1 Web Engineering – Die Disziplin zur systematischen Entwicklung von Web-Anwendungen

Gerti Kappel, Birgit Pröll, Siegfried Reich, Werner Retschitzegger

Moderne Web-Anwendungen stellen vollwertige, komplexe Softwaresysteme dar. Die Entwicklung dieser Web-Anwendungen erfordert daher eine ingenieurmäßige und methodisch fundierte Herangehensweise. Ausgehend vom Software Engineering umfasst Web Engineering die Anwendung systematischer und quantifizierbarer Ansätze, um Spezifikation, Implementierung, Betrieb und Wartung qualitativ hochwertiger Web-Anwendungen durchführen zu können. Abhängig von der Entwicklungshistorie und dem Komplexitätsgrad weisen Web-Anwendungen dokumentenzentrierte, interaktive, transaktionale oder ubiquitäre Merkmale oder auch solche des semantischen Web auf. Die besonderen Anforderungen an Web Engineering werden aus den Charakteristiken von Web-Anwendungen abgeleitet, die sich aus der Sicht des Softwareprodukts, seiner Entwicklung und seiner Nutzung ergeben. Die Evolution wird dabei als Querschnittcharakteristikum betrachtet, das die drei anderen Bereiche umspannt.

1.1 Motivation

Das World Wide Web beeinflusst unser aller Leben nachhaltig. Von Wirtschaft und Industrie über Bildungs- und Gesundheitswesen bis hin zu öffentlicher Verwaltung und dem Freizeitsektor gibt es kaum einen Bereich des täglichen Lebens, in den das World Wide Web, kurz Web genannt, nicht Eingang gefunden hätte [GiMu01b]. Der Grund für diese Omnipräsenz liegt insbesondere in der Natur des Web, die gekennzeichnet ist durch globale und permanente Verfügbarkeit sowie komfortablen und einheitlichen Zugriff auf beliebig verteilte, von jedermann erstellbare Informationen in Form von Webseiten [Bern96, MDHG99].

Ursprünglich war das Web als reines Informationsmedium konzipiert. Eine *Website* wurde als eine Menge logisch zusammengehöriger und durch Verweise verbundener statischer Webseiten betrachtet. In letzter Zeit entwickelt sich das Web immer mehr zu einem Anwendungsmedium, Websites oder – in diesem Zusammenhang besser – *Web-Anwendungen* sind mittlerweile vollwertige, komplexe Softwaresysteme [BaGP00, GiMu01a, MDHG99]. Sie stellen interaktive,

datenintensive und personalisierbare Dienste über verschiedene Endgeräte zur Verfügung, arbeiten zur Realisierung von Benutzertransaktionen zustandsbasiert und legen die verwendeten Daten in der Regel in einer zugrunde liegenden Datenbank ab [HKRS02]. Was Web-Anwendungen von traditionellen Softwareanwendungen unterscheidet, ist der Einsatz des Web, d.h. seiner Technologien und Standards, sowohl als Entwicklungsplattform als auch als Nutzungsplattform. Eine Web-Anwendung kann daher wie folgt definiert werden:

> Eine Web-Anwendung ist ein Softwaresystem, das auf Spezifikationen des World Wide Web Consortium (W3C) beruht und Web-spezifische Ressourcen wie Inhalte und Dienste bereitstellt, die über eine Benutzerschnittstelle, den Web-Browser, verwendet werden.

Diese Definition inkludiert explizit sowohl den Softwareaspekt als auch die Interaktion mit den Benutzern. Die »Einschränkung« auf softwareintensiv *und* interaktiv stellt de facto eine Erweiterung des Problembereichs dar, weil sowohl der Software- als auch der Benutzerschnittstellenaspekt im Zusammenhang mit dem Web untersucht werden muss, was mit ein Ziel dieses Buchs ist.

Trotz der fundamentalen Änderung der Ausrichtung des Web von einem Informationsmedium zu einem Anwendungsmedium erinnert die heutige Situation der Ad-hoc-Entwicklung von Web-Anwendungen an Praktiken der Softwareentwicklung in den 60er Jahren, bevor man erkannte, dass die Entwicklung von Anwendungen mehr erfordert als reine Programmierexpertise [Gaed00, GrKn02, Muru00, Pres00a, ReSc00]. Die Entwicklung von Web-Anwendungen wird oftmals als ein einmaliges Ereignis angesehen, sie erfolgt vielfach in einer spontanen Art und Weise, sie basiert zumeist auf dem Wissen, den Erfahrungen und den Entwicklungspraktiken individueller Entwickler, beschränkt sich auf Wiederverwendung im Sinne des »Copy&Paste-Paradigmas« und ist letztendlich gekennzeichnet durch unzureichende Dokumentation von Entwurfsentscheidungen. Auch wenn diese Vorgehensweise pragmatisch erscheint, führen solche schnellen und unsauberen Entwicklungen zu umfangreichen Qualitätsmängeln und in weiterer Folge zu schwerwiegenden Problemen bei Betrieb und Wartung. Die entwickelten Anwendungen sind meist stark technologieabhängig und fehleranfällig, durch unzureichende Leistungsfähigkeit, Zuverlässigkeit und Skalierbarkeit gekennzeichnet und weisen eine mangelnde Benutzerfreundlichkeit und dadurch fehlende Akzeptanz auf [Frat99]. Aufgrund der hochgradigen Vernetzung von Web-Anwendungen steigt darüber hinaus die Gefahr, dass auftretende Probleme nicht lokal beschränkt bleiben.

Die Ursachen für diese Situation sind vielschichtig (vgl. z.B. [BaPB02, Gini00, Lowe99, Muru00]):

- **Dokumentenzentrierte Sichtweise**
  Die Entwicklung von Web-Anwendungen wird in vielen Fällen noch immer *dokumentenzentriert* gesehen, d.h., als eine rein redaktionelle Aufgabe, die die Erstellung von Webseiten, deren Verlinkung und das Einbinden von Grafiken umfasst [GiLR95]. Auch wenn bestimmte Web-Anwendungen (private Homepages, Onlinezeitungen etc.) in diese Kategorie fallen, ist diese Sichtweise zur Entwicklung softwareintensiver Web-Anwendungen nicht adäquat.
- **Vermeintliche Einfachheit der Web-Anwendungsentwicklung**
  Durch die Verfügbarkeit verschiedenster Werkzeuge wie HTML-Editoren und Formulargeneratoren (vgl. [Frat99]) wird es ermöglicht, auch ohne entsprechendes Fachwissen einfache Web-Anwendungen zu erstellen. Dabei wird oftmals der Präsentation mehr Aufmerksamkeit geschenkt als der internen Strukturierung und Programmierung. Inkonsistenzen und Redundanzen sind die Folge.
- **Nicht-Nutzung von Know-how aus relevanten Disziplinen**
  Es wird oftmals angenommen, dass die Entwicklung von Web-Anwendungen analog zur Entwicklung traditioneller Anwendungen zu behandeln wäre und damit die gleichen Methoden des Software Engineering anwendbar seien. Dies erscheint jedoch aufgrund der speziellen Charakteristika einer Web-Anwendung in vielen Fällen nicht adäquat (siehe dazu Unterkapitel 1.3). Gleichzeitig werden vielfach Konzepte und Techniken aus relevanten Bereichen wie Hypertext oder Mensch-Maschine-Kommunikation nicht konsequent eingesetzt [DeHM99]. Standards zur Entwicklung von qualitativ hochwertigen Web-Anwendungen sind, nicht zuletzt durch die relativ kurze Geschichte des Web, nicht existent.

Die gängige Praxis bei der Entwicklung von Web-Anwendungen sowie die gleichzeitig stark wachsende Komplexität und Bedeutung von Web-Anwendungen für viele Bereiche unserer Gesellschaft, insbesondere für die effiziente Abwicklung unternehmenskritischer Prozesse (wie beispielsweise im E-Commerce) [DeHa01], gibt Anlass zu wachsender Besorgnis über diese Art der Entwicklung und die langfristige Qualität von Web-Anwendungen, die bereits einen Großteil der heute entwickelten Individualsoftware ausmachen. Nach einer Studie des Cutter-Consortium [Cutt00] liegen die Hauptprobleme großer Web-Anwendungsprojekte in verfehlten geschäftlichen Zielsetzungen (84% der Fälle), Projektverzögerungen (79%), Budgetüberschreitungen (63%), Mangel an Funktionalität (53%) sowie Mangel an Qualität (52%). Als Konsequenz kann man von einer Neuauflage der Softwarekrise [NaRa68] sprechen, nämlich der Web-Krise (*web crisis* [GiMu01a]). Der Umfang der Web-Krise könnte jedoch aufgrund der heutigen Omnipräsenz von Web-Anwendungen und deren hochgradiger Vernetzung die proklamierte Softwarekrise der 60er Jahre um ein Vielfaches übertreffen [Muru00, LoHa99, ReSc02]. Dieser Herausforderung stellt sich Web Engineering.

Die Disziplin des *Web Engineering* tritt also einer Neuauflage der Softwarekrise entgegen. Web Engineering ist kein einmaliges Ereignis, sondern erstreckt sich ähnlich wie Software Engineering über den gesamten Lebenszyklus einer Web-Anwendung. Inwieweit unterscheidet sich nun Web Engineering von Software Engineering und ist es gerechtfertigt, von einer (eigenen) Disziplin zu sprechen?

Eine *Disziplin* ist definiert als Wissenschaftszweig, d.h. ein mehr oder weniger in sich abgeschlossenes Gebiet der Wissenschaft, das Forschung, Lehre und begründetes Wissen in Form von Veröffentlichungen umfasst [Hein93]. Die Vielzahl an Publikationen, Lehrveranstaltungen, entstehenden Curricula, Workshops und Konferenzen zeigt im Sinne dieser Definition, dass Web Engineering als ein eigenständiger Zweig des Software Engineering gesehen werden kann.[1] *Engineering* bezeichnet dabei die ingenieurmäßige Anwendung wissenschaftlicher Erkenntnisse mit dem Ziel, Systeme korrekt, sicher, schnell und kostengünstig zu bauen. *Software Engineering* ist definiert als »the application of science and mathematics by which the capabilities of computer equipment are made useful to man via computer programs, procedures, and associated documentation« [Boeh76]. In Anlehnung an diese Definition und in Anlehnung an [DMG+02] definieren wir daher Web Engineering wie folgt:

(1) Web Engineering ist die Anwendung systematischer und quantifizierbarer Ansätze (Konzepte, Methoden, Techniken, Werkzeuge), um Anforderungsbeschreibung, Entwurf, Implementierung, Test, Betrieb und Wartung qualitativ hochwertiger Web-Anwendungen kosteneffektiv durchführen zu können.
(2) Web Engineering bedeutet auch die wissenschaftliche Disziplin, die sich mit der Erforschung dieser Ansätze beschäftigt.

Aus Sicht des Software Engineering stellt die Entwicklung von Web-Anwendungen eine neue Anwendungsdomäne dar. Wenngleich Gemeinsamkeiten mit traditionellen Anwendungen existieren, müssen aufgrund der besonderen Charakteristika von Web-Anwendungen viele Ansätze aus dem Bereich des Software Engineering entsprechend angepasst oder neue Ansätze entwickelt werden [DeHM99, Glas03, MDHG99].

1. Für eine Übersicht siehe auch *www.webengineering.org*

Die Grundprinzipien des Web Engineering[2] können aber ähnlich wie jene des Software Engineering folgendermaßen charakterisiert werden (vgl. u.a. [DLWZ03, Lowe99]):

- Klar definierte Ziele und Anforderungen
- Systematische Entwicklung einer Web-Anwendung in Phasen
- Sorgfältige Ausgestaltung dieser Phasen
- Kontinuierliche Überwachung des gesamten Entwicklungsprozesses

Web Engineering ermöglicht die Planbarkeit und Wiederholbarkeit des Entwicklungsprozesses und damit auch eine kontinuierliche Weiterentwicklung von Web-Anwendungen. Dadurch kann nicht nur Kostenreduktion und Risikominimierung bei der Neuentwicklung und Wartung erreicht werden, sondern auch eine Qualitätssteigerung sowie die Messbarkeit der Qualität von Ergebnissen der einzelnen Phasen [GiMu01b].

Dieses Buch folgt in seinem Aufbau dem Buch *Software Engineering Body of Knowledge* (SWEBOK, [BoDu01]), d.h., die einzelnen Kapitel sind analog dem traditionellen Software Engineering angeordnet. Jeder der Beiträge nimmt dabei auf die besonderen Charakteristika der jeweiligen Thematik im Web Bezug. Im folgenden Unterkapitel werden Kategorien von Web-Anwendungen definiert. Darauf aufbauend beschreibt Unterkapitel 1.3 die besonderen Charakteristika von Web-Anwendungen. Im Anschluss daran gibt Unterkapitel 1.4 einen Überblick über die Struktur des Buchs.

1.2 Kategorien von Web-Anwendungen

Web-Anwendungen weisen unterschiedlichste Komplexitätsgrade auf. In Abbildung 1–1 werden anhand der *Entwicklungshistorie* und des *Komplexitätsgrads* verschiedene Kategorien von Web-Anwendungen identifiziert sowie jeder Kategorie konkrete Beispiele zugeordnet (vgl. [Muru00])[3]. Dabei gilt zu beachten, dass es einen Zusammenhang zwischen zeitlicher Entwicklung und Komplexität gibt. So sind Workflow-basierte Anwendungen beispielsweise transaktionsunterstützt, d.h., die höhere Entwicklungsstufe erfordert die zeitlich vorangegangene Entwicklung einer weniger komplexen Kategorie.

---

2. Verwandte Begriffe aus der Literatur, die sich mit einer ähnlichen Thematik auseinander setzen, sind beispielsweise *Web Site Engineering* [PoJC98, Schw97], *Hypermedia Engineering* [LoHa99], *Document Engineering* [GlMc02] und *Internet Software Engineering* [BaPB02]. Web Engineering ist im Vergleich dazu knapp und griffig, wenngleich als Begriff streng genommen nicht ganz korrekt – es wird ja nicht das Web, sondern eine Web-Anwendung ingenieurmäßig gebaut. Aber wollen Sie ein Buch über Web-Anwendungs-Engineering lesen ... ☺
3. Ähnliche Kategorisierungen von Web-Anwendungen finden sich z.B. in [Cona00, KPR+02b, PoJC98, Pres00d, Weit02].

Die Web-Auftritte von Organisationen, die bereits in den Anfängen des Web präsent waren, weisen oftmals eine ähnliche Entwicklungshistorie auf, wie in Abbildung 1–1 dargestellt. Gleichzeitig kann die Entwicklung einer Web-Anwendung in einer beliebigen Kategorie begonnen werden und in weiterer Folge entsprechend der angeführten Komplexitätsstufen erweitert werden. Jüngere Kategorien weisen in der Regel eine höhere Komplexität auf, was aber nicht gleichzusetzen ist mit einer Ersetzung der älteren Generation durch eine jüngere. Vielmehr haben nach wie vor alle Kategorien ihre spezifisch vorteilhaften Anwendungsgebiete. Dementsprechend können insbesondere komplexe Web-Anwendungen typischerweise mehreren der vorgestellten Kategorien zugeordnet werden. Online Shopping Malls beispielsweise integrieren nicht nur Anbieter verschiedenster Dienste, sondern bieten darüber hinaus verschiedenste Suchmöglichkeiten, Statusüberwachung von getätigten Bestellungen und in manchen Fällen auch Onlineauktionen an.

Es zeigt sich auch, dass die verschiedenen Kategorien von Web-Anwendungen viele traditionelle Anwendungsbereiche abdecken, wie beispielsweise Onlinebanking, dass aber auch völlig neue Anwendungsbereiche geschaffen werden, wie etwa das Angebot ortsabhängiger Dienste. Im Folgenden werden die wesentlichen Merkmale dieser Kategorien näher beschrieben.

Aus historischer Sicht stellen *dokumentenzentrierte Websites* gemäss der Definition in Unterkapitel 1.1 eigentlich die Vorläufer von Web-Anwendungen dar. Da jedoch einige Themenbereiche in diesem Buch auch für derartige »einfache« Web-Anwendungen Gültigkeit haben, werden diese im Folgenden kurz erläutert. Bei dokumentenzentrierten Websites werden Webseiten in Form von vorgefertigten, also statischen HTML-Dateien auf einem Webserver abgelegt und im Falle einer Anfrage als Antwort an den Web-Client geschickt. Die Aktualisierung der Webseiten erfolgt häufig manuell mit Hilfe entsprechender Werkzeuge. Dies stellt insbesondere bei änderungsintensiven Webseiten einen kostenintensiven Faktor dar und führt wegen der manuell nachzuführenden Änderungen von Realweltgegebenheiten häufig zu veralteten Informationen. Darüber hinaus besteht die Gefahr von Inkonsistenzen, da oftmals zum Zwecke eines komfortablen Informationszugangs bestimmte Inhalte redundant auf verschiedenen Webseiten gespeichert werden. Der hauptsächliche Vorteil liegt in der Einfachheit und Stabilität derartiger Websites sowie den niedrigen Antwortzeiten, da die Webseiten bereits vorgefertigt auf dem Webserver vorliegen. Statische Homepages, Web-Radio und einfache Firmenpräsentationen fallen in diese Kategorie.

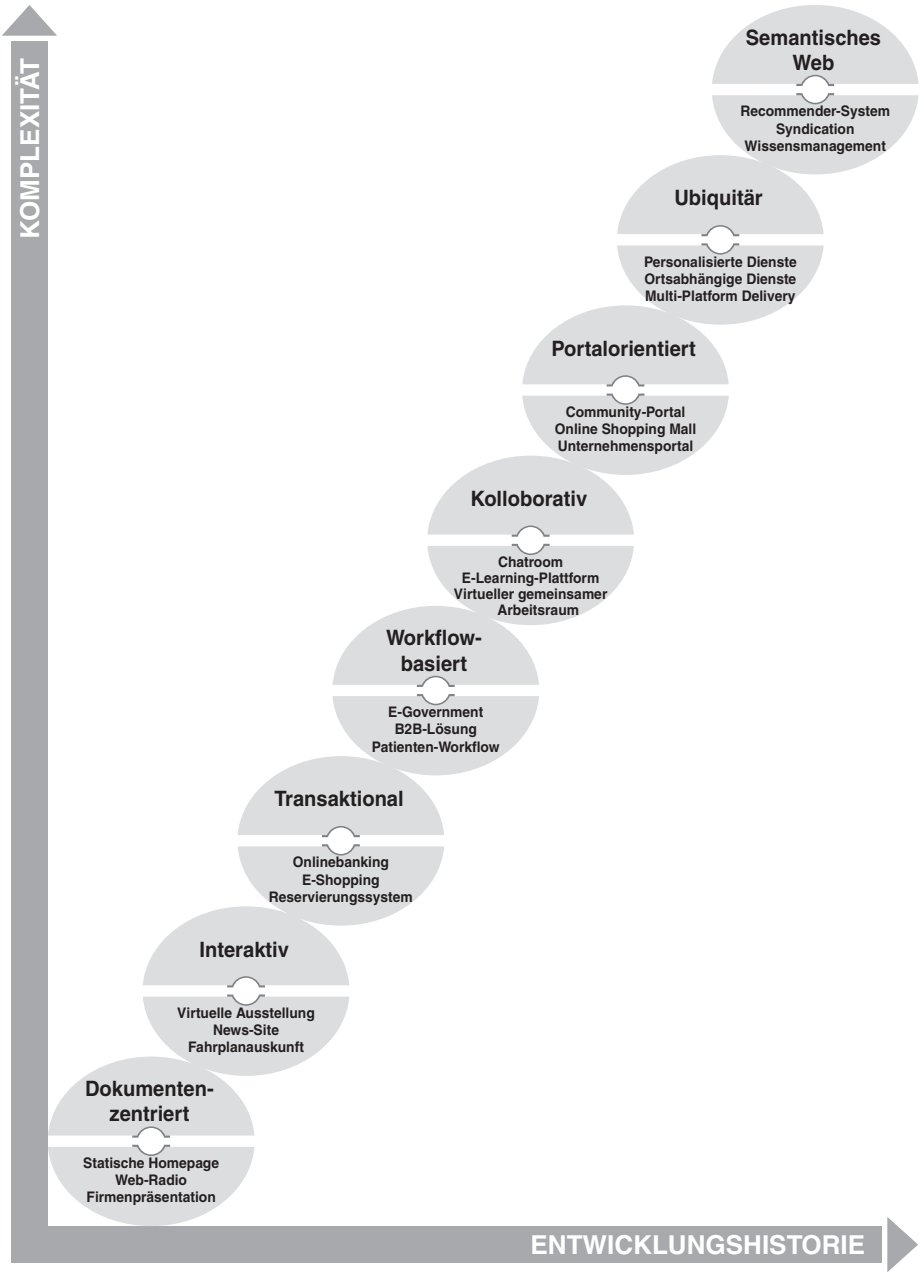

Abb. 1–1 *Kategorien von Web-Anwendungen*

Interaktive Web-Anwendungen bieten dem Benutzer (ursprünglich) durch die Einführung des Common Gateway Interface [CGI01] und der HTML-Formular-Technik eine erste einfache Form der Interaktivität über Eingabebereiche, Radio-

Buttons und Auswahllisten. Webseiten, aber auch Verweise auf andere Webseiten können dadurch abhängig von Benutzereingaben dynamisch generiert werden. Beispiele für diese Kategorie von Web-Anwendungen sind virtuelle Ausstellungen, News-Sites, aber auch Fahrplanauskünfte.

Die Realisierung *transaktionaler Web-Anwendungen* trägt dem Wunsch nach erhöhter Interaktivität Rechnung, indem Benutzer nicht nur lesend mit einer Web-Anwendung interagieren, sondern darüber hinaus auch Modifikationen vornehmen können. Dadurch können beispielsweise in einem Tourismusinformationssystem Datenbestände dezentral aktualisiert oder aber Zimmerreservierungen getätigt werden (vgl. z.B. [PrRW01]). Die Voraussetzung dafür schaffen Datenbanksysteme, die eine effiziente und konsistente Verwaltung der ständig zunehmenden Datenmenge von Web-Anwendungen erlauben sowie dem Benutzer die Möglichkeit strukturierter Abfragen bieten. Beispiele für diese Kategorie von Web-Anwendungen sind Systeme für Onlinebanking, E-Shopping und Reservierungssysteme.

Workflow-basierte Web-Anwendungen erlauben die Abwicklung von Geschäftsprozessen (*workflows*) innerhalb oder zwischen verschiedenen Unternehmen, Behörden und privaten Benutzern [GrRa03]. Eine treibende Kraft hierfür ist die Verfügbarkeit von entsprechenden Web-Services mit bekannten Schnittstellen [Leym03], um Interoperabilität zu gewährleisten. Die Komplexität der einzubindenden Dienste, die Autonomie der beteiligten Unternehmen und die Notwendigkeit zur Robustheit und Flexibilität der Geschäftsprozesse stellen dabei wesentliche Herausforderungen dar. Beispiele sind *Business-to-Business*-Lösungen (B2B) im E-Commerce-Bereich, E-Government-Anwendungen im Bereich der öffentlichen Verwaltung oder die Web-basierte Unterstützung von Patienten-Workflows im Gesundheitsbereich.

Während Workflow-basierte Web-Anwendungen eine gewisse Strukturierung der zu automatisierenden Prozesse und Vorgänge voraussetzen, werden *kollaborative Web-Anwendungen* insbesondere zur Kooperation bei unstrukturierten Vorgängen eingesetzt (*groupware*), wobei der Bedarf an Kommunikation zwischen den kooperierenden Benutzern besonders hoch ist. Kollaborative Web-Anwendungen unterstützen zum einen gemeinsame Informations- und Arbeitsräume (*shared workspaces*) (z.B. WikiWiki, *http://c2.com/cgi/wiki*, oder BSCW, *http://bscw.gmd.de*), um gemeinsame Informationen generiereren, bearbeiten und verwalten zu können. Zum anderen dienen sie zur Sitzungs- und Entscheidungsunterstützung (z.B. Argumentationssysteme wie QuestMap, *www.compendiuminstitute.org/tools/questmap.htm*, oder einfache Chatrooms), zur Gruppenterminplanung oder als E-Learning-Plattform.

Eine *portalorientierte Web-Anwendung* vereint den Zugriff auf verteilte, potenziell heterogene Informationsquellen und Dienste im Sinne eines *Single Point of Access* [Wege02]. Browserhersteller wie Microsoft und Netscape, Suchdienste wie Yahoo, Onlinedienste wie AOL, Medienkonzerne und andere Unter-

nehmen haben diesen Bedarf erkannt und bieten zentrale Anlaufseiten, so genannte Portale, als Einstiegshilfe ins Web an. Neben solchen allgemeinen Portalen existiert eine Reihe spezialisierter Portale wie beispielsweise Unternehmensportale, Marktplatzportale in Form von Online Shopping Malls und Community-Portale [ScSc99]. Unternehmensportale ermöglichen in Form eines Intranets oder Extranets eigenen Mitarbeitern bzw. Partnerunternehmen einen fokussierten Zugriff auf verschiedene Informationsquellen und Dienste. Bei Marktplatzportalen unterscheidet man zwischen horizontalen und vertikalen Marktplätzen. Horizontale Marktplätze zeichnen sich dadurch aus, dass sie im Business-to-Consumer-Bereich einem Massenpublikum Konsumgüter anbieten und im Business-to-Business-Bereich branchenfremden Unternehmen ihre Produkte. Vertikale Marktplätze fassen Unternehmen einer Branche zusammen, zum Beispiel auf der einen Seite Lieferanten und auf der anderen Seite weiterverarbeitende Betriebe. Community-Portale sind auf bestimmte Zielgruppen wie etwa junge Leute ausgerichtet und versuchen durch Interaktion zwischen den Besuchern Kundenloyalität zu schaffen oder aber durch eine entsprechende Benutzerverwaltung individuelle Angebote zu ermöglichen (o*ne-to-one marketing*).

Die zunehmend an Bedeutung gewinnende Kategorie von *ubiquitären Web-Anwendungen* stellt personalisierte Dienste zu jeder Zeit an jedem Ort und für eine Vielzahl von Endgeräten zur Verfügung, womit ein allgegenwärtiger Zugriff möglich wird. Ein Beispiel wäre die Anzeige von Mittagmenüs auf den mobilen Endgeräten jener Benutzer, die zwischen 11^{00} Uhr und 14^{00} Uhr ein Restaurant betreten. Eine wesentliche Voraussetzung dafür ist die Kenntnis des *Kontexts,* in dem die Web-Anwendung gerade benutzt wird, um bei Änderungen dieses Kontexts dynamisch, d.h. zur Laufzeit, entsprechende *Anpassungen* an der Web-Anwendung vornehmen zu können [KaRS01, KPR+02a]. Existierende Web-Anwendungen dieser Kategorie bieten häufig noch eine sehr eingeschränkte Form der Ubiquität, indem beispielsweise nur ein Aspekt unterstützt wird, entweder Personalisierung oder ortsabhängige Dienste oder endgerätespezifische Ein- und Ausgabe (*Multi-Platform Delivery*) [KPRS03].

Aktuelle Weiterentwicklungen, vor allem die fortschreitende Konvergenz von Informationstechnologie und Telekommunikation bewirken in naher Zukunft eine Situation, in der ubiquitäre Anwendungen den Markt beherrschen werden. Zu diesen Weiterentwicklungen zählt auch die aktuellste und zukunftsweisende Kategorie von Web-Anwendungen, die auf dem *semantischen Web* aufbaut. Ziel des semantischen Web ist, Informationen im Web nicht nur für den Menschen verständlich aufzubereiten, sondern für Maschinen automatisch verarbeitbar zu machen [BeHL01]. Wissensmanagement im Web, vor allem die Vernetzung und Wiederverwendung von Wissen, auch Syndication genannt, und das Finden von neuem relevantem Wissen, z.B. durch Recommender-Systeme, sollen dadurch ermöglicht werden. Die Zukunft wird uns weitere Anwendungen des semantischen Web bringen, die heute noch nicht einmal angedacht sind.

1.3 Charakteristika von Web-Anwendungen

Web-Anwendungen weisen im Unterschied zu traditionellen, nicht Web-basierten Anwendungen eine Reihe von Charakteristika auf, die es Wert sind, näher untersucht zu werden. Dabei handelt es sich zum einen um Charakteristika, die in herkömmlichen Anwendungen gänzlich fehlen (z.B. nichtlineare Navigation), und zum anderen um Eigenschaften, die in Web-Anwendungen besonders stark ausgeprägt sind (z.B. Häufigkeit von Änderungen) [BaPB02, McWe01b, Whit02]. Das Vorhandensein eines bestimmten Charakteristikums sowie dessen Intensität sind zum Teil abhängig von der Art der Web-Anwendung. So müssen transaktionale Web-Anwendungen wie E-Commerce-Systeme mehr Augenmerk auf die Aktualität und Konsistenz des Content legen als reine Informationsanbieter wie digitale Bibliotheken. Oftmals sind diese Charakteristika der Grund dafür, dass Konzepte, Methoden, Techniken und Werkzeuge des traditionellen Software Engineering nur in angepasster Form zur Entwicklung von Web-Anwendungen eingesetzt werden können oder sich sogar als gänzlich ungeeignet erweisen. Abbildung 1–2 gibt einen Überblick über diese Charakteristika und gruppiert sie anhand der drei Dimensionen *Produkt*, *Nutzung* und *Entwicklung* sowie deren *Evolution*.

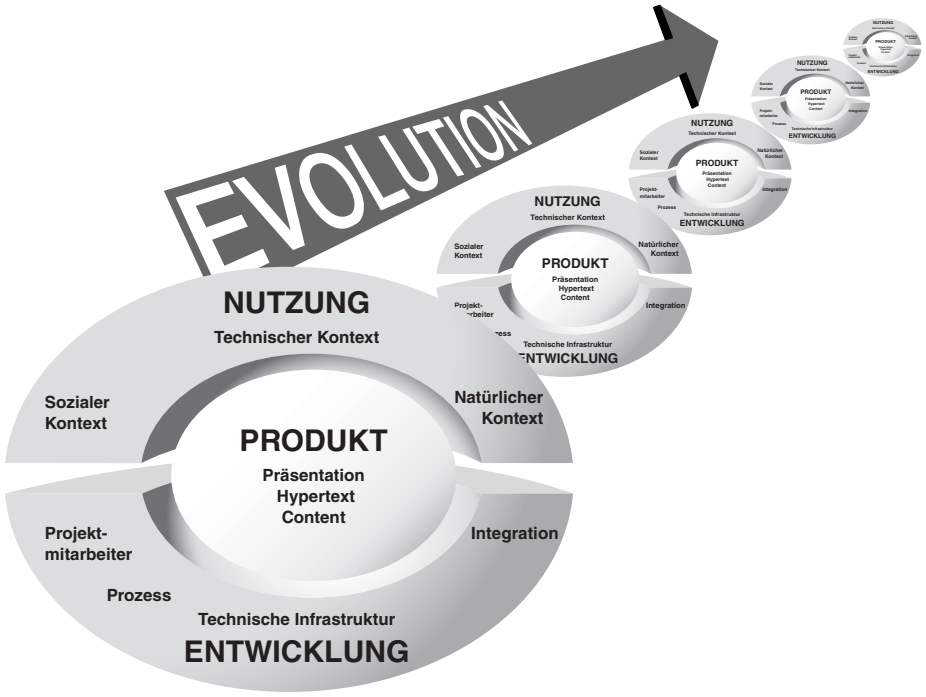

Abb. 1–2 *Gruppierungsdimensionen nach ISO/IEC 9126-1*

Diese Dimensionen basieren auf dem ISO/IEC-9126-1-Standard zur Bewertung der Qualität von Softwareanwendungen. Durch die Zuordnung der Charakteristika von Web-Anwendungen zu diesen Dimensionen lässt sich auch deren Einfluss auf die Qualität der Anwendungen aufzeigen, womit die Charakteristika wiederum als Ausgangspunkt zur Ableitung der Anforderungen an Web Engineering herangezogen werden können (siehe auch Unterkapitel 1.4). Neben den produktbezogenen, nutzungsbezogenen und entwicklungsbezogenen Charakteristika existiert die Evolution als ein »Querschnittcharakteristikum«, dem alle drei Dimensionen, Produkt, Nutzung und Entwicklung, unterworfen sind. Produkte sollten anpassbar sein, neue Kontextinformation sollte bei der Nutzung berücksichtigt werden und die Entwicklung ist mit sich ständig ändernden Rahmenbedingungen konfrontiert, um nur einige Beispiele zu nennen. Im Folgenden werden die einzelnen Charakteristika beschrieben. Dabei wird punktuell auf jene Kapitel verwiesen, die sich in ihren Ausführungen auf das Charakteristikum beziehen. Für eine zusammenfassende Übersicht des Einflusses dieser Charakteristika auf die Kapitel dieses Buchs wird auf Tabelle 1-1 verwiesen.

1.3.1 Produktbezogene Charakteristika

Produktbezogene Charakteristika finden sich in den wesentlichen Bestandteilen einer Web-Anwendung bestehend aus *Content* (den Inhalten), *Hypertext* (der Navigationsstruktur) und *Präsentation* (der Benutzerschnittstelle). Jeder dieser Bestandteile umfasst im Sinne des objektorientierten Paradigmas neben einem strukturellen oder statischen Aspekt auch einen Verhaltens- bzw. dynamischen Aspekt.

Content

Ebenso zentral wie die Entwicklung und Bereitstellung der eigentlichen Software einer Web-Anwendung sind das Erzeugen, die Bereitstellung, die Integration und die Aktualisierung des Content, d.h. der Inhalte, die eine Web-Anwendung zur Verfügung stellt. Web-Anwendungen werden ursächlich wegen des zur Verfügung gestellten Content verwendet – nach dem Motto *Content is King*. Besondere Aspekte, die dabei betrachtet werden müssen, sind das unterschiedliche Ausmaß der Strukturiertheit des Content sowie die Qualitätsansprüche, die die Benutzer an den Content stellen.

- **Dokumentenzentrierter Charakter und Multimedialität**
  Abhängig von der Strukturiertheit wird Content z.B. in Form von Tabellen, Text, Grafiken, Animationen, Audio oder Video zur Verfügung gestellt. Der Dokumentencharakter von Web-Anwendungen bezieht sich darauf, dass *Inhalte* kommuniziert werden, d.h., Dokumente erzeugt werden, die Informationen »didaktisch« für bestimmte Benutzergruppen aufbereiten (z.B. touris-

tische Information über die Region Salzkammergut). Dies bedingt u.a. besondere Anforderungen an die Usability (siehe Kapitel 11, Usability). Die Inhalte werden zum Teil auch dynamisch durch definiertes Verhalten erzeugt und aktualisiert, z.B. die Anzahl der verfügbaren Zimmer in einem Tourismusinformationssystem. Darüber hinaus wird das Web zur Übertragung multimedialer Inhalte verwendet, z.B. in Form von Videokonferenzen oder Real-Audio-Anwendungen.

- **Qualitätsansprüche**
  Abhängig vom Anwendungsbereich unterliegt der Content einer Web-Anwendung nicht nur unterschiedlichen *Änderungshäufigkeiten*, sondern auch unterschiedlichen *Qualitätsansprüchen* hinsichtlich Aktualität, Genauigkeit, Konsistenz, Verlässlichkeit und Umfang [WaSF95]. Dies erfordert nicht nur die Berücksichtigung dieser Qualitätsansprüche in der Anforderungsdefinition (siehe Kapitel 2, Requirements Engineering), sondern auch die Evaluierung deren Einhaltung (siehe Kapitel 7, Testen).

  News-Sites beispielsweise sind durch eine hohe Änderungshäufigkeit einerseits und hohe Ansprüche der Benutzer an die Aktualität andererseits gekennzeichnet. Das Web als eigenständiges Medium, neben Fernsehen, Radio und Printmedien, kann diesen Ansprüchen beispielsweise durch Individualisierung besser Rechnung tragen als traditionelle Medien.

  Eine Sonderstellung hinsichtlich Qualität nehmen z.B. Preis- und Verfügbarkeitsinformation in E-Shopping-Systemen ein, da diese die Grundlage eines Geschäftsabschlusses bilden (vgl. z.B. [PrRW01]). Inkorrekte Preise können eine Annullierung des Verkaufs nach sich ziehen, nicht aktuelle Verfügbarkeitsinformationen können dazu führen, dass tatsächlich vorhandene Produkte nicht verkauft werden oder es zu Engpässen kommt, da als verfügbar geführte Produkte in Wirklichkeit nicht verfügbar sind.

  Unabhängig vom Anwendungsbereich stellt Qualität des Content einen besonders kritischen Faktor für die Akzeptanz einer Web-Anwendung dar, wobei die Herausforderung darin besteht, Benutzern trotz großer Datenmengen und hoher Änderungshäufigkeiten die Qualität der publizierten Daten zusichern zu können.

Hypertext

Web-Anwendungen basieren prinzipiell auf Hypertext-Dokumenten [Conk87] und unterscheiden sich dadurch wesentlich von herkömmlichen Anwendungen. Das Hypertext-Paradigma als Grundlage zur Aufbereitung von Information geht zurück auf Vannevar Bush [Bush45]. Es existiert eine Vielzahl unterschiedlicher Hypertext-Modelle [McCa03], wobei das Web ein eigenes, sehr einfaches Modell benutzt. Generelle Grundbestandteile von Hypertext-Modellen sind *Knoten*, *Links* und *Anker*. Ein Knoten (*node*) ist eine nicht weiter unterteilte und eindeutig

identifizierbare Informationseinheit, im Web z.B. ein HTML-Dokument, die über eine URL (*Uniform Resource Locator*) adressierbar ist. Links (*links*) realisieren Verweise zwischen Knoten, wobei im Web diese ausschließlich unidirektional sind und die Bedeutung der Verweise nicht explizit festgelegt ist. Mögliche Bedeutungen wären »nächster Knoten gemäß empfohlener Lesereihenfolge« oder »Diagramm zu mathematischer Formel«. Anker (*anchors*) sind Quell- bzw. Zielbereiche von Links innerhalb von Knoten, z.B. eine Wortfolge im Text oder ein grafisches Objekt in einer Zeichnung. Im Web sind Anker nur in HTML-Dokumenten möglich.

Das essenzielle Merkmal des Hypertext-Paradigmas ist die *Nicht-Linearität* in der Produktion durch die Autoren und in der Konsumtion der Inhalte durch die Benutzer, verbunden mit den potenziellen Problemen der *Desorientierung* und *kognitiven Belastung*.

- **Nicht-Linearität**
  Hypertexte implizieren Stereotype des mehr oder weniger systematischen Lesens, wodurch sich Web-Anwendungen grundlegend von traditionellen Softwareanwendungen abheben. Dabei unterscheidet man unter anderem Stöbern (*browsing*), z.B. in E-Shopping-Anwendungen, Suchen (*query*), z.B. in virtuellen Ausstellungen, und geführtes Navigieren (*guided tour*), z.B. in E-Learning-Anwendungen. Das individuelle, den Bedürfnissen der Benutzer und deren Tätigkeiten angepasste Lesen von Inhalten kommt dem natürlichen Auffassungsvermögen des Menschen entgegen. Der Benutzer kann sich abhängig von Interesse oder Vorwissen im Informationsraum bewegen. Anker (und damit Links) werden nicht nur statisch durch die Autoren vordefiniert, sondern auch dynamisch durch entsprechend definiertes Verhalten erzeugt (*computed link*). Generell stellt das Erstellen von Hypertexten eine besondere Herausforderung für die Autoren dar, da bei der Konsumtion Desorientierung und kognitive Belastung vermieden werden sollen (siehe Kapitel 3, Modellierung, und Kapitel 11, Usability).
- **Desorientierung und kognitive Belastung**
  Eine besondere Aufgabe bei der Entwicklung von Web-Anwendungen besteht nun darin, diesen beiden grundsätzlichen Problemen des Hypertext-Paradigmas zu begegnen. Die *Desorientierung* bezeichnet die Tendenz, den Orts- und Richtungssinn in einem nichtlinearen Dokument zu verlieren. Unter *kognitiver Belastung* versteht man den zusätzlichen Aufwand und die zusätzliche Konzentration, die notwendig sind, um mehrere Wege oder Aufgaben gleichzeitig zu überblicken. Übersichtsdiagramme (*Sitemaps*), Schlagwortsuche, Rückverfolgen von »gegangenen Wegen« (*history mode*) und das Anzeigen von Zugriffszeit und Verweildauer unterstützen den Leser bei der Orientierung in der Anwendung. Gezielte Linksetzung und intelligente Benennung der Links vermindern die kognitive Belastung [Conk87]. Darüber hinaus wird

versucht, diesen Problemen durch wiederkehrende Muster bei der Modellierung des Hypertext-Aspekts entgegenzuwirken [GeCo00, LyRo01] (siehe Kapitel 3, Modellierung, und Kapitel 11, Usability).

Präsentation

Besonderheiten von Web-Anwendungen im Bereich der Präsentationsebene, also der Benutzerschnittstelle, betreffen insbesondere deren *Ästhetik* sowie die *Selbsterklärbarkeit*.

- **Ästhetik**
  Die *Ästhetik* der Präsentationsebene einer Web-Anwendung im Sinne des *Look and Feel* der Benutzerschnittstelle ist im Unterschied zu traditionellen Anwendungen nicht zuletzt aufgrund des hohen Konkurrenzdrucks im Web ein zentraler Faktor. Das Aussehen von Webseiten ist Modetrends unterworfen und insbesondere für E-Commerce-Anwendungen erfolgsentscheidend [Pres00d].
- **Selbsterklärbarkeit**
  Neben der Ästhetik ist bei Web-Anwendungen die *Selbsterklärbarkeit* von essenzieller Bedeutung, d.h., die Bedienung der Web-Anwendung muss weitgehend ohne Dokumentation möglich sein. Die Benutzungslogik, d.h. das Interaktionsverhalten, muss über die gesamte Web-Anwendung hinweg einheitlich gestaltet werden, so dass die Benutzer rasch Routine bei der Bedienung aufbauen können und ihnen das Arbeiten mit der Web-Anwendung vertraut ist (siehe Kapitel 11, Usability).

1.3.2 Nutzungsbezogene Charakteristika

Die Nutzung von Web-Anwendungen weist im Unterschied zu traditionellen Anwendungen starke *Heterogenitäten* auf. Benutzer unterscheiden sich in Anzahl und Kultur, Endgeräte variieren in Hardware- und Softwareeigenschaften, und auch Zeit und Ort der Nutzung sind nicht vorhersehbar [KaRS00]. Darüber hinaus besteht aus der Sicht der Anwendungsentwickler nicht nur Unkenntnis über die mögliche Vielfalt der Ausprägungen dieser so genannten Kontextfaktoren, sondern auch aufgrund deren Autonomie keine Möglichkeit, diese zu beeinflussen. Beispielsweise lässt sich die Anzahl der Zugriffe auf eine Web-Anwendung nur schwer vorhersagen (siehe auch Kapitel 2, Requirements Engineering, und Kapitel 12, Performanz).

Damit ist die Nutzung von Web-Anwendungen durch die Notwendigkeit der permanenten Anpassung an konkrete Nutzungssituationen, genannt Kontexte, gekennzeichnet. Die Anpassung an diese Kontexte kann dabei alle Bestandteile des Produkts, d.h. Content, Hypertext und Präsentation, gleichermaßen betreffen (siehe Kapitel 3, Modellierung). Aufgrund der zentralen Bedeutung der Anpas-

sung an Kontexte werden daher nutzungsbezogene Charakteristika nach der Art der Kontexte in *sozialen Kontext*, *technischen Kontext* und *natürlichen Kontext* unterteilt [KaRS00, KoWi01, KPRS03].

Sozialer Kontext: Benutzer

Der soziale Kontext umfasst Aspekte der Benutzer einer Web-Anwendung, wobei in diesem Zusammenhang insbesondere *Spontaneität* und *Multikulturalität* in hohem Ausmaß Heterogenität verursachen.

- **Spontaneität**
  Benutzer können jederzeit eine Web-Anwendung aufsuchen bzw. wieder verlassen, vielleicht sogar zur unmittelbaren Konkurrenz wechseln. Von einem Web-Anwender kann keine Loyalität zum Web-Anbieter erwartet werden, das Web ist ein unverbindliches Medium [HoCl02]. Benutzer verwenden Web-Anwendungen nur dann, wenn sie diese unmittelbar als Erfolg versprechend empfinden, denn Konkurrenzanwendungen sind über Suchmaschinen einfach auffindbar und sofort nutzbar.

  Die Spontaneität der Nutzung bedeutet auch, dass die Anzahl der Benutzer im Unterschied zu traditionellen Anwendungen nur schwer vorhergesagt werden kann und daher auf Skalierbarkeit besonderes Augenmerk gelegt werden muss [HeFo02] (siehe Kapitel 4, Architektur, und Kapitel 12, Performanz).

- **Multikulturalität**
  Web-Anwendungen werden für unterschiedliche Benutzergruppen entwickelt. Wird eine Web-Anwendung für einen geschlossenen Benutzerkreis im Sinne eines Intranets oder Extranets entwickelt, so entspricht dies am ehesten einer Nicht-Web-Anwendung. Wird eine Web-Anwendung für einen anonymen Benutzerkreis entwickelt, bestehen große und streng genommen nicht bestimmbare Heterogenitäten der Benutzer in Bezug auf Fähigkeiten (z.B. Behinderungen), Wissen (z.B. Domänenexpertise) und Präferenzen (z.B. inhaltliche Interessen) [Kobs01]. Um eine entsprechende Personalisierung durchführen zu können, werden bei der Entwicklung von Web-Anwendungen Annahmen über Benutzerkontexte getroffen, die in der Adaptierung der Bestandteile einer Web-Anwendung Berücksichtigung finden. Beispielsweise werden für Stammkunden spezielle Rabatte berechnet (Adaptierung des Content), Novizen werden über eine »guided tour« durch die Web-Anwendung geleitet (Adaptierung des Hypertexts), und Benutzer mit Sehbeeinträchtigungen werden durch geeignete Schriftgrößen unterstützt (Adaptierung der Präsentation). Personalisierung baut in der Regel auf der Angabe von Präferenzen durch den Benutzer auf, wie zum Beispiel die Bekanntgabe der bevorzugten Zahlungsart bei *www.amazon.com*.

Das Spektrum möglicher Benutzergruppen bedeutet auch, dass es schwierig ist, einen repräsentativen Benutzerkreis zur Erhebung der Anforderungen zu definieren (siehe Kapitel 2, Requirements Engineering).

Technischer Kontext: Netzwerk und Endgeräte

Der technische Kontext umfasst Eigenschaften bezüglich der Netzwerkverbindung im Sinne der *Dienstgüte* sowie der Hardware und Software der verwendeten Endgeräte, mit denen auf die Web-Anwendung zugegriffen wird, im Sinne eines *Multi-Platform Delivery.*

- **Dienstgüte**
  Web-Anwendungen basieren auf dem Client/Server-Prinzip. Die Eigenschaften des Übertragungsmediums wie Bandbreite, Latenzzeiten, Zuverlässigkeit, Gefahr von Verbindungsabbrüchen etc. existieren als autonome Größen, die bei der Entwicklung der Web-Anwendung berücksichtigt werden müssen, um eine entsprechende Dienstgüte (*quality of service*) zu gewährleisten [BFK+00, Pres00d]. Um beispielsweise die zu übertragende Datenmenge zu optimieren, kann der gegebene Parameter »maximale Bandbreite« dahingehend berücksichtigt werden, dass bei kleinerer Bandbreite multimediale Inhalte wie z.B. Videos in geringerer Auflösung übertragen werden. Während bei traditionellen Anwendungen die Leistungsmerkmale des Netzwerks in den meisten Fällen von vornherein bekannt sind, müssen bei Web-Anwendungen darüber hinaus Annahmen über diese Eigenschaften getroffen werden (siehe Kapitel 7, Testen, und Kapitel 12, Performanz).
- **Multi-Platform Delivery**
  Web-Anwendungen stellen in vielen Fällen Dienste nicht nur einer bestimmten Klasse von Endgeräten zur Verfügung, sondern beliebigen, auch mobilen Endgeräten, die sich durch unterschiedlichste Leistungsmerkmale auszeichnen (z.B. Bildschirmgröße, Speicherausstattung oder installierte Software) [EiVP01]. Eine Herausforderung stellt in diesem Zusammenhang die große Anzahl verschiedener Browserversionen dar, die durch unterschiedliche Funktionalitäten und Einschränkungen gekennzeichnet ist. Dies erschwert in hohem Maße die Realisierung einer konsistenten Benutzerschnittstelle und damit auch das Testen von Web-Anwendungen (siehe Kapitel 7, Testen).

  Ferner sind Browser vom Benutzer autonom konfigurierbar. Sowohl Darstellung, z.B. Bilder ausblenden, als auch Rechte, z.B. Zugriffsrechte für Java-Applets, und Funktionsumfang, z.B. Cookies und Caching, können stark verändert werden und beeinflussen so u.a. Performanz, Transaktionsfunktionalität und Interaktionsmöglichkeiten (siehe Kapitel 4, Architektur, Kapitel 5, Technologiebewusstes Design, und Kapitel 6, Implementierungstechnologien).

  Entwickler von Web-Anwendungen können aufbauend auf Annahmen über typische Endgeräte-Klassen u.a. Inhalte an PDAs (*Personal Digital Assis-*

tants) anpassen, indem Bilder oder Videos nicht übertragen (*web clipping*) und durch Verweise oder textuelle Erklärungen repräsentiert werden. Auf der Hypertext-Ebene können z.B. Hypertext-Dokumente für eine Druckversion aufbereitet werden. Um schließlich auf der Präsentationsebene den unterschiedlichen Varianten von JavaScript bei Browsern Rechnung zu tragen, können Bibliotheken eingesetzt werden, die von der jeweiligen Zielplattform abstrahieren (siehe z.B. *www.domapi.com*).

Natürlicher Kontext: Ort und Zeit

Der natürliche Kontext umfasst Aspekte des Zugriffsorts und des Zugriffszeitpunkts auf eine Web-Anwendung, wobei in diesem Zusammenhang insbesondere *Globalität* und *Verfügbarkeit* in hohem Ausmaß Heterogenität verursachen.

- **Globalität**
  Der Standort, von dem aus auf eine Web-Anwendung zugegriffen wird, z.B. in Form einer geografischen Position, dient einer entsprechenden Internationalisierung von Web-Anwendungen im Hinblick auf regionale, kulturelle und linguistische Unterschiede. Darüber hinaus kann der Standort zur Bestimmung einer logischen Position wie Wohnort oder Arbeitsplatz herangezogen werden, um damit ortsabhängige Dienste zu ermöglichen. Die globale Verfügbarkeit stellt gleichzeitig auch erhöhte Anforderungen an die Sicherheit von Web-Anwendungen, indem Benutzer weder unabsichtlich noch unberechtigt auf private oder vertrauliche Bereiche zugreifen dürfen (siehe Kapitel 13, Sicherheit).
- **Verfügbarkeit**
  Der durch die Natur des Web implizierte »Auslieferungsmechanismus« ermöglicht eine sofortige Verfügbarkeit auch von Teilen des Produkts. Darüber hinaus wird damit die Web-Anwendung sofort nutzbar, weshalb eine ausreichende Qualität des entwickelten Produkts sichergestellt sein muss. Gleichzeitig stellt die Möglichkeit einer permanenten Verfügbarkeit (24x7) erhöhte Anforderungen an die Ausfallsicherheit von Web-Anwendungen (siehe z.B. Kapitel 7, Testen). Neben der sofortigen und permanenten Verfügbarkeit von Web-Anwendungen wird durch die Berücksichtigung des Zeitaspekts auch die Realisierung zeitabhängiger Dienste ermöglicht (z.B. Fahrplanauskunft unter Berücksichtigung von Tageszeit und Wochentag).

1.3.3 Entwicklungsbezogene Charakteristika

Die Entwicklung von Web-Anwendungen wird geprägt durch die notwendigen Ressourcen wie *Projektmitarbeiter* und *technische Infrastruktur*, durch den *Entwicklungsprozess* selbst sowie durch die Notwendigkeit zur *Integration* bestehender Lösungen.

Projektmitarbeiter

Multidisziplinarität und *Juvenilität* des Entwicklungsteams kennzeichnen in besonderem Maße die Entwicklung von Web-Anwendungen und stellen gemeinsam mit der so genannten *Community-Entwicklung* eine völlig neue Form zur Organisation der Zusammenarbeit zwischen verschiedenen Entwicklergruppen dar. Die unterschiedlichen Sichtweisen und Schwerpunktsetzungen müssen nicht zuletzt durch ein entsprechendes Projektmanagement und durch einen angepassten Entwicklungsprozess integriert werden (siehe Kapitel 9, Web-Projektmanagement, und Kapitel 10, Entwicklungsprozess).

- **Multidisziplinarität**
  Web-Anwendungen können als eine Mischung aus Printpublishing und Softwareentwicklung, Marketing und Informatik, Kunst und Technologie charakterisiert werden [PoJC98]. Die Entwicklung von Web-Anwendungen sollte demnach auch als ein *multidisziplinäres Vorgehen* betrachtet werden, im Zuge dessen Wissen und Sachkenntnis aus verschiedenen Bereichen erforderlich ist. Neben IT-Experten, die für die technische Realisierung der Funktionalität des Systems verantwortlich zeichnen, sollten Hypertext-Experten und Designer für die Gestaltung von Hypertext und Präsentation herangezogen werden, während Domänenexperten für die Inhalte verantwortlich sind. Das Entwicklungsteam ist daher stärker als bei traditioneller Softwareentwicklung durch unterschiedliche Fähigkeiten und unterschiedliches Wissen gekennzeichnet (siehe auch Kapitel 5, Technologiebewusstes Design).

  Abhängig von der Art der Web-Anwendung, wird die eine oder andere Disziplin mehr in den Vordergrund treten. Während E-Commerce-Anwendungen stärker auf traditioneller Datenbank- und Programmierexpertise aufbauen, steht bei der Entwicklung einer virtuellen Ausstellung die Domänen- und Designexpertise im Vordergrund.

- **Juvenilität**
  Entwickler von Web-Anwendungen sind im Schnitt deutlich jünger und daher auch unerfahrener als traditionelle Softwareentwickler. Sie entsprechen in der Regel dem Stereotyp des Technik-Freak, der sich um Altbewährtes nicht kümmert und großes Interesse an neuen Technologien und Werkzeugen zeigt [McWe01b].

- **Community-Entwicklung**
  Ein vollkommen neues Phänomen stellt die Entwicklung von im Web frei verfügbarer Software inklusive Quellcode (*open source*) und deren Einbindung in »echte« Anwendungen dar. Entwickler verwenden diese Software und stellen ihre eigenen Entwicklungen wieder im Web zur Verfügung. Das bewusste Einbinden fremder Entwickler bzw. Entwicklungsgruppen mit ihren ungeschriebenen Gesetzen der Zusammenarbeit prägt diese neue Form der so

genannten Community-Entwicklung (siehe auch Kapitel 6, Implementierungstechnologien).

Technische Infrastruktur

Die *Inhomogenität* der verwendeten Komponenten sowie deren *Immaturität* sind wesentliche Charakteristika der technischen Infrastruktur von Web-Anwendungen (siehe Kapitel 4, Architektur, und Kapitel 5, Technologiebewusstes Design).

- **Inhomogenität**
  Bei der Entwicklung von Web-Anwendungen wird auf zwei wesentlichen Fremdkomponenten aufgebaut – dem Webserver und dem Web-Browser. Während man den (eigenen) Webserver zumeist selbst konfiguriert und bedient, kann auf die verwendeten Web-Browser und ihre Einstellungen kein Einfluss genommen werden. Diese Situation wird noch erschwert durch unterschiedliche, in Verwendung befindliche Browserversionen und ihr Zusammenspiel mit Plug-ins (siehe auch Abschnitt 1.3.2, Technischer Kontext).
- **Immaturität**
  Aufgrund des steigenden *Time-to-Market*-Drucks sind Komponenten, die in Web-Anwendungen verwendet werden, vielfach noch unreif, d.h., sie sind entweder fehlerbehaftet oder verfügen nicht über die gewünschte Funktionalität. Darüber hinaus ist ein Versionswechsel der Web-Anwendung oftmals mit einem Wechsel der Entwicklungsumgebung verbunden. Die Konsequenz ist, dass Entwicklungswissen verloren geht bzw. gar nicht erst aufgebaut werden kann.

Prozess

Der Entwicklungsprozess stellt zum einen den Rahmen für alle entwicklungsbezogenen Charakteristika dar, zum anderen ist er selbst durch *Flexibilität* und *Parallelität* geprägt (siehe Kapitel 9, Web-Projektmanagement, und Kapitel 10, Entwicklungsprozess).

- **Flexibilität**
  Die Entwicklung von Web-Anwendungen läuft nicht nach einem starr vorgegebenen Prozessschema ab. Flexibles Reagieren auf geänderte Rahmenbedingungen, wie zum Beispiel das Einhalten von Terminen mit flexiblen Inhalten, steht im Vordergrund.
- **Parallelität**
  Aufgrund der Notwendigkeit von kurzen Entwicklungszeiten und unterstützt durch die Tatsache, dass Web-Anwendungen oftmals in autonome Komponenten strukturiert werden können (Authentifizierung, Suchfunktion, Newsticker etc.), werden viele Web-Anwendungen parallel von Subgruppen entwi-

ckelt. Diese Subgruppen sind somit nach Anwendungen strukturiert und nicht nach Expertise der Projektmitarbeiter (GUI-Entwickler, Datenmodellierer etc.), wie vielfach bei traditioneller Softwareentwicklung [McWE01b].

Neben dieser *Parallelisierung nach (Teil-)Anwendungen* erfolgen oftmals auch die methodischen Tätigkeiten wie Design, Implementierung und Qualitätssicherung gleichzeitig für verschiedene Versionen. Während die Qualitätssicherung für eine frühere Version läuft, hat z.B. die Implementierung der nächsten Version bereits begonnen und für die übernächste Version beschäftigt man sich bereits mit dem Design. Aus dieser *Parallelisierung nach Phasen* ergeben sich neue Anforderungen für die Planung des Einsatzes der Mitarbeiter in Web-Projekten.

Integration

Ein besonderes Charakteristikum vieler Web-Anwendungen stellt die Notwendigkeit zur *internen* und *externen Integration* dar. Integration umfasst dabei nicht nur technische Aspekte (siehe Kapitel 4, Architektur, Kapitel 5, Technologiebewusstes Design, und Kapitel 6, Implementierungstechnologien), sondern auch inhaltliche Aspekte (siehe Kapitel 14, Semantisches Web) und organisatorische Aspekte (siehe Kapitel 10, Entwicklungsprozess).

- **Interne Integration**
  In vielen Fällen müssen Web-Anwendungen mit existierenden Legacy-Systemen integriert werden, indem existierende Inhalte, z.B. Produktkataloge, durch eine Web-Anwendung zur Verfügung gestellt werden sollen.
- **Externe Integration**
  Neben der angesprochenen internen Integration stellt die Integration von Inhalten und Diensten externer Web-Anwendungen ein besonderes Charakteristikum von Web-Anwendungen dar. Auch wenn sich in diesem Zusammenhang große Ähnlichkeiten zu heterogenen Datenbanksystemen ergeben, ist die Integration im Web darüber hinaus mit einer Reihe von Besonderheiten verbunden [LoHa99, SaCS02]. Zunächst handelt es sich oftmals um eine große Anzahl von Quellen, die häufig wechseln und durch hohe Autonomie hinsichtlich Verfügbarkeit und Änderung der Schemata gekennzeichnet sind. Darüber hinaus liegen meist nur wenige Detailinformationen über die Eigenschaften der Quellen vor, z.B. zu deren Inhalt oder zu deren funktionalen Fähigkeiten. Schließlich sind die verschiedenen Quellen meist durch Heterogenität auf den unterschiedlichsten Ebenen gekennzeichnet, von der Datenebene über die Schemaebene bis hin zur Datenmodellebene.

  Die Integration von externen Diensten, wie sie zum Beispiel bei portalorientierten Web-Anwendungen zum Einsatz kommt, zielt auf die sich schnell etablierende Entwicklungsform des Bereitstellens und Verwendens von *Web-Services* ab [Leym03]. Das Zusammenspiel unterschiedlicher Web-Services,

das Vermeiden von Seiteneffekten und die Dienstgüte sind nur einige der in diesem Zusammenhang relevanten Fragen.

1.3.4 Evolution

Evolution ist ein Querschnittcharakteristikum, dem, wie anfangs erwähnt, alle drei behandelten Dimensionen Produkt, Nutzung und Entwicklung unterworfen sind (siehe insbesondere Kapitel 8, Betrieb und Wartung). Die Notwendigkeit für Evolution lässt sich mit der *Unbegrenztheit* der Rahmenbedingungen und Anforderungen, dem *Konkurrenzdruck* und der sich generell abzeichnenden *Schnelllebigkeit* zusammenfassen.

- **Unbegrenztheit**
  Web-Anwendungen zeichnen sich durch eine hohe Änderungsdynamik aus und unterliegen damit einer permanenten Evolution aufgrund geänderter Anforderungen oder Rahmenbedingungen [Scha00]. Insbesondere die hohe Änderungsrate von Technologien und Standards im Web-Bereich machen es notwendig, Web-Anwendungen permanent an diese Technologien und Standards anzupassen bzw. diese anzuwenden. Dies deshalb, weil erstens die Erwartungshaltung der Benutzer auf den aktuellsten Hype im Web abzielt und zweitens die eingesetzten Werkzeuge ebenfalls technologiegetrieben sind. Gerade diese Unbegrenztheit der Rahmenbedingungen und Anforderungen ist ein zentrales Charakteristikum einer Web-Anwendung. Änderungen können dabei alle drei Dimensionen einer Web-Anwendung betreffen, das eigentliche Produkt, seine Nutzung und vor allem seine Entwicklung.
- **Konkurrenzdruck**
  Der außerordentlich starke Konkurrenzdruck im Bereich des Web, der *Time-to-Market-Druck* und die damit verbundene Notwendigkeit, im Web präsent zu sein (vergleichbar mit dem Goldrausch Ende der 40er Jahre [Muru00]), erzwingen immer *kürzere Produktlebenszyklen* und sehr *kurze Entwicklungszyklen* und lassen scheinbar keinen Spielraum für einen systematischen Entwicklungsprozess. Unverzüglicher Web-Präsenz wird höhere Priorität beigemessen als langfristiger Perspektive [Pres98].
- **Schnelllebigkeit**
  Der hohe Termindruck bei Web-Anwendungsentwicklungen basiert auf dem raschen Wandel im Web und der damit verbundenen kurzen Lebenszeit von Web-Anwendungen bzw. deren Änderungshäufigkeit. Tsichritzis hat es in [Tsic00] pointiert, aber treffend mit »either you are fast or irrelevant« ausgedrückt.

  Während die Evolution bei konventioneller Software in einer geplanten Reihe von Versionen realisiert wird, erfolgt sie bei Web-Anwendungen kontinuierlich, wodurch sich diese Anwendungen in einem permanenten War-

tungsprozess befinden. Die Änderungszyklen von Web-Anwendungen betragen oftmals nur wenige Tage oder Wochen [Pres00d]. Web-Anwendungen erfordern daher eine »Verschlankung« klassischer Verfahren des Software Engineering mit besonderer Betonung auf Anforderungsanalyse und Spezifikation einerseits (Kapitel 2) und Betrieb und Wartung (Kapitel 8) andererseits.

1.4 Ziele und Aufbau des Buchs

Die Ziele dieses Buchs können folgendermaßen charakterisiert werden:

- Einblick in aktuelle Konzepte, Methoden, Techniken, Werkzeuge und Erfahrungen zur ingenieurmäßigen Entwicklung von Web-Anwendungen
- Herausarbeitung von Gemeinsamkeiten und Unterschieden der Entwicklung nicht Web-basierter und Web-basierter Anwendungen
- Analyse der Eignung von Konzepten, Methoden, Techniken und Werkzeugen des traditionellen Software Engineering zur Entwicklung von Web-Anwendungen
- Aufzeigen potenzieller Risiken bei der Entwicklung von Web-Anwendungen
- Ausblick auf zukünftige Entwicklungen im Bereich des Web Engineering

Der Aufbau dieses Buchs orientiert sich am *Guide to the Software Engineering Body of Knowledge* (SWEBOK, [BoDu01]), in dem die verschiedenen Aktivitäten des traditionellen Software Engineering zusammengefasst werden. Diese Aktivitäten sind auch bei der Entwicklung von Web-Anwendungen anwendbar, wenngleich – wie in diesem Buch gezeigt wird – deren konkrete Ausgestaltung sowie auch deren zeitliche Abfolge zum Teil adaptiert werden müssen, um auf die in den Unterkapiteln 1.2 und 1.3 diskutierten Kategorien und Charakteristika von Web-Anwendungen Rücksicht zu nehmen.

Die verschiedenen Charakteristika von Web-Anwendungen beeinflussen in unterschiedlichem Ausmaß die verschiedenen Themengebiete des Web Engineering und stellen unterschiedliche Anforderungen an die Ausgestaltung von Konzepten, Methoden, Techniken und Werkzeugen. Diese Charakteristika haben daher auch den Aufbau dieses Buchs und den Inhalt der einzelnen Kapitel wesentlich mitgeprägt, wobei der Anspruch verfolgt wurde, das Gebiet des Web Engineering möglichst umfassend aufzuspannen.

Tabelle 1–1 zeigt den Einfluss der in Kapitel 1.3 diskutierten Charakteristika von Web-Anwendungen auf die in diesem Buch behandelten Themengebiete des Web Engineering. Eine umfassende Behandlung der unterschiedlichen Anforderungen, die sich aus dieser Vielfalt von Einflüssen ergeben, sowie der in Literatur und Praxis existierenden Lösungsmöglichkeiten ist Gegenstand der einzelnen Kapitel dieses Buchs.

Charakteristika von Web-Anwendungen			Charakteristikum \ Kapitel	Web Engineering												
				2: Requirements Engineering	3: Modellierung	4: Architektur	5: Technologiebew. Design	6: Implementierungstechn.	7: Testen	8: Betrieb und Wartung	9: Web-Projektmanagement	10: Entwicklungsprozess	11: Usability	12: Performanz	13: Sicherheit	14: Semantisches Web
Produkt	Content		Dokumentcharakter/ Multimedialität		x	x		x					x			
			Qualitätsansprüche	x	x				x	x			x			
	Hypertext		Nicht-Linearität		x		x		x				x	x		
			Desorientierung/ kognitive Belastung		x								x	x		
	Präsentation		Ästhetik	x	x		x				x		x			
			Selbsterklärbarkeit	x	x						x		x			
Nutzung	Sozialer Kontext		Spontaneität		x	x					x			x		
			Multikulturalität	x	x				x				x	x		
	Technischer Kontext		Dienstgüte	x	x				x				x	x		
			Multi-Platform Delivery	x	x	x	x	x	x				x	x		
	Natürlicher Kontext		Globalität		x	x			x						x	
			Verfügbarkeit		x				x			x				
Entwicklung	Projekt-mitarbeiter		Multidisziplinarität	x					x		x					x
			Juvenilität						x		x					
			Community-Entwicklung				x	x							x	x
	Technische Infrastruktur		Inhomogenität		x	x	x	x	x				x			x
			Immaturität			x	x	x		x	x					
	Prozess		Flexibilität	x					x		x	x				
			Parallelität								x	x				
	Integration		Interne Integration	x		x	x					x	x			
			Externe Integration	x		x	x					x	x			x
Evolution			Unbegrenztheit	x		x			x	x		x	x			
			Konkurrenzdruck			x	x		x	x		x				
			Schnelllebigkeit			x			x	x	x	x	x	x		

Tab. 1–1 *Anforderungen an Web Engineering*

Abbildung 1–3 zeigt in Anlehnung an [Balz00] die Struktur des Buchs als Komponenten eines Hauses. Die Kernkapitel werden dabei in die drei Säulen Vorgehensweise, Produktentwicklung und Qualitätsaspekte gegliedert. Kapitel 1 bildet das Fundament, auf dem alle weiteren Kapitel aufbauen. Kapitel 14 über das semantische Web bildet den Ausblick.

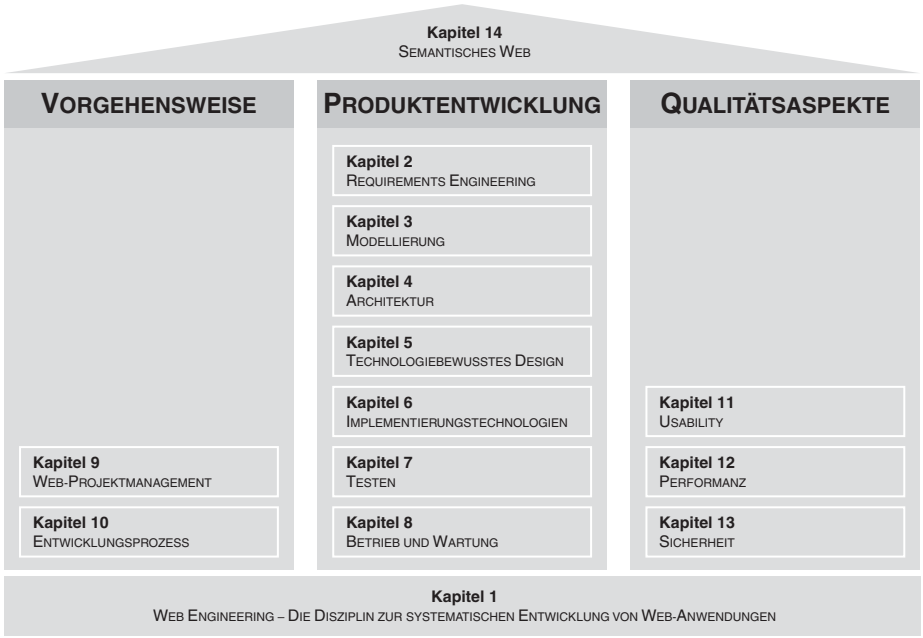

Abb. 1–3 *Aufbau des Buchs*

Generell kann dieses Buch sequenziell von vorne nach hinten gelesen werden. Gleichzeitig gibt es aber auch die Säulen und diese können durchaus als Cluster – und daher auch nicht sequenziell – gelesen werden. So gehören beispielsweise die Kapitel 9, Web-Projektmanagement, und 10, Entwicklungsprozess, zusammen, weil beide die Vorgehensweise behandeln. Dabei werden immer wieder die einzelnen Phasen der Entwicklung thematisiert. Der Begriff »Phase« wird in diesen Kapiteln, wie auch im Rest des Buchs, als Synonym für Tätigkeit bzw. Aktivität benutzt. Das heißt, nicht der sequenzielle Ablauf von Phasen steht im Vordergrund, sondern welche Ziele und damit welche Aktivitäten mit einer Phase verfolgt werden. Die Kapitel 2, Requirements Engineering, und 3, Modellierung, beschreiben Anforderungen und konzeptionelles Design. Darauf aufbauend beschreiben die Kapitel 4, Architektur, 5, Technologiebewusstes Design, und 6, Implementierungstechnologien, vor allem technische Merkmale von Web-Anwendungen. Das Kapitel 6, Implementierungstechnologien, hat dabei eine Art

Querschnittfunktion, da in vielen anderen Kapiteln auf Implementierungstechnologien Bezug genommen wird. Kapitel 7 widmet sich der Einhaltung bzw. Sicherstellung sowohl funktionaler als auch nicht funktionaler Anforderungen inklusive Fehlerbehandlung. Drei essenzielle Qualitätsaspekte werden in eigenen Kapiteln behandelt, und zwar in Kapitel 11, Usability, 12, Performanz, und 13, Sicherheit. Den Abschluss und Ausblick bildet Kapitel 14, Semantisches Web, das die Dimensionen künftiger Web-Anwendungen aufspannt.

Im Anhang folgen ein Glossar der wichtigsten Begriffe und Kurzbiographien der Autoren. Ein gemeinsames Literaturverzeichnis aller Kapitel sowie ein umfangreicher Index beschließen das Buch.

Die einzelnen Kapitel weisen folgende strukturelle Gemeinsamkeiten auf:

- Eine Zusammenfassung beschreibt die Essenz des jeweiligen Kapitels.
- Ein Unterkapitel fasst die generellen Konzepte der jeweiligen Aktivität zusammen, ohne auf die Charakteristika von Web-Anwendungen einzugehen.
- Darauf aufbauend werden unter Berücksichtigung der Charakteristika von Web-Anwendungen die Besonderheiten der jeweiligen Aktivität im Web Engineering aufgezeigt.
- Der Kern jedes Kapitels umfasst aktuelle Konzepte, Methoden, Techniken und Werkzeuge des Web Engineering.
- Den Abschluss bildet ein Ausblick auf zukünftige Entwicklungstendenzen.

Im Detail werden folgende Inhalte in den jeweiligen Kapiteln beschrieben:

- In **Kapitel 2: Requirements Engineering** wird von Grünbacher ausgeführt, dass Requirements Engineering (RE) für Web-Anwendungen vor besonderen Herausforderungen steht. Dazu zählen nicht verfügbare Stakeholder, dynamische Randbedingungen, schwer prognostizierbare Einsatzumgebungen, die besondere Bedeutung von Qualitätsaspekten und die oft fehlende Erfahrung mit Technologien. Beim Einsatz existierender Methoden des RE im Web Engineering sind daher wichtige Prinzipien zu beachten, wie die konsequente Einbeziehung wichtiger Stakeholder, die iterative Ermittlung der Anforderungen sowie eine konsequente Ausrichtung des Vorgehens an den Projektrisiken.
- Schwinger und Koch beschreiben in **Kapitel 3: Modellierung** die modellbasierte Entwicklung von Web-Anwendungen, wobei aufgrund der existierenden Ansätze eine Content- und Hypertext-zentrierte Behandlung im Vordergrund steht. Der Einbeziehung von Kontextinformation sowie der daraus abzuleitenden Anpassung wird zunehmend während der Modellierung Beachtung geschenkt, eine Folge der vorherrschenden ubiquitären Web-Anwendungen. Das Spektrum existierender Methoden für die Modellierung von Web-Anwendungen und ihre Schwerpunkte werden aufgezeigt sowie drei aktuelle Modellierungswerkzeuge vorgestellt, um dem Leser Hilfestellung bei der Auswahl einer geeigneten Modellierungsmethode zu geben.

- In **Kapitel 4: Architektur** argumentiert Eichinger, dass die Qualität einer Web-Anwendung maßgeblich von der ihr zugrunde liegenden Architektur bestimmt wird. Schlechte Performanz, unzureichende Wartbarkeit und Erweiterbarkeit sowie geringe Verfügbarkeit lassen sich häufig auf eine unzureichende Architektur zurückführen. Der Einsatz mehrschichtiger, flexibler Architekturen, die Berücksichtigung multimedialer Inhalte sowie die Integration existierender Datenbestände und Anwendungen stellen die Herausforderungen in der Entwicklung erfolgreicher Architekturen für Web-Anwendungen dar.
- Während Kapitel 3 die »modellgetriebene« Top-down-Entwicklung von Web-Anwendungen behandelt, beschreiben Austaller, Lauff, Lyardet und Mühlhäuser als Alternative dazu in **Kapitel 5: Technologiebewusstes Design** eine »technologiegetriebene« Bottom-up-Entwicklung von Web-Anwendungen. Die drei identifizierten Wurzeln, Hypertextdesign, Informationsdesign und objektorientiertes Softwaredesign, können für sich genommen keine befriedigende Lösung bieten, daher sind Ansätze für Web-spezifisches Design gefragt. Entwickler von Web-Anwendungen sollten diese in drei logische Schichten gliedern, die jeweils nach demselben Prinzip, nämlich Knoten und Anordnung in ein Geflecht, zweigeteilt sind. Folgende drei Ebenen sind dabei zu unterscheiden: 1. Präsentationsdesign, wo das *Look and Feel* bestimmt wird und multimodale Benutzeroberflächen ins Spiel kommen, 2. Interaktionsdesign, wo die Navigation durch Geflechte und der konkrete Dialog mit Komponenten entworfen werden, 3. Funktionales Design, das den »Kern« der Web-Anwendung festlegt. Bei zunehmender Konkretisierung des Entwurfs auf jeder Ebene und für beide Teile (Knoten, Geflecht) muss auf Werkzeuge des Hypertext-, Informations- und Softwaredesigns zurückgegriffen werden. Nach heutigem Stand steht für den integrativen Teil des Entwurfs noch keine befriedigende technische Unterstützung bereit.
- Gaedke, Nussbaumer, Jung und Dieckmann argumentieren in **Kapitel 6: Implementierungstechnologien**, dass es notwendig ist, die Charakteristika entsprechender Technologien zu kennen, um zu wissen, wann ihr Einsatz sinnvoll ist. Weiterhin erfordert die Implementierung von Web-Anwendungen oftmals nicht nur die Kenntnis einzelner Technologien, sondern vielmehr das Zusammenspiel unterschiedlicher Technologien innerhalb einer vorhandenen Architektur. Das Kapitel gibt einen Überblick über diverse Technologien und deren Zusammenspiel sowie die Verwendung innerhalb einiger ausgewählter Architekturbeispiele. Dabei stehen Empfehlungen des World Wide Web Consortium (W3C) im Vordergrund.
- **Kapitel 7: Testen** von Steindl, Ramler und Altmann führt aus, dass bestehende Testmethoden sich weitgehend auf den Test von funktionalen Anforderungen von Web-Anwendungen konzentrieren. Sie berücksichtigen aber eine breite Palette von nicht funktionalen Qualitätsanforderungen nur unzureichend, die für den Benutzer einer Web-Anwendung von Bedeutung sind, wie

Usability, Zuverlässigkeit und Sicherheit. Das vorgestellte Testschema berücksichtigt eine breite Palette von Qualitätsmerkmalen von Web-Anwendungen und fördert das Verständnis für ein systematisches, vollständiges und risikobewusstes Vorgehen beim Testen.

- In **Kapitel 8: Betrieb und Wartung** beschreiben Ebner und Werthner, dass unter dem Vorwand, eine Web-Anwendung stelle ohnedies einen evolutionären Prozess der laufenden Weiterentwicklung dar, häufig bereits während der Entwicklungsphase notwendige Aufgaben in die Betriebs- und Wartungsphase verlagert werden. Das stellt zwar keine Lösung des ursächlichen Problems dar, steigert jedoch die Verzahnung der beiden Phasen. Dies wird zusätzlich durch neue Aufgaben für Betrieb und Wartung von Web-Anwendungen verstärkt, die eine direkte Rückkopplung auf die (Weiter-)Entwicklung der Web-Anwendung haben. Dazu gehören die Bewerbung von Web-Anwendungen, die Content-Pflege und die Zugriffsanalyse.
- In **Kapitel 9: Web-Projektmanagement** führt Mayr in die Bedeutung der ganzheitlichen Betrachtungsweise der Herausforderungen an das Web-Projektmanagement ein. Projektmanagement ist eine Tätigkeit von Menschen zur Gestaltung des Handelns anderer Menschen. Diese Humanzentriertheit benötigt eine große Konfliktlösungskompetenz von Web-Projektleitern und interdisziplinäres Verständnis in Web-Teams. Das Vorgehensmodell bei der Entwicklung von Web-Anwendungen muss konsequenterweise sehr flexibel sein und eine stark iterativ-inkrementelle Entwicklung unter häufiger Einbeziehung des Auftraggebers zulassen. Werkzeuge und Techniken des Web-Projektmanagements sind daher besonders vom derzeitigen Übergang von klassischen Softwareentwicklungsmethoden hin zu agilen Vorgehensmethoden geprägt.
- **Kapitel 10: Entwicklungsprozess** von Engels, Lohman und Wagner diskutiert die Möglichkeit der Nutzung klassischer Softwareentwicklungsprozesse für die Entwicklung von Web-Anwendungen. Dazu werden sechs grundlegende Anforderungen an den Entwicklungsprozess von Web-Anwendungen formuliert. Diese Anforderungen werden zur Evaluation des Rational Unified Process (RUP) und von Extreme Programming (XP) verwendet. Es zeigt sich, dass keiner der Prozesse in der Lage ist, alle Anforderungen zu erfüllen. Die Stärken des RUP liegen in seiner Anpassbarkeit an den Grad der Komplexität der zu entwickelnden Anwendung. Die Stärken von XP dagegen liegen im Umgang mit kurzen Entwicklungszeiten und sich erst entwickelnden bzw. sich ändernden Anforderungen.
- Kapitel 11 (Usability), 12 (Performanz) und 13 (Sicherheit) beschreiben drei Qualitätsaspekte, die für Web-Anwendungen von besonderer Bedeutung sind. Dabei wird in **Kapitel 11: Usability** (Gebrauchstauglichkeit) von Hitz und Leitner ausgeführt, dass nicht gebrauchstaugliche Anwendungen speziell im Web zu Nutzungsverweigerung führen. Darüber hinaus stellen aktuelle Entwicklungen wie mobile Web-Anwendungen und die Berücksichtigung von

Benutzern mit Behinderungen immer härtere Anforderungen an die Usability. Das Kapitel zeigt, dass Usability nicht »auf einen Schlag« erreicht werden kann, sondern vielmehr über den gesamten Entwicklungsprozess hinweg berücksichtigt werden muss.

- In **Kapitel 12: Performanz** spannt Kotsis den Bogen von modellierenden Methoden zur Leistungsfeststellung hin zu messenden Methoden. Bei den Modellierungstechniken werden exemplarisch operationale Modelle, Warteschlangennetzwerke und allgemeine Simulationsmodelle vorgestellt. Messende Ansätze erfordern den Zugriff auf ein existierendes System und haben den Vorteil, dass das System auch unter realer Last beobachtet werden kann. Sind nun durch Messung oder Modellierung Performanzprobleme erkannt worden, so sind Maßnahmen zur Verbesserung der Performanz der nächste Schritt (Ausbau der Hardware, Software-Tuning und Caching bzw. Replikation). Neben dem traditionellen Analyse- und Verbesserungszyklus stellt Performance Management einen neuen Ansatz aus der Forschung dar, in dem versucht wird, messende, analytische und verbessernde Maßnahmen miteinander zu kombinieren und deren Zusammenspiel zu automatisieren.
- In **Kapitel 13: Sicherheit** führt Buchholz in die Sicherheitsprobleme existierender Web-Technologien ein. Angefangen mit CGI- und API-Skripte bis hin zu mobilen Agenten und Web-Services, beschäftigen sich Web-Anwendungen mit vielen Technologien, die entsprechend sicher genutzt werden sollen. Das Kapitel gibt einen Überblick über die verschiedenen Sicherheitslücken dieser Mechanismen und zeigt auch Möglichkeiten auf, um sich dagegen zu schützen. Ein spezielles Augenmerk wird auf die Sicherheit von Web-Services gelegt.
- In **Kapitel 14: Semantisches Web** wird von Behrendt das semantische Web als die logische Weiterentwicklung des Web vorgestellt. Dabei werden drei wesentliche Stützpfeiler ausgemacht. Erstens, von den Informationslieferanten, also jenen, die Web-Inhalte produzieren, müssen in Zukunft semantisch gekennzeichnete Webseiten kommen. Das heißt, jede Webseite trägt eine für Maschinen verständliche Beschreibung ihrer Inhalte mit sich. Zweitens jene, die Information suchen, müssen irgendeine Form von intelligenter Suchmaschine benützen, die solche semantisch gekennzeichneten Webseiten erfassen kann. Und drittens wird vorgeschlagen, autonome intelligente Softwareagenten für diese Aufgabe zu entwickeln. Sowohl die Produzenten von Web-Inhalten als auch die Softwareagenten müssen sich zu jeweils einer gemeinsamen Begriffswelt bekennen, genannt Ontologie. Insgesamt steckt das semantische Web zweifellos noch in den Kinderschuhen, aber bei Umfragen unter Forschern und Technologen aus dem industriellen Umfeld ist die Meinung vorherrschend, dass es sich hier um sehr zukunftsträchtige Technologien handelt, die in den nächsten 10 Jahren massiven Einfluss insbesondere auf die Arbeitswelt von »Wissensarbeitern« haben werden.