Ein konzeptionelles Architekturrahmenswerk für Web-Anwendungen

Stefan Jablonski, Ilia Petrov, Christian Meiler

Lehrstuhl für Datenbanksysteme - Institut für Informatik
Universität Erlangen-Nürnberg
Martensstraße 3, D-91058 Erlangen
{Stefan.Jablonski, Ilia.Petrov, Christian.Meiler}@informatik.uni-erlangen.de

Abstract: *Das Web gleicht einer sich beständig ändernden Technologielandschaft. Standards und Techniken, die zur Implementierung von Web-Anwendungen herangezogen werden, unterliegen ebenso wie die Plattformen, auf denen sie betrieben werden, einem kontinuierlichen Wandel. Um trotz dieser Dynamik strukturiert und systematisch Web-Anwendungen entwickeln zu können, fehlt eine saubere Entwurfsmethodik zur Entwicklung von Web-Anwendungen unter Berücksichtigung der zentralen Ziele Erweiterbarkeit und Flexibilität. Dieser Beitrag unterbreitet eine Begriffsdefinition für Web-Architekturen und ein konzeptionelles Architekturrahmenswerk für Web-Anwendungen. Mit diesem einhergehend werden wichtige Charakteristika eines solchen Rahmenswerks untersucht und ein Vorgehensmodell zur Anwendung vorgestellt.*

1 Einführung

Das World Wide Web (WWW) ist ein sich ständig änderndes Medium, das nicht nur die Kommunikation und Interoperabilität heterogener Informationsquellen und unterschiedlicher Typen von Systemen ermöglicht, sondern auch den Zugriff auf unterschiedlichste Ressourcen auf einheitliche Art und Weise gestattet.

Das Web als Umgebung ist wegen der Grundsätze Offenheit, Einfachheit und Verfügbarkeit gleichermaßen äußert attraktiv für Lösungsanbieter und Nutzer. Standards und Technologien des Webs müssen für eine breite Palette von Hard- und Software und oft aus Basis anderer Technologien entworfen werden. Voraussetzung, dieses zu realisieren, ist eine offene, allgemein bereitgestellte Dokumentation dieser Technologien. Die Einfachheit der Anwendung ist das zweite Prinzip des Webs als Umgebung. Mit nur sehr geringer oder fast gar keiner Vorbildung und praktischer Erfahrung sollte es jedem möglich sein, das volle Potenzial des Webs auszuschöpfen. Die hohe Verfügbarkeit ist der dritte konzeptionelle Grundpfeiler des Webs, der insbesondere deswegen schwer umzusetzen ist, da er viele insbesondere auch technische Anforderungen nach sich zieht: das Web muss sich prinzipiell über viele heterogene Systeme erstrecken, es muss ein universelles Transport- und Kommunikationsprotokoll angewandt werden und schließlich müssen Interoperabilitätsprobleme im Hinblick auf Web-Anwendungen und Anforderungen an die Web-Plattform überwunden werden [Me02]. Das Web ist ein Forum für neue Technologien, die sich im wahrsten Sinn des Worts beinahe monatlich verändern. Bei einer solch rasanten Entwicklung ist es nicht verwunderlich, dass selbst erfahrende Entwickler nicht den Überblick über alle neuen Trends behalten können. Eine erweiterbares Architekturrahmenswerk auf der einen Seite und eine Klassifikation von Standards und

Technologien auf der anderen Seite sind nötig, um in dieser dynamischen Umgebung Orientierung zu bewahren. Ein solcher Ansatz ermöglicht:

- Einordnung bestehender Standards und Technologien und somit Vergleich mit verwandten Ansätzen
- Erkennen von Lücken in der vorhandenen Technologielandschaft und somit Hinweis auf erforderliche Ergänzungen
- Identifikation neuer Integrationskonzepte zwischen den Technologien basierend auf erkannte Lücken und einem Architekturvergleich

Die Hauptziele dieses Aufsatzes sind es, zunächst ein generelles und erweiterbares Architekturrahmenswerk für Web-Anwendungen zu definieren, das möglichst alle zur Zeit bestehenden Web-Anwendungsarchitekturen einschließt und Erweiterbarkeitsmechanismen anbietet. Wir unterscheiden hierbei zwischen dem Web als Ausführungsumgebung, woraus sich die Architektur der Web-Anwendungsplattform (Web-Anwendungsplattformarchitektur, WPA) ableitet und der Architektur darauf aufsetzender Web-Anwendungen (Web-Anwendungsarchitektur, WAA). Diese Unterscheidung ist die unumstößliche Basis für die folgende Diskussion. Des Weiteren unternehmen wir den Versuch einer Klassifikation von Web-Standards und Technologien, um diese anschließend mit dem Architekturrahmenswerk zu verknüpfen. Ergebnis ist ein Rahmenswerk zur strukturierten und systematischen Entwicklung und Gestaltung von Architekturen für die Web-Anwendungsplattform und für Web-Anwendungen. Ohne ein solches Rahmenswerk bleibt dem Entwickler nur die Möglichkeit, sich nach besten Wissen und Gewissen durch einen Urwald von „Bestandteile“ zu navigieren, die für die Umsetzung von Web-Anwendungen grundsätzlich zur Verfügung stehen. Am Ende steht die vage Hoffnung, eine passende und befriedigende Lösung gefunden zu haben. Die Vorgehensweise ähnelt dann eher einem „Trial & Error“ Prinzip als einer strukturierten und geplanten Vorgehensweise. Wir erheben nicht den Anspruch, dass unser Vorgehensmodell ausschließlich auf „messbaren Entscheidungen“ beruht, bewegen uns aber mit diesem Ansatz mit Sicherheit in Richtung geplanter und durchdachter Entwicklung von Web-Anwendungen. In diesem Sinn ist der hier vorgestellte Ansatz dem Gebiet der Entwicklung von Software-Architekturen zuzuordnen (vgl. [CLF00]).

Im nächsten Abschnitt wird ein Überblick über die Eigenschaften bestehender Web-Architekturen gegeben. In Abschnitt 3 schlagen wir ein Architekturrahmenswerk für Web-Anwendungsplattformen und Web-Anwendungen vor. Hierfür werden die Charakteristika bestehender Client-Server-Architekturen als Basis für weitere Entwicklungen untersucht. In diesem Abschnitt geben wir auch eine mögliche Klassifikation von Standards und Technologien an, die sich direkt oder indirekt auf das Web beziehen (Abschnitt 3.4). In Abschnitt 3.5 wird ein Beispiel diskutiert; dies zeigt, wie das hier vorgestellte Architekturrahmenswerk anzuwenden ist.

