Gesuchsverwaltung mit Datenbanken
Möglichkeiten und Perspektiven

Dokumentation zum Vortrag vom 10.9.04 anlässlich des Seminares „Strategisches Stiftungsmanagement“
von Stefan Bürer, stefan.buerer@gmx.net

Analyse

Datenanalyse

Die Datenanalyse stellt die Grundlage dar für den effektiven Einsatz einer Datenbank. Nachfolgend finden Sie eine geraffte Analyse der bei der Gesuchsbearbeitung anfallenden und erhobenen Daten. Diese Analyse lässt sich selbstverständlich noch weiter vertiefen.

Gesuchverwaltung

- **Administrative Grunddaten**: Laufnummer, Gesuchstitel, Beschreibung, Eingangs- und Gesuchsdatum
- **Gesuchsklassifikation**: hierarchische, geographische, sprachliche Klassifikation
- **Behandlung**: Zuständigkeit für die Behandlung, Sitzungszuordnung, Datum der Behandlung, Behandlungsstatus, Erledigungsdatum
- **Finanzielles**: Beantragter Betrag, bewilligter Betrag, bezahlte Beträge
- **Kommunikation** mit dem Gesuchsteller: Datumsangaben der Bestätigung, Rückfrage, Vertröstung, Zu-/Absage etc.
- **Bewertung**: Kriterien zur Gesuchsbewertung, Qualitätskontrolle

Adressverwaltung

Mehrfache Bezüge zur Gesuchsverwaltung: GesuchsstellerIn/Mitunterzeichner, Nutzniessung, etc.

- **Benennung**: Institutionsbezug/name, Name, Vorname
- **Adresse**: Adresse, Ortsangabe, Anrede
- **Kontaktdaten**: Telefon, Mail, Internet-Adresse
- **Klassifikation**: Beschlagwortung

Kategorien

Ein Klassifikationssystem zur eindeutigen Einteilung von Gesuchen fördert die Übersicht.

- **Hierarchische Gesuchskategorien** mit mehreren Gliederungsebenen

Sitzungsverwaltung

Behandlung von Gesuchen anlässlich einer Sitzung eines Gremiums

- **Sitzungskennung**
- **Sitzungsdatum**, behandelte Gesuche, gesprochene Beträge

Budgetverwaltung

- **Vergabungsplafond** pro Geschäftsjahr mit Finanzanalyse

Zahlungsverwaltung

Erfassung von Teil-Zahlungen an ein Projekt

- **Zahlungsadresse**
- **Ausgabekonten**: Vergabung oder Nebenkosten
- Zuordnung zu **Budgetjahr**

Ereignisverwaltung

Die Ereignisverwaltung ermöglicht eine Terminplanung und gibt einen Überblick über die Vorgeschichte eines Gesuchsteller/einer Gesuchstellerin.

- Zuordnung von **Personen** zu Ereignissen
- **Kontaktform**: Telefon, E-Mail, Persönlich etc.

Ablauforganisation

Die Analyse der Ablauforganisation ist die Basis für die Qualitätssicherung mit Terminkontrolle und Dokumentenmanagement. Die Applikation sollte die Erzeugung von Standarddokumente und deren geordnete Ablage unterstützen. Offene, gängige Formate ermöglichen den Austausch mit Standardapplikationen. Strategische Weiterentwicklungen, wie Gesuchseingabe über das Internet, können die Ablauforganisation nachhaltig beeinflussen.

Dokumententypen

- **Gesuchsverwaltung**: Aktenzeichen und Etiketten, Gesuchsübersicht
- **Korrespondenz** mit den Gesuchsteller: Bestätigungsbrief, Nachfrage, Absage, Vertröstung, Zusage; auf Vorlagen basierte editierbare Dokumente.
- **Sitzungsdokumente**: Arbeitsunterlagen, Protokolle
- **Jahresberichte**: gruppierte Gesuchslisten, Finanzberichte, grafische Auswertungen
- **Buchhaltung**: Finanzberichte, Zahlungsanweisungen

Konzepte

Unterschiedliche Datenbank-Technologien können für eine Applikation zur Gesuchsverwaltung verwendet werden. Je nach verwendeter Technologie sind die Schwerpunkte eher bei Dokumentenmanagement, bei der Vorgangsbearbeitung oder bei der Datenkonsistenz gesetzt. Zu diesen Technologien gehören:

- XML Datenbank Systeme
- Workflow Systeme (Notes)
- Relationale Datenbank Systeme

Im Folgenden sollen verschiedene Architekturen von *Relationalen Datenbank Management Systeme* (RDBMS) untersucht werden. Relationale Datenbanken gelten als erprobte und bewährte Technologie, die für kleine, wie auch für die allergrössten Datenbanken zum Einsatz kommen. Die Vorteile liegen in der Flexibilität der Datenorganisation, in der Wiederverwendbarkeit der Daten, in der Schnelligkeit und Offenheit. Aufwändig ist diese Technologie bei der Abbildung sehr heterogener Datenstrukturen. RDBMS können von Einzelarbeitsplätzen bis hin zu verteilten Datenbanken im Internet betrieben werden.

Desktop Datenbanken

Desktop Datenbanken (Bsp. MS-Access oder Filemaker) stellen wohl die geläufigste Entwicklungsplattform für kleinere Datenbanken dar. Diese Datenbank-Technologie wurde für den Einsatz auf einzelnen Arbeitsplatzrechnern konzipiert, kann aber auch im Netzwerk eingesetzt werden.

+ einfacher Einstieg in die Datenbank-Technologie
+ schnelle Entwicklung
+ optimale Integration in Büroautomation
- Netzwerkprobleme
- Skalierbarkeit
- Schwierige Integration in andere Systeme
- Pflege der Applikation aufwändig

Client-Server Architektur

Client-Server Datenbank Systeme sind für den Einsatz im Netzwerk und für mehrere gleichzeitige Benützer konzipiert. Eine Client-Applikation sendet an einen Datenbank-Server eine Anfrage, der die geforderten Daten an den Client zur Verarbeitung zurück schickt, worauf der Client das Resultat der Bearbeitung wiederum an den Server zurück gibt.

+ Einbindung im Netzwerk
+ Skalierbarkeit
+ Verbindung mit anderen Systemen
- Server-Client Interaktion zumeist plattformspezifisch
- Server-Pflege aufwändig
- Client-Unterhalt umständlich

3-Schichten Architektur

Im 3-Schichten Modell („Three Tier Model“) wird zwischen Client und Datenbank-Server eine weitere Schicht mit der Applikationslogik eingeschoben. „Internet-Applikationen“, also Applikationen, die im Web-Browser bewirtschaftet werden, sind typische 3-Schichten Applikationen.

+ Universell einsetzbar
+ Skalierbarkeit
+ Weitgehend Plattformunabhängig
+ Einfacher Unterhalt der Applikation
+ Erweiterung des Webinterfaces
- Benützerkomfort
- Integration in Büroautomation

Realisierungs-Philosophien/Strategien

Bei den Realisierungsstrategien werden zwei Ansätze kontrovers diskutiert: *Closed Source vs. Open Source*.

- Bei *Open Source Software* (OSS) ist der Quellcode (Programmcode) Bestandteil der Software und kann eingesehen werden. OSS kann sowohl kommerziell lizenziert sein wie auch als *Free Software* ohne Lizenzzahlungen verwendet werden; sie unterliegt dabei der GNU Lizenz (www.fsf.org). Bei dieser Form wird die Software in einer Gemeinschaft weiterentwickelt, Änderungen und Weiterentwicklungen von Free Software unterliegen ebenfalls dieser Lizenz müssen der Allgemeinheit zugänglich gemacht werden (z.B. über www.sourceforge.net). OSS verwendet aus naheliegenden Gründen vorwiegend offene Protokolle und Standards für die Software, nicht zuletzt um Lizenz- und Patentprobleme mit proprietärer Software zu vermeiden. Dieses Entwicklungsmodell hat zu sicherer, flexibler, anpassbarer Software geführt, die universell einsetzbar ist, sowohl von grossen Computerfirmen ebenso wie von Privatpersonen. Das Geschäftsmodell basiert auf den erbrachten Dienstleistungen und nicht auf den Lizenzabgaben für Software-Pakete.

- *Closed Source Software* liegt dem gängigen Lizenzmodell (EULA) für Software zugrunde, bei dem für die Benützung einer Software Lizenzabgaben gezahlt werden. Bei dieser Softwaregattung ist der Quellcode das Betriebsgeheimnis des Herstellers. Der Hersteller bestimmt den Funktionsumfang der Software hat als Einziger die Möglichkeit, diese weiter zu entwickeln; er haftet in sehr beschränktem Masse für das Funktionieren der Software. Die Exponenten von Closed Source Software haben grosses Interesse an der Patentierbarkeit von Software und verwenden proprietäre Software-Komponenten, um Systemabhängigkeiten zu erzeugen. Der grösste Teil der verkauften Software entspricht diesem Modell: die kommerzielle Ausrichtung und die hohe Systemintegration ermöglichte sehr benützerorientierte und einfach zu bedienende Software, die über sehr umfangreiche Funktionen verfügen kann. Raubkopien sind die grössten Herausforderung für die Hersteller dieser Software-Gattung, was sich in teilweise aufwändiger Lizenzverwaltung bei den Benützern niederschlägt.

Merkmalsmatrix	**Open Source**	**Closed Source**
Programmcode	Öffentlich zugänglich	Geheim
Lizenzmodell	Frei (GNU) bis kommerziell	Kommerziell
Änderungen Programmcode	Allgemein möglich, Änderungen müssen ebenfalls öffentlich verfügbar gemacht werden (GNU)	Nur Hersteller autorisiert
Weiterentwicklung	Verteilt in Benützergruppen (Community Modell)	Hersteller
Abhängigkeit	Von Entwicklergemeinschaft	Von Hersteller
Verteilung	Meist frei über Internet	Nach Lizenzierung über kommerzielle Vertriebswege
Haftung	Benützer	Hersteller
Patentierbarkeit	Nicht erforderlich	Gefordert
Geschäftsmodell	Dienstleistungszentriert	Lizenzabgaben, Value-added
Schnittstellen	Offen	proprietär