E-GOVERNMENT PLATTFORM KONZEPT DER STADT GRAZ

Juni 2006

Zweitausend Verfahren wurden innerhalb weniger Monate von den BürgerInnen über die Grazer E-Government Plattform abgewickelt. Die intelligente Erweiterung der existierenden Plattform um eine Business Process Engine bringt nun zusätzliche Vorteile sowohl bei der Verfahrensentwicklung als auch bei der Verfahrensausführung. Mittels einer standardisierten Definitionssprache wird der Ablauf eines Verfahrens mit den jeweils anzuwendenden Regeln, Prüfungen, Diensten und deren technischer Repräsentation (in der Regel Softwarekomponenten, Web Services) beschrieben. Vorgefertigte Bausteine und Standard-Prozess-Definitionen vereinfachen das Modellieren der E-Government Verfahren, wodurch sich der Aufwand für das Umsetzen erheblich reduziert. Die Verfahrensausführung mit Orchestrierung der erforderlichen Services erfolgt ablaufgesteuert den beschriebenen Workflows entsprechend. Durch die somit konfigurierbare Ablaufsteuerung entsteht hohe Flexibilität hinsichtlich organisatorischer oder auch rechtlicher Veränderungen bei der Verfahrensabwicklung.

1. Die Grazer E-Government-Plattform - eGraz

Die Geschichte der Grazer E-Government-Plattform reicht zurück ins Jahr 2003, als mit der Erstellung des Grazer E-Government-Masterplans begonnen wurde. Ebenfalls in diesem Jahr wurde der E-Government Masterplan von Bund und Ländern fertig gestellt [1]. Generell war diese Zeit von einer Vielzahl von Aktivitäten im Rahmen der österreichischen E-Government Initiative sowie vom bevorstehenden Beschluss des E-Government Gesetzes [2] gekennzeichnet. Aus der Sicht der Software-Entwicklung war – und ist – E-Government ein typischer „Emerging Market“ in dem das Verständnis für die benötigte Funktionalität erst reifen muss.

Die bis dahin stattgefundenen und laufenden Anstrengungen zum Umsetzen von E-Government Diensten waren entlang beispielhafter Verfahren organisiert; in der Regel wurde pro Verfahren ein eigenes Projekt gestartet, um dieses für E-Government tauglich zu machen. Jedes dieser Projekte wurde weitgehend isoliert betrachtet. Im klaren Gegensatz dazu steht die Zielsetzung des Grazer E-Government-Masterplans: Die Realisierung einer universellen Plattform, über die alle Verfahren der Stadt Graz einheitlich abgewickelt werden können. Die Anforderungen an dieses System ergaben sich nicht aus den Vorgaben eines spezifischen Verfahrens, sondern aus den Funktionalitäten, die zur Abwicklung elektronischer Verfahren generell benötigt werden.

Die Ziele und Einschränkungen des Grazer E-Government-Masterplans können wie folgt zusammengefasst werden:

- Abwicklen aller Verfahren über eine einheitliche Plattform mit minimalem Aufwand zum Einbinden neuer Verfahren
- Weitgehende Nutzung von Open Source Software und deren Vorteile[3]
- Integration existierender und noch entstehender Standards im Bereich E-Government
- „Stage 4" – Reifegrad der angebotenen Leistungen. Das bedeutet die durchgehende elektronische Abwicklung der Verfahren [4] [5].

2. Die Systemarchitektur

Um die selbst gesetzten Vorgaben auch erfüllen zu können, wurde von Anfang an ein iteratives Vorgehensmodell, das es gestattet, schnell auf geänderte Rahmenbedingungen oder neue Entwicklungen zu reagieren, angewendet.

Für die Systemarchitektur wurde ein komponentenorientierter Software-Entwicklungs-Ansatz gewählt[6]. So bleiben die Auswirkungen absehbarer Änderungen am System auf andere Teile minimal und der notwendige Änderungsbedarf wird gering gehalten. Damit die Vorteile dieses Ansatzes auch im praktischen Einsatz zur Wirkung kommen, müssen die einzelnen Komponenten möglichst unabhängig von einander realisiert sein. Dies wird durch Einhalten folgender Vorgaben im System-Design erreicht:

1. Entkoppeln der Komponenten durch einen möglichst vereinheitlichten Nachrichtenaustausch zwischen den Komponenten
2. Stark abstrahierte Beschreibung der Komponenten-Interfaces, damit das damit definierte Protokoll von möglichst vielen unterschiedlichen Implementierungen erfüllt werden kann
3. Festlegen von Applikationsschichten mit wohl definierten Aufgaben

Das Umsetzen dieser Rahmenbedingungen führte zu einer durchgehend komponentenbasierten, dreischichtigen E-Government Plattform[7] (siehe auch Abbildung 1). Die Entkopplung der Komponenten wird mit einer standardisierten Darstellung der wichtigsten in den Abläufen der Verwaltung vorkommenden Geschäftsobjekte erreicht. Als Datenaustauschformat wird derzeit noch XML-a [8]verwendet, es ist jedoch geplant, dieses in Zukunft durch EDIAKT II [9] abzulösen. Innerhalb des Systems werden damit Technologien und Datenformate verwendet, die ursprünglich für den Austausch von Informationen zwischen unterschiedlichen Organisationseinheiten und Systemen gedacht waren. Diese Vorgehensweise ermöglicht einerseits das klare Entkoppeln der eingesetzten Komponenten, andererseits wird aber auch größtmögliche Offenheit für Fremdsysteme erreicht.

Die Aufgaben des Backoffice werden innerhalb der Stadt Graz durch den großflächig eingesetzten ELAK und eine Reihe von Fachapplikationen unterstützt. Eine E-Government-Plattform, über die alle Verfahren abgewickelt werden, muss daher unterschiedliche Fachapplikationen ansprechen können. Damit die Schnittstellen dieser unterschiedlichen

Anwendungen keinen Einfluss auf andere Bausteine der Plattform haben, werden deren Eigenarten durch die Transaktionsebene ausgeglichen.

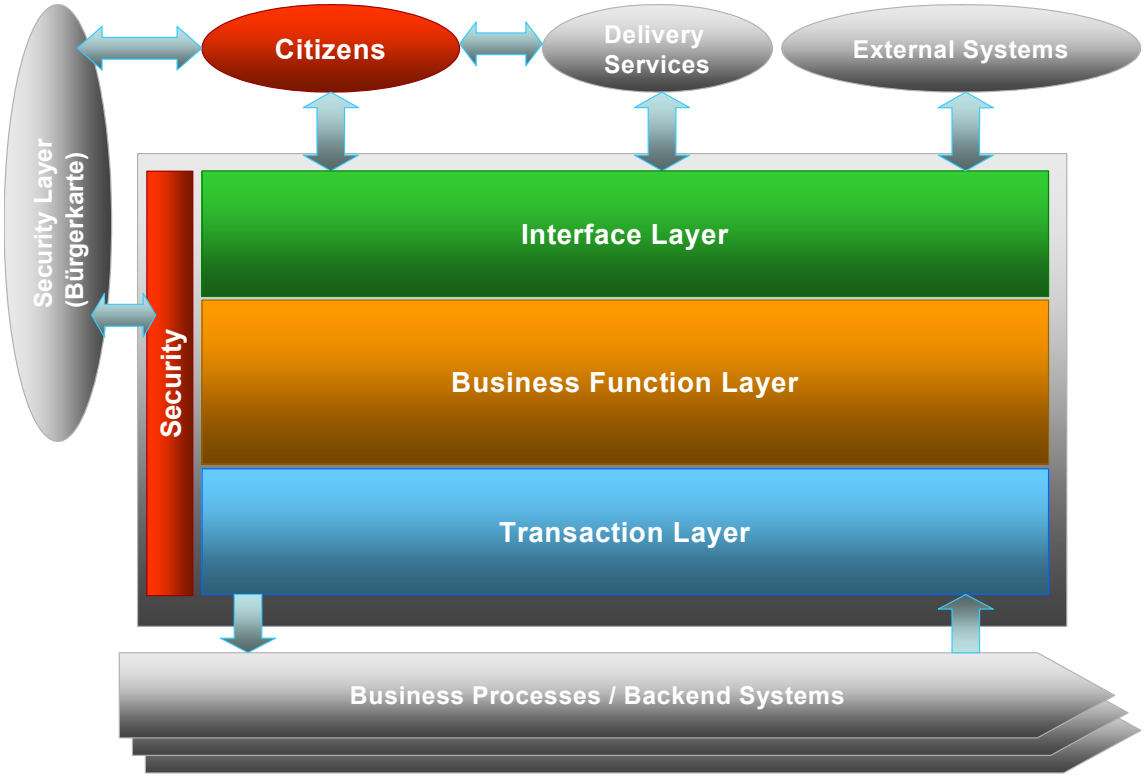

Abbildung 1: Die drei Ebenen der E-Government-Plattform Graz

Die Transaktionsebene bildet die klare Trennlinie zwischen der Plattform und den „Backend-Systemen“. Im Wesentlichen wird auf Basis der in dieser Schicht definierten Interfaces von einem Backend-System folgende Mindestfunktionalität erwartet:

- Erzeugen eines neuen Akts
- Hinzufügen von Einbringen zu einem bereits existierenden Akt

In Gegenrichtung bietet die Plattform für die Backend-Systeme Schnittstellen zur Übermittlung von Dokumenten (z.B. Erledigungen), Nachrichten, Zahlungsaufträgen, etc. die an die AntragstellerInnen weitergeleitet werden.

3. Die Funktionsebene

Kernschicht der gesamten Plattform ist die so genannte Funktionsebene (Abbildung 2). In dieser werden jene Komponenten zusammengefasst, welche die Basisfunktionalität zur Abwicklung von E-Government-Verfahren anbieten.

Dazu zählen Komponenten zum Erstellen und Anzeigen von Formularen, zum Aufbringen bzw. Überprüfen von Digitalen Signaturen, Plausibilitätsprüfungen eingegebener Daten sowie die Metadatenverwaltung für die Konfiguration der auf der Plattform angebotenen Verfahren .

Abbildung 2: Die Funktionsebene der E-Government-Plattform

Dank des komponentenorientierten Ansatzes kann die Funktionalität der Plattform durch das Hinzufügen zusätzlicher, neuer Komponenten einfach erweitert werden. Diese Erweiterbarkeit war eines der wesentlichen Design-Ziele, da die Plattform als wichtiges Leistungsmerkmal den Anspruch auf Universalität stellt und es unmöglich ist, alle Anforderungen aller zukünftig über die Plattform abzuwickelnden Verfahren zu kennen.

4. Integration einer Business Process Engine

Die für die Abwicklung eines Online-Verfahrens benötigte Funktionalität wird durch das koordinierte Zusammenspiel einzelner Komponenten erreicht. Die Koordination dieser Abläufe übernimmt der so genannte Plattform-Controller. Damit dieser Controller die unterschiedlichen Funktionskomponenten gleichförmig behandeln kann, müssen diese ein bestimmtes Interface implementieren. Für den Nachrichtenaustausch zwischen den einzelnen Komponenten steht ein Verfahrens-Kontext, über den auf alle verfahrensrelevanten Dokumente zugegriffen werden kann, zur Verfügung.

Anzahl und Typ der benötigten Funktionalität - ebenso wie deren Reihenfolge - können von Verfahren zu Verfahren unterschiedlich sein. Der Controller ruft daher je Verfahren die erforderlichen Bausteine in der zugeordneten Reihenfolge auf.

Um die Definition dieser unterschiedlichen Abläufe so einfach wie möglich zu gestalten, wurde entschieden, den Controller auf Basis einer so genannten Business Process Engine zu implementieren. Bei diesen Überlegungen stand weniger die klassische Ablaufsteuerung aus Sicht der involvierten Benutzer (=Workflow) im Vordergrund, sondern die Steuerung des Programmablaufs innerhalb der Plattform.

Gemäß den allgemeinen Rahmenbedingungen des Projekts, wurde zunächst nach einer Open Source Lösung im Bereich Workflow/Ablaufsteuerung gesucht. Dazu wurden in einem Subprojekt die beiden Produkte Enhydra Shark[10] und JBoss jBPM[11] evaluiert. Das Ergebnis dieser Untersuchung war, dass funktional beide Produkte für den geplanten Einsatz geeignet sind. Die Entscheidung viel letztlich zu Gunsten von jBPM auf Grund folgender Kriterien:

- Bessere und aktuellere Dokumentation verfügbar
- „Leichtgewichtiger“ als Enhydra Shark

Die Vorteile von Enhydra Shark liegen sicherlich in den unterstützten Standards[12] und der besseren Ausstattung an Tools. Beides war jedoch für den vorgesehenen Einsatz nicht relevant.

Um trotz der Entscheidung für ein konkretes Produkt auch für andere Lösungen offen zu sein, wurden die benötigten Konzepte („Prozess“, „Aktivität“,…) in Form von Interfaces abstrahiert und die entsprechenden konkreten Klassen der jBPM-Engine über das Adapter-Entwurfsmuster[13] eingebunden. So ist es jederzeit möglich eine alternative Workflow-Engine zu verwenden.

5. Prozessmodellierung und -bereitstellung

In einem Vorprojekt wurden in der Stadt Graz ca. 2800 IST-Geschäftsprozesse erhoben. Genaue Untersuchungen haben gezeigt, dass nach Transformation dieser Prozesse in E-Government-Verfahren nur wenige unterschiedliche Ablaufmuster existieren. Diese Abläufe werden durch vormodellierte Prozessschemata auf der Plattform bereitgestellt.

Um ein E-Government-Verfahren auf der Grazer E-Government Plattform zu implementieren, sind nur wenige Schritte nötig. Es wird das entsprechende Online-Formular entwickelt und über Konfigurationsparameter die zugehörige Prozessdefinition und das Backendsystem zugeordnet.

6. Ablaufsteuerung durch Prozessbeschreibungen

Durch die Integration der Business Process Engine wird der interne Ablauf des Aufrufs der unterschiedlichen Komponenten über eine Prozessdefinition gesteuert.

Abbildung 3 zeigt einen Ausschnitt aus einem Musterprozess. Nachdem die Bürgerin bzw. der Bürger einen Antrag ausgefüllt, ggf. digital signiert und abgeschickt hat, wird der zugeordnete Geschäftsprozess gestartet. Die Workflow-Engine ruft nun gemäß dieser Prozessdefinition die verschiedenen Komponenten auf. Diese Komponenten werden über Adapter in die Prozesse als Aktivitäten (Rechtecke im Diagramm) integriert und deren Funktionalität ausgeführt.

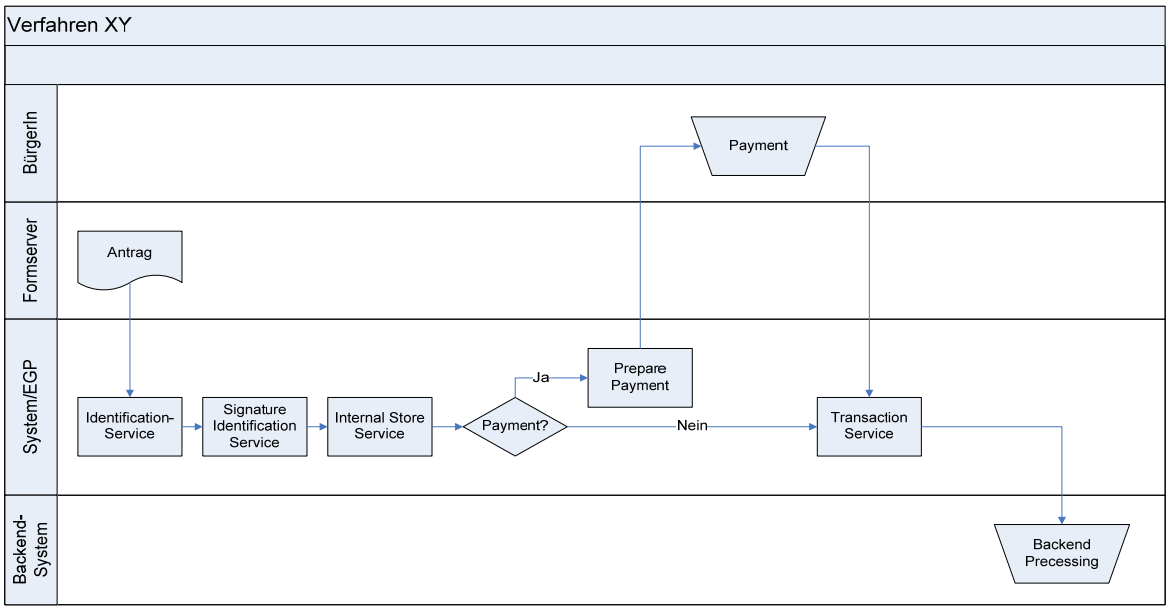

Abbildung 3: Teil eines Musterprozesses

Die vertikale Unterteilung in so genannte „Swimlanes“ definiert Zuständigkeiten. Trapezförmige Kästchen sind so genannte „Tasks“. Das sind Aufgaben, bis zu deren Erledigung der Geschäftsprozess in einen Wartezustand übergeht. In diesem Beispiel bewirkt dies, dass mit dem Prozess erst dann fortgefahren wird, wenn eine Zahlung erfolgt ist (falls eine Zahlung notwendig ist). Gleiches gilt für die Abarbeitung des Verfahrens innerhalb des Back-Office. Erst wenn hier eine Rückmeldung an die E-Government-Plattform erfolgt, wird der Prozess fortgesetzt (z.B. mit der Zustellung der Erledigung).

7. Zusammenfassung

Der Einsatz von Workflow- bzw. Business Process Engines erlaubt eine einfache und rasche Anpassung des Systemablaufs an sich ändernde Anforderungen. Dadurch können über eine universelle, einheitliche Plattform alle Verfahren abgewickelt werden.

In Verbindung mit der komponentenorientierten Architektur wurde so ein ausgesprochen leistungsfähiges Gesamtsystem entwickelt, welches die selbst gesetzten Anforderungen (siehe 1) erfüllt. Derzeit kann ein neues Verfahren – vorausgesetzt, es verwendet ein bereits unterstütztes Backend-System – binnen zwei Tagen online gestellt werden. Den größten Teil des Aufwandes nehmen dabei das Design des Formulars und dessen Test in Anspruch. Die eigentliche Installation des Verfahrens bei Vorhandensein eines fertigen Formulars kann in ca. 15 Minuten durchgeführt werden.

8. Literatur / References

[1] e-Government Masterplan und Roadmap - Stand Oktober 2003, http://reference.e-government.gv.at/Veroeffentlichte_Entwuerfe.354.0.html, zuletzt abgerufen am 8.5.2006

[2] E-Government Gesetz, http://ris1.bka.gv.at/authentic/findbgbl.aspx?name=entwurf&format=pdf&docid=COO_2026_100_2_30412 , zuletzt abgerufen am 8.5.2006.

[3] Peter Salhofer, Martin Pickl, Using Open Source Software Development Models for e-Government Solutions, Proceedings of the International Conference on e-Government, Ottawa, Canada, 27.-28.10.2005, p.363-374

[4] Layne K,. Lee J., Developing Fully Functional E-government: A Four Stage Model., Government Information Quarterly, 18(2), pp122-136, 2001

[5] Zahir Irani, Madi Al-Sebie and Tony Elliman, Transaction Stage of e-Government Systems: Identification of its Location & ImportanceI, Proceedings of the 39th Hawaii International Conference on System Sciences - 2006

[6] D. Gruntz, S. Murer, Component Software – Beyond Object-Oriented Programming, Pearson Education Limited, Second Edition, 2002

[7] Peter Salhofer, Friedrich Steinbrucker, E-Componment – komponentenbasierende E-Government Architekturen, in: Wimmer (Hrsg), E-Government 2005: Knowledge Transfer und Status, Österreichische Computer Gesellschaft, Wien, 2005

[8] E-Governemnt XML-Strukturen für Antragsdaten, http://reference.e-government.gv.at/XML-Strukturen_fuer_Antragsdat.438.0.html, zuletzt abgerufen am 8.5.2006

[9] Freitter, Gradwohl, Dehner, XML-Schema EDIAKT II, http://reference.e-government.gv.at/XML-Schema_zu_Ediakt_II__ediak.739.0.html, zuletzt abgerufen am 8.5.2006.

[10] Enhydra Shark – Open Source Java XPDL worklow, http://shark.enhydra.org, zuletzt abgerufen am 8.5.2006

[11] JBoss jBPM, http://www.jboss.com/products/jbpm , zuletzt abgerufen am 8.5.2006

[12] Workflow Management Coalition, XML Process Definition Language Specification, 2002, abrufbar unter http://www.wfmc.org/standards/docs/TC-1025_10_xpdl_102502.pdf

[13] Erich Gamma et al, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Professional, 1995