2 Bestehende Web-Architekturen

Eine Web-Anwendung kann folgendermaßen definiert werden: ein identifizierbares und eigenständiges Stück Anwendungsfunktionalität, das auf Basis einer Web-Anwendungsplattform in einer Web-Umgebung betrieben wird. Die Tatsache, dass Web-Anwendungen „identifizierbar“ sind, bedeutet, dass sie sowohl registriert und damit auffindbar sind, als auch, dass sie als ganzes identifizierbar und als Einheit einer größeren Anwendung verwendbar sind. Ihre „Eigenständigkeit“ impliziert auf der einen Seite,

dass ihre Implementierung durch eine Schnittstelle versteckt wird, und auf der anderen Seite, dass durch diese Schnittstelle, die Anwendungsfunktionalität und andere begleitende Eigenschaften beschrieben werden (z.B. transaktionales Verhalten). Diese Beschreibung kann beim Prozess des Auffindens und der Interaktion genutzt werden. Der Begriff „Anwendungsfunktionalität“ steht für die Anwendungslogik, also das, was eine Anwendung tatsächlich erbringt.
Der Begriff „Web-Anwendungsplattform“ beschreibt eine Kombination von Softwarekomponenten, welche die Grundlage dafür erbringen, dass darauf unterschiedliche Web-Anwendungen entwickelt, eingesetzt und ausgeführt werden können. Eine Web-Anwendungsplattform stellt die notwendigen Basisdienste zur Verfügung, die von einer Anwendung im Betrieb im Web benötigt werden. Plattformen können von Web-Anwendung zu Web-Anwendung abhängig von deren funktionalen Anforderungen variieren. Typische Plattformkomponenten sind beispielsweise ein http-Server, ein Applikationsserver oder eine Datenbank auf der Server-Seite und typischerweise ein Web-Browser auf der Client-Seite. Nehmen wir beispielsweise an, dass die Entwickler einen Teil einer Web-Anwendung als Java-Applets implementieren wollen. In diesem Fall muss die Web-Anwendungsplattform um eine Java Virtual Machine auf der Seite des Clients erweitert werden, damit dort eine Ausführungsumgebung zur Verfügung steht.

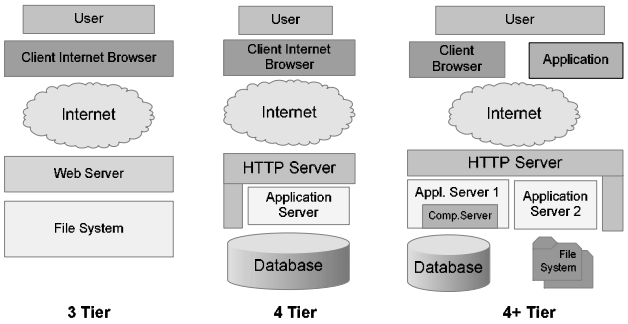

Abb. 1. 3-, 4- und 4+-Architekturen

Charakteristisch für alle existierenden Architekturen im Bereich des Webs ist, dass die Begriffe Plattformarchitektur und Anwendungsarchitektur vermischt werden. Dies verdeutlicht Abb. 1, in der typische Beispiele in Gestalt von 3-,4- und 4+-tier Architekturen dargestellt sind. Diese Architekturen sind Repräsentanten dafür, wie Komponenten, die zur Plattform gehören, und Komponenten, die Anwendungsfunktionalität implementieren, vermischt werden. Beispielsweise gehören die meisten Bestandteile der in Abb. 1 gezeigten 4+-Architektur zur Web-Anwendungsplattform; allerdings ist die Architekturkomponente „Application“ sicherlich Bestandteil der Web-Anwendung. Nichtsdestotrotz lässt sich in Abb. 1 ein wichtiges Architekturprinzip erkennen: Schichtenbildung. Jede Schicht stellt eine Abstraktion eines konkreten Problembereichs dar [No00] . Da Schichtenbildung ein extrem mächtiger Mechanismus ist (vgl. [HR99]), wird er in den hier folgenden Ausführungen zur Definition und Formalisierung des Architekturrahmenswerks eine entscheidende Rolle spielen.

3 Das Architekturrahmenswerk

In diesem Abschnitt wird unser Ansatz eines Architekturrahmenswerks für Web-Anwendungen vorgestellt, welches auf den Prinzipien Erweiterbarkeit, Allgemeingültigkeit, Klarheit und Umfassendheit aufbaut. Das Client/Server-Konzept fungiert hierbei als grundlegendes Architekturprinzip (vgl. [So00]), das mehrere Aspekte umfasst:

- Hardwarearchitektur: Eine Rechnerinfrastruktur, in der ein schneller zentraler Rechner (Server) über ein Kommunikationsnetzwerk mit vielen kleinen Rechnern (Clients) verbunden ist.
- Softwarearchitektur: Das wichtige Konzept dieser Technologie ist, dass Teile der Software, die von vielen Clients gemeinsam verwendet werden, auf dem Server implementiert werden; andere Teile, welche im einfachsten Fall nur client-spezifische Funktionalitäten übernehmen, werden auf dem Client implementiert.
- Kommunikationsmodell: Die Client/Server-Softwarearchitektur beruht auf einem Kommunikationsmechanismus, in dem Nachrichten (Anfragen und Antworten) über ein Kommunikationsnetzwerk ausgetauscht werden.

Das Client/Server-Konzept spielt eine Schlüsselrolle in dem hier vorgestellten Architekturrahmenswerk. Obwohl es als Basis für Web-Pattformarchitektur (WPA) verwendet wird, kann es ebenso als Entwurfsansatz für eine Web-Anwendungsarchitektur (WAA) verstanden werden.

3.1 Web-Plattformarchitektur

Eine Web-Plattformarchitektur (WPA) ist eine in Schichten aufgebaute Architektur, die ursprünglich aus der Client/Server-Architektur entstammt. Letztendlich besteht sie aber aus einer Reihe von verbundenen Softwarekomponenten, die eine Umgebung schaffen, in der Web-Anwendungen entwickelt, gespeichert und ausgeführt werden können.

Die Vorstellung einer Plattform und der damit verbundenen Anforderungen variiert stark in Abhängigkeit von der Art der Anwendung. Eine standardisierte ANSI C/C++ Anwendung benötigt z.B. eine Menge von Funktionen einer Ausführungsbibliothek, die zusammen mit dem Betriebsystem dann die Ausführungsumgebung darstellen. Die Java Virtual Machine ist ein noch besseres Beispiel für eine Plattform, da sie im Vergleich zu einer Ausführungsbibliothek nicht nur eine gewisse begrenzte Sicht auf das Betriebsystem darstellt, sondern selbst – wie der Name bereits impliziert – als Ausführungsumgebung fungiert. Eine Plattform kann gleichwohl als Schicht im Sinne einer Abstraktion und Standardisierung als auch als funktionale Brücke angesehen werden.

Eine Schicht kann als Stufe der Abstraktion aufgefasst werden, wobei eine darüber liegende Schicht ein generelleres und somit Nutzer zentriertes Problem adressiert und eine darunter liegende Schicht spezifischere und somit eher Server-orientierte Dienstleistungen erbringt. In einer traditionellen Schichtenarchitektur können nur nebeneinander liegende Schichten mittels wohldefinierter Schnittstellen miteinander kommunizieren (einfache Schichtenbildung). Es ist im Kontext von Web-Plattformarchitekturen weit verbreitet, dass mehr als eine einfache Schichtenbildung angeboten wird, also dass auch nicht direkt benachbarte Schichten miteinander kommunizieren können. Die bietet auf der einen Seite Vorteile hinsichtlich Performanz, ist auf der anderen Seite wohl auch auf Grund der Einfachheit und fehlender Technologien entstanden [Bu96].

Eine Plattform für Web-Anwendungen ist eine Kombination von Software (Modulen der Plattform), die nach entsprechender Konfiguration zusammenarbeiten, wobei alle diese Module installierbar und entfernbar sind. Entscheidend in diesem Kontext sind offene

Standards oder Kommunikationsmodule, wie z.B. Konverter oder Konnektoren. Dies ist die Grundlage dafür, dass beispielsweise ein Applikationsserver von Hersteller X so konfiguriert werden kann, dass er mit einem Datenbankprodukt des Herstellers Y eine Anwendung auf Basis von Enterprise Java Beans der Firma Z ausführen kann. Diese Konfiguration kann sowohl auf einen einzelnen Rechner vorhanden sein, als auch in einem Netz dedizierter Rechner realisiert werden.
Abb. 2 zeigt einen evolutionären Ansatz zur Definition von Schichten für Web-Plattformen; gleichzeitig wird hier die wichtigste Eigenschaft einer Web-Plattformarchitektur, nämlich die Erweiterbarkeit unterstützt. In den frühen Stufen der Entwicklung von Rechnersystemen agierte das ganze System als ein monolithisches Stück Funktionalität. Diese Struktur hat sich mit der Weiterentwicklung der Netzwerktechnologie dramatisch verändert und die Konsequenzen haben sich in zwei Richtungen ausgewirkt. Auf der einen Seite haben sie der Idee Vorschub geleistet, konventionelle PCs an Stelle von Terminals einzusetzen, was zu gesteigerter Rechenleistung und somit gesteigerten Fähigkeiten auch auf Seiten der Clients geführt hat. Auf der anderen Seite hat sich die Softwarearchitektur drastisch verändert: Es haben sich drei klar unterscheidbare Schichten herauskristallisiert: Server, Kommunikation und Client. An dieser Stelle ist es wichtig zu erwähnen, dass die Schichten Server und Client als Plattformschichten betrachtet werden, auch wenn sie leicht mit der korrespondierenden Organisation der Anwendungssoftware assoziiert werden können. Ein gutes Beispiel für eine solche Plattform stellt eine Datenbank dar, von der ein Teil auf der Server-Seite und ein anderer Teil auf der Client-Seite installiert ist. Beide arbeiten auf dem vom Betriebsystem bereitgestellten Teil der Kommunikationsinfrastruktur (TCP/IP LAN, DNS Server, Router usw.) in einem Netzwerk zusammen, das Teil der Kommunikationsschicht ist. Die darauf aufgesetzte Datenbankanwendung (z.B. ein Abrechungssystem oder ein Data Warehouse), die auf dieser Plattform läuft, kann – muss aber nicht zwingend – die gleiche Struktur von Client- und Server-Modulen aufweisen. Eine ausführliche Diskussion dieser Beobachtung folgt im nächsten Abschnitt.

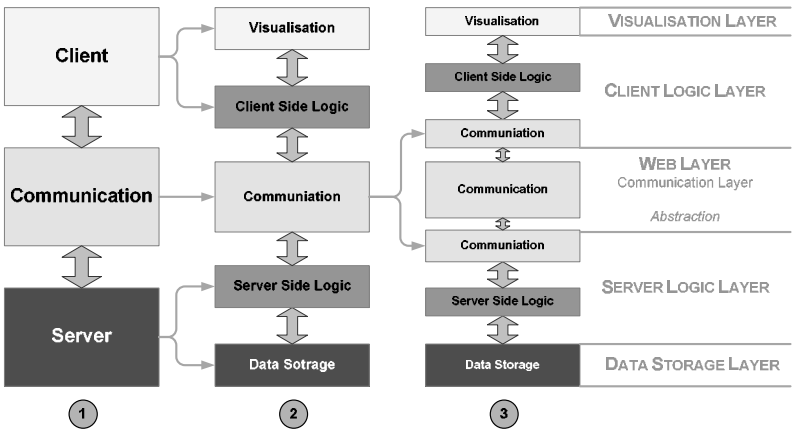

Abb. 2. Evolutionäre Entwicklung der Schichten

In vielen Publikationen wird die in Fall 2 der Abb. 2 dargestellte einfache Struktur verwendet und diese für das Web (vgl. Abb. 1) adaptiert. Diese Vorgehensweise wird von

dem Prinzip getrieben, die unterschiedlichen Server (z.B. http-Server, Applikationsserver, Datenbank, Betriebssystem) den verschiedenen Schichten zuzuordnen, wobei die Kommunikation vernachlässigt und eher als ein der Plattform innewohnender Dienst aufgefasst wird. In der hier vorgeschlagenen Architektur verfolgen wir einen flexibleren Ansatz, indem eine mehr an den Funktionalitäten ausgerichtete Perspektive verwendet wird (vgl. Fall 3 in Abb. 2).

Wir nehmen an, dass die serverseitigen Plattformmodule Datenspeicherung, Logik und Kommunikationsdienste organisiert in den jeweiligen Schichten zur Verfügung stellen. Die Datenspeicherungsschicht beinhaltet dabei mehrere unterschiedliche Module, wie z.B. Datenbankverwaltungssysteme oder Dateisysteme, und ermöglicht es Anwendungen, ihre Daten persistent zu machen. Die serverseitige Logikschicht beinhaltet Plattformmodule, auf deren Basis Anwendungsmodule ausgeführt werden können. Typische Vertreter dieser Schicht sind gängige Applikationsserver, wie z.B. Apache, IBM WebSphere, BEA Web Logic Application Server. Die Kommunikationsschicht nimmt kommunikationsorientierte Server auf (z.B. http-Server). Obwohl ein http-Server oft als Teil eines Applikationsservers angesehen wird, ist es auf Gründen der Übersichtlichkeit und Klarheit der Architektur sinnvoll, diesen in einer separaten Schicht zu organisieren. Somit kommt seine Bedeutung klarer zum Ausdruck.

Auf ähnliche Art und Weise kann die clientseitige Plattform in drei Schichten unterteilt werden: Visualisierung, clientseitige Logik und Kommunikation. Obwohl typischerweise ein Internet Browser die Visualisierung in weiten Teilen übernimmt, kommen auch andere Module wie Ausführungsumgebungen (z.B. die Java Virtual Machine im Falle eines Java-basierten Ansatzes) zum Tragen. Die clientseitige Logikschicht enthält Plattformmodule in Form von Ausführungsumgebungen.

Ein entscheidender Vorteil, der hier vorgeschlagenen Architektur, ist ihre Erweiterbarkeit und Anpassbarkeit. Gehen wir beispielsweise von einer Anwendung aus, die auf Basis der Enterprise Java Beans Technologie entwickelt wurde. Schon aus diesem Grund muss der Einsatz eines EJB-Containers – wie z.B. Tomcat, JBoss oder JOnAS – als Teil des Applikationsservers erfolgen. Da dieser Komponentenserver eine entscheidende Rolle in der gesamten Architektur einnimmt, können wir ihn explizit als Architekturkomponente identifizieren. Somit teilen wir die serverseitige Logikschicht in zwei benachbarte Schichten auf: eine für den Applikationsserver und eine für den eigentlichen Komponentenserver. Ein weiteres Beispiel bezieht sich auf die clientseitige Visualisierungsschicht. Auf den ersten Blick erscheint es nicht klar, wieso Visualisierungsdienste betrachtet werden müssen, da sie im allgemeinen Fall standardmäßig vorhanden sind. Gehen wir von einer web-basierten Datenbankanwendung aus, die nur zur Erstellung rein textueller Berichte verwendet wird. In diesem Fall kann ein rein textbasierter http-Browser verwendet werden, wodurch eine eigenständige Visualisierungsschicht praktisch aufgegeben werden kann. Werden diese Berichte nun aber durch Grafikdiagramme unterstützt, müssen clientseitige Visualisierungstechniken eingesetzt werden – insbesondere dann, wenn diese Diagramme interaktiv sein sollen und z.B. auf Basis von Java Applets implementiert sind. Jede Erweiterungsentscheidung muss subjektiv durch die Architekten getroffen werden, wobei die Kriterien Klarheit im Design, hoher Grad an Strukturierung und eine intuitive Entwurfsveranschaulichung beachtet werden müssen. Ziel ist es eine möglichst genaue Identifikation funktionaler Komponenten als Grundlage für die Auswahl der Implementierungen.

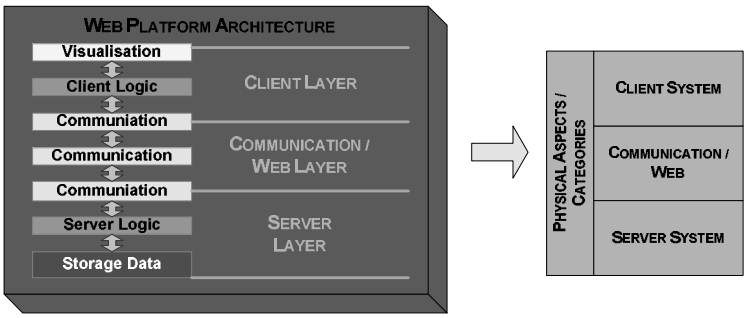

Abb. 3. Die Web-Plattformarchitektur wird auf der vertikalen Achse der Web-Architekturmatrix aufgetragen

3.2 Web-Anwendungsarchitektur

In diesem Abschnitt wird der Begriff Web-Anwendungsarchitektur im Detail betrachtet. Fast alle web-orientierten Anwendungsarchitekturen, die in der Informatik bekannt sind, lassen eine ausreichende Betrachtung ihres Einsatzkontexts vermissen. Diesen haben wir aber durch die Diskussion der Web-Plattformarchitektur in Abschnitt 3.1 ausreichend beschrieben. Um die Bestandteile einer Web-Anwendungsarchitektur zu analysieren, bietet sich eine äquivalente Vorgehensweise wie dort gezeigt in Form eines evolutionären Ansatzes an. Im generellen kann jedes Modul einer Anwendung als „Funktionalität" betrachtet werden. In Zeiten monolithischer Programme, gab es nur ein nicht weiter zu zerlegendes Stück, das die gesamte Funktionalität einer Anwendung umfasste (vgl. Abb. 4, Fall 0).

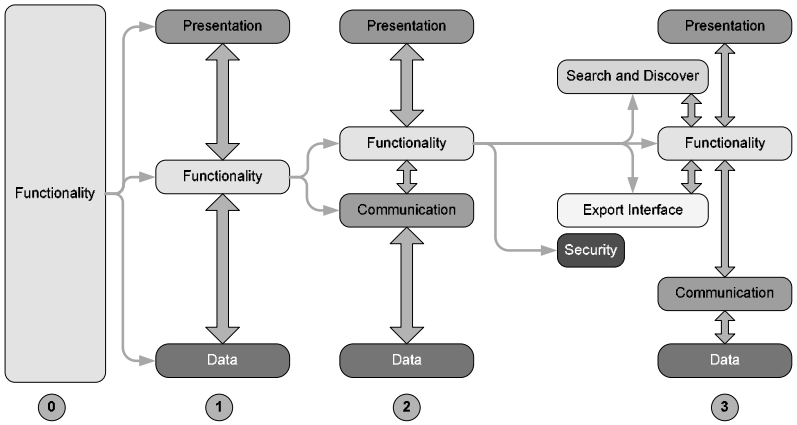

Abb. 4. Evolution der Module einer Web-Anwendungsarchitektur

Mit wachsenden Fähigkeiten der Rechner und deren breiten Verwendung in Industrie und Forschung wurden Anwendungen mit immer neuen Anforderungen belegt. Diese haben die Systemanalytiker dazu veranlasst bzw. sogar gezwungen, diese monolithischen Architekturen in dedizierte Teile zu zerlegen (vgl. Abb. 4, Fall 1). Eine erste Strukturierung identifiziert explizit Präsentation (Nutzerschnittstelle), Basisfunktionalität

und Datenspeicherung. Ein weiterer entscheidender Schritt wird im Rahmen der Einführung von Rechnernetzwerken durch die zusätzliche Einführung eines Kommunikationsmoduls getan (vgl. Abb. 4, Fall 2). Da alle Module einer Anwendung zumindest theoretisch miteinander kommunizieren, kann die Kommunikation auch als Bestandteil der anderen Module angesehen werden. In der Web-Umgebung ist Kommunikation jedoch von so herausragender Bedeutung, dass sie im Folgenden als separates Modul betrachtet wird; sie ist im Sinne von anwendungsspezifischer Kommunikation zu verstehen. Damit soll der Bezug zu spezifischen Kommunikationsmustern, verwendeten Nachrichten- und Datenstrukturen herausgearbeitet werden; dies geschieht im Kontrast zur Kommunikationsschicht der Plattformarchitektur, die sich mehr oder weniger auf die technische Kommunikationsinfrastruktur bezieht.
Eine umfassendere und weiterführende Erweiterung ist in Fall 3 der Abb. 4 dargestellt, wo drei weitere Module einer Web-Anwendungsarchitektur identifiziert werden. Diese sind nicht in jedem Fall notwendig und müssen auch nicht in explizit der hier vorgestellten Terminologie erfolgen, sind jedoch ein gutes Beispiel für die Erweiterbarkeit des hier vorgestellten Architekturrahmenswerks. Namentlich sind diese Module „Suchen&Auffinden“ (search & discover), „Exportschnittstelle“ (export interface) und „Sicherheit“ (security). Das Modul „Suchen&Auffinden“ adressiert dabei die Registrierung von Web-Anwendungen in einer globalen und online verfügbaren Registratur sowie das Suchen und Auffinden von Web-Anwendungen, die eventuell auch als Bestandteil komplexer kompositer Anwendungen verwendet werden können. Das Sicherheitsmodul umfasst dabei alle Sicherheitsfunktionen von Web-Anwendungen, wie z.B. Authentifizierung, Autorisierung, sichere Übertragung und Nutzerverwaltung. Es gibt dabei wieder mindestens zwei Wege auf denen Sicherheit in ein Web-Anwendungen eingebracht werden kann: man kann diese entweder als Charakteristikum und Bestandteil jedes Anwendungsmoduls auffassen (z.B. kann eine Datenverwaltungssystem bestimmte Sicherheitsvorkehrungen besitzen oder ein Kommunikationsmodul unter verschiedenen Sicherheitsparametern betrieben werden) oder eben als wirklich separates Modul betrachten. Jeder dieser beiden Ansätze hat sein Vor- und Nachteile, die hier nicht näher betrachtet werden sollen. In der weiteren Ausführung wird aus Gründen der expliziten Benennung der Funktionalität die zweite Variante bevorzugt.
Erweiterbarkeit und Flexibilität sind wiederum die grundlegenden Prinzipien der hier vorgestellten Web-Anwendungsarchitektur. Keines der bisher diskutieren Anwendungsmodule ist zwingend notwendig. Jeder Architekt kann subjektiv selbst entscheiden, ob er hier eingeführte Module verwerfen oder neue einführen will. Messlatte sind die Belange der konkret umzusetzenden Anwendung. Beispielsweise sehen wir oft die Notwendigkeit, ein zusätzliches Beschreibungsmodul einzuführen („Description“). Dessen Zweck ist es die vielseitigen Aspekte einer Web-Anwendung zu beschreiben: Welche Aufgaben erfüllt diese? Wie arbeitet diese? Welche Daten werden von dieser benötigt? Dieses Beschreibungsmodul kann als Kombination aus einer Beschreibungssprache (z.B. WSDL), Meta-Daten und Ontologien (vgl. [CP01] , [Ne02], [JPM02]) betrieben werden..

3.3 Architekturrahmenswerk für Web-Anwendungen

Das Architekturrahmenswerk für Web-Anwendungen vereint die Web-Anwendungsarchitektur und die Web-Plattformarchitektur in einer erweiterbaren Matrix. Zu diesem Zweck werden die identifizierten und als wichtig erachteten Komponenten

der Web-Plattformarchitektur (Abb. 3) als eine Dimension einer Matrix dargestellt (vertikale Dimension). Die interessanten Komponenten der Web-Anwendungsarchitektur bilden die horizontale Dimension der Matrix. Jedes Feld der Matrix korrespondiert mit einem WPA- und einem WAA-Element und wird im Folgenden als Quadrant bezeichnet. Wir bevorzugen auf Grund der Klarheit und Einfachheit die Darstellung in einer Matrix. Die Verwendung einer alternativen Notation, die ein ähnliches Problem angeht, ist als „pole shoe"-Notation in [No00] illustriert.

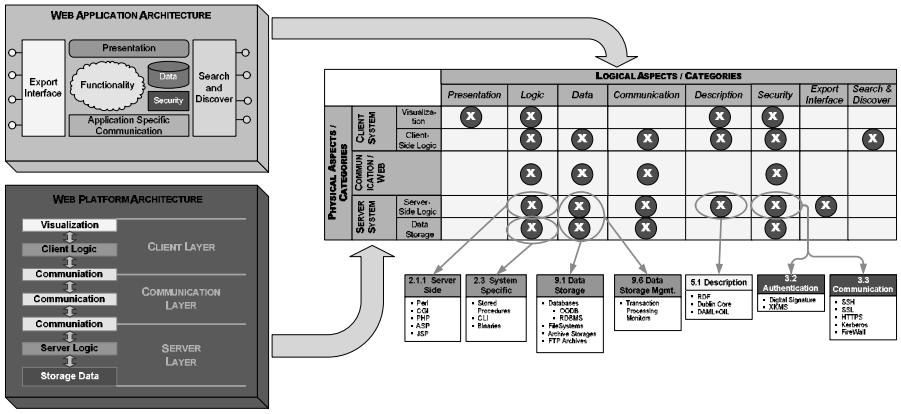

Abb. 5. Architekturrahmenswerk für Web-Anwendungen

Wie bereits erwähnt, ist das Architekturrahmenswerk aus zweierlei Gesichtspunkten nützlich. Auf der einen Seite kann es verwendet werden, um Architekturen für Web-Anwendungen zu definieren. Dazu setzen wir gekreuzte Kreise in die Matrixelemente (vgl. Abb. 5). Ein Kreuz in einem Quadranten zeigt die Verknüpfung der entsprechenden Module beider Architekturen an. Zum Beispiel steht das Kreuz in der Spalte „Daten" (Data) und der Zeile „Kommunikation/Web" (Communication/Web) für die Tatsache, dass hier Datenelemente mit spezifischem Bezug auf die Kommunikation im Web – wie z.B. Zwischenspeicher wie Caches oder Proxies – Verwendung finden. Analog dazu bedeutet die fehlende Markierung im Quadranten der Spalte „Präsentation" (Presentation) und der Zeile „serverseitige Logik" (Server-Side Logic), dass keine Nutzerschnittstelle mit clientseitiger Logik benötigt wird, um die Serverfunktionalität zu koordinieren. Auf der anderen Seite kann das Architekturrahmenswerk im weitesten Sinn dazu verwendet werden, relevante Standards und Technologien zu klassifizieren. Um in diesem Sinn handeln zu können, benötigt man eine Klassifikation der Technologien, wie sie im Abschnitt 3.4 erarbeitet, vorgestellt und diskutiert wird. Unter der Vorraussetzung, dass ein Quadrant bereits mit einem Kreuz markiert ist, können diesen Kategorien aus der Klassifikation zugewiesen werden, indem eine Klassifikationskategorie mit einem Quadranten verbunden wird. Zum Beispiel kann auf Grund des Kreuzes im Quadranten der Spalte „Daten" (Data) und der Zeile „Datenspeicherung" (Data Storage) die Klassifikationskategorie „Datenspeicherung" (Data Storage) zugeordnet werden. Nach diesem Schritt kann der Architekt aus den sich anbietenden Alternativen Datenbank, Dateisystem, FTP-Archiv usw. wählen. Auch mehrere Klassifikationskategorien können einem Quadranten zugeordnet werden. Diese Vorgehensweise wird insbesondere im Rahmen des Beispiels in Abschnitt 3.5 näher erläutert.

3.4 Klassifikation bestehender Technologien

In diesem Abschnitt wird eine beispielhafte Klassifikation unterschiedlicher mit dem Web verbundener Technologien und Standards eingeführt. Das Ziel ist es, einige Prinzipien für einige Klassifikation aufzuzeigen, die bei der Entwicklung einer Taxonomie mit dem Ziel einer Vervollkommnung des Architekturrahmenswerks Verwendung finden können.

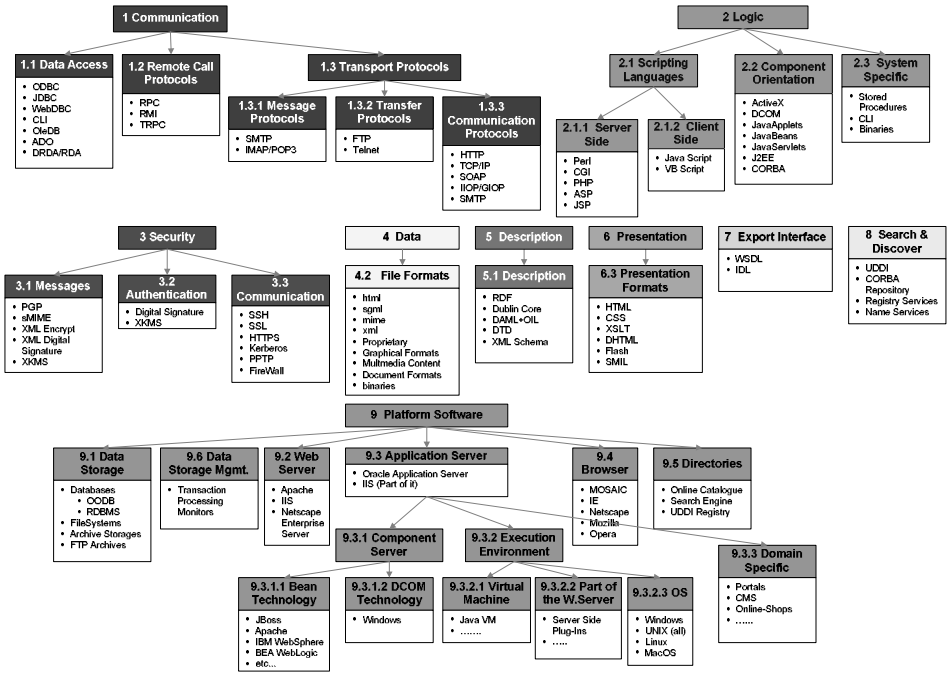

Abb. 6. Beispielklassifikation von Web-Standards und Technologien

Das anfängliche Problem, dem man bei der Entwicklung einer solchen Klassifikation aus dem „Nichts“ begegnet, ist die Auswahl einer geeigneten Semantik für die Kategorien der Wurzelknoten. Wir konnten empirisch feststellen, dass die Verwendung der Module aus der Web-Anwendungsarchitektur als oberste Klassifikationskategorien Sinn macht und wohl nahe der Realität ist. Unser Ziel ist es, etablierte Standards und Technologien aus dem Web-Umfeld zu klassifizieren, um diese bei der Verwendung von Web-Anwendungen gezielt einsetzen zu können. Die Verwendung von konkreten Softwareprodukten sollte in der Klassifikation vermieden werden, außer wenn sich diese zu so genannten De-facto-Standards entwickelt haben und dadurch fast zwangsweise eine Kategorie repräsentieren. Eine Beispielklassifikation findet sich in Abb. 6.

Große Technologien bzw. Technologieprodukte (z.B. Oracle Application Server) müssen möglichst detailliert und in feiner Granularität betrachtet werden, damit eine sinnvolle Klassifikation möglich ist. Dies liegt vor allem in deren Vielseitigkeit begründet, also der Tatsache, dass in einem Produkt zahlreiche Technologien vereint sind. So finden sich im „Oracle Application Server“ z.B. ein http-Server, ein Verzeichnisdienst und eine Java-Ausführungsumgebung. Eine Klassifikation nur unter „Application Server“ ohne wei-

tere Verfeinerung kann für das Vorgehensmodell in diesem Fall offensichtlich keinen Gewinn bringen. Ähnliche Technologien (z.B. DCOM und Enterprise Java Beans) sollten unabhängig von der Tatsache, dass sie zu unterschiedlichen Ansätzen gehören oder von unterschiedlichen Ansätzen stammen, unter der gleichen Kategorie subsumiert werden. Um die Klassifikation so einfach wie möglich zu halten, haben wir noch keine Methode zur Festlegung von Abhängigkeiten zwischen einzelnen Technologien und Standards eingesetzt. In der Kategorie Plattform-Software (Platform Software) sind Softwaremodule, aus denen eine Web-Plattform zusammengesetzt werden kann, in einer einfachen Klassifikation dargestellt. Darin enthalten ist eine Mischung aus gängigen Standards und kommerziellen Softwareprodukten.
Die Klassifikation in Abb. 6 ist nur eine Beispielklassifikation, die sich als Orientierungsrahmen in unserer aktuellen Forschung bewährt hat. Wir wollen auf dieser Basis nicht einen breiten Nutzen dieser Klassifikation für jeden Architekten in jedem beliebigen Kontext ableiten, sondern vielmehr jeden dazu motivieren, sich auch seiner Sicht mit dieser Klassifikation auseinanderzusetzen, sie zu verbessern und auf seinen Bedarf anzupassen.

3.5 Beispielhafte Anwendung des Architekturrahmenswerks

Dieser Abschnitt ist einem kompakten Beispiel gewidmet, mit dem Ziel das in den vorherigen Abschnitten vorgestellte Architekturrahmenswerk einer praktischen und konkreten Anwendung zuzuführen. Insbesondere stellen wir in diesem Beispiel das Vorgehensmodell zur Findung einer Architektur für Web-Anwendungen und Web-Anwendungsplattformen vor. Das hier vorgestellte Beispiel beschäftigt sich mit dem Entwurf einer kleinen Shop-Lösung. Die Vorgehensweise zur Erstellung der Architektur ist iterativ. Wird ein Arbeitsschritt abgeschlossen, ist es notwendig die Ergebnisse, die im vorangegangenen Schritt erzielt worden sind, auf Basis der im aktuellen Schritt erzielten Ergebnisse nochmals zu verifizieren.
Im ersten Schritt modellieren wir eine konkrete Architektur in der Bedeutung von WPA und WAA. Die Aufgabe kann auf die einfache Tätigkeit des Einordnens der WAA-Komponenten in die Schichten der WPA herunter gebrochen werden (vgl. Abb. 7.a links). Dabei ist es wichtig, mit einer kleinen Anzahl von Schichten zu beginnen, um nicht den Überblick zu verlieren – wir empfehlen die drei Basisschichten. Sobald eine weitere Schichtenunterteilung innerhalb einer bereits vorhandenen Schicht zu erkennen ist, muss der Architekt entscheiden, ob er diese Schicht zerlegen soll. Der Prozess des Anordnens ist dabei weitgehend subjektiv und von den Vorstellungen des Architekten abhängig, wobei es vor allem wichtig ist, den genauen Zweck eines jeden eingeordneten Elements zu dokumentieren. So steht z.B. die Anwendungslogikkomponente in der Visualisierungsschicht (Abb. 7.a) für ein Applet, das ein statistisches Diagramm der Erlöse über der Zeitachse ausgibt. Der Architekt muss sich ebenso um die konkreten Beziehungstypen zwischen verschiedenen Komponenten Gedanken machen. Gerade weil hier eine Modellierung auf konzeptioneller Ebene das Ziel ist, muss eine niedrige Komplexität angestrebt werden. Die Komplexität kann dann mit der Zahl der Iterationen erhöht und damit das Modell verfeinert werden.

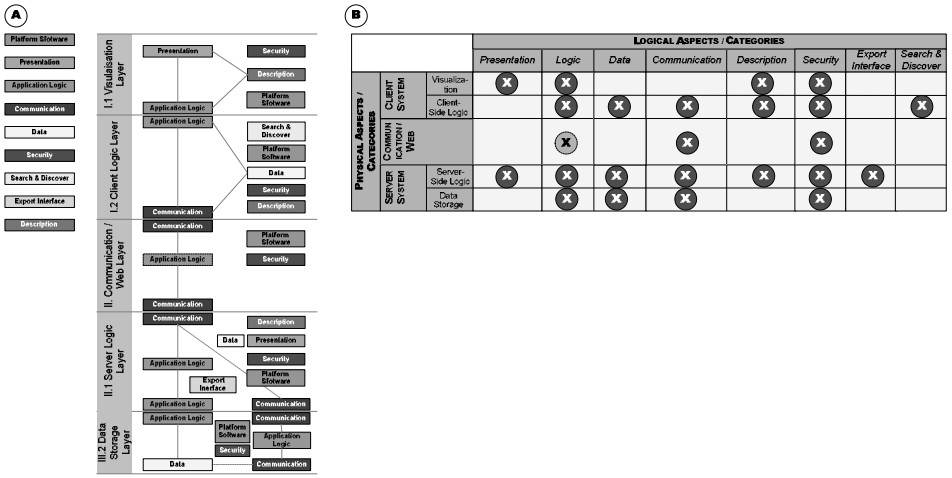

Abb. 7. (a) Web-Plattformarchitektur mit Web-Anwendungskomponenten
(b) Die Architektur aus (a) dargestellt in der Web-Ramenswerkarchitekturmatrix

Im nächsten Schritt muss die Struktur aus Abb. 7.a in die Matrix aus Abb. 7.b überführt werden. Die Transformation ist einfach und erfordert nur das Setzen eines Kreuzes in jedem Quadranten in dem eine Übereinstimmung und somit Verwendungsabsicht vorliegt. Nachdem die Matrix vollständig ausgefüllt ist, kann es von Interesse sein, die Architektur erneut zu überdenken. Wenn festgestellt wird, dass noch weitere neue Komponenten benötigt werden oder das alternative oder optionale Komponenten existieren, dann müssen diese in einer auffälligen Art und Weise markiert werden (in Abb. 7 Verwendung einer gestrichelten Linie). Nun empfiehlt es sich noch einmal über die Ergebnisse von diesem und den vorhergehenden Schritten zu iterieren, um eventuell noch vorhanden Abhängigkeiten aufzulösen.
Im nächsten Schritt wird nun die Matrix (Abb. 7.b) basierend auf der Beispielklassifikation und den darin enthaltenen Technologien und Standards aus Abb. 6 erweitert. Eine Einheit der Klassifikation ist dabei innerhalb einer möglichst minimalen Anzahl von Quadranten zu platzieren. Da jede dieser Kategorieeinheiten mehr als eine konkrete Technologie oder einen Standard beinhaltet, entstehen für den Architekten eine Vielzahl von Alternativen zur Lösung der vorhandenen Probleme. Es ist nun Aufgabe der Architekten eine geeignete Entscheidung zu treffen.
Die Anwendungslogikkomponente aus Abb. 7.a und die dazu vorhandene Markierung in Abb. 7.b implizieren, dass die für die Konzeption verantwortliche Person einen Repräsentanten aus den Unterkategorien von Kategorie 2 (Logik) aus Abb. 6 auswählen muss. Abhängig von den vorhandenen Rahmenbedingungen und des ausgewählten Ansatzes kann eine Wahl zwischen 2.1.2 (clientseitige Logik) und 2.2 (Komponentenorientierung) getroffen werden. Im Folgenden wollen wir von der Annahme ausgehen, dass eine Entscheidung zu Gunsten clientseitiger Logik getroffen wird und damit die Entscheidung auf eine Implementierung mittels Java Applets fällt. In diesem Fall muss dann auch die Java Virtual Machine als dazu passende Ausführungsumgebung gewählt werden (Klassifikationsbox 9.3.2). Der grundlegende Vorteil des Architekturrahmenswerks liegt in diesem Fall darin, dass die Entwurfsentscheidung auf konzeptioneller Ebene getroffen wird und erst in einem nächsten Schritt mit einer konkreten Technologie ausgefüllt wird. An

der eben erwähnten Stelle könnten beispielsweise an Stelle von Java Applets z.B. ActiveX-Steuerelemente herangezogen werden. In diesem Fall müsste als Ausführungsumgebung dann Windows anstatt der Java Virtual Machine gewählt und der „Internet Explorer“ als Repräsentant aus Kategorie 9.4 (Browser) gewählt werden.
Das hier vorgestellte Architekturrahmenswerk für Web-Anwendungen ist ein typischer Vertreter einer Top-Down-Methode verbunden mit einer Schichtenarchitektur. Als solcher erlaubt dieser Ansatz einen systematischen Entwurf von Web-Anwendungen aus gut entworfenen und strukturierten Modulen mit einer typischen Vielfalt und Vielzahl von Abstraktionsschichten. Der Ansatz ist dabei möglichst flexibel und nicht eng an einen bestimmten Implementierungsansatz gebunden. Die Nachteile, die durch den Ansatz an den Tag gelegt werden, sind ein gewisser Mangel an Korrektheit und Präzision, die Architekturen für komponentenorientierte und Middleware-basierte Systeme erwarten.

4 Bewertung

Dieser Aufsatz stellt kein Vorgehensmodell vor, dass anhand objektiv messbarer Größen den Weg in Richtung optimal entworfener Web-Anwendungen weist. Wir sind davon überzeugt, dass es keinen solchen Weg geben kann. Was der Leser hier aber finden kann, ist eine strukturierte Vorgehensweise zum Entwurf von Web-Anwendungen, die basierend auf einer orthogonalen Trennung von Plattform und Anwendung, sinnvolle Entwurfsschritte mit einer Klassifikation und einer klaren Terminologie verbindet. Abgesichert wird die Vorgehensweise durch ihren iterativen Charakter, der das wiederholende Überprüfen von Entwurfsentscheidungen einfordert.

Literaturverzeichnis

[Bu96] Buschmann, F.; Meunier, F.; Rohnert, H.; Sommerlad, P.; Stal, M.: Pattern-oriented Software-Architecture. John Wiley and Sons Ltd, 1996.

[CLF00] Caroli, P.; de Lucena, C.; Fontoura, M.: An Architecture for the Evolution of Web Applications. 2000.

[CP01] Corcho, O.; Gómez-Pérez, A.: Solving integration problems of e-commerce standards and initiatives through ontological mappings. 2001.

[HR99] Härder, T., Rahm, E.: Datenbanksysteme – Konzepte und Techniken der Implementierung. Springer, Berlin Heidelberg New York, 1999.

[JPM02] Jablonski, S.; Petrov, I.; Meiler, C.: Repositories as Integration Enabler for the Web. 2002. Technical Report. University of Erlangen- Nuernberg, 2002

[Me02] Merz, M.: E-Commerce und E-Business – Marktmodelle, Anwendungen und Technologien. dpunkt.verlag, Heidelberg, 2002.

[Ne02] Newcomer, E.: Understanding Web Services. Addison-Wesley, Boston, 2002.

[No00] Noack, J.; Mehmaneche, H.; Zendler, A.: Architectural Patterns for Web Applications, 2000.

[So00] Sommerville, I.: Software Engineering. Addison-Wesley, Boston, 2000.