

Otto-von-Guericke-Universität Magdeburg



Fakultät für Informatik
Institut für Verteilte Systeme
Arbeitsgruppe Softwaretechnik

Diplomarbeit

Konzeption und prototypische Implementation
einer agentenbasierten Web-Qualitätssicherung

Verfasser:

Michael Weiß

29. November 2004

Betreuer:

Prof. Dr.-Ing. habil. Reiner R. Dumke

Dipl.-Inf. Cornelius Wille

Otto-von-Guericke-Universität Magdeburg

Fakultät für Informatik

Institut für Verteilte Systeme

AG Softwaretechnik

Weiß, Michael:

*Konzeption und prototypische Implementation
einer agentenbasierten Web-Qualitätssicherung*

Diplomarbeit,

Otto-von-Guericke-Universität Magdeburg 2004

Kurzfassung

Im Internet ist innerhalb weniger Jahre eine schier unendliche Zahl an Webseiten geschaffen worden. Viele dieser Webseiten erfüllen mehr als nur Informationszwecke und sind ein wichtiges Standbein im Leben eines Unternehmens oder einer Organisation. Deshalb ist es nachzuvollziehen, dass die Verantwortlichen für Firmenauftritte im Internet eine Möglichkeit finden wollen, die Qualität ihrer Webseite zu bewerten.

Diese Diplomarbeit beschäftigt sich mit der Entwicklung einer Web-Qualitätssicherung, welche es verschiedenen Nutzergruppen ermöglichen soll, Webseiten miteinander zu vergleichen und diese qualitativ zu bewerten. Hierzu werden Qualitätsaspekte aus unterschiedlichen Interessenkreisen beleuchtet. Es ergibt sich die Notwendigkeit, ein integriertes System zu benutzen, welches flexibel an die Anforderungen des jeweiligen Nutzers angepasst werden kann und eine komfortable Informationsgewinnung von Messwerten der Webseiten zulässt.

Der Aufgabenschwerpunkt dieser Diplomarbeit liegt in der Herleitung sinnvoller Qualitätsmerkmale sowie deren empirischer Bewertung anhand beispielhafter, als „gut“ oder „schlecht“ titulierter, Webseiten.

Als Implementationsbasis wird das in der Diplomarbeit von Uwe Schäfer [37] entwickelte Agentensystem *WebTomix* erweitert, um das Messen und Bewerten von Qualitätsmerkmalen zu ermöglichen. Die Modellierung der Sachverhalte erfolgt unter Verwendung der Modellierungssprache *Unified Modeling Language* (UML). Hierbei wird die Struktur des Programms erklärt, ohne ins Detail der Implementation zu gehen.

In einer ersten Anwendung des Messprogramms zeigen beispielhaft gewählte Webseiten die Funktionsweise und den Umfang der Software und geben so einen Überblick über die Exaktheit der Web-Qualitätssicherung.

Schlüsselwörter: Web-Qualitätssicherung, Messen, Webmetriken, Webmaße, Qualitätsmaße, Qualität, WWW, Internet, Webseiten, WebTomix, Web Measurement Suite, Agentensystem, Softwareagenten, Agenten

Abstract

In the last few years the number of web pages has increased dramatically. Many of them do not only serve informational purposes, but are a main pillar of companies and organizations. Thus it can easily be conceived that Chief Information Officers want to find possibilities to evaluate the quality of their web pages.

This diploma thesis deals with the development of a Web Quality Assurance System which enables various user groups to compare web pages and to evaluate their quality. Therefore aspects of web quality are examined from the point of view of different interest groups. It is necessary to use an integrated system which can be customized for the requirements of respective users and allows for a comfortable retrieval of information from web pages.

The main focus of this diploma thesis is the derivation of reasonable criteria of quality and their empirical evaluation using exemplary - high and low quality - web pages.

The implementation is based on the agent system *WebTomix* introduced in the diploma thesis of Uwe Schäfer [37]. In order to allow for the measurement and evaluation of quality metrics, an adaptation of the agent system is carried out. The modeling of the software is accomplished using *Unified Modeling Language* (UML). The structure of the program is explained without a detailed description of the implementation.

In a first application of the measurement software exemplary web pages demonstrate the functionality and size of the Web Quality Assurance System. This way an overview of the system's accuracy is given.

Keywords: Web Quality Assurance, Measurement, Measures, Metrics, Web Metrics, Quality Metrics, Quality, WWW, Internet, Websites, WebTomix, Web Measurement Suite, Agent System, Software Agents, Agents

Selbständigkeitserklärung

Ich versichere, dass ich die Diplomarbeit selbständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Magdeburg, 29. November 2004

Inhaltsverzeichnis

1	Einleitung	1
2	Technologische Grundlagen des World Wide Web	3
2.1	TCP/IP - Übertragung von Webdokumenten	4
2.1.1	IP	4
2.1.2	TCP	5
2.2	Webserver - Bereitstellung von Webdokumenten	6
2.3	Identifizierung - Benennung von Webdokumenten	7
2.3.1	URI	7
2.3.2	URL	9
2.4	HTTP - Abruf von Webdokumenten	12
2.4.1	HTTP-Requests	13
2.4.2	HTTP Response-Codes	16
2.4.3	HTTPS - Hypertext Transfer Protocol Secure	17
2.5	HTML & CSS - Formatierung von Webdokumenten	18
2.5.1	HTML	18
2.5.2	CSS	19
2.6	Webbrowser - Anzeigen von Webdokumenten	19
2.6.1	Browserkrieg	20
2.6.2	Internet Standards und Sicherheit	20
2.7	Skript & Programm - Dynamik in Webdokumenten	21
2.7.1	Clientseitige Skriptsprachen	21
2.7.2	Serverseitige Schnittstellen	21
3	Qualität und Messtheorie	23
3.1	Terminologie und Begriffsdefinitionen	24

3.2	Messtheorie	26
3.2.1	Maße und Metriken	26
3.2.2	Forderung an Maße	26
3.2.3	Maßskalen	27
3.2.4	Definition der Ziele von Maßen	29
3.2.5	Empirische Bedeutung	29
3.3	Bewertung von Softwareprodukten mit ISO 9126	30
4	Web Measurement	31
4.1	Arten des Web Measurement	32
4.1.1	Messung der Nutzerakzeptanz	33
4.1.2	Messung wirtschaftlicher Aspekte	36
4.1.3	Messung performance-technischer Aspekte	37
4.1.4	Messung des Wartungsaufwands	40
4.1.5	Messung der Webseiten-Eigenschaften	42
4.1.6	Messung der Standardkonformität	44
4.2	Probleme von Messprogrammen im WWW	46
4.2.1	Variabilität des WWW	46
4.2.2	Spoofing	47
4.2.3	Reloadsperre	48
4.2.4	Differenzierung von Webressourcen	48
4.2.5	Statische Seiten versus Content-Management-Systeme	49
4.3	Zusammenfassung	49
5	Agentensysteme	53
5.1	Grundlagen von Software-Agenten	53
5.2	Multiagentensysteme	54
5.3	Objektorientierung des Agentensystems	56
5.4	Der Web-Tomograph WebTomix	58
5.4.1	Arten von Technologien und ihre Merkmale	59
5.4.2	Erkennungskette	60
5.4.3	Das Agentensystem von WebTomix	61
5.4.4	Die Benutzerschnittstelle von WebTomix	62
5.4.5	Anwendungsbeispiele von WebTomix	63
5.5	Das Agentensystem der Web-Qualitätssicherung	64
5.5.1	Generische Rollen des Agentensystems	64
5.5.2	Klassifikation des Agentensystems	68
5.6	Zusammenfassung	69

6	Entwicklung einer Web-Qualitätssicherung	71
6.1	Herleitung der Messwerte und Qualitätsmaße	72
6.2	Spezifikation & Entwurf der Web-Qualitätssicherung	75
6.2.1	Verwaltung von Messprojekten	76
6.2.2	Erzeugung von Messreihen	78
6.2.3	Aufbau des Suchbaumes	81
6.2.4	Der Messvorgang	85
6.2.5	Aufnahme der Messwerte und Qualitätsmaße	89
6.2.6	Auswertung durch Report-Erzeugung	90
6.2.7	Webseitenüberwachung mit dem Monitoring Service	95
6.2.8	Datenhaltung	97
6.3	Agenten und Nutzer im System	101
6.4	Implementationsbeschreibung	106
6.4.1	ICEBrowser SDK	107
6.4.2	JavaHelp	108
6.4.3	JBYTE - JavaBY Template Engine	108
6.4.4	JavaMail	109
6.4.5	Software-Struktur der Web Measurement Suite	110
7	Beispielhafte Anwendung der Web-Qualitätssicherung	111
8	Zusammenfassung und Ausblick	119
A	Abkürzungen	121
B	Messwerte und Qualitätsmaße	125
B.1	Messwerte	125
B.2	Qualitätsmaße	132
C	Werbe-Pattern in URLs	145
D	Endnutzerdokumentation der Web Measurement Suite	147
D.1	Inhalt der CD	147
D.2	Systemvoraussetzungen	148
D.3	Installation	148
D.4	Nach der Installation	149
D.5	Bedienung	150
D.6	Die Web Measurement Suite als Verteiltes System	152
	Literaturverzeichnis	155

Abbildungsverzeichnis

2.1	TCP-Handshake	5
2.2	HTTP-Verbindungsarten	13
2.3	Ausführung eines serverseitigen Skriptes	22
3.1	Empirie-basierte Mess- und Schätzebenen	29
3.2	Softwarequalität im Qualitätsmodell	30
3.3	Qualitätsmodell nach ISO/IEC 9126: 1991	30
4.1	Linkgraph mit PageRank-bewerteten Knoten	34
4.2	HTTP-Anfrage	38
4.3	Performance-Maße eines Webdienstes	39
4.4	Wartungstypen und deren Zeitverhältnis	40
4.5	Durchschnittliches Änderungsintervall von Webseiten	41
5.1	Eigenschaften von Software-Agenten	53
5.2	Komponenten eines universellen Software-Agenten	55
5.3	Lebenszyklus von Software-Agenten	55
5.4	Die Klassifikationsmatrix eines Agentensystems	56
5.5	Erkennungskette von Webtechnologien in WebTomix	60
5.6	Klassifikation des Agentensystems von WebTomix	61
5.7	Das Agentensystem von WebTomix	62
5.8	Die Benutzerschnittstelle von WebTomix	62
5.9	WebTomix: Ergebnis der Messung von www.uni-magdeburg.de	63
5.10	Generische Rollen im Messtool	64
5.11	Die Klassifikationsmatrix des Messtool-Agentensystems	68
6.1	Qualitätsmodell der Web Measurement Suite	73
6.2	Klasse ProjectData	76
6.3	Öffnen von Messprojekten als Sequenzdiagramm	77
6.4	Messprojekte	77

6.5	Verknüpfung zwischen ProjectData und ConfigurationData	79
6.6	Verwaltung von Messkonfigurationen als Sequenzdiagramm	80
6.7	Messreihen	80
6.8	Webstruktur als Beispiel für den Aufbau des Suchbaumes	81
6.9	Test auf fehlenden Slash am Ende der URL	84
6.10	Lebenszyklus von Konfigurationen eines Messprojektes	85
6.11	Webressourcen im Zusammenhang mit dem Messvorgang I	85
6.12	Lebenszyklus eines eine Webressource abbildenden Objektes	87
6.13	Webressourcen im Zusammenhang mit dem Messvorgang II	88
6.14	Speicherung von Messwerten	89
6.15	Allgemeine Bewertungsskalen von Qualitätsmaßen	91
6.16	Abstände der Grenzen	91
6.17	Report-Typen und Evaluation von Qualitätsmaßen	94
6.18	Verknüpfung zwischen Monitor-Objekten und deren Messkonfiguration	96
6.19	Datenstruktur des Meßtools	98
6.20	Klassendiagramme aus dem XML-Paket	100
6.21	Zugriff der Nutzer auf das System	102
6.22	Agentenstruktur der Messagenten	103
6.23	Kommunikation der Agenten über RMI	103
6.24	Auswertungskomponente	104
6.25	Der Monitoring Service im vollständigen Agentensystem	105
6.26	Komponenten des ICEBrowser SDK	107
6.27	Beispielcode für die Template-Engine JBYTE	109
6.28	Komponenten der Web Measurement Suite	110
C.1	Elemente einer URL für Werbe-Pattern	145
D.1	Web Measurement Suite: Laden eines Messprojektes	150
D.2	Web Measurement Suite: Einen Qualitätsreport anschauen	151
D.3	Web Measurement Suite: WebUni.de im integrierten Browser	152
D.4	Web Measurement Suite: lokaler HTMLParser auf Notebook	154
D.5	Web Measurement Suite: entfernter HTMLParser auf PC	154

Tabellenverzeichnis

2.1	TCP/IP-Modell	5
3.1	Skalentypen	28
4.1	Web-Qualitätseigenschaften nach ISO 9126	42
5.1	Vergleich von AOP und OOP	57
5.2	Einteilung der Webtechnologien	59
6.1	Präfixe von Messwerten und Qualitätsmaßen	72
6.2	Messbare Strukturmerkmale einer Webseite	74
6.3	Startbedingungen einer Messkonfiguration	78
6.4	Aufbau des Suchbaumes in Szenario 1	82
6.5	Aufbau des Suchbaumes in Szenario 2	82
7.1	Startbedingungen Messprojekt W3C	112
7.2	Startbedingungen Messprojekt cssZenGarden	113
7.3	Startbedingungen Messprojekt Wikipedia	114
7.4	Startbedingungen Messprojekt WebUni	115
7.5	Startbedingungen Messprojekt Bundesregierung	116
7.6	Startbedingungen Messprojekte von Universitäten	116
7.7	Messergebnisse von Universitäten	116
7.8	Startbedingungen Messprojekte von B2C-Seiten	117
7.9	Messergebnisse von B2C-Seiten	118
7.10	Zusammenfassung der Messergebnisse	118

1 Einleitung

Internet, E-Mail, Online, World Wide Web sind längst keine Modewörter mehr. Das Internet boomt. Die eigene E-Mail-Adresse und Webseite sind auf Visitenkarten fast schon wichtiger als Postadresse und Faxnummer. Im Januar 2003 hatten bereits 46 Prozent der deutschen Haushalte die Möglichkeit zu Hause ins Internet zu gehen [41] und diese Zahl wächst von Jahr zu Jahr.

Das große Interesse an der neuen Technologie ist den Unternehmen nicht entgangen. Schon seit einigen Jahren gehört die Internetadresse mit auf den Briefkopf. In den Fernsehwerbespots wird immer häufiger die Adresse der Firmen-Webseite abgebildet. Marketing-Profis bringen inzwischen bereits Kinowerbung heraus, die kurz und witzig ist, aber keinen Hinweis auf ihren Auftraggeber enthält. Einzig eine einprägsame, bis dato unbekannte Internetseite weckt das Interesse des Kinogängers. Besucht die Person zu Hause die Internetadresse, erwartet ihn eine unaufdringliche Webseite mit einem kleinen Spiel und natürlich Daten über das Produkt und den Auftraggeber der Werbung - ein Automobilhersteller, der seinen neuen Kleinwagen gekonnt für verspielte und jung gebliebene Erwachsene anpreist.

Zudem lassen sich viele Aufgaben per Internet einfacher und schneller erledigen. Es bieten sich völlig neue Möglichkeiten wie E-Commerce, Internetauktionen und Online-Banking - eine gewaltige Zeitersparnis für den Nutzer. Kommunikation und der Zugang zu Informationen aus aller Welt innerhalb von wenigen Augenblicken machen das Internet für seine Nutzer unentbehrlich.

Im Gegensatz zu Privatpersonen haben Unternehmen natürlich andere Erwartungen. Die Selbstdarstellung im Internet ist für jedes Unternehmen bereits heute Standard. Manche von ihnen nutzen das Internet als Absatzkanal. Global agierende Unternehmen wissen die nahezu unbegrenzten Möglichkeiten zu schätzen. Weltweite Vernetzung, Telekooperation, Informationsmanagement, Groupware, Intranet-Portale, Workflow und Optimierung der gesamten Geschäftsprozesse in allen Filialen verhelfen den Firmen zu Einsparungen und Verbesserungen ihrer Arbeit.

Durch die Einführung des Internet wurden vollkommen neue Betätigungsfelder und Berufe geschaffen. Eine Internetseite zu entwerfen ist einfach und die Vielfalt an Webseiten ist geradezu explodiert. Die bis dahin gesammelten Erfahrungen des Software Engineering wurden lange Zeit bei der Entwicklung von Webseiten ignoriert. Erst seit kurzer Zeit wird dieses Defizit aufgeholt.

Diese Diplomarbeit beschäftigt sich mit der Bestimmung von Qualität, Komplexität und Wartungsaufwand von Webseiten. Hierzu werden Bewertungsgrundlagen geschaffen und diese prototypisch in Form eines Messtools implementiert. Mit diesem Messtool ist es möglich, Eigenschaften von Webseiten automatisiert zu sammeln und zu bewerten. Dadurch können Webseiten unter Qualitätsaspekten miteinander verglichen und frühzeitig Engpässe und Defizite erkannt werden.

Kapitel 2 beginnt mit grundlegenden Fakten, die im Zusammenhang mit dem WWW und dem Begriff Qualität erklärt werden müssen. Den Begriff Qualität selbst definiert Kapitel 3 und stellt zusätzlich Fakten rund um diesen Begriff vor. Beide Kapitel bilden die Motivation, Webseiten messen zu wollen, damit Aussagen über sie getroffen werden können. Mit dem Ziel die Qualität zu verbessern, soll Fehlern und Schwächen vorgebeugt werden. Unterstützt wird die Motivation durch das unterschiedliche Verständnis von Qualität seitens verschiedener Nutzergruppen. Deshalb bietet Kapitel 4 eine Übersicht über Qualitätsauffassungen und findet erste Lösungen, wie die jeweiligen Qualitätsmerkmale gemessen werden können.

Ab Kapitel 5 wird der Leser an die Struktur der Web-Qualitätssicherung herangeführt. In diesem Kapitel werden zuerst einige grundlegende Fakten zu Software-Agenten genannt, bevor einzelne Aufgaben spezifischen Agentenrollen zugeteilt werden. Dieser Faden wird in Kapitel 6 aufgegriffen, in dem das implementierte System anhand seiner Anforderungen und Ziele strukturiert hergeleitet und spezifiziert wird. Eine kurze Einschätzung der erbrachten Leistung in dieser Diplomarbeit ist am Ende dieses Kapitels zu finden.

Das Kapitel 7 wendet das Messprogramm schließlich erstmalig an. Durch gezielte Wahl „guter“ Webseiten werden Anforderungen an Webseiten gebildet und Werte von Kriterien anhand empirischer Beobachtung justiert. Eine Zusammenfassung der Ergebnisse ist in Kapitel 8 zu finden, welches zugleich einen Ausblick bietet.

2 Technologische Grundlagen des World Wide Web

Das World Wide Web (kurz: WWW) ist ein architektonisches Rahmenwerk, welches das Internet nutzt, um auf beliebige, weltweit verteilte Webseiten zuzugreifen. Das WWW ist hierbei nur ein Teil des Internet. Neben dem WWW stellt das Internet unter anderem folgende Dienste bereit: E-Mail, FTP, IRC, Usenet, Telnet und SSH, Web Services sowie Peer-to-Peer-Systeme. Einen kurzen Überblick über die technischen Grundlagen des Internet und somit des WWW ist im Abschnitt 2.1 zu finden.

Webseiten bestehen neben Texten auch aus Bildern, Sound und Videos. Die Speicherung dieser Webdokumente übernehmen Webserver. Webserver sind ständig an das Internet angeschlossene Computer (siehe Abschnitt 2.2), die bei Anfragen eines Nutzers Webseiten via *Hypertext Transfer Protocol* (HTTP) ausliefern (siehe Abschnitt 2.4). Die Adressierung von Webseiten wird in Abschnitt 2.3 näher erläutert.

Webseiten sind in *Hypertext Markup Language* (HTML) geschrieben. HTML ist eine Auszeichnungssprache, welche über Steuerkommandos die Formatierung und Positionierung von Elementen innerhalb der Webseite bestimmt (siehe Abschnitt 2.5). Am wichtigsten sind jedoch Hyperlinks. Dies sind in der Webseite eingebettete Textpassagen oder Bilder, die Webseiten miteinander verknüpfen. Das WWW wird über Webbrowser (siehe Abschnitt 2.6) bedient.

In Abschnitt 2.7 wird erläutert, wie das WWW dynamisiert wird. Es wird auf server- und clientseitige Skriptsprachen eingegangen, die bis dahin statische Webseiten auf Nutzerantworten reagieren lassen. Im Vergleich mit konventionellen Programmen ist die Bedienbarkeit von Webseiten eingeschränkt. Dennoch lässt sich der Trend erkennen, dass immer mehr Internet-Dienste über das WWW angeboten werden. So kann man E-Mails über WebMail abrufen, Web-Foren ersetzen das Usenet und Web-Chats das IRC.

2.1 TCP/IP - Übertragung von Webdokumenten

Das Internet¹ basiert auf dem TCP/IP-Protokoll, welches einen Standard für die Adressierung und den Datenaustausch zwischen verschiedenen Computern und Netzwerken festlegt.

2.1.1 IP

Das *Internet Protocol* (IP) definiert die Adressierung des Computers und die Struktur beim Transport von Daten.

IP-Adresse

Während der Nutzungsdauer wird ein Computer im Internet eindeutig adressiert und identifiziert. Hierzu erhält der Computer eine IP-Adresse, welche in der Version IPv4 aus 32 Bits besteht. In der *dotted decimal notation* werden sie als 4 durch Punkte voneinander getrennte Dezimalzahlen von 0 bis 255 (je 8 Bit) dargestellt. Die Internetadresse **141.44.1.20** zeigt zum Beispiel auf einen Webserver der Otto-von-Guericke-Universität Magdeburg.

IPv4 wurde 1981 in RFC791 [34] zur Diskussion gestellt und die Spezifikation zum neuen IPv6 1995 in RFC1883 [11]. IPv6 wird zur Zeit eingeführt, um die latente Adressenknappheit zum Beispiel im IT-Wachstumsmarkt Asien zu beseitigen und auch in Zukunft genügend Spielraum zu haben [30]. Aus circa 4 Milliarden IPv4-Adressen werden $3,4 \cdot 10^{38}$ IPv6-Adressen. Rein rechnerisch könnten jedem Quadratmeter der Erdoberfläche $6,5 \cdot 10^{23}$ Adressen zugeteilt werden. Den Diskussionen über chronischen Adressenmangel ist somit vorerst ein Riegel vorgeschoben - zumindest bis die nächste größere technische Neuerung eingeführt wird, die heute noch nicht vorstellbar ist.

Struktur beim Transport der Daten

Die IP-Adresse wird nun verwendet, um Daten im weltweiten Netz diesem einen Computer zuzuordnen. Um Daten über das *Internet Protocol* zu versenden, werden sie in Pakete aufgeteilt und mit einem Header versehen, der unter anderem die Sende- und Ziel-IP-Adresse enthält. Die Datenpakete werden durch das Netzwerk geleitet (geroutet) und am Ziel-Rechner wieder zu einem Datenstrom zusammengesetzt.

¹Das Wort **Internet** entstand durch die Zusammenfassung von **Inter**connected **Net**works. Dies lässt sich dadurch erklären, dass das Internet ein hierarchisch übergeordnetes Computernetzwerk ist, welches verschiedene lokale Netzwerke miteinander verbindet.

2.1.2 TCP

Der andere Teil von TCP/IP ist das *Transmission Control Protocol* (TCP) [35]. TCP setzt (in den meisten Fällen) auf IP auf (siehe Tabelle 2.1).

Anwendung	FTP	SMTP	HTTP	DNS	...
Transport	TCP			UDP	
Netzwerk	IP (IPv4,IPv6)				
Netzzugang	Ethernet	Token Bus	Token Ring	FDDI	...

Tabelle 2.1: TCP/IP-Modell [7]

Bei einer TCP-Verbindung verbinden sich zwei entfernte, über IP-Adressen gekennzeichnete Computer miteinander und tauschen Daten bidirektional aus. Um gleichzeitig pro Netzanschluss mehrere Verbindungen zu ermöglichen, werden zusätzlich Ports benutzt. Ports sind 16-bit Zahlen und reichen von 1 bis 65535. Ein Endpunkt einer TCP-Verbindung wird also eindeutig von IP-Adresse und Port definiert. Eine TCP-Verbindung wird immer durch zwei Endpunkte definiert. Somit ist es möglich, dass ein Webserver mehrere Verbindungsanfragen annehmen und mehrere Nutzer gleichzeitig bedienen kann. Der Webserver hält dabei den Port 80 ständig offen und wartet auf eingehende Verbindungen. Dies wird als *passive open* bezeichnet, da keine vollständige TCP-Verbindung vorliegt.

Ein Internetnutzer baut die Verbindung zum Webserver über einen lokal offenen Port auf und stellt eine TCP-Verbindung zwischen der Webserver-IP mit dem Port 80 und seiner eigenen IP-Adresse mit dem gewählten Port her. Für den Verbindungsaufbau sind drei Pakete erforderlich (3-Wege-Handshake - siehe Bild 2.1), wobei sowohl Server als auch Client jeweils eine Anfrage des anderen bestätigen müssen. *Syn* markiert einen Synchronisationsvorgang, der mit *Ack* (Acknowledge) vom Kommunikationspartner bestätigt wird.

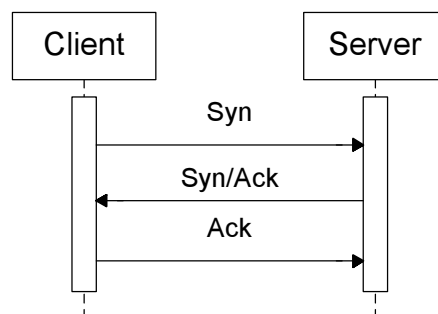


Abbildung 2.1: TCP-Handshake

Ist eine TCP-Verbindung hergestellt, wird diese *active open* genannt. Daten, die über TCP übertragen werden, beinhalten eine Prüfsumme und eine Sequenznummer im Header des Datenpakets. Hierdurch wird sichergestellt, dass der Empfänger alle Daten vom Sender empfangen hat und dass die Daten in der richtigen Reihenfolge zusammengesetzt werden. Der Sender verschickt die Pakete wiederholt, wenn keine Bestätigung innerhalb einer bestimmten Zeitspanne eintrifft.

Die Qualität einer Verbindung wird durch die Zeit für den Verbindungsaufbau bestimmt und durch die Geschwindigkeit während der Datenübertragungsphase. Hierbei spielt auch die Anzahl der Pakete eine Rolle, die während des Transports verloren gegangen sind und somit erneut gesendet werden müssen. Man spricht von einem *Lag*, wenn es zu einer vorübergehenden Verzögerung kommt und die Pakete unerwartet lange zur Übertragung zwischen den Teilnehmern der Verbindung benötigen.

2.2 Webserver - Bereitstellung von Webdokumenten

Ein Webserver ist ein Server-Programm, welches Dateien speichert und diese auf Anfrage an den Nutzer ausliefert. Der Webserver hält dafür ständig einen Port offen, zu dem sich die Internetnutzer verbinden. Die ersten 1024 Ports sind Standardports (*Well Known Port Numbers*) und werden von der Internet Assigned Numbers Authority (IANA) verwaltet (siehe [26]). Für Webserver wurde hierzu Port 80 als Standard festgelegt. Es sind natürlich auch beliebige andere frei zu definierende Ports möglich. Weitere oft von Webservern benutzte Ports sind 8080 und 8088. Diese Ports müssen dann bei der Anfrage an den Server speziell angegeben werden (siehe Abschnitt 2.3).

Es werden Webseiten, Stylesheets, Javascript-Dateien sowie Multimedia-Dateien (Bilder, Audio-Dateien, ...) vom Webserver bereitgestellt. Im Prinzip können alle Arten von Dateien von Webservern ausgeliefert werden; nur eignen sich andere Internetdienste (FTP, Peer-to-Peer-Systeme, ...) besser zum Transfer großer Datenmengen.

Welcher Webserver bei einer Webseite verwendet wird, kann nicht in jedem Fall herausgefunden werden. Der Administrator des Webserver kann entscheiden, ob die Kennung des Webserver-Programms bei einer Antwort mit zurückgesendet wird oder nicht.

Der Webserver ist grundsätzlich dafür verantwortlich, dass eine Webseite schnell und effizient ausgeliefert wird. Er muss gleichzeitig verschiedene Anfragen beantworten können. Deshalb laufen im Normalfall immer mehrere Subprozesse in einem Serverpool.

Die Anzahl der Subprozesse, die Timeout-Zeit und andere Einstellungen können in der Webserver-Konfigurationsdatei eingestellt werden. Eine optimierte Konfiguration eines Apache-Webserver ist in [38] auf Seite 41 zu finden. Ein schlecht konfigurierter Webserver kann die Qualität einer Webseite drastisch verringern und dadurch zu nicht oder schlecht erreichbaren Webseiten führen, die für seine Besucher praktisch uninteressant sind.

Die Anzahl der Besucher und der Zugriffe werden in Logfiles gespeichert. Durch Logfile-Analyse können unterschiedliche Statistiken generiert und somit Aussagen über die Akzeptanz und Nutzung der Webseite getroffen werden (siehe Abschnitt 4.1.1 auf Seite 33).

2.3 Identifier - Benennung von Webdokumenten

Das World Wide Web ist ein verteiltes Hypertext-System, welches hauptsächlich von zwei architektonischen Merkmalen bestimmt wird: den Informationen bzw. Ressourcen und den Hyperlinks, welche die Ressourcen miteinander verknüpfen und eine Navigation zwischen den Ressourcen ermöglichen. Ein Aspekt des WWW ist der Zugriff auf weltweit verteilte Informationen. Daher muss es möglich sein diese Ressourcen eindeutig zu referenzieren. Dies wird über *Identifier* bzw. Bezeichner gelöst [50, S.35].

2.3.1 URI

Identifier werden im Internet allgemein *Uniform Resource Identifier* (URI) genannt. URIs werden benutzt um Ressourcen im Internet zu lokalisieren und zu identifizieren. Allgemein sind die Bedingungen einer URI die Erweiterbarkeit, Vollständigkeit und Druckfähigkeit. Es muss gewährleistet werden, dass immer neue Arten von URI (*Schemata*) für die Identifikation von neuen Informationsressourcen hinzugefügt werden können (Erweiterbarkeit). Jedes neue Schema muss der allgemein definierten Struktur der URI genügen (Vollständigkeit) und die URI muss auch geschrieben oder gedruckt für den Menschen verständlich sein (Druckfähigkeit).

Die Erweiterbarkeit wird dadurch gewährleistet, dass die Definition einer URI in zwei Teile getrennt wurde - einen Teil für das Schema und einen Teil für die Identifikation, welche je nach Schema-Typ entschlüsselt wird.

Um vollständig zu sein, muss eine URI im zweiten Teil alle binären Zeichenfolgen maskieren und gültige Zeichen benutzen. Dadurch können theoretisch alle Zeichenfolgen dargestellt werden. Da alle Zeichen kodiert darstellbar sind, ist das Konzept der URIs langfristig für alle vorstellbaren Schemata anwendbar.

Indem man einen festen Zeichensatz definiert (z.B. ASCII) wird eine URI druckfähig. Das %-Zeichen wurde als *Escapezeichen* (engl.: Ausgleichs-Zeichen) gewählt. Die Umschreibung %20 maskiert zum Beispiel das Leerzeichen². Da das Prozent-Zeichen selber als Escapezeichen verwendet wird, muss bei Bedarf %25 statt % verwendet werden³.

Syntax einer URI

Die Syntax einer URI lautet:

```
scheme ":" scheme-specific-part
```

Das Schema (*scheme*) definiert die Art der Identifizierung. Die Bedeutung des *scheme-specific-part* hängt vom Schema ab und definiert einige allgemeine Festlegungen zur Syntax.

Viele URI-Schemata wie http oder ftp verwenden einen hierarchischen Aufbau. Als Trennzeichen von verschiedenen hierarchischen Zeichenketten wurde das Slash-Zeichen (/) definiert. Die Hierarchie wird von links nach rechts definiert. Je weiter links eine Zeichenkette ist, desto näher ist das durch die Zeichenkette definierte Objekt der Wurzel der Hierarchie. Ein Slash-Zeichen in der URI, welches keine Hierarchie definiert, muss mit %2F umschrieben werden.

Das Doppelkreuz (#) definiert in

```
scheme ":" scheme-specific-part "#" fragment
```

ein Fragment. Die Referenz vor dem Doppelkreuz zeigt auf ein Objekt und das Fragment zeigt auf einen Punkt innerhalb des Objektes. Das Escapezeichen für das Doppelkreuz ist %23.

Das Fragezeichen übergibt in

```
scheme ":" scheme-specific-part "?" query
```

eine Anfrage an das Objekt. Das Objekt kann auf die übergebenen Argumente reagieren und die Ausgabe dementsprechend anpassen. Dies wird in Abschnitt 2.7 verwendet. Die Umschreibung für das Fragezeichen ist %3F. Um Leerzeichen innerhalb des Query

²Während URIs bis 2003 nur aus ASCII-Zeichen bestehen durften, ist mittlerweile die Erweiterung *Internationalized Resource Identifiers* (IRIs) eingeführt, welche Sonderzeichen in URIs erlaubt. Näheres hierzu unter [12] sowie RFC3490, RFC3491 und RFC3492.

³Das %-Zeichen belegt hexadezimal den Wert 25 im ASCII Zeichensatz.

darzustellen werden Pluszeichen verwendet. Die Umschreibung für das Pluszeichen ist `%2B`.

Wenn eine URI komplett ausgeschrieben wird, nennt man dies absolute Referenz. Wurden Teile der URI weggelassen, spricht man von relativer Referenzierung. In diesem Fall werden die weggelassenen Teile von der referenzierenden URI geerbt [50, S.42-44].

Die in Verbindung mit dem WWW am häufigsten verwendeten Schemata sind `http`, `https`, `ftp` und `mailto`. Das IANA stellt unter [25] eine vollständige Liste aller offiziell definierten Schemata mit den definierenden RFC-Dokumenten zur Verfügung. Freie Schemata zum Beispiel im Peer-to-Peer-Umfeld sind *ed2k* und *torrent*.

Semantik einer URI

Semantisch werden URIs im Normalfall in zwei Unterarten eingeteilt. Die *Uniform Resource Locators* (URLs) identifizieren eine Ressource über ihren Ort. Die *Uniform Resource Names* (URNs) mit dem URI-Schema `urn` identifizieren eine Ressource ohne sie zu lokalisieren (zum Beispiel `urn:isbn`). Da eine Ressource per URL nur über den Ort identifiziert wird, ist es möglich, dass ein und dieselbe Ressource verschiedene URLs haben kann und dass diese URLs grundsätzlich als verschiedene Ressourcen aufgefasst werden⁴. Ursprünglich sollte jede URI einer der beiden Unterarten zugeordnet werden, was aber aufgegeben wurde. Es gab Schemata, die einfach in keine der beiden Klassen passten. Einige Schemata (wie `mailto`) sind heute keiner der beiden Unterarten mehr zugeteilt, obwohl sie früher als URL behandelt wurden.

In RFC2396 [4] ist die momentane Definition von URIs zu finden und in [5] wird diese derzeit überarbeitet.

2.3.2 URL

Im WWW werden ausschließlich URLs als Identifikation von Ressourcen verwendet. Das URI-Schema einer URL wird zumeist von dem verwendeten Internet-Protokoll abgeleitet. Ressourcen, die über HTTP abgerufen werden (siehe Abschnitt 2.4), haben das Schema `http`. Sichere Seiten werden über `https` abgerufen. Zugriff auf FTP-Server erhält man über das Schema `ftp`.

⁴Es gibt Ansätze, die einer Ressource einen eindeutigen Namen zuordnen, der sich auch durch Wechsel der Lokalität nicht ändert. Beispiele sind *Digital Object Identifier* (DOI) durch Voranstellen des Proxyservers `http://dx.doi.org/` und *Persistent Uniform Locator* (PURL) als Verzeichnisdienst (`http://purl.oclc.org/`).

Syntax des Schema-spezifischen Teils einer URL

Der folgende spezifische Teil des URI-Schemas gilt allgemein für alle URLs. Nicht alle Elemente werden für alle URL-Schemata verwendet [50, S.47,48].

```
"//" [user [ ":" password ] "@" ] host [ ":" port ] "/" url-path
```

Nutzer und Passwort

Der Teil `user` definiert den Nutzer, mit dem man sich zusammen mit dem `password` zum Beispiel an einem FTP-Server anmeldet. Dieser Teil ist optional, was durch den Einschluss in eckigen Klammern dargestellt wird.

Host

Der Host einer URL kann als Domain-Name oder als IP-Adresse angegeben werden. Domain-Namen müssen den Internet Standards RFC1034 und RFC1123 genügen und werden durch das *Domain Name System* (DNS) oder das *Open Root Server Network* (ORSN) - der europäischen Alternative zu DNS (<http://european.orsn.net/>) - aufgelöst. Die Konzepte zu DNS sind in RFC1034 und Details zur Umsetzung in RFC1035 zu finden.

Ein Domain-Name ist zum Beispiel `www.uni-magdeburg.de`. Im Prinzip ist ein Domain-Name nur ein mnemonischer Ausdruck für eine IP-Adresse und zeigt eindeutig auf diese. Ein Domain-Name enthält Subdomains. Eine Subdomain ist eine Domain, welche in der Hierarchie unterhalb einer anderen Domain liegt.

An oberster Stelle in der Hierarchie ist die *Top Level Domain* (TLD). Die TLDs werden von der *Internet Corporation for Assigned Names and Numbers* (ICANN) verwaltet und werden in allgemeine TLDs (generic TLD; gTLD) und länderspezifische TLDs (country-code TLD; ccTLD) unterteilt. Die in dem Beispiel verwendete ccTLD `de` ist hierbei nach ISO3166 [22] Deutschland zugeteilt und nach der gTLD `com` die zweithäufigste TLD aller registrierten Domains⁵.

In der Hierarchie unter der TLD ist die (im allgemeinen Sprachgebrauch eigentlich Domain genannte) Subdomain `uni-magdeburg`. Domains (bzw. Subdomains erster Ordnung) werden von den jeweiligen Registraren an die Domain-Inhaber vermietet. In dem Beispiel mietet die Universität Magdeburg von dem Registrar DENIC (<http://www.denic.de/>) die Domain `uni-magdeburg` unter der TLD `de`.

⁵http://www.denic.de/de/domains/statistiken/domainvergleich_tlds/

Da Subdomains erster Ordnung normalerweise nur Domains genannt werden, spricht man von Subdomains erst ab der zweiten Ordnung. In dem oben genannten Beispiel ist das `www` eine Subdomain unter `uni-magdeburg.de`.

Port

Der Port im URL-Schema ist ebenso optional, da die meisten Internetdienste eine Standard-Portnummer besitzen, mit der die TCP-Verbindung zum Server aufgebaut wird. Bei Webservern ist dies standardmäßig der Port 80. Laufen aber gleichzeitig zwei Webserver (zum Beispiel ein Standard-Webserver und ein experimentieller Webserver zum Testen), so könnte der zweite Server nicht auch unter Port 80 laufen.

Manche Webserver - wie zum Beispiel der Tomcat (<http://jakarta.apache.org/tomcat/>) - haben, wenn sie Stand-alone laufen, standardmäßig den Port 8080 eingestellt. Application-Server (z.B. J2EE-Server) haben gleichzeitig einen offiziell zugänglichen Webserver für Deployments unter Port 80 laufen und belegen außerdem einen zweiten Port zur Administration der Applikationen.

URL-Pfad

Der Rest der URL gibt an, wo die Ressource auf dem Host gefunden werden kann. Die genaue physische Struktur hängt von der Serverkonfiguration ab. Normalerweise muss die Eingangs-Datei `index.html` nicht mit angegeben werden - kann also weggelassen werden. Es ist aber auch möglich, dass die Pfade in dem Beispiel

```
http://www.server.com/path/query/arg1/value/arg2/value
```

Argumente für das Serverskript `query` sind, welches im physischen Verzeichnis `path` liegt.

Beispiel

Ein vollständiges Beispiel aller syntaktischen Möglichkeiten einer URL ist:

```
http://www.server.no-ip.com:8080/~user/path/file.jsp?arg1=value
&arg2=value1+value2#anker
```

Wenn diese URL in den Browser eingegeben wird, versucht der Webbrowser den Host `www.server.no-ip.com` per DNS aufzulösen. Die IP des DNS-Rootservers, der die TLD `com` bedient, ist bekannt. Dort wird nach `no-ip` nachgeschaut, von wo aus man Schritt für Schritt bis zum Ende durchsucht.

Ist die IP bekannt, wird eine TCP-Verbindung mit dem Webserver auf Port 8080 aufgebaut. Dieser Webserver sucht im Nutzerverzeichnis des Nutzers `user` (auf Unix-Systemen physisch im Normalfall `/home/user/public_html/`) im Verzeichnis `path` nach der Datei `file.jsp`. Dies wäre dann `/home/user/public_html/path/file.jsp`. Letztendlich ruft er diese Datei auf und übergibt 2 Argumente - `arg1` mit dem Wert "value" und `arg2` mit dem Wert "value1 value2". Innerhalb des zurückgelieferten Dokumentes wird zum Markierungspunkt `anker` gesprungen.

2.4 HTTP - Abruf von Webdokumenten

Das Standardtransferprotokoll im WWW ist das *Hypertext Transfer Protocol* (HTTP), welches momentan in der Version HTTP/1.1 definiert ist. HTTP verwendet ein *Request-Response Model*. Eine HTTP-Transaktion folgt dabei immer demselben Schema. Der Nutzer baut eine Verbindung mit dem Server auf und stellt eine Anfrage (*Request*), worauf der Server antwortet (*Response*).

Die Verbindung wird meistens via TCP/IP aufgebaut, was aber vom HTTP-Standard nicht formell gefordert wird. Sollten beispielsweise Netze mit *Asynchronous Transfer Mode* (ATM)⁶ zuverlässig genug sein, dann könnte HTTP problemlos mit diesen Netzen verwendet werden [45, S.726].

Die Verbindung selbst ist zustandslos und nur kurzzeitig. Zustandslos bedeutet, dass HTTP zwischen zwei Anfragen nicht unterscheidet und sich den momentanen Zustand einer Verbindung nicht merkt⁷. Kurzzeitig ist eine HTTP-Verbindung, da ein Server zumeist nach dem Senden der Antwort die Verbindung wieder trennt. Der Vorteil von kurzzeitigen Verbindungen ist, dass der Server jederzeit genügend freie Verbindungen für Nutzeranfragen hat und es werden keine sinnlosen Verbindungen zu „toten“ Nutzern gehalten. Nachteilig an dieser Methode ist der Aufwand um eine medienstarke Webseite zu empfangen. Bei einer kurzzeitigen Verbindung wird für jedes Bild eine neue Verbindung zum Webserver aufgebaut. Deshalb wurden in HTTP/1.1 permanente Verbindungen mit und ohne Pipelining eingeführt (siehe Bild 2.2).

⁶ *Asynchronous Transfer Mode* (ATM) hat im Gegensatz zu TCP/IP den Vorteil, dass die Größe der Datenpakete nicht variabel ist und damit eine effizientere Weiterleitung in großen Netzen (z.B. dem Internet) gewährleistet ist. Weitere Informationen zu ATM gibt es unter <http://www.atmforum.com/>

⁷ Ein Webserver kann sich Zustände mit Hilfe von Cookies oder Sessions merken. Bei Cookies werden die Daten beim Client gespeichert und bei jeder Anfrage in den Anfrageheader des Requests geschrieben. Sessions speichern die Daten auf dem Server und einzig eine Session-ID wird beim Client gespeichert.

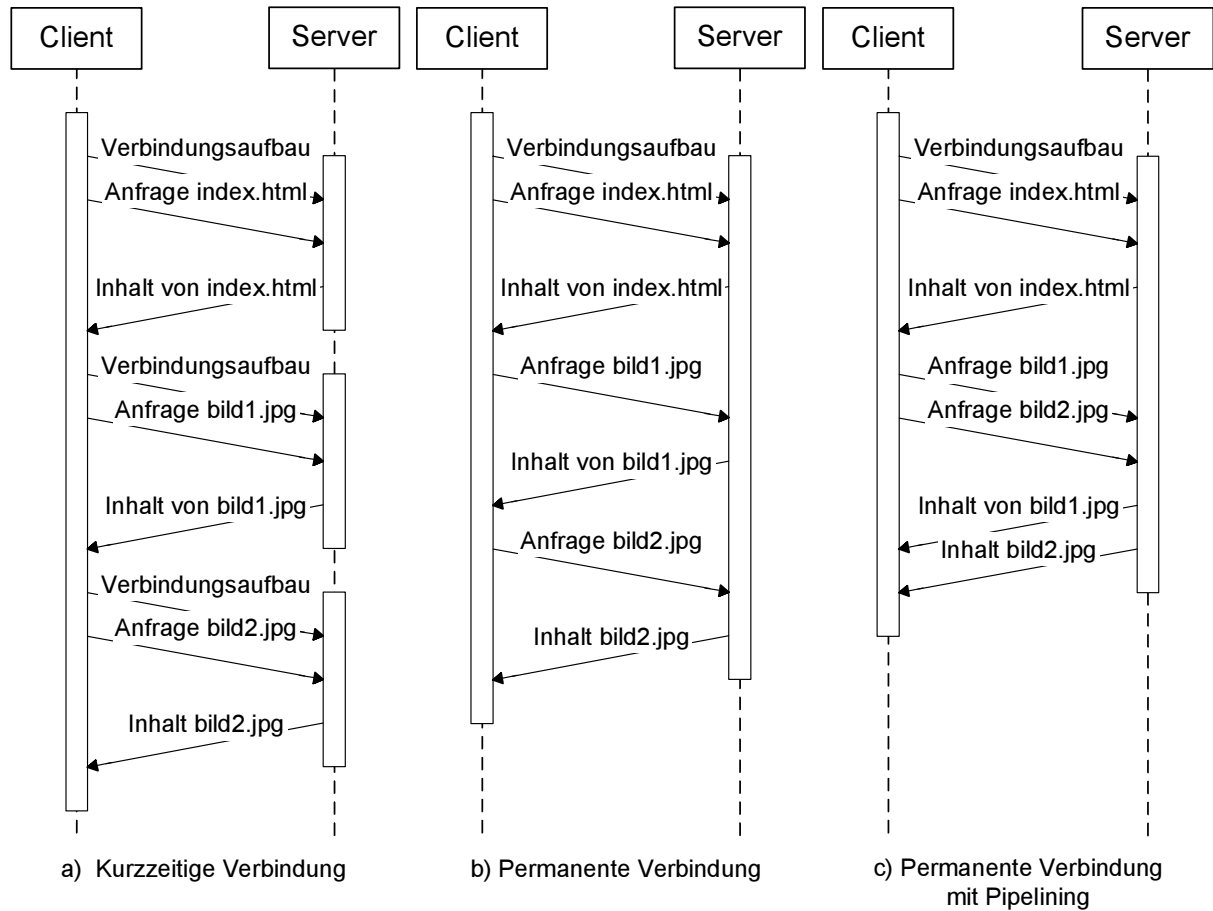


Abbildung 2.2: HTTP-Verbindungsarten

Ist die Verbindung zum Server aufgebaut, wird über HTTP die Anfrage gestellt. Die Anfrage besteht dabei aus alphanumerischen Zeichen sowie einigen Sonderzeichen. Der Server kann daraufhin eine Antwort senden. Die Antwort könnte dann zum Beispiel die angeforderte Webseite enthalten oder aber eine Fehlermeldung. Bricht ein Datentransfer ab, bevor alles gesendet wurde, muss die Verbindung erneut geöffnet und alles von vorn angefordert werden.

2.4.1 HTTP-Requests

Da HTTP nicht nur für das WWW spezifiziert wurde, unterstützt es für die verschiedenen Anwendungen auch verschiedene Request-Methoden. Im Folgenden sind vor allem die Methoden näher erklärt, die für Webseiten wichtig sind.

GET-Methode

Im WWW ist GET die wohl wichtigste und elementarste Methode im HTTP-Repertoire. GET existiert seit HTTP/0.9⁸. Damit ein Nutzer zum Beispiel die Webseite

```
http://www.server.com/index.htm
```

abrufen kann, verbindet er sich mit dem Host `www.server.com` und sendet die Anfrage:

```
GET /index.htm HTTP/1.1
Host: www.server.com
Connection: close
```

Eine mögliche Antwort könnte zum Beispiel sein:

```
HTTP/1.1 200 OK
Server: Apache/2.0.50 (Debian GNU/Linux) PHP/4.3.6
Content-Length: (Größe von index.htm in Byte)
Content-Language: de
Content-Type: text/html
Connection: close
```

```
(Inhalt von index.htm)
```

Der Inhalt der Datei `index.htm` muss nicht zwingend auf dem Server als Datei abgelegt sein. Die zurückgelieferten Informationen können auch dynamisch generiert worden sein, wie es bei serverseitigen Skriptsprachen der Fall ist.

HEAD-Methode

Ähnlich der GET-Methode lautet die Anfrage der HEAD-Methode. HEAD ist seit der Version HTTP/1.0 verfügbar. Eine Anfrage mittels HEAD hat dieselben Anfrageargumente wie GET und liefert dieselbe Antwort abzüglich des Inhalts der angeforderten Datei.

HEAD kann also verwendet werden, um Informationen zu sammeln ohne jedes Mal die eigentliche Datei abzurufen. So kann zum Beispiel im Vorfeld die Dateigröße (*Content-Length*) der Ressource in Erfahrung gebracht werden, bevor die Datei vom Server abgerufen wird.

⁸Genauer gesagt war GET die einzige Methode in HTTP/0.9.

POST-Methode

Seit HTTP/1.0 kann die POST-Methode verwendet werden, um Formulardaten an den Webserver zu senden. Eine mögliche Anfrage wäre zum Beispiel:

```
POST /index.jsp HTTP/1.1
Host: www.server.com
Content-Length: 39
Connection: close

user=Name&code=qwerty&mail=a%40bc%2Ede
```

An das Serverskript `http://www.server.com/index.jsp` werden die Argumente `user` (*Name*), `code` (*qwerty*) und `mail` (*a@bc.de*) übermittelt - in den Klammern jeweils die übergebenen Werte.

Die Antwort einer POST-Anfrage enthält die angeforderte Datei.

Weitere Methoden

Einige weitere Methoden sind:

CONNECT wird beim *SSL Proxying* angewendet, um eine sichere Direktverbindung zwischen Client und Server aufzubauen, die über Proxy getunnelt wird (HTTP/1.1).

DELETE bittet den Webserver eine Datei zu löschen (HTTP/1.1). Die Methode ist bei den meisten Webservern deaktiviert.

OPTIONS fordert Methoden und Voraussetzungen der Kommunikation sowie Fähigkeiten des Servers an (HTTP/1.1).

PUT sendet einen Datenstrom zum Server und bittet ihn diese Daten in eine Datei zu speichern (HTTP/1.1). Die Methode ist bei den meisten Webservern deaktiviert.

LINK erstellt eine semantische Verbindung zwischen zwei Dokumenten (HTTP/1.0). Der Server ist angewiesen bei Anfragen an das Dokument weitere Header-Informationen auszugeben.

UNLINK löst eine semantische Verbindung zwischen zwei Dokumenten (HTTP/1.0).

TRACE verfolgt den Pfad, den eine Anfrage durch ein Netzwerk wählt (HTTP/1.1).

2.4.2 HTTP Response-Codes

HTTP Response-Codes werden vom Server zurückgeliefert und geben an, ob die Anfrage erfolgreich oder nicht erfolgreich war. Die Response-Codes sind dreistellige Zahlen zwischen 100 und 599. Sie sind in bestimmte Bereiche gegliedert.

Response-Codes 100-199: Informelle Codes

Dieser Bereich ist rein informeller Natur. Zum Beispiel besagt der Response-Code 100, dass ein Teil einer mehrteiligen Nachricht empfangen wurde und Response-Code 101, dass das aktuelle Protokoll erfolgreich gewechselt wurde.

Response-Codes 200-299: Request erfolgreich

Dieser Bereich teilt dem Client mit, dass seine Anfrage erfolgreich war. Wichtige Response-Codes sind:

- 200 OK** Der Request war erfolgreich und die angeforderten Daten wurden gesendet.
- 204 No Content** Der Request war erfolgreich, es konnten aber keine Daten zurückgesendet werden.
- 205 Reset Content** Der Request wurde ausgeführt und der Client soll die Daten des Requests zurücksetzen (meist Formulardaten).

Response-Codes 300-399: Weiterleitung

Dieser Bereich teilt dem Client mit, dass eine Umleitung zu einer anderen URL erfolgen muss. Wichtige Response-Codes sind:

- 301 Moved Permanently** Die angeforderte URL existiert nicht mehr, die Ressource wurde dauerhaft verschoben. Die neue Position wird mitgeliefert.
- 304 Not Modified** Wenn der Client noch eine alte Version der angeforderten Ressource im Cache hat und sich diese Version seit dem letzten Request nicht geändert hat, wird 304 zurückgesendet.
- 307 Temporary Redirect** Die URL wurde zeitweilig verschoben. Die neue URL wird mitgeliefert. In Zukunft soll aber weiterhin die alte URL verwendet werden.

Response-Codes 400-499: Client-Fehler

Dieser Bereich informiert den Client, dass er einen Fehler verursacht hat. Wichtige Response-Codes sind:

- 400 Bad Request** Die Anfrage enthält einen Syntaxfehler.
- 401 Unauthorized** Eine Authentifizierung ist erforderlich. Ein WWW-Authenticate Header wird mit zurückgeliefert.
- 403 Forbidden** Der Request wurde vom Server abgelehnt. Der Webserver hat zum Beispiel keine Leserechte für die angeforderte Datei.
- 404 Not Found** Die angeforderte Ressource wurde nicht gefunden.
- 408 Request Timeout** Der Server konnte die Anfrage nicht innerhalb der zur Verfügung stehenden Zeitspanne verarbeiten.
- 414 Request URI Too Long** Der Request konnte nicht verarbeitet werden, da die angeforderte URI zu lang war.

Response-Codes 500-599: Server Fehler

Dieser Bereich teilt dem Client mit, dass der Server einen Fehler verursacht hat. Wichtige Response-Codes sind:

- 500 Internal Server Error** Der Server konnte den Request wegen eines internen Fehlers nicht verarbeiten.
- 505 HTTP Version Not Supported** Der Server unterstützt nicht die verwendete HTTP-Version.

Eine ausführliche Liste aller Response-Codes ist im RFC2616 [3] zu finden. Nähere Informationen zu HTTP gibt es in [45, S.726-728] und [50, Kap.3]. Alle für HTTP relevanten RFCs sind unter <http://www.w3.org/Protocols/Specs.html> aufgelistet.

2.4.3 HTTPS - Hypertext Transfer Protocol Secure

HTTPS wird überall dort im WWW verwendet, wo sichere Verbindungen vonnöten sind (z.B. beim E-Commerce, Online-Banking, ...). Die Daten werden über SSL/TLS verschlüsselt und damit abhörsicher gemacht. HTTPS-Verbindungen laufen ebenso wie HTTP-Verbindungen über TCP. Der Standard-Port für HTTPS-Verbindungen ist 443. Ein Webserver, der HTTPS unterstützen will, muss diesen Port ebenso wie Port 80 offen halten.

HTTPS über TCP ist technologisch gesehen identisch zu HTTP via SSL über TCP.

2.5 HTML & CSS - Formatierung von Webdokumenten

2.5.1 HTML

Die *Hypertext Markup Language* (HTML) ist eine für Hypertext spezialisierte Auszeichnungssprache, die auf der im ISO-Standard 8879 definierten Metasprache SGML basiert. Um ein gültiges Dokument in SGML zu beschreiben, benötigt man eine *Document Type Definition* (DTD), in der die Dokumentstruktur beschrieben wird.

Die letzte Version des HTML-Standards (HTML 4.01) wurde in der Meta-Sprache XML neu formuliert und wird nun XHTML 1.0 genannt. XML genügt den von SGML geforderten Bedingungen, verfügt aber über strengere syntaktische Regeln.

Ein Überblick über die in HTML gebräuchlichen Sprachelemente ist nicht Gegenstand dieser Diplomarbeit. Bei näherem Interesse sei hier auf das Projekt SelfHTML [39] oder die HTML 4.01 Spezifikation [55] verwiesen.

Die drei in XHTML 1.0 definierten DTD-Varianten sind:

Strict Diese DTD war notwendig um an XML anzuknüpfen und bildet die Grundlage für XHTML. Viele Elemente und Attribute fehlen, jedes geöffnete Element muss ein schließendes Element besitzen, die Reihenfolge der Elemente muss eingehalten werden und Text muss grundsätzlich innerhalb eines Containers sein. Um abwärtskompatibel zu sein, ist diese DTD nicht zwingend vorgeschrieben.

Transitional Diese DTD akzeptiert noch ältere Elemente und Attribute. Sie stellt die Alternative für HTML-Autoren dar, die noch nicht Inhalt und Layout getrennt haben.

Frameset Das Frameset enthält zusätzlich zu allen Elementen der Transitional-Methode noch die Elemente zur Erzeugung eines Framesets.

Die Struktur einer Webseite besteht immer aus drei Bereichen:

1. Definition der verwendeten DTD (z.B. HTML 4.01 Strict) (kann die Verarbeitungsgeschwindigkeit des Browsers beeinflussen),
2. HTML-Header mit Metaangaben (Autoren, Schlüsselwörter, Beschreibung, ...), Dateiimporten und Definitionen (beide Male Stylesheets, Javascript, ...),
3. HTML-Body, der den anzuzeigenden Inhalt enthält.

Eine Beispiel-HTML-Seite sieht somit folgendermaßen aus:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Titel der Webseite</title>
    <link href="stylesheets.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <span>Dies wird im Browser angezeigt.<br />Zeilenumbruch</span>
  </body>
</html>
```

2.5.2 CSS

Cascading Style Sheets (CSS) ist eine deklarative Stylesheet-Sprache, die verwendet wird, um das Layout vom Inhalt in HTML zu trennen. Die Syntax von CSS-Klassen besteht aus Regeln, die bestimmten Eigenschaften Werte zuweisen. Eigenschaften können zum Beispiel Farbe, Schrift, Position auf der Seite und Ausrichtung sein.

CSS verfolgt dabei das Prinzip der Vererbung der Eigenschaftswerte an untergeordnete Elemente. Durch Kombination mehrerer Stylesheets können die Eigenschaften der CSS-Klassen vermischt werden. Das W3C betreibt unter [54] eine Übersichtsseite zu CSS.

2.6 Webbrowser - Anzeigen von Webdokumenten

Webbrowser (kurz: Browser) sind die Präsentations- und Steuerungsprogramme des WWW. Ein Browser kümmert sich um die Annahme der Nutzeranfragen und verbindet sich mit den Webservern um Webressourcen abzufragen. Treten Fehler während des Transportes auf, versucht der Browser den Vorgang zu wiederholen oder informiert letztendlich den Nutzer über den Fehlschlag.

Wurde eine Webseite erfolgreich angefordert, ist es Aufgabe des Browsers, dass alle *Inline*-Elemente (eingebettete Elemente wie Bilder, ...) ebenfalls geladen werden und dass die Seite richtig dargestellt wird. Die Darstellung ist aber ein Problem. Eine Webseite, die auf einem Browser richtig dargestellt wird, stellt ein anderer Browser möglicherweise falsch dar.

2.6.1 Browserkrieg

Anfang der 1990er Jahre diente das WWW fast ausschließlich nur den Akademikern als Informationsquelle. Damals novellierte die Firma Netscape den Begriff Webbrowser, als sie mit dem Netscape Navigator den HTML-2.0 Standard drastisch erweiterte, welcher weder Formatierungen noch Tabellen oder Ähnliches kannte. Der neue Browser erlaubte den WWW-Autoren das Einbinden von Tabellen, Farben und später (1995) sogar Frames, Javascript und Multimedia. Mit Netscape begann die Popularität des WWW.

Im August 1995 wurde Microsoft aktiv und veröffentlichte seinen Internet Explorer. Microsoft sah die Marktführerschaft auf dem Gebiet der Betriebssysteme gefährdet, denn auf dem Browsermarkt dominierte Netscape mit über 80 Prozent und Netscape war nicht zwingend an das Betriebssystem von Microsoft gebunden. Anfänglich war der Internet Explorer keine Gefahr für Netscape. Es arbeiteten gerade einmal 6 Mitarbeiter an diesem Browser und er war dem Navigator technisch hoffnungslos unterlegen.

In den folgenden Jahren (bis 1999) versuchten beide Hersteller sich gegenseitig zu übertrumpfen und erweiterten den vom W3C definierten Webstandard eigenmächtig und natürlich inkompatibel zum Konkurrenten.

Durch die zusätzliche Integration des Internet Explorers in das Betriebssystem und alle Office-Produkte von Microsoft verlor Netscape im Frühjahr 1999 erstmalig die Marktführerschaft an Microsoft. Am Ende konnte Netscape personell⁹ und finanziell nicht mithalten und gab den Programmcode im Sommer 1999 der Open-Source-Gemeinde frei. Der Programmcode wurde komplett neu implementiert und diente als Grundlage für die Entwicklung aller heutigen Mozilla-Klone. Diese konnten den verlorenen Boden zum Internet Explorer mit 90 Prozent Marktanteil bisher noch nicht wieder einholen [46].

2.6.2 Internet Standards und Sicherheit

Negative Auswirkungen des hohen Marktanteils vom Internet Explorer zeigen sich in der niedrigen Frequenz der Updates und Patches. Im Durchschnitt dauert es mehrere Monate, bis gravierende Sicherheitslöcher im Browser geschlossen werden und dann zumeist nur ungenügend. Auch die Kompatibilität zu aktuellen technischen Standards von HTML lässt zu wünschen übrig. Der Internet Explorer ist zwar sehr fehlertolerant, was die Syntax von HTML angeht, aber vor allem CSS2-Standards werden nicht unterstützt und machen es WWW-Autoren teilweise unmöglich, Webseiten standardkonform zu schreiben [27].

⁹Bei Microsoft arbeiteten im Jahr 1999 inzwischen 1000 Mitarbeiter am Internet Explorer. Dies waren mehr Angestellte, als die Firma Netscape für ihre gesamte Produktpalette je hatte.

2.7 Skript & Programm - Dynamik in Webdokumenten

Die Möglichkeit, allein statische Informationen im WWW abzurufen, hätte das WWW nicht zu dem gemacht, was es heute ist. Mittels client- und serverseitigen Skripten und Programmen ist es möglich eine Webseite je nach Nutzerwunsch anzupassen und zu adaptieren oder Daten aus einer Datenbank zu holen.

2.7.1 Clientseitige Skriptsprachen

Clientseitige Skriptsprachen werden vom Server zum Client übertragen und erst beim Nutzer interpretiert und ausgeführt.

Die erste clientseitige Skriptsprache Javascript wurde von Netscape eingeführt und von der *European Computer Manufacturers Association* (ECMA) in *ECMAScript* standardisiert [16].

Clientseitige Skriptsprachen lassen die Webseite auf Nutzeraktionen reagieren. Beispielsweise öffnen sich Menüs, wenn eine Maus über ein Objekt fährt oder Stylesheets ändern sich. Zumeist werden Aktionen bei Javascript getriggert ausgeführt. Das bedeutet, dass der Nutzer agieren muss, damit die Javascript-Routine ausgeführt wird.

Eine weitere clientseitige Skriptsprache ist VBScript von Microsoft. Clientseitige Skriptsprachen sind nicht Inhalt dieser Diplomarbeit, da es nicht das Ziel ist, die Syntax der verschiedenen Sprachkonstrukte zu überprüfen und deren Korrektheit festzustellen [50, S.375ff] [20, S.134ff].

2.7.2 Serverseitige Schnittstellen

Im Gegensatz zu clientseitigen Skriptsprachen sind serverseitige Schnittstellen Programme, die bei Bedarf vom Webserver aufgerufen werden. Die bekannteste Programmierschnittstelle für Webserver ist das *Common Gateway Interface* (CGI). Parameter vom Client werden über Umgebungsvariablen oder über Standard-Eingabe an ein CGI-Programm übergeben. Das CGI-Programm kann in Perl, Python, C/C++ oder Java geschrieben sein.

ColdFusion von Allaire (heute Macromedia) besitzt eine eigene Programmiersprache namens CFML (*ColdFusion Markup Language*). CFML bietet sowohl eine einfache Sammlung an Tags und Funktionen als auch Zugriff auf komplexe Technologien.

Server Side Includes (SSI) stellen eine Ergänzung zum HTML-Befehlssatz dar. SSI-Befehle werden in HTML als Kommentare eingefügt und beim Aufruf interpretiert.

PHP (*PHP: Hypertext Preprocessor*) ist eine (seit Juli 2004) objektorientierte, an die Syntax von C bzw. Perl angelehnte Skriptsprache, die serverseitig eine breite Unterstützung an Funktionsbibliotheken anbietet.

Java Server Pages (JSP) bzw. Servlets sind Produkte von Sun in der Serverskriptsparte, wobei auf die volle Funktionalität der Java-Klassenbibliotheken zurückgegriffen werden kann. Servlets sind vorkompilierte, speicherresident geladene Prozesse und gewährleisten eine im Gegensatz zu anderen Sprachen gute Performance. JSP-Skripte sind in HTML integrierte Servlets und werden beim ersten Aufruf in Servlets umgewandelt.

Active Server Pages (ASP) sind serverseitige Skripte von Microsoft, die ähnlich JSP auf deren Klassenbibliotheken zurückgreifen. Der Nachteil von ASP ist natürlich die Plattformgebundenheit an Microsofts Betriebssystem.

Mit Serverskripten und Programmen können mächtige Anwendungen geschrieben werden. Zum Beispiel bietet ein *Content-Management-System* (CMS) dem Webautor die Möglichkeit, Inhalte schnell und einfach per Administrations-Oberfläche in die Webseite einzufügen, ohne den Quellcode der Seite zu ändern. CMS ermöglichen die Trennung von Inhalt, Gestaltung, Funktion und verschiedenen Navigationsstrukturen. Nach außen hin ist der Einsatz von Serverskripten zumeist nicht nachweisbar. Deshalb gibt es ein Problem den Wartungsaufwand von dynamischen Webseiten einzuschätzen. Bild 2.3 zeigt den Ablauf bei der Anforderung einer dynamischen Seite.

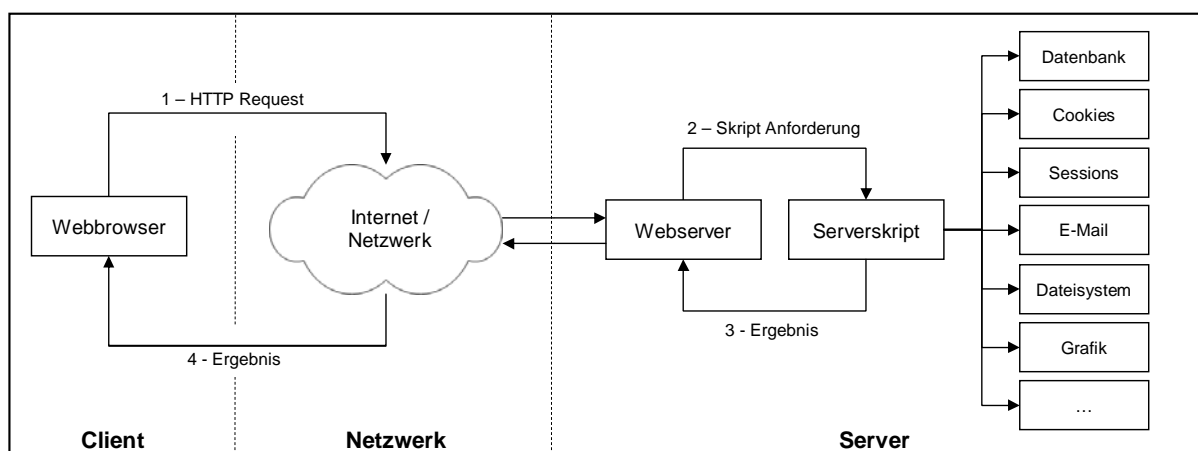


Abbildung 2.3: Ausführung eines serverseitigen Skriptes

3 Qualität und Messtheorie

Der Begriff Qualität ist ein wichtiges Kriterium um ein Produkt einzuschätzen und es zu bewerten. Es soll immer die beste Qualität erzielt werden. Um dieses Ziel zu erreichen, muss die Qualität aber eindeutig und sehr präzise definiert werden.

Die Qualität eines Produktes ist nicht einfach zu beschreiben. Sie setzt sich aus vielen Eigenschaften zusammen, welche der Prüfgegenstand erfüllen muss. Diese Eigenschaften können aber von Produkt zu Produkt unterschiedlich definiert sein und sich teilweise sogar gegenseitig ausschließen.

Es ist also falsch, wenn man sich als Ziel setzt die „beste Qualität“ zu erreichen. Viel eher sollte gesagt werden, dass man die „richtige Qualität“ erreichen will, die sich aus den jeweils gewünschten Eigenschaften bildet. Hierzu müssen die Eigenschaften von Produkt zu Produkt jedes Mal neu angepasst werden. Es muss eine Qualitätszielbestimmung erfolgen und es müssen Methoden gefunden werden, wie diese Ziele zu erreichen sind [29, Kap.1].

Die Bestimmung der Ziele sowie die Messung der Qualität sind dabei immer mit dem Zeit- und Kostenaspekt abzugleichen. Durch die Vielfalt der Anforderungen und die unterschiedliche Ausprägung der Qualitätsziele kann es keine universell geeignete Lösung geben. Der kleinste gemeinsame Nenner wäre eine sehr allgemein gehaltene Sammlung an bewertenden Methoden, die bei jedem Projekt vor der Benutzung angepasst werden müsste.

In diesem Kapitel werden einige Grundlagen von Qualität und Messungen behandelt. Abschnitt 3.1 definiert grundlegende Termini und Begriffe aus dem Gebiet der Software-Qualität. Abschnitt 3.2 behandelt erforderliche messtheoretische Grundlagen. Der Standard ISO 9126 zur Bewertung von Softwareprodukten wird in Abschnitt 3.3 erläutert. Einen tieferen Einblick in mögliche Messmethoden des WWW liefert das Kapitel 4.

3.1 Terminologie und Begriffsdefinitionen

Die folgenden Definitionen sind für das Gesamtverständnis des Begriffes *Qualität* wichtig.

Qualitätssicherung

Zur Qualitätssicherung gehören Maßnahmen zur Aufrechterhaltung und Bewahrung von Qualität. Die frühzeitige Erkennung von Qualitätsrisiken und das Ergreifen von Maßnahmen zur Vermeidung von Fehlern ist wichtig.

Qualität

Qualität ist ein abstrakter Begriff und beschreibt den Grad, in dem ein Satz inhärenter Qualitätsmerkmale entsprechende Qualitätsanforderungen erfüllt (ISO 9000 [23]).

Qualitätsanforderung

Der Begriff der Qualitätsanforderung bezeichnet die Gesamtheit der einzelnen, betrachteten Anforderungen an die Beschaffenheit eines Produktes [17].

Qualitätsmerkmal

Qualität wird konkret durch Qualitätsmerkmale festgelegt. Diese stellen Eigenschaften der Eignung von Anforderungen unter einem bestimmten Blickwinkel dar, ohne jedoch eine Aussage über den Grad der Ausprägung zu enthalten.

Ein Anwender präferiert zum Beispiel die Merkmale Funktionalität, Benutzbarkeit, Verfügbarkeit, Sicherheit und Zuverlässigkeit. Ein Entwickler sieht die Merkmale Wartbarkeit, Erweiterbarkeit, Portabilität, Prüfbarkeit, Speicher- und Laufzeiteffizienz als wichtig an [40, S.554].

Es werden zwischen funktionalen und nicht-funktionalen Anforderungen unterschieden [13, S.24f]. Hierbei treten oft Wechselwirkungen auf und viele Qualitätsmerkmale schließen sich gegenseitig aus. Zum Beispiel kann Software entweder speichereffizient mit geringstmöglichem Speicherbedarf oder laufzeiteffizient mit minimaler Laufzeit aber erhöhtem Speicherbedarf sein.

Eine Verbesserung eines Merkmals zieht oft die Verschlechterung eines anderen mit sich. Somit gibt es keine perfekte Kombination. Mittels einer Qualitätszielbestimmung muss die optimale Lösung gefunden werden.

Qualitätsmaß

Ein Qualitätsmaß ist die konkrete Feststellung der Ausprägung eines Qualitätsmerkmals. Hierzu wird dem Qualitätsmerkmal eine Zahl zugeordnet, welche durch das Qualitätsmaß bewertet bzw. messbar gemacht werden kann [13, S.176f].

Fehlverhalten, Fehler, Irrtum

Die Bedeutungen der Begriffe Fehlverhalten, Fehler und Irrtum sind nicht identisch.

Ein Fehlverhalten oder Ausfall (engl.: *failure*) tritt bei der Benutzung des Produktes auf und ist durch einen Fehler verursacht. Bei Software basieren Fehlverhalten zumeist auf Programmierfehlern, bei Hardware auf Verschleiß.

Ein Fehler oder Defekt (engl.: *fault, defect*) ist dagegen direkt im Produkt verankert. Bei Software könnte dies zum Beispiel ein Tippfehler sein, bei Hardware ein Konstruktionsfehler wie zum Beispiel ein falsches Bauteil im Server.

Die Ursache eines Fehlers könnte ein Irrtum (engl.: *error*) sein. Ein Irrtum ist zum Beispiel die falsche Benutzung eines Sprachkonstrukts oder eine zu gering kalkulierte Bandbreite beim Webserver.

Korrektheit

Ein Produkt ist korrekt, wenn es seinen vorher spezifizierten Eigenschaften entspricht bzw. wenn es eine Übereinstimmung zwischen beobachtetem und gewünschtem Verhalten gibt. Das Produkt tut also genau das, was es tun sollte und ist somit frei von Fehlverhalten.

Ein Produkt kann entweder korrekt sein oder nicht - eine graduelle Abstufung gibt es dabei nicht. Existiert zu dem Produkt keine Spezifikation, so ist eine Überprüfung auf Korrektheit nicht möglich.

Ein komplett fehlerfreies Produkt ist oberhalb einer bestimmten Komplexität kaum möglich. Deshalb werden Fehler priorisiert und das Ziel ist die Vermeidung von schweren Fehlern.

Vollständigkeit

Ein Produkt ist funktional vollständig, wenn alle in der Spezifikation geforderten Funktionen realisiert sind.

3.2 Messtheorie

Erst durch Messungen werden die Qualitätsmerkmale quantifiziert. Durch das Abbilden der Eigenschaften auf eine Skala von gut bis schlecht wird Qualität „sichtbar“.

3.2.1 Maße und Metriken

Software-Maße beschreiben bestimmte Eigenschaften des Produktes. Maße können in folgenden Gebieten angewendet werden [29, S.211]:

- Kontrolle der Qualität und der Komplexität
- Kontrolle und Überwachung des Entwicklungsprozesses
- Aufwands-, Kosten- und Zeitabschätzung
- Kontrolle der Einhaltung von Standards
- Problemidentifikation
- Vergleich und Beurteilung von Produkten
- Nachweis, ob Änderungen den gewünschten Erfolg hatten

Während im Englischen der Begriff *software metrics* gebräuchlich ist, müsste im deutschen Sprachgebrauch von Maßen geredet werden. Das Wort Metrik entstammt dem Griechischen, kommt aus der Verslehre und beschreibt die Lehre von dem Versbau und den Versmaßen [2]. In der Mathematik beschreibt es ein axiomatisches Maß für Abstände im metrischen Raum. Obwohl in der Praxis häufig von Software-Metriken geredet wird, wird in dieser Diplomarbeit korrekterweise das Wort Software-Maß verwendet [56].

3.2.2 Forderung an Maße

Nach [29] müssen Maße folgende Voraussetzungen erfüllen, um sinnvoll zu sein:

Einfachheit Das Ergebnis muss einfach genug sein, dass es interpretierbar ist.

Eignung (Validität) Das Maß sollte für die zu messende Eigenschaft geeignet sein.

Stabilität Der Wert des Maßes muss stabil genug sein um nicht von nebensächlichen Eigenschaften beeinflusst zu werden.

Rechtzeitigkeit Das Maß sollte zu einem Zeitpunkt gebildet werden, an dem ein Fehlverhalten noch behoben werden kann.

Analysierbarkeit Der Messwert muss statistisch analysierbar sein.

Reproduzierbarkeit Das Maß muss präzise definiert sein und unabhängig von der Art der Bildung muss der Messwert identisch sein.

3.2.3 Maßskalen

Maßskalen beschreiben, welche Operationen auf den Messwerten sinnvoll sind. Ein Beispiel aus der Physik zeigt die Notwendigkeit von Maßskalen.

Messung der Länge Wenn das Brett a einen Meter und das Brett b zwei Meter lang ist, dann kann man sagen, dass das Brett b doppelt so lang ist wie das Brett a.

Messung der Temperatur Es ist inkorrekt zu sagen, dass die Temperatur 20°C doppelt so warm ist wie 10°C.

Bei beiden Beispielen wurde dieselbe Operation ausgeführt. Während diese Operation bei der Längenmessung praktikabel ist, würde sie bei Temperaturen keinen Sinn machen, da kein Verhältnis zwischen einem positiven und einem negativen Temperaturwert gebildet werden könnte.

Die Länge in Metern wird auf einer Rationalskala abgebildet und die Temperatur auf einer Intervallskala. Auf Intervallskalen sind Divisionen nicht sinnvoll. Insgesamt unterscheidet man zwischen fünf aufeinander aufbauenden Skalentypen [29, S.216f] [13, S.179]:

Nominalskala

Freie Bezeichnung bestimmter Eigenschaften mit Entitäten

- Checkliste: Ja oder Nein
- Auswählen der Serverskriptsprache: PHP, JSP, Perl, ...

Bijektive Abbildung ist eine zulässige Transformation.

Ordinalskala

Zuordnung der Messwerte zu sortierten Entitäten ohne Abstände.

- Schulnoten: 1 ist besser als 2 ist besser als ... 6
- Zufriedenheit: sehr zufrieden, zufrieden, unzufrieden, sehr unzufrieden

Transformationen mittels streng monoton steigender Funktion sind zulässig.

Erstmalig kann man durch Größenvergleiche zwei Werte vergleichen.

Intervallskala

Abbildung des Messwertes auf eine kontinuierliche Skala. Es existiert kein Nullpunkt auf der Skala.

- Temperatur auf der Celsius- oder Fahrenheitskala
- Jahreszahlen

Lineare Transformationen der Art $g(x) = \alpha \cdot x + \beta$ sind erlaubt. Somit ist es möglich zwischen der Celsius- und der Fahrenheitskala umzurechnen ohne die Relation zwischen Temperaturen zu verlieren ($T^{in\ ^\circ C} = \frac{5}{9} \cdot T^{in\ ^\circ F} - \frac{160}{9}$).

Neben den Größenvergleichen sind Differenzen und Summen möglich, wodurch man den Durchschnitt mehrerer Werte errechnen kann.

Rationalskala

Abbildung des Messwertes auf eine kontinuierliche Skala. Es existiert ein natürlicher Nullpunkt auf der Skala.

- Temperatur in Kelvin (Nullpunkt: 0 K)
- Entfernung (Nullpunkt: keine Entfernung)

Zusätzlich zu in den vorherigen Skalen erlaubten Transformationen sind multiplikative Transformationen der Art $g(x) = \alpha \cdot x$ erlaubt.

Erlaubte Operationen sind zusätzlich Multiplikationen und Divisionen.

Absolutskala

Zuordnung der Messwerte auf eine Skala, die den Sachverhalt darstellt. Eine verhältniserhaltende Skalierung der Merkmalsausprägung würde die mit dem Messwert verbundene Aussage verändern.

- Zahlen
- Häufigkeiten
- Wahrscheinlichkeiten

Einen kurzen Überblick über die fünf Skalentypen liefert Tabelle 3.1.

Bezeichnung	Relevanz	mgl. Transformationen
<i>Nominalskala</i>	Unterscheidung der Entitäten	bijektive Abbildung
<i>Ordinalskala</i>	Anordnung der Entitäten	streng monotone Funktionen
<i>Intervallskala</i>	Verhältnis von Intervallen	$g(x) = \alpha \cdot x + \beta$
<i>Rationalskala</i>	Verhältnis von Messwerten	$g(x) = \alpha \cdot x$
<i>Absolutskala</i>	Messwerte	$g(x) = x$

Tabelle 3.1: Skalentypen [57]

3.2.4 Definition der Ziele von Maßen

Damit Maße zielgerichtet eingesetzt werden, kann die Methode *Goal Question Metric* (GQM) verwendet werden. Dabei sind drei Fragen zu beantworten:

1. Welches Ziel soll durch die Messung erreicht werden (*Goal*)?
2. Was muss gemessen werden, damit das Ziel erreicht wird (*Question*)?
3. Wie kann gemessen werden, d.h. welche Maße sind besonders geeignet die Eigenschaften zu erfassen (*Metric*)?

3.2.5 Empirische Bedeutung

Da ein Maß allein ohne Vergleichswerte relativ nutzlos ist, kann man davon ausgehen, dass das Maß an sich uninteressant ist und vielmehr eine andere Eigenschaft, die mit dem Maß korreliert, von Interesse ist.

Empirie (von griech.: *empereia* = Erfahrung) ist eine gewonnene Erkenntnis, die auf methodischem Weg z.B. durch Induktion und Analogieschlussmethode aufgestellt werden kann. Mittels empirischer Beobachtungen ist es möglich, Maße auswertbaren Eigenschaften zuzuordnen. Ein relativ einfaches Beispiel ist die folgende Zuordnung:

Je größer die Anzahl der LOC, desto größer ist der Wartungsaufwand.

Die Gegenüberstellung von Maß und Empirie ist dabei nicht immer einfach. Die Exaktheit und Bedeutung variiert je nach Genauigkeit (Bild 3.1) von unbedeutend bis außerordentlich bedeutend [13, S.187].

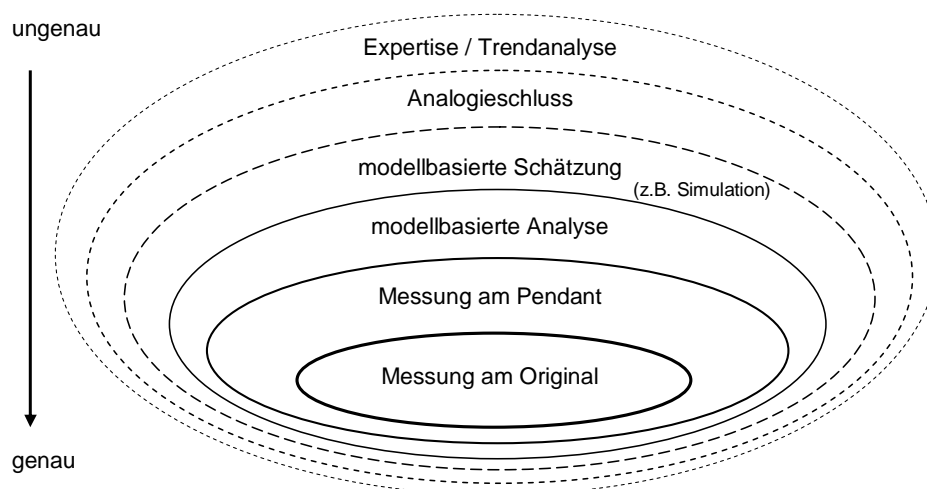


Abbildung 3.1: Empirie-basierte Mess- und Schätzebenen [15]

3.3 Bewertung von Softwareprodukten mit ISO 9126

Aus der allgemeinen Definition von Qualität wurde in der ISO 9126 [24] eine Definition für Software-Qualität abgeleitet.

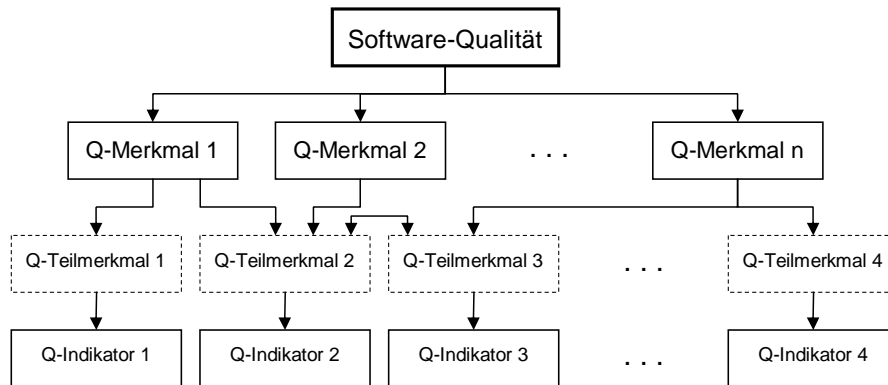


Abbildung 3.2: Softwarequalität im Qualitätsmodell

ISO 9126 spezifiziert in Abbildung 3.2 die Qualitätsmerkmale Funktionalität, Zuverlässigkeit, Effizienz, Benutzerfreundlichkeit, Portabilität und Wartbarkeit. Die Norm enthält jedoch keine Teilmerkmale oder Qualitätsindikatoren¹⁰, die in der Phase der Qualitätszielbestimmung für jedes Produkt separat definiert werden müssen.

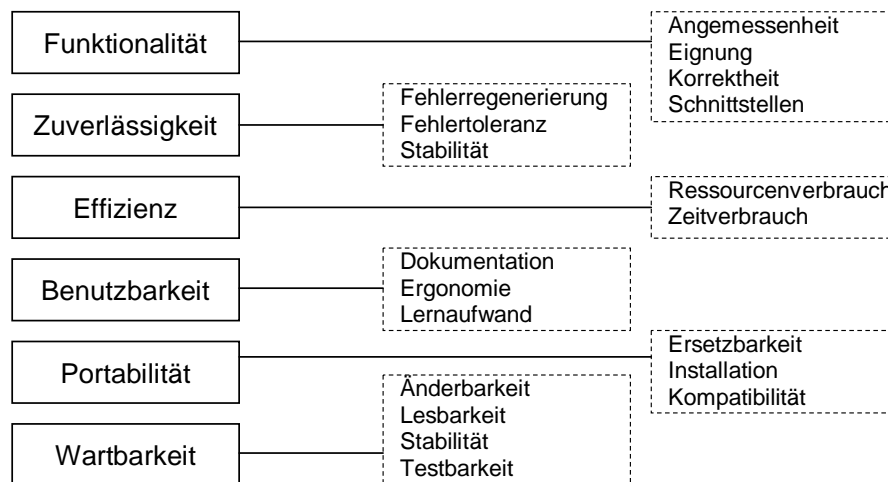


Abbildung 3.3: Qualitätsmodell nach ISO/IEC 9126: 1991 [13, S.29]

Abbildung 3.3 zeigt ein mögliches Qualitätsmodell nach ISO/IEC 9126: 1991. In den gestrichelten Rahmen befinden sich Qualitätsteilmerkmale, die zusammen das jeweils von ISO 9126 definierte Qualitätsmerkmal ergeben.

¹⁰Indikatoren sind Eigenschaften eines Softwareproduktes, die nicht weiter verfeinert werden können und mit Hilfe von Qualitätsmaßen quantitativ messbar sind. Indikatoren werden zu den Qualitätsmerkmalen in Beziehung gesetzt.

4 Web Measurement

Während in den letzten Kapiteln Grundlagen des World Wide Web, von Qualität und Messtheorie behandelt wurden, beschreibt dieses Kapitel aktuelle Verfahren, um die Qualität von Webseiten zu messen und zu bewerten.

Wenn von einer qualitativ hochwertigen Webseite die Rede ist, handelt es sich um eine subjektive Einschätzung. Im Folgenden sind fünf Szenarien dargestellt:

WWW-Nutzer

Ein WWW-Nutzer sucht nach speziellen Informationen. Hierzu wird er entsprechende Suchbegriffe in eine Suchmaschine eintragen und erwarten, dass dazu passende Webseiten in der Ergebnisliste sind. Die Webseite muss also suchmaschinenfreundlich sein. Hat der Nutzer eine Webseite gefunden, will er, dass die gesuchten Informationen gut strukturiert und übersichtlich präsentiert sind. Die Webseite soll sowohl mit Modem als auch mit Breitbandanschlüssen schnell geladen werden. Die Struktur der Webseite soll intuitiv und lückenlos sein und keine Hyperlinks enthalten, die ins Leere führen. Zudem sollte die Seite browserfreundlich und barrierefrei¹¹ sein.

Manager

Einen Manager interessiert weniger das Aussehen der Firmenwebseite noch die Funktionalität. Für ihn sind vielmehr wirtschaftliche Fakten, farbige Charts und Zahlen von Bedeutung, die die Akzeptanz der Seite belegen. Außerdem interessiert ihn, ob die Kosten für Werbung auch den gewünschten Mehrgewinn bringen.

Webmaster

Der Webmaster ist derjenige, der direkt an der Webseite arbeitet, sie weiterentwickelt und wartet. Er benötigt Aussagen über Merkmale wie Wartbarkeit, Komplexität und Fehleranfälligkeit.

¹¹Eine barrierefreie Webseite schließt körperlich eingeschränkte Menschen nicht aus. Barrierefreiheit ist innerhalb des W3C das Thema der *Web Accessibility Initiative* (WAI), welche 1999 die Richtlinie WCAG 1.0 verabschiedete. Die deutsche BITV (Barrierefreie Informationstechnik Verordnung; siehe <http://www.wob11.de/>) baut auf dieser Richtlinie auf (siehe auch [6]).

Server-Administrator

Die Aufgabe des Administrators ist die Bereitstellung der Plattform und der technischen Voraussetzungen. Ihn interessieren benötigte Techniken und deren Stabilität. Nicht zu unterschätzen ist die Anzahl der Sicherheitslöcher der verwendeten Software sowie die Zeit, bis ein Security Exploit behoben wird. Die Stabilität und Korrektheit der Web-Applikationen sowie die verwendete Backup-Strategie müssen wohl definiert sein. Ebenso ist er für performancetechnische Aspekte verantwortlich.

Webbrowser

Obwohl ein Webbrowser sich nicht direkt über eine faktisch qualitativ wertlose Webseite beschweren kann, hat auch ein Webbrowser beziehungsweise der Hersteller des Browsers ein Bild von Qualität. Dieses Bild wird von Institutionen wie dem World Wide Web Consortium geprägt und durch Standards wie XHTML 1.0 spezifiziert. Eine qualitativ hochwertige Seite muss für einen Webbrowser also standardkonform und syntaktisch korrekt sein.

Die Beispiele zeigen, dass jede Perspektive unterschiedliche Eigenschaften von Webseiten als Qualitätsmerkmale definiert. Diese Eigenschaften hängen oft voneinander ab und beeinflussen sich gegenseitig. Zum Beispiel wird eine hohe Serverlast und damit ein einhergehender Performanceverlust langfristig zu einem Verlust der Nutzerakzeptanz führen.

Nicht alle Eigenschaften sind messbar, viele sind an den Messort gebunden. Es ist deshalb nur möglich ein Messtool zu entwickeln, welches eine bestimmte Menge an Eigenschaften misst. In dieser Diplomarbeit wird eine breite Auswahl an Maßen getroffen, die durch Messungen direkt über das WWW realisiert werden können. Das Messen auf dem Server oder von nicht im WWW verfügbaren Daten wird nicht berücksichtigt.

4.1 Arten des Web Measurement

Web Measurement kann man in folgende Rubriken einteilen:

- Messung der Nutzerakzeptanz
- Messung wirtschaftlicher Aspekte
- Messung performance-technischer Aspekte
- Messung des Wartungsaufwands
- Messung der Eigenschaften der Webseite
- Messung der Standardkonformität

4.1.1 Messung der Nutzerakzeptanz

Server-Logfiles

Ein weithin verbreitetes Mittel zur Bestimmung der Nutzerakzeptanz ist die Auswertung der vom Webserver geschriebenen Nutzungs-Logfiles. In diesen Logfiles steht zum Beispiel, welcher Nutzer wann was gemacht hat. Hierbei wird der Nutzer über die verwendete IP-Adresse identifiziert [14, Kap.9.2] [42, Kap.5].

Diese zumeist sehr langen Logfiles werden serverseitig von Analyseprogrammen automatisiert ausgewertet und in Form von Diagrammen und Tabellen dargestellt. Webserver-Logs können benutzt werden, um grundlegende Daten der Nutzung zu erfahren. Allgemeine Informationen erhält man über Zählmaße¹². Will man detailliertere Informationen erfahren, muss man die Logfiles etwas gezielter untersuchen. Durch die Beobachtung eines Nutzers kann man zum Beispiel genau erkennen, wohin er geklickt hat und wo er die Seite verlassen hat. Hiermit kann man erkennen, wo die Webseite noch verbessert werden könnte. Ebenso ist es möglich den Erfolg einzelner Seiten zu bestimmen, um beispielsweise zu erfahren, ob eine Werbung erfolgreich war.

Dem in dieser Diplomarbeit entwickelten Messtool fehlt der Zugriff auf die Server-Logs. Im Allgemeinen sind Server-Logfiles nicht für das Internet zugänglich, weshalb dieser Ansatz hier nicht weiterverfolgt wird.

PageRank-Algorithmus

Durch Server-Logfiles erfährt man etwas über die Nutzung der Webseite, aber nicht zwingend etwas über die Wertung der Nutzer in Bezug auf diese Seite. Eine hohe Nutzerakzeptanz kann man indirekt über den Verlinkungsgrad der Webseite erkennen.

Ein verbreitetes Verfahren ist der PageRank-Algorithmus, dessen Grundprinzip ist: *„Eine Webseite ist wichtig, wenn sie oft von anderen Webseiten verlinkt wurde“*. Das Verfahren bewertet also nicht direkt die Qualität der Webseite, sondern überlässt die Bewertung den Autoren anderer Webseiten, die eine Verknüpfung angelegt haben. Die einzelnen Autoren geben subjektive Bewertungen ab, aber alle subjektiven Bewertungen zusammen ergeben ein objektives Bewertungsbild [28, S.53f] [9].

¹²Allgemeine, aus Webserver-Logfiles gewonnene Zählmaße sind: Anzahl der Requests, ausgesendete Bytes, Anzahl der eindeutigen Besucher (durch IP gekennzeichnet), Anzahl der abgerufenen Dokumente pro Besucher... [36, S.102]

Hierbei gilt: Je mehr Links auf eine Webseite verweisen, desto bedeutender ist diese Seite. Je weniger Links eine Seite enthält, desto bedeutender ist jeder einzelne Link. Je bedeutender eine Seite ist, desto bedeutender sind die auf ihr enthaltenen Links. Je bedeutender die Links sind, die auf eine bestimmte Seite zeigen, desto bedeutender ist diese Seite.

Um den PageRank r_v einer Seite v zu erhalten, muss dieser iterativ berechnet werden:

$$r_v^{t+1} = \frac{\varepsilon}{n} + (1 - \varepsilon) \cdot \sum_{u:(u,v) \in E} \frac{r_u^t}{d_u^{out}}$$

Die einzelnen Werte sind:

- r_v^{t+1} - aktueller Iterationsschritt bei der Berechnung des PageRanks r_v
- ε - Wahrscheinlichkeit, dass ein Nutzer die Lust verliert und zu einer völlig anderen Webseite springt
- $(1 - \varepsilon)$ - Wahrscheinlichkeit, dass ein Nutzer einen Link auf der momentanen Webseite weiterverfolgt
- n - Anzahl aller Webseiten in der Ergebnismenge E
- r_u - Webseiten in der Ergebnismenge E , die r_v verlinken
- d_u^{out} - Anzahl der ausgehenden Links auf der Webseite r_u

In Abbildung 4.1 ist ein einfaches Beispiel abgebildet, in dem die Kanten- und Knotengewichte im Gleichgewicht sind. Für das gewählte Webseiten-System sind unter anderem je nach Initial-Werten folgende Lösungen möglich

$$r(A, B, C, D) = \{ 0,2 \quad 0,6 \quad 0,5 \quad 0,4 \},$$

$$r(A, B, C, D) = \{ 0,1 \quad 0,3 \quad 0,25 \quad 0,2 \}.$$

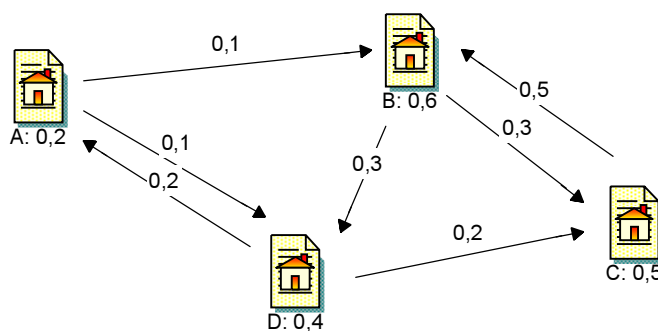


Abbildung 4.1: Linkgraph mit PageRank-bewerteten Knoten

Nachteile des PageRank-Algorithmus sind:

1. Damit der PageRank-Algorithmus korrekt bewerten kann, muss die Ergebnismenge eine gewisse Größe erreicht haben. Der Suchmaschinenbetreiber Google, welcher PageRank als Basis verwendet, hat zum Vergleich 8.058.044.651 Webseiten indiziert (November 2004).
2. 1999 untersuchte Albert-László Barabási das Netz auf die Verlinkungsstruktur. Er erwartete eine Gaußsche Glockenkurve mit wenigen kaum verlinkten, wenigen oft verlinkten und vielen mittelmäßig oft verlinkten Webseiten. Stattdessen beobachtete er eine potenzförmige Kurve - viele kaum verlinkte Webseiten und wenige sehr oft verlinkte Webseiten. Die Schlussfolgerung ist, dass neue, unbekannte Webseiten unabhängig von Qualität und Inhalt immer benachteiligt sind [1].
3. Der PageRank-Algorithmus ist anfällig gegenüber Linkfarmen, Cloaking und anderen unseriösen Methoden, die darauf abzielen, einer bestimmten Webseite durch Manipulationen einen hohen Punktwert im PageRank-Algorithmus zu verschaffen.

Bewertungsportale

Alternativ - aber nicht automatisierbar - ist das Prüfen der Nutzerakzeptanz auf Bewertungsportalen im Internet, auf denen Nutzer ihre Berichte über Webseiten veröffentlichen. Beispiele hierfür sind <http://www.dooyoo.de/> und <http://www.ciao.com/>.

Zusammenfassung

Grundsätzlich sind Maße über Nutzerakzeptanz nur ein Indikator dafür, ob die Webseite von den Nutzern angenommen wird. Im Grunde genommen sagt Nutzerakzeptanz nichts über die Qualität der Webseite aus. Das folgende Gegenbeispiel zeigt, dass eine hohe Nutzerakzeptanz nicht mit Qualität gleichzusetzen ist:

Der Marktführer und Monopolist XYZ vertreibt ein beliebtes, oft verlinktes Software-Produkt. Durch den Erfolg dieses Produktes gerät es in das Visier von Hackern. Daher ist die Firma XYZ gezwungen, regelmäßig Software-Updates anzubieten. Hierzu benutzen sie ihre Firmenwebseite, die trotz des großen Erfolges der Firma in allen Punkten qualitativ schlecht ist. Leider haben die Nutzer des Produktes keine Wahl und benutzen deshalb trotzdem die ständig überlastete, nicht standardkonforme Webseite und laden die Updates herunter. Aus den Server-Logfiles kann die Firma entnehmen, dass ihre Webseite eine hohe Klickrate und viele Besuche täglich hat. Doch ist sie deshalb qualitativ gut?

4.1.2 Messung wirtschaftlicher Aspekte

Der wirtschaftliche Aspekt beim Web Measurement (speziell für Firmen-Webseiten) beschreibt den wirtschaftlichen Erfolg der Webseite. Hierbei muss beachtet werden, dass es verschiedene Arten von Webseiten und damit verbunden verschiedene Gewinnabsichten gibt [33].

B2C - Business-to-Consumer

B2C-Webseiten benutzen typischerweise Firmen, die ihre Produkte über den Vertriebsweg Internet an den Kunden bringen wollen. Dieser kann der einzige Vertriebskanal sein oder nur ein zusätzlicher Service. Beispiele für B2C-Webseiten¹³ sind <http://www.amazon.de/>, <http://www.dell.de/> und <http://www.mindfactory.de/>.

Um den Erfolg einer B2C-Webseite zu messen, kann zuerst einmal die **durchschnittliche Auftragszahl** bewertet werden. Allerdings sagt die Auftragszahl nicht sehr viel über das Verhältnis von tatsächlichem und möglichem Erfolg aus, da zumindest noch die Anzahl der Interessenten begutachtet werden sollte. Das Maß **Net dollar per visitor**

$$NetDollar = \frac{Umsatz}{\#Besucher}$$

sagt aus, wie viel jeder Besucher der Firma im Durchschnitt an Umsatz bringt.

Der **Clickstream**¹⁴ bietet einen Überblick, wie ein Nutzer auf die Seite kam. Durch die Analyse des Clickstreams kann man so zum Beispiel den Erfolg von Werbemaßnahmen belegen. Dazu passend lassen sich mit den **Drop-Off Rates** Schwachstellen im Bestellvorgang nachweisen. Hierbei wird überprüft, an welcher Stelle die Nutzer einen Bestellvorgang abgebrochen haben.

B2B - Business-to-Business und B2E - Business-to-Employee

B2B-Webseiten sind Portale, deren Mittelpunkt der Geschäfts- und Datenverkehr zwischen zwei Unternehmen ist, also z.B. zwischen Unternehmen und Zulieferer. B2E-Portale erlauben Mitarbeitern eines Unternehmens firmenspezifische Informationen abzufragen, wie z.B. Interna, Stellenausschreibungen, aktuelle Projekte, ... Bei beiden Portal-Typen stehen Sicherheitsaspekte, Performance und Effizienz im Vordergrund.

¹³Ein anderer Begriff für *Business-to-Consumer*-Webseiten ist *E-Shops*.

¹⁴Ein Clickstream enthält jeden Klick eines Nutzers auf der Webseite und ermöglicht somit eine detaillierte Analyse des Nutzerverhaltens.

B2B- und B2E-Portale sollen funktioneller Natur sein. Das Ziel besteht darin, dass die Nutzer der Systeme in möglichst kurzer Zeit das finden, was sie suchen. Das Maß **Average Time spent on system** misst hierbei, ob die durchschnittliche Session-Dauer möglichst gering ist. Die **Effizienz** wird gemessen, indem in den Logfiles ermittelt wird, ob viele Nutzer an denselben Stellen scheinbar ziellos herumklicken.

Content

Content-Webseiten bieten direkt keine Dienstleistungen für ihre Nutzer an. Beispiele sind alle Arten von Informationsportalen, Online-Enzyklopädien oder Wörterbüchern. Deren Gewinnabsichten sind primär im Supporting und in der Werbung zu finden. Kritische Eigenschaften für Content-Seiten sind **Loyalität** und **Nutzerzufriedenheit**. Maße für Loyalität sind Anzahl der Besuche, Anzahl aktiver, registrierter Nutzer und Karteteilchen bei Nutzeraccounts. Nutzerzufriedenheit kann man über Evaluationsumfragen, Feedbackformulare oder einfache Daumenhoch- / Daumenrunter-Funktionalität bestimmen.

Zusammenfassung

Die Qualität von Webseiten aus wirtschaftlicher Sicht ist zusammenfassend nur ein gezieltes Auswerten der Server-Logfiles oder der speziell von der Seite gesammelten Messwerte. Eine automatisierte Sammlung dieser Messwerte aus dem WWW ist somit nicht möglich. Die Auswertung und Bewertung durch ein Tool ist dadurch erst recht nicht gewährleistet.

4.1.3 Messung performance-technischer Aspekte

Bewertet ein Nutzer die Performance einer Webseite, vergleicht er deren Geschwindigkeit mit der von anderen Webseiten. Die Zeit wird hierbei aus den Einzelzeiten der folgenden Aspekte zusammengesetzt (siehe Bild 4.2 auf Seite 38):

1. Der Nutzer übermittelt eine Webadresse an den Browser.
2. Der Browser überprüft, ob die Webseite im Browser-Cache zwischengespeichert ist (R'_C). Ist dies der Fall, werden die folgenden Schritte übersprungen und die gecachte Webseite in Schritt 8 verwendet.
3. Der Browser schlüsselt die URI in seine Einzelteile auf und fragt die IP-Adresse der verwendeten Domain ab. Der Abruf der IP-Adresse beim DNS-Service kann verhältnismäßig viel Zeit in Anspruch nehmen und ist somit ein integraler Bestandteil der Gesamtperformance.

4. Der Browser öffnet eine TCP-Verbindung zu der IP-Adresse. Die Geschwindigkeit und benötigte Zeit hängt in der Netzwerkebene enorm vom gewählten Netzwerkpfad ab. Jeder einzelne Knoten im Netzwerk trägt zur Gesamtzeit bei¹⁵.
5. Der Endpunkt der Verbindung ist das Netzwerk, das den Computer beinhaltet, auf dem der Webserver läuft. Die Geschwindigkeit innerhalb dieses Netzwerkes wird unter anderem auch von der Verwendung von Sicherheits-Komponenten wie Firewall und Intrusion Detection Systems beeinflusst. Auch sind Lastverteilungssysteme möglich, die durch Redundanz positiv zur Performance beitragen können. Die Zeit, die die Daten vom Client zum Server benötigen, ist R'_{N1} .
6. R'_S ist die Zeit, die der Webserver zum Bearbeiten einer Anfrage benötigt. R'_S hängt sowohl von der Art und Größe des angeforderten Dokumentes ab als auch von der Serverarchitektur - also Drehgeschwindigkeit, Fragmentierung und Suchzeit der Festplatte, Busgeschwindigkeit, ...
7. Schliesslich wird das angeforderte Dokument wieder zurückgesendet (R'_{N2}). Die Gesamtzeit, die das Dokument von Anfordern bis Empfangen benötigt, wird nach [31] *Round-Trip Time* (RTT) genannt.
8. Der letzte Schritt ist das Parsen der HTML-Syntax im angeforderten Dokument und das Rendering der Seite im Browser. Zusätzlich müssen noch alle Inline-Elemente (also Bilder, eingebundene Include-Dateien, ...) abgerufen werden.

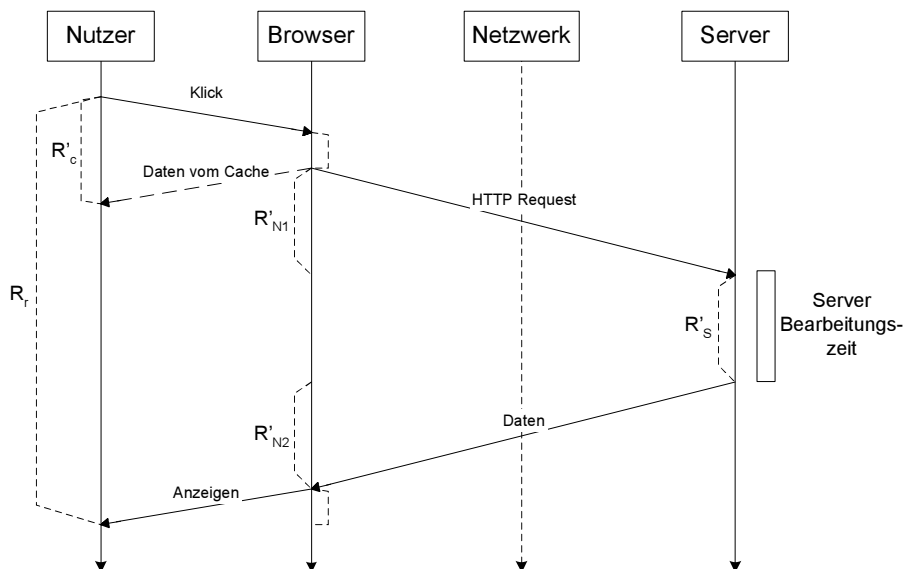


Abbildung 4.2: HTTP-Anfrage [31, S.78]

¹⁵Unter <http://www.internettrafficreport.com/> ist der momentane, weltweite Netzwerkfluss zu finden.

Die Zeit, die für den Transport verwendet wird, ist $R'_{Network} = R'_{N1} + R'_{N2}$. Da das zu implementierende Messtool keinen direkten Zugang zum Server hat, kann nur die Antwortzeit $R_{Receive} = R'_{Network} + R'_{S(server)}$ gemessen werden. Leider ist diese Messung vom verwendeten Netzwerkpfad und den momentanen Gegebenheiten abhängig und der Wert dadurch mit Vorsicht zu genießen.

Die Zeit $R'_{S(server)}$ ist nach [14] von den in Bild 4.3 abhängigen Faktoren abhängig.

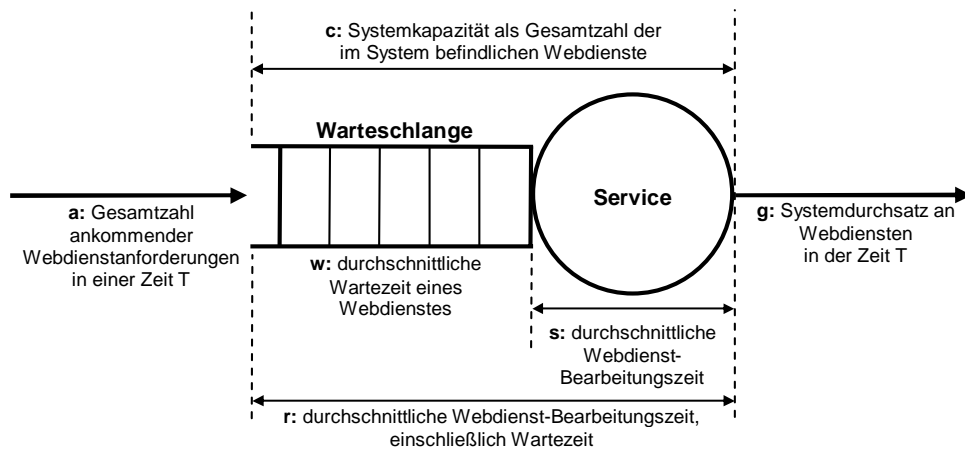


Abbildung 4.3: Performance-Maße eines Webdienstes [14, S.134]

Die Kennzahl a wird auch *Workload* bzw. Arbeitslast und die Kennzahl c Systemkapazität genannt. Darauf aufbauend können weitere Performance-Maße hergeleitet werden, wie zum Beispiel:

- die durchschnittliche Webservice-Rate: $\mu = \frac{1}{s}$
- die durchschnittliche Ankunftsrate der Web-Dienstanforderungen: $\lambda = \frac{a}{T}$
- die Webdienst-Verkehrsdichte¹⁶: $\rho = \frac{\lambda}{\mu} = \frac{a \cdot s}{T}$

Ein anderes Konzept ist die Berechnung der theoretischen Übertragungszeit des Webdokumentes und aller eingebetteten Elemente. Hierzu wird die Gesamtgröße benötigt und für verschiedene Geschwindigkeiten berechnet. Als mögliche Internetanschlüsse könnten zum Beispiel ein 28k-Modem (28800 Bits/s), ein 56k-Modem (56000 Bits/s), ein ISDN-Anschluss (128000 Bits/s) und ein DSL-Anschluss (768000 Bits/s) gewählt werden. Die minimale Übertragungszeit ΔT_{min} für die Geschwindigkeit v könnte mit der Formel

$$\Delta T_{min} = \frac{8}{v} \cdot \sum_{\forall i \in D} \beta_i$$

für die Summe der Dokumentengrößen β aller Dokumente D bestimmt werden¹⁷.

¹⁶ *Traffic Intensity*: Ist ρ größer als 1, erhält der Webdienst mehr Anforderungen als er verarbeiten kann.

¹⁷ Einheiten: ΔT_{min} in Sekunden, β_i jeweils in Bytes, v in Bits pro Sekunde.

4.1.4 Messung des Wartungsaufwands

In der Entwicklung eines Softwareproduktes ist die Wartung der kostspieligste Prozess. Dasselbe gilt auch oder vor allem für Webseiten. Webseiten werden ständig geändert, erweitert und ergänzt.

Herkömmliche Software wird im Normalfall bis zur Marktreife (dem sogenannten *Release Candidate*) im Entwicklungsteam spezifiziert, implementiert und getestet. Im Gegensatz dazu stellt man Webseiten zumeist schon vor dem Fertigstellungszeitpunkt auf den Webserver und erst nach und nach werden alle Funktionen nachgeliefert [14, S.50]. Bild 4.4 zeigt vier verschiedene Arten der Wartung und welche Zeit im Verhältnis zur normalen Entwicklung für die Wartungstätigkeit benötigt wird.

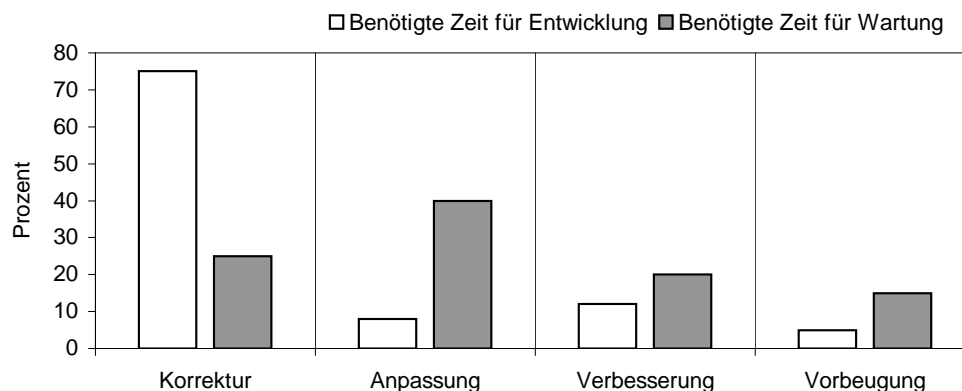


Abbildung 4.4: Wartungstypen und deren Zeitverhältnis [36, S.35]

Die Korrektur beseitigt Fehler im Produkt. Hier sind die Wartungskosten eher gering. Teurer sind dagegen Anpassungen. Hier wird die Funktionalität grundlegend geändert oder an neue Bedingungen angepasst. Dies geschieht, wenn zum Beispiel neue technische Voraussetzungen oder geänderte Rahmenbedingungen auftreten. Verbesserungen dienen der Optimierung von Leistung und Benutzerfreundlichkeit. Vorbeugungen beheben rechtzeitig entdeckte Fehler im Produkt, bevor durch diese Fehlverhalten auftritt.

Änderungen an Webseiten und Verbesserungen können auf Satz- oder Textebene geschehen durch Änderungen am Inhalt oder an der Struktur und Funktionalität der Seite. Doch die meisten Wartungstätigkeiten an Webseiten sind vom WWW aus nicht zu erkennen. Die Webserversoftware muss vorbeugend regelmäßig auf den neusten Sicherheitsstandard gebracht werden. Kleine Änderungen in der HTML-Struktur sind für die Nutzer zumeist nicht erkennbar. Fehlerkorrekturen an den Skripten und Programmen von dynamischen Webseiten sind nicht unmittelbar erkennbar.

Ebenso wie Nutzerakzeptanz und wirtschaftliche Faktoren ist der Wartungsaufwand von Webseiten aus dem WWW nur über Indikatoren messbar. Ein möglicher Indikator für Wartungsarbeiten an der Webseite ist das *Model of Page Change*, welches von [28] vorgeschlagen wurde. Hierbei wird betrachtet, wann eine Änderung im Websystem auftritt. Eine Änderung kann hierbei das Hinzufügen, Ändern oder Löschen einer Webseite sein. Somit können folgende Daten gesammelt werden:

- Der Zeitstempel des Zugriffs auf die Seite durch das Messtool: *visit*
- Der Zeitstempel der letzten Änderung oder falls nicht vorhanden die ausgelieferte Webseite an sich, um sie mit einer älteren Kopie zu vergleichen: *modified*
- Der Zeitstempel, wann eine Seite hinzugefügt wurde: *created*
- Der Zeitstempel, seitdem eine Webseite nicht mehr verfügbar ist: *deleted*

Damit lassen sich folgende Maße berechnen:

- Alter: $visit - modified$
- Lebensdauer *lifespan*: $deleted - created$
- Anzahl der Änderungen während der Lebensdauer: *changes*
- Änderungs-Intervall: $lifespan / changes$

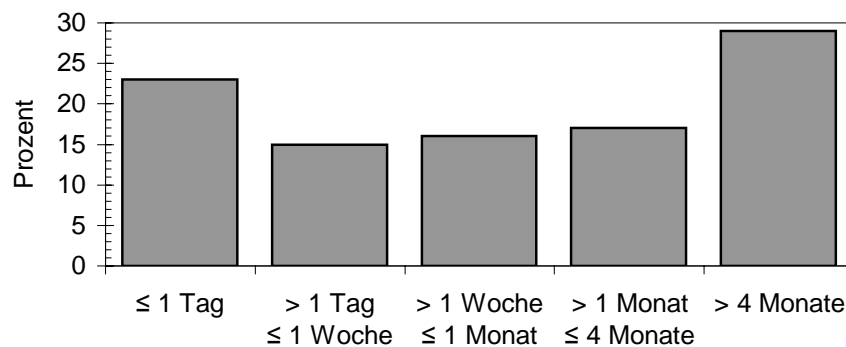


Abbildung 4.5: Durchschnittliches Änderungsintervall von Webseiten [10]

Nach einer Studie von 2000 werden zumindest 70 Prozent aller Webseiten innerhalb von vier Monaten geändert (siehe Bild 4.5). Es wird also Änderungsaufwand betrieben, jedoch kann keine Aussage über die Größe des Aufwands gemacht werden. Die Genauigkeit der Studie wird zudem auch dadurch negativ beeinflusst, dass es immer mehr dynamisierte Webseiten gibt, die ohne Wartungsaufwand allein durch Nutzer der Webseite (z.B. im Forum) oder durch Web Services geändert werden.

Unter gewissen Umständen kann man aus seitenspezifischen Eigenschaften Aussagen über den Wartungsaufwand machen. Eine Seite mit vielen externen Links benötigt zum Beispiel eine regelmäßige Prüfung auf Aktualität der Links. Mehr Informationen zu Messungen der Webseiten-Eigenschaften gibt es im nächsten Abschnitt.

4.1.5 Messung der Webseiten-Eigenschaften

Qualitätseigenschaften können nach ISO 9126 in die Rubriken Funktionalität, Zuverlässigkeit, Effizienz, Benutzerfreundlichkeit, Portabilität und Wartbarkeit unterteilt werden. Die auf [32] basierende Tabelle 4.1 zeigt Eigenschaften, die auf dieser Unterteilung aufbauen:

Rubrik	Eigenschaften
Funktionalität	<p>Suchfähigkeiten in Rubriken / global</p> <p>Navigationsaspekte Navigationspfad vorhanden — momentane Position ausgewiesen</p> <p>Personalisierbarkeit Abbruch der Intro-Sequenz — Globale Deaktivierung von Werbung — An- / Abschaltbarkeit verschiedener Bereiche — Einhaltung von datenschutzrechtlichen Belangen bezüglich der Nutzerdaten</p>
Benutzerfreundlichkeit	<p>Seitenstruktur Sitemap vorhanden — Inhaltsverzeichnis — Navigationshilfen — Anzahl Links</p> <p>Feedback und Hilfen Impressum mit Informationen über Webseitenbetreiber — Antworten auf häufig gestellte Fragen (FAQ) — Feedbackformulare für Kommentare und Fragen / Support — Kennzeichnung des letzten Updates (global und Seite für Seite)</p> <p>Interfacedesign Verwendung von globalen Stylesheets — einheitliches Layout — keine Frames</p> <p>Sonstiges versch. Sprachen — Was ist neu? - Anzeige — Anpassung an aktuelle Bildschirmauflösung</p>
Effizienz	<p>Geschwindigkeit Seitengröße — Global definierte Stylesheet- / Javascript-Datei — Sparsamer Umgang mit Medien — kleine Bilder</p> <p>Zugänglichkeit Browserunabhängigkeit — Barrierefreiheit (Bild-Titel, keine Frames, strikte Trennung von Content und Layout)</p>
Zuverlässigkeit	<p>Fehlerhafte Referenzen Linkfehler (40x) — Ungültige Links (301, 307) — Links mit inzwischen geändertem Inhalt (Handylogo, Warez, Erotikseiten)</p> <p>Inkompatibilität Fehlende Features aufgrund der Wahl anderer Browser — Fehlende Features aufgrund des Deaktivierens von Javascript, ActiveX, Flash, Shockwave, ...</p> <p>Sonstiges unlogische Redundanzen in Navigation — Unerwartetes „Under construction“ — Fehlende Verlinkung in Navigation — Fehlermeldungen von Serverskripten und Datenbank — Javascript-Fehler</p>

Tabelle 4.1: Web-Qualitätseigenschaften nach ISO 9126

In der Tabelle sind keine Eigenschaften zu Wartbarkeit und Portabilität aufgeführt. Wartbarkeit wurde bereits im vorherigen Abschnitt behandelt. Eigenschaften der Portabilität sind aufgrund der Eigenheiten von Webseiten eher unwichtig, da HTML weder an bestimmte Webserver noch an Betriebssysteme gebunden ist. Ausnahmen bilden dynamisierte Webseiten, die beispielsweise die Verfügbarkeit von Webserver-Plugins erfordern.

Viele dieser Eigenschaften sind semantischer Natur und somit nicht messbar. Ein Messtool kann zum Beispiel nicht erkennen, ob eine Webseite ein nach deutschem Recht korrektes Impressum anbietet oder ob die Navigation schlecht konzipiert oder nicht durchgängig ist. Im Zweifelsfall muss also immer ein Mensch die Eigenschaften auf dieser Liste durch Ankreuzen bewerten.

Im Gegensatz zu semantischen Eigenschaften sind syntaktische Eigenschaften oder **Strukturmerkmale** messbar. Strukturmerkmale können aus dem Quellcode der Webseite gewonnen werden. Ein einfaches Beispiel hierfür ist die Anzahl der Bilder, welche in HTML zum Beispiel wie folgt eingebunden werden:

```
<body BACKGROUND="hintergrund.gif">
<table BACKGROUND="muster.jpg"><tr>
  <td><IMG src="bild.png" alt="Skizze des Gegenstandes"></td>
  <td>...</td>
</tr></table>
</body>
```

Der HTML-Parser sucht nun in den Webseiten nach dem Vorkommen dieser syntaktischen Bausteine und zählt somit die Anzahl der Bilder. Eine große Bilderanzahl beeinflusst Geschwindigkeitsmerkmale, wie beispielsweise die benötigte Zeit für den Download sowie den Seitenaufbau und damit die Kundenzufriedenheit. Aus genannten Eigenschaften bildet sich ein Regelwerk, worauf ein Webseiten-Autor achten muss.

Strukturmerkmale können in die Kategorien *Inhalt*, *Interaktion* und *Navigation* unterteilt werden.

- Die Kategorie *Inhalt* umfasst alle Messwerte zu den Webseiten und deren Inhalten.
- *Interaktion* enthält Eigenschaften, mit denen der Nutzer interagieren kann (zum Beispiel Feedbackmöglichkeiten wie E-Mailadressen und Formulare).
- *Navigation* beinhaltet Navigations-Eigenschaften wie Links, Frames, ...

Durch die Analyse der Strukturmerkmale können Aussagen über die in ISO 9126 definierten Qualitätseigenschaften getroffen werden, insbesondere aus den Rubriken

Benutzerfreundlichkeit

Seitenstruktur, Interfacedesign

Effizienz

Geschwindigkeit, Zugänglichkeit

Zuverlässigkeit

Fehlerhafte Referenzen

Wartbarkeit

HTML-Komplexität, Webseitengröße, Änderungsintervall

Es lassen sich empirische Schlüsse ziehen, beispielsweise dass mit der Anzahl der Dokumente in einem Webprojekt auch der Wartungsaufwand steigt. Diese Aussage ist allerdings ungenügend, da eine Bewertung nicht aufgrund einer allgemeinen Beobachtung gebildet werden kann. Es ist nicht möglich für diese Messwerte einen exakten Grenzwert zu nennen, der eine gute von einer schlechten Webseite unterscheidet. Daher ist es notwendig, dass die Grenzwerte variabel sind, um die Messung an die jeweilige Webseite im Rahmen einer Qualitätszielbestimmung anzupassen.

4.1.6 Messung der Standardkonformität

Die syntaktische Form der Webseitenstruktur wird durch Standards festgelegt. Im Gegensatz zu objektorientierten Sprachen mit fester Sprachsyntax ist es nicht zwingend vorgeschrieben, den Quelltext einer Webseite durch ein Tool laufen zu lassen, das die Syntax auf Korrektheit testet. HTML und die damit verbundenen Technologien werden auch fehlerbehaftet von den Webbrowsern akzeptiert. Eventuelle Fehler werden einfach ignoriert oder bestmöglich automatisch korrigiert. Allerdings können sie zu Geschwindigkeitsverlusten beim Seitenaufbau und zu Kompatibilitätsproblemen mit verschiedenen Webbrowsern führen.

In den letzten Jahren des Browserkriegs zwischen Microsoft und Netscape wurde kein Wert auf Standards gelegt und es dominierten proprietäre Lösungen. Im Nachhinein haben Standardisierungsgremien die Neuerungen aufgearbeitet, Fehler und Inkompatibilitäten beseitigt und neue Standards festgelegt. 1998 wurde das *Web Standards Project* (WaSP) gegründet, um die Webstandards populär zu machen und den Browserherstellern die Unterstützung dieser Standards nahezu legen.

Aktuelle Standards für das WWW sind [48]:

Strukturelle Sprachen

- Extensible Hypertext Markup Language (XHTML) 1.0 - <http://www.w3.org/TR/xhtml1>
- XHTML 1.1¹⁸ - <http://www.w3.org/TR/xhtml11>
- Extensible Markup Language (XML) 1.0 - <http://www.w3.org/TR/2000/REC-xml-20001006>

Darstellende Sprachen

- Cascading Style Sheets (CSS) Level 1 - <http://www.w3.org/TR/REC-CSS1>
- CSS Level 2 - <http://www.w3.org/TR/REC-CSS2>
- CSS Level 3 - <http://www.w3.org/Style/CSS/current-work>

Objektmodelle

- Document Object Model (DOM) Level 1 (Core) -
<http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001>
- DOM Level 2 - <http://www.w3.org/DOM/DOMTR#dom2>

Skriptsprachen

- ECMAScript 262 (Javascript) - <http://www.ecma-international.org/publications/standards/Ecma-262.htm>

Zusätzliche darstellende Sprachen

- Mathematical Markup Language (MathML) 1.01 - <http://www.w3.org/1999/07/REC-MathML-19990707>
- MathML 2.0 - <http://www.w3.org/TR/MathML2>
- Scalable Vector Graphics (SVG) 1.0 - <http://www.w3.org/TR/SVG/>

... sowie neu entstehende Standards, wie die für TV- und PDA-basierte Browser

Für viele der genannten Standards gibt es öffentliche Validatoren, die eine Webseite auf Standardkonformität überprüfen¹⁹. Eigenschaften, die Standards betreffen und geprüft werden können, sind nach [47] zum Beispiel:

1. die Verwendung eines korrekten Doctypes aus der Liste
<http://www.w3.org/AQ/2002/04/valid-dtd-list.html>
(Verwendung zur Validierung der Webseite)
2. die Verwendung einer Zeichensatzkodierung
<http://www.w3.org/International/tutorials/tutorial-char-enc/>
(Benötigt zur Vermeidung von unlesbarem Text)
3. die mehrmalige Verwendung von Individualformaten (ID-Attribute)
(Wenn ein Element damit ausgezeichnet wird, sollte der id-Name eindeutig sein im Dokument.)

¹⁸An XHTML 2.0 wird momentan unter <http://w3.org/TR/2004/WD-xhtml2-20040722/> gearbeitet.

¹⁹HTML-Validator - <http://validator.w3.org/>, CSS-Validator - <http://jigsaw.w3.org/css-validator/>
sowie weitere Validatoren unter <http://www.w3.org/QA/Tools/#validators>

4.2 Probleme von Messprogrammen im WWW

Eine Web-Qualitätssicherung hat neben den teilweise begrenzten Möglichkeiten der Informationsgewinnung weitere grundlegende Probleme, die im Zusammenhang mit den Techniken des WWW auftreten. Diese sind bei jedem automatisierten Web-Programm zu beachten, u.a. auch bei den Suchbots von Suchmaschinen. Die Probleme können die Aussagekraft und Qualität der Messergebnisse eines Messtools weiter einschränken.

Probleme bei Messungen im Internet entstehen durch die Komplexität des dem Internet zugrunde liegenden Netzwerks und durch die hohe Dynamik bei der Entstehung und Weiterentwicklung des WWW. Dies betrifft sowohl den normalen Wandlungsprozess des Netzes als auch gezielte Manipulationen der Messprogramme, die darauf abzielen, eine bestimmte Seite besser zu positionieren oder Inhalte zu verbergen.

4.2.1 Variabilität des WWW

Das Netz lebt davon, dass Webseiten untereinander verknüpft werden und somit ein weltumspannendes Netzwerk aufbauen. Leider ändert sich die Struktur der Webseiten ständig und es ist für einen Webseitenbetreiber schwer, seine Webseite aktuell zu halten. Existiert eine verlinkte Webseite nicht mehr, sinkt die Qualität, da dies fehlende Aktualität impliziert.

Um Linkfehler aufzudecken, ist ein Messtool darauf angewiesen, dass beim HTTP-Request an den Webserver der Seite ein 404-Fehler zurückgesendet wird. Dieser Effekt wird automatisch vom Webserver ausgelöst. Der Webserver kann jedoch auch so konfiguriert werden, dass statt des HTTP-Fehlercodes eine spezifische 404-Fehlerseite ausgegeben wird. Das Messtool empfängt somit eine normale HTML-Seite und keinen 404-Fehler. Der Fehler wird nicht erkannt, wodurch die Erkennungsgenauigkeit des Messtools sinkt. Die Qualität einer Webseite wird unter dem Aspekt der Aktualität in diesem Fall als zu gut eingeschätzt.

Eine mögliche Lösung ist die Durchführung einer semantischen Analyse der Webseite. Hierbei wird die Webseite nach Textmustern durchsucht. Die Suche nach dem Wort 404 ist allerdings nicht ratsam, da dieses als Text oder Bild auftreten kann. Statt des Fehlercodes kann auch eine Fehlerbeschreibung im Freitext in jeder beliebigen Sprache ausgegeben werden.

Obwohl spezifische Fehlerseiten das Messergebnis der verlinkten Webseiten verfälschen, haben sie einen positiven Effekt. Sie steigern die Benutzerfreundlichkeit, da auf ihnen zumeist Suchmöglichkeiten und Hinweise stehen, die es ermöglichen, die verlorene Seite wiederzufinden.

Ist im Gegensatz zu lediglich fehlenden Webdokumenten im Webserverpfad der komplette Webserver nicht erreichbar, kann auch keine spezifische Fehlerseite zurückgegeben werden. In diesem Fall sollte immer ein Fehler festgestellt werden. Problematisch ist es jedoch, wenn ein Registrar Anfragen an nicht registrierte Domains zu einer Webseite weiterleitet und keine Fehlermeldungen zurückliefert. So wie es im September 2003 geschehen ist, als der für die .com und .net-Domains zuständige Registrar VeriSign alle fehlerhaften Anfragen an die hauseigene Suchseite SiteFinder weitergeleitet hat. Der Service wurde zwar knapp einen Monat später auf Klage der ICANN und vieler in ihrer Existenzgrundlage bedrohten Firmen wieder abgeschaltet, aber das Beispiel zeigt, welche Auswirkungen eine Veränderung im Netzwerk des WWW bewirken kann [19].

Obwohl es nicht immer möglich ist, fehlerhafte Referenzen in Hyperlinks zu erkennen, müssen auftretende Fehlermeldungen ausgewertet und evaluiert werden. Die Web-Qualitätssicherung bewertet fehlerhafte Verlinkungen in den Maßen *RAT_FAIL_LINKS*, *RAT_FAIL_IMG* und *RAT_FAIL_DOC*.

4.2.2 Spoofing

Spoofing kommt aus dem Englischen und bedeutet soviel wie Manipulation oder Verschleierung.

Ein Client kann seine TCP/IP-Pakete so manipulieren, dass anstatt der eigenen eine fremde IP-Adresse eingesetzt wird (beispielsweise eine auf dem Webserver als vertraut markierte IP-Adresse). Dies wird **IP-Spoofing** genannt. Beim **Browser-Spoofing** wird das HTTP-Feld User-Agent entsprechend so manipuliert, dass der Webserver bestimmte Daten ausgibt, die normalerweise nicht zugänglich sein sollen. **Server-Spoofing** oder **ARP-Spoofing** werden für das Abhören oder Manipulieren einer Kommunikation verwendet. Der Datenverkehr zwischen zwei Hosts in einem Netzwerk wird manipuliert, indem sich ein dritter Host in der Leitung dazwischenhängt und sich gegenüber den beiden als der jeweils andere Kommunikationspartner ausgibt (*Man-In-The-Middle*). Diese drei Verfahren beeinflussen jedoch nicht bzw. kaum die Arbeit eines Messtools, da dieses selber der Client ist und auch keine geheimen Daten übertragen werden, die ein Abhören begründen würde.

Der für Messprogramme eklatanteste Betrugsversuch ist das **Webseiten-Spoofing**. Da Messprogramme sich durch ihre Kennung verraten²⁰, können Webmaster den Inhalt ihrer Seite verschleiern, indem sie alternativen Content ausliefern. Statt einer Hackerseite wird ein Tutorial zum Einrichten von Squid²¹ für Linux ausgegeben. Die Messergebnisse variieren damit natürlich.

Dieses Problem kann man allerdings durch Browser-Spoofing umgehen. Das Messprogramm gaukelt der Webseite schlichtweg vor, es sei ein gängiger Browser, wodurch wieder die Hackerseite ausgegeben wird. Hierzu soll es möglich sein, den Wert des User-Agent-Feldes frei anzugeben²².

4.2.3 Reloadsperre

Ein Messprogramm baut innerhalb kurzer Zeit viele Verbindungen zu demselben Webserver auf. Besitzt der Server eine Reloadsperre, so wird bei zu häufigem Konnektieren alternativer Inhalt angezeigt. Dies ist zumeist eine Warnung mit dem Hinweis, einige Minuten zu warten. Da das Messprogramm die Semantik der Nachricht nicht versteht, wird ein verfälschtes Ergebnis präsentiert. Lösen kann man dieses Problem prinzipiell nur, