tabs wiederum als eigenständige Benutzersteuerelemente erstellt werden. Die Bearbeitung wird dadurch erleichtert aufgrund von kürzeren Ladezyklen. Zudem sind die jeweils nötigen Codebestandteile besser voneinander getrennt.

Im WPF – Modell kann die eigentliche Darstellung der Komponenten in Templates definiert werden (Ordner Resources). Diese funktionieren ähnlich wie die Stylesheets im Webumfeld. Im Hauptprojekt wurden alle Templates definiert. Die Bezeichnung der Dateien entspricht den Steuerelementtypen. In einer zusätzlichen Datei wurden die zugehörigen Farbvorlagen definiert,

welche von den anderen Templates referenziert werden. Die Farben wurden an das Corporate Design Konzept der Universität St.Gallen angelehnt.

Darstellung der Prüfung im Editiermodus

Die Prüfung wird in der aktuellen Fassung mit verschachtelten Benutzersteuerelementen dargestellt (Dateien in Projekt CustomControls):

- Prüfungsdetail: Beinhaltet eine Liste mit den Prüfungsteilen
- Prüfungsteilliste: Beinhaltet die Anzeige der Daten zum Prüfungsteil und eine Liste mit den Aufgabenblöcken
- Aufgabenblockliste: Beinhaltet die Angaben zum Aufgabenblock und eine Aufgabenliste.
- Aufgabenliste: Beinhaltet eine einzelne Aufgabe (nicht mehrere).

Alle Benutzersteuerelemente unterhalb von Prüfungsdetail wurden als Datentemplate der Liste des übergeordneten Steuerelements hinzugefügt.

Diese Lösung erwies sich aufgrund von Schwierigkeiten mit der Datenaktualisierung in der Anzeige als unpraktisch und benötigt viel Rechenaufwand für die Anzeige. Es müssen mittels Programmcode an Events angedockt werden und die Datenbindung sichergestellt werden. Speziell auch die Berechnung und Darstellung der Summierungen von Punktzahl und Bearbeitungszeit bereitete Schwierigkeiten. Die jetzige Lösung sollte gründlich überdacht werden und eine bessere Struktur erarbeitet werden!

Mittels eines Datentemplates sollte ein bestehendes Standardsteuerelement verwendet werden, welches die Datenaktualisierung beherrscht. Allenfalls kommt hier ein Treeview in Frage. Auf Codeproject.net und anderen Homepages finden sich Beispiele, wie solche Steuerelemente angepasst wurden.

Sonstige Verbesserungsmöglichkeiten

Verwendung eines besseren Oberflächen – Templates

Für die Gestaltung des GUI wurde ein Standard-Template aus dem Internet verwendet. Es hat sich im Verlaufe der Gestaltung der Farbvorlagen gezeigt, dass ein erfahrener Designer aus den grafischen Gestaltungsmöglichkeiten noch einiges mehr herausholen kann. Es sollte abgeklärt werden, ob bessere Vorlagen existieren und eingekauft werden können. Hier muss jedoch Wert darauf gelegt werden, dass der Anbieter verfügbar ist. Denn beim Einbauen des Standardtemplates mussten verschiedene Änderungen vorgenommen werden.

Zentralisierung der Datenbank

Momentan existiert eine einzelne lokale Datenbank in Form einer XML – Datei. Dies brachte beim Entwickeln des Demonstrators verschiedene Probleme mit sich. Zusätzlich kommen im Falle einer richtigen Software weitere Probleme auf dieses System hinzu:

- Kein zentraler Pool von Prüfungsfragen: Die Verteilung der Fragen zu allen Programm Benutzern würde unmöglich werden, da alle Benutzer prinzipiell Änderungen vornehmen könnten, welche zusammengeführt werden müssen.

- Keine Möglichkeiten der Versionskontrolle: Da jeder Benutzer auf einer eigenen Datei arbeitet gibt es keine Möglichkeit paralleles Ändern der Daten zu verhindern. Diese könnten nur umständlich wieder zusammengeführt werden.
- Umständliches Weitergeben der Dateien: Für die Dateiweitergabe werden externe Systeme benötigt (z.B. E-Mail oder Dateiablage). Dies verhindert einerseits eine Zugriffskontrolle auf die Dateien und birgt andererseits die Gefahr, dass nicht autorisierte Benutzer Zugriff auf die Prüfungsfragen erhalten (z.B. Datei falsch abgelegt, falscher E-Mailempfänger)
- Sicherheit: Zugriff auf zukünftige Prüfungsfragen ist ein potentielles Ziel von Studenten. Falls dies gelingt birgt dies verschiedene Probleme: Aufwendungen wegen der nötigen Wiederholung der Prüfung, Imageschaden der Universität St.Gallen. Eine zentrale Datenbank würde das Sicherheitsrisiko verkleinern.

Workflowsteuerung

Das Durchführen einer Prüfung betrifft verschiedene Stellen und Organisationseinheiten. Eine Workflowsteuerung bietet sich hier ein. Dies erlaubt eine bessere Kontrolle über den Prozess der Prüfungserstellung. Die nötigen Daten werden automatisch an die korrekten Stellen weitergeleitet. Grosses Potential besteht auch in den Automatisierungsmöglichkeiten. Beispielsweise können nach dem Bidding direkt die Prüfungen angelegt werden und an die verantwortlichen Personen zur weiteren Bearbeitung zugeteilt werden. Die verantwortliche Person für die Prüfungsdurchführung hat anschliessend einen direkten Überblick über den Status der einzelnen Prüfungen.

Vordefiniertes Layout für die Prüfungen

Bisher wurden die verschiedenen Anspruchsgruppen einer Prüfung noch wenig einbezogen in die Gestaltung des Prüfungslayouts (Textgestaltung, Farbgestaltung, Aufbau der Prüfungsfragen, etc.). Auch hier herrscht Verbesserungspotential mit einer Standardisierung. Hierbei sollte besonders auch auf Studenten mit Sehbehinderung eingegangen werden. Für diese Gruppen käme beispielsweise eine separate Ausgabemöglichkeit der Prüfung mit grösseren Schriften oder besseren Farbkontrasten in Betracht.

Datenaustausch zum Online-Prüfungstool der Universität Zürich

Allenfalls sind in Zukunft auch Online-Prüfungen an der Universität St.Gallen ein Thema. Hierbei kommt eine Kooperation mit der Universität Zürich in Betracht. Diese sind bereit s mit der Fachhochschule Rapperswil (?) eine Art Datenaustauschformat für Prüfungen am definieren. Dies sollte je nachweiterem Vorgehen ins Projekt einbezogen werden. Eine Schnittstelle kann notfalls auch später eingebaut werden.

Hilfestellung bei der Korrektur / Notengebung

In einem späteren Tool könnte in einem weiteren Schritt auch die eigentliche Korrektur unterstützt werden. Denn alle nötigen Angaben wurden bereits erfasst (Anzahl Punkte, Musterlösung). Eine erste einfache Variante wäre die Ausgabe einer Exceldatei. Das andere Extrem wäre bei der vollständigen digitalen Erfassung der Prüfungsergebnisse mit Integration in die bestehende Notendatenbank.