AUSWAHL EINER TEMPLATE ENGINE FÜR EINE FINTECH-ANWENDUNG

DANIEL SCHNEIDER (BACHELORSTUDIUM INFORMATIK)

Betreuer: Prof. Dr. Gerd Beneken, Prof. Dr. Florian Künzner

Die Auswahl von Standardsoftware ist eine wichtige Aufgabe, da der Einsatz von geeigneten Anwendungen mitentscheidend für den Erfolg von Unternehmen sein kann, unabhängig davon, ob die Software allein oder als Teil einer größeren IT-Anwendung eingesetzt wird. In dieser Arbeit wurde ein solches Auswahlprojekt für die Identifikation neuen Template Engine zur Dokumenterzeugung innerhalb der FinTech Anwendung RAQUEST durchgeführt. Dieses Programm wird von Banken zur automatisierten Quellensteuerrückforderung für Ihre Kunden eingesetzt und dabei müssen komplexe Antragsdokumente sowie Anschreiben im Corporate Design der Banken von der Anwendung erstellen werden. Technische Limitierungen der gegenwärtig dafür genutzten SQL Server Reporting Services (SSRS) führen zu einem hohen Entwicklungs- und Wartungsaufwand für Dokumente, woraus Probleme für das weitere Wachstum der RAQUEST GmbH resultieren. Diese Probleme sollten mit Hilfe einer neuen Template Engine gelöst werden. Um den Erfolg des durchgeführten Auswahlprojekts nach Abschluss überprüfen zu können wurden zum Projektstart vier zu erreichende Ziele als Bewertungskriterien definiert.

Schritt wurde zunächst das Vorhandensein aller Features mit hoher Priorität in den einzelnen Softwarepakten überprüft. Dabei zeigte sich, dass lediglich die Template Engine von Telerik® alle diese Kriterien im gewünschten Ausmaß erfüllte und somit als vielversprechendster Kandidat aus der ersten Phase der Leistungsevaluation hervorging. Eine weitere Analyse zeigte, dass dieses Anwendungspaket zusätzlich auch die meisten Features mit mittlerer und niedriger Priorität enthielt. Somit konnte die Telerik® Reporting Engine als geeignetstes Anwendungspaket identifiziert werden und wurde schließlich auch erworben.

Auf Basis dieser Software wurde im zweiten Teil des Projekts ein Webservice zur Dokumentenerzeugung entwickelt. Dieser sollte unabhängig sein und sowohl Aufträge von RAQUEST und als auch von anderen Anwendungen bearbeiten können. Dafür wurde ein Webanwendung entworfen, die alle für die Dokumenterzeugung benötigten Informationen über eine REST API-Schnittstelle im JSON-Format entgegennehmen kann. Sobald die Anwendung mit den entsprechenden Informationen aufgerufen wird, beginnt die Erzeugung aller angefragten Dokumente. Dafür sind intern mehrere auf-

Zielnummer	Zielbeschreibung
Z1	Schnellerer und effizienterer Designprozess für neue Dokumente
Z2	Veränderung und Wartung bestehender Dokumente wird vereinfacht
Z3	Design und Wartung von Dokumenten teilweise durch Nichtentwickler durchführbar
Z4	Entwicklung eines vielfältig einsetzbaren *Documentservice* mit der Template Engine möglich

Abbildung 1: Definierte Ziele für die Auswahl einer neuen Template Engine.

Mit Hilfe einer ersten Marktanalyse wurden zunächst sieben mögliche Kandidatenanwendungen für die neue Template Engine identifiziert. Von diesen initialen Kandidaten konnten mittels einer Grobbewertung anhand vorher definierter, gewünschter Features der neuen Software drei Anbieter schnell ausgeschlossen werden. Eine Feinbewertung und detaillierte Leistungsevaluation wurde anschließend mit den vier verbliebenen Softwarepaketen durchgeführt. Hierfür wurde eine Liste von insgesamt 26 gewünschten Eigenschaften erstellt, die gemäß ihrer Wichtigkeit, in Features mit hoher, mit mittlerer und mit niedriger Priorität unterteilt wurden. Im ersten

einander aufbauende Schritte nötig, die vollständig im Webservice gekapselt sind. Nach Fertigstellung der Dokumente werden diese über einen anderen Endpunkt der Restschnittstelle dem Aufrufer zur Verfügung gestellt (Abbildung 2).

Um die Erzeugung von Dokumenten in dem Webservice parallel und nicht nur sequentiell durchführen zu können, erfolgte die Entwicklung der Anwendung mittels eines Aktorenmodells. Dabei handelt es sich um ein Netzwerk von unabhängigen, nebenläufigen Einheiten, den sogenannten Aktoren, die untereinander nur mittels Nachrichten kommunizieren und nicht in der Lage sind den internen Status

ROSENHEIMER INFORMATIKPREIS INF-BACHELOR

anderer Aktoren zu beeinflussen. Mit Hilfe eines Aktorenmodells ist es sehr einfach ein gleichzeitiger Zugriff auf Daten von unterschiedlichen, nebenläufigen Prozessen verhindert und so eine stabile, asynchrone und nebenläufige Anwendung, wie hier gewünscht, zu realisieren.

Die Implementierung des Webservice wurde mit dem ASP.NET Framework und dem Akka.NET Framework in C# durchgeführt, wobei das Programm gegenwärtig aus etwa 3000 Zeilen Code besteht. Die funktionelle Analyse dieser Anwendung ergab, dass Beispieldokumente wie gewünscht und in hoher Qualität mit ihr erzeugt werden konnten. Detaillierte Performancetests demonstrierten darüber hinaus die Geschwindigkeitsvorteile des parallelen Renderings durch die Aktoren gegenüber einer sequenziellen Bearbeitung der Anfragen, gerade bei einer hohen Auslastung.

Durch die Auswahl der Telerik® Reporting Engine konnten alle zu Beginn des Projekts definierten Ziele erreicht und das Projekt erfolgreich bearbeitet werden. Darauf basierend konnte in dieser Arbeit zusätzlich Prototyp für einen Webservice zur Dokumentenerzeugung entwickelt werden, welcher im kommenden Jahr komplettiert werden wird und anschließend im neuen Druckprozess von RAQUEST, sowie in einer weiteren Anwendung der RAQUEST GmbH eingesetzt werden wird.

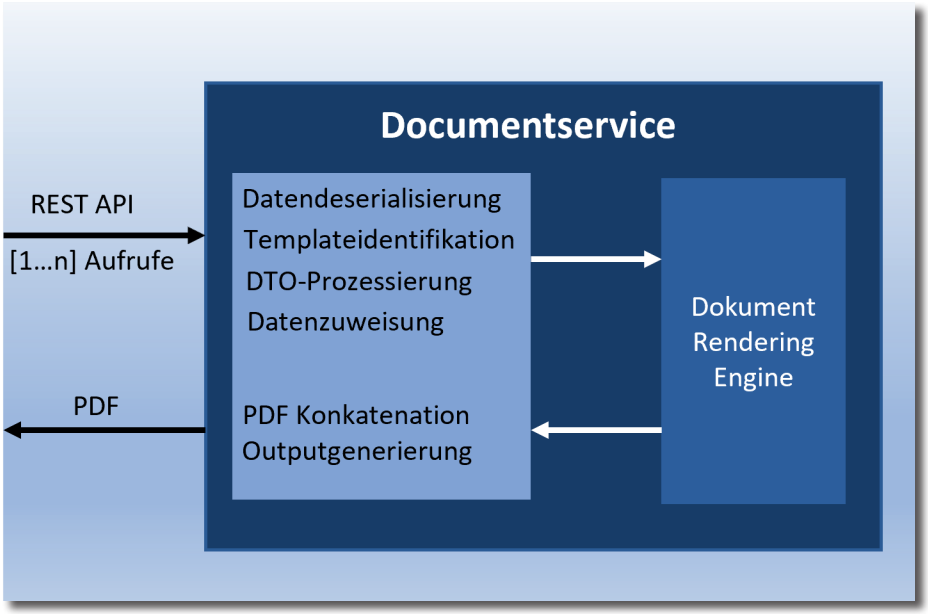

Abbildung 2: Schematische Darstellung des Documentservices inklusive Schnitstellen und der durchzuführenden internen Aufgaben.