H. Wandke, S. Kain & D. Struve (Hrsg.): Mensch & Computer 2009: Grenzenlos frei!?
München: Oldenbourg Verlag, 2009, S. 431-434

Fassets: Ein webbasiertes Präsentationssystem für den Lehr-Lern-Kontext

Martin Christof Kindsmüller, Jan Krüger, Michael Herczeg

Institut für Multimediale und Interaktive Systeme; Universität zu Lübeck

Zusammenfassung

Das hier vorgestellte System *Fassets* kombiniert Vorteile bewährter template-basierter Desktop-Präsentationssysteme mit den Vorteilen webbasierter Ansätze. Die Grundlage bildet die Modellierung aller Visualisierungselemente, wie beispielsweise Skripte, Grafiken, Multimediadateien, Artikel oder URLs in Form so genannter Assets. Als leistungsfähige Informationsstruktur für den aufgabenoptimierten Zugriff auf die im System verwalteten Assets wird eine Facettenklassifikation genutzt. Das System wurde als erweiterbare Software-Architektur realisiert, die eine einfache Informationsaufnahme aus verschiedensten Quellen (z. B. Datei-Uploads, Anbindung an Legacy-Systeme) ermöglicht. Die Umsetzung erfolgt in einem agilen, nutzerzentrierten Entwicklungsprozess. Als technische Basis wird das Web-Application-Framework Ruby on Rails verwendet.

1 Einleitung und Stand der Technik

Die computerunterstützte visuelle Präsentation wird gegenwärtig als eines der wichtigsten Instrumente bei der Wissensvermittlung angesehen. Pötzsch und Schettler (2006) zeigen, dass ein erheblicher Teil der in Vorträgen dargestellten Information in Form von computerunterstützten Visualisierungen kommuniziert wird, weshalb diese von den Autoren als paradigmatisches Kommunikationsinstrument der so genannten Wissensgesellschaft angesehen wird. Ähnlich wie Präsentationen in einem kommerziellen Kontext wird heute ein Großteil der Vorlesungen an Hochschulen von computergestützten Visualisierungen begleitet.

Das vorliegende System *Fassets* ist der Prototyp eines erweiterbaren, webbasierten Autoren- und Präsentationswerkzeugs für Visualisierungen im Lehr-Lern-Kontext, das zusätzlich Komponenten zur Verwaltung und Distribution dieser Inhalte bereitstellt. Visualisierungen bestehen üblicherweise aus einer sequentiellen Folge von bildschirmfüllenden „Folien“, die synchron zu einem Vortrag mit Hilfe eines Datenprojektors dargeboten werden. „Folien“ be-

stehen aus einer Anordnung von digitalen Medien wie Texte, Fotografien, Grafiken, Diagrammen sowie Audio- und Video-Inhalten. Jedes dieser Elemente, aber auch die komplette Visualisierung für beispielsweise eine Vorlesung („Foliensatz“) wird als Asset aufgefasst. Assets werden nach Jacobsen et al. (2005) als: (1) digitale Dateien plus Nutzungsrechte und (2) digitale Dateien plus Metadaten definiert. Beide Teildefinitionen sind notwendig: Ohne Nutzungsrecht darf ein als wertvoll erkanntes Asset nicht genutzt werden. Ein Nutzungsrecht hingegen kann nur ausgeübt werden, wenn ein Asset, aufgrund einer adäquaten Auszeichnung mit Metadaten, gefunden und als werthaltig erkannt werden kann.

„PowerPoint-Präsentation“ wird in einigen Bereichen bereits als synonym zu mediengestütztem Vortrag gesehen (Pötzsch & Schettler 2006), was zweifelsohne der Marktdurchdringung des Microsoft Office Pakets geschuldet ist. Vergleichbare Produkte anderer Hersteller sind OpenOffice Impress oder Apple Keynote. Die Aufgabenabdeckung dieser desktopbasierten Systeme beschränkt sich zumeist auf die Produktion und Präsentation dieser Visualisierungen. Die Verwaltung der Assets, die bei deren Produktion Verwendung finden, obliegt i. d. R. nicht der Anwendung selbst, sondern erfolgt mit Mitteln des Betriebssystems. Zur Distribution der Visualisierungen sind separate Systeme (wie beispielsweise HTTP-, WebDAV-, FTP-Server etc.) notwendig.

Neben diesen desktopbasierten Anwendungen zu Visualisierungsgestaltung existieren einige webbasierte Ansätze (z. B. das XMendeL-System, vergleiche Hartwig et al. 2003), die gegenüber desktopbasierten Systemen insbesondere Stärken in den Bereichen Verwaltung und Distribution aufweisen. Darüber hinaus können diese Systeme i. d. R. plattformunabhängig und kollaborativ eingesetzt werden. So kann im Lehr-Lern-Kontext ein webbasiertes System als zentrales Repository für das gesamte Lehrmaterial genutzt werden. „Foliensätze“ sind so stets online und aktuell. Sie können ohne zusätzliche Anwendung direkt im Browser editiert werden. Vorraussetzung hierfür sind: (1) eine Informationsstruktur, mit der umfangreiche Bestände von verschiedenartigen Assets verwaltet und durchsucht werden können, und (2) ein Editierwerkzeug, mit dem die Assets aufgabengerecht zu Visualisierungen zusammengestellt werden können. Insbesondere der zweite Punkt wird von desktopbasierten Systemen gut, von webbasierten Systemen bislang jedoch nur unzureichend unterstützt (Krüger 2008). Beim System *Fassets* sollen Vorteile beider Systemklassen vereint werden.

2 Konzeption und Implementierung

Im Folgenden werden Architektur und Benutzungsschnittstelle (UI) des Systems dargestellt, das auf Basis eines agilen, nutzerzentrierten Entwicklungsprozesses konzipiert und realisiert wurde. In klassischen desktopbasierten Visualisierungssystemen setzt die Informationsstruktur auf der Ebene eines einzelnen Dokuments an. Die Verwaltung der Präsentationen und der Assets, die für die Produktion der Präsentationen genutzt werden, findet außerhalb der Anwendung statt. Die bislang verfügbaren webbasierten Lösungen beruhen auf hierarchischen Strukturen. Da diese den Erfordernissen im Lehr-Lern-Kontext oft nur unzureichend gerecht werden (Krüger 2008), ist eine attribut-orientierte Ansicht der Informationsstruktur, mit Attributen wie Semester, Fach, Lehrveranstaltung oder Autor sinnvoll. Da diese Attribute in

keiner natürlichen Hierarchie zueinander stehen, wurde als Informationsstruktur eine Facettenklassifikation (Garfield 1984) eingesetzt (Abb. 1).

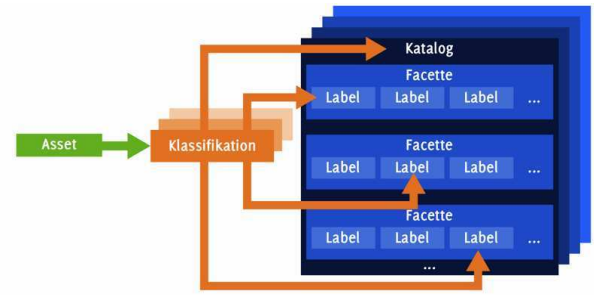

Abbildung 1: Über Klassifikationen wird ein Asset mit einem Katalog und den Labels der jeweiligen Katalog-Facetten assoziiert. Ein Katalog bildet eine unabhängige Facettenklassifikation.

Fassets ist als Web-Applikation konzipiert. Der Benutzer interagiert mit dem System über einen Web-Browser, der als Benutzungsschnittstelle von Fassets die Eingaben des Benutzers als HTTP-Requests an den Web-Server schickt. Dieser reicht die Anfragen an einen Applikations-Server weiter, auf dem die Anwendung ausgeführt wird. Die Struktur der Anwendung basiert auf dem MVC-Architekturmuster (*Model, View, Controller*, Thomas & Heinemeier Hansson 2007). Das API von Fassets realisiert die Prinzipien des REST-Architekturstils (*Representational State Transfer*), d. h., jede Entität, auf die Benutzer zugreifen können, bildet eine Ressource der Anwendung. Hinter einer Ressource steht ein Verbund aus MVC-Komponenten.

Für die Realisierung von Fassets wird das Web-Application-Framework Ruby On Rails eingesetzt. Rails-Anwendungen sind konsequent nach dem MVC-Architekturmuster strukturiert, wobei sich die Modellebene durch eine Datenbankabstraktionsschicht mit integriertem OR-Mapping (*Object Relational*) auszeichnet. Um Nutzer beim Erstellen und Editieren von Visualisierungen aufgabengerecht zu unterstützen wird das jQuery-Framework eingesetzt. jQuery unterstützt asynchrone Server-Kommunikation (AJAX), Event-Handling, Animation und Interaktionsmethoden wie Drag & Drop, wovon im Rahmen von *Fassets* stark Gebrauch gemacht wird. Die dynamischen Komponenten der Benutzungsschnittstelle bilden – neben Struktur und Inhalt (HTML) sowie Gestaltung (CSS) – eine unabhängige Verhaltensschicht.

3 Demonstration typischer Use-Cases

Da Editierwerkzeuge in webbasierten Systemen oft als defizitär angesehen werden, konzentriert sich die Darstellung im Folgenden auf die diesbezüglichen Use-Cases. Abb. 2(a) demonstriert, wie ein Bild vom Tray auf einen Slot-Editor gezogen wird. Der Slot-Editor für den Slot „*right*“ wird zur Eingabe von Textinhalt eingesetzt. Der Text wurde mit der Mar-

kup-Sprache von Fasset ausgezeichnet. Im Beispiel wird eine geschachtelte Liste und ein Link verwendet. Abb. 2(b) zeigt die Präsentationsansicht der gleichen Folie.

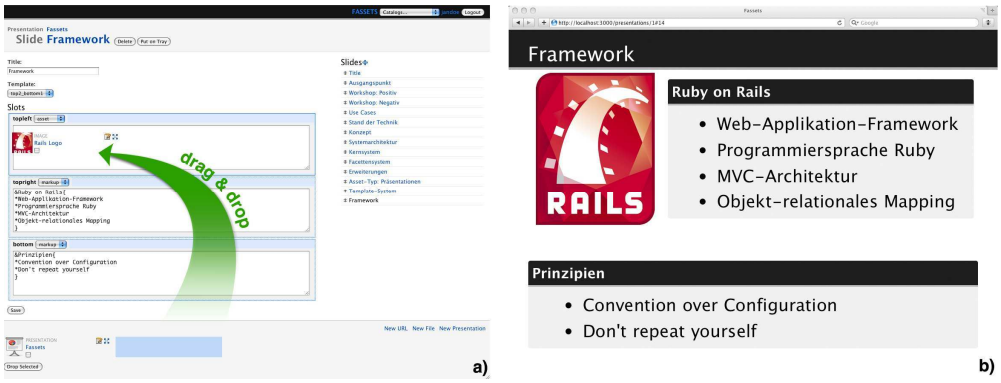

Abbildung 2: (a) Drag&Drop-Interaktion zum Einfügen eines Bildes in den Slot einer Folie (Pfeil nachträglich eingefügt). (b) Präsentationsansicht der Folie. Das Template der Folie, ordnet die beiden Slots nebeneinander an. Per JavaScript wird die Größe des Browserfensters überwacht und die Ausrichtung der Slot-Inhalte sichergestellt.

Literaturverzeichnis

Garfield, E. (1984). A Tribute to S.R. Ranganathan, the Father Library Science. Part 2. Contribution to Indian and International Library Science. *Essays of an Information Scientist 7*, S. 45-49.

Hartwig, R., Herczeg, M. & Hadley, L. (2003). XMendeL – A web-based semantic Web Tool for e-Learning Production Processes. Proceedings of the ICCE 2003. Hong Kong: ACCE, S. 556-563.

Jacobsen, J., Schlenker, T. & Edwards, L. (2005). *Implementing a Digital Asset Management System. For Animation, Computer Games, and Web Development*. Boston, MA: Focal Press.

Krüger (2008). *Fassets: Entwicklung eines webbasierten Asset-Management- und Präsentationssystems für den Lehr-Lern-Kontext*. Unveröffentlichte Bachelorarbeit am Institut für Multimediale und Interaktive Systeme der Universität zu Lübeck.

Pötzsch, F. S. & Schnettler, B. (2006). Bürokraten des Wissens? Denkstile computerunterstützter visueller Präsentationen. In: Gebhard, W. & Hitzler, R. (Hrsg.): *Nomaden, Flaneure, Vagabunden. Wissensformen und Denkstile der Gegenwart*. Wiesbaden: Verlag f. Sozialwissenschaften, S. 186-203.

Thomas, D. & Heinemeier Hansson, D. (2007). *Agile Web Development with Rails, 2nd Edition*. Raleigh, NC: Pragmatic Bookshelf.

Kontaktinformationen

Martin Christof Kindsmüller
E-Mail: mck@imis.uni-luebeck.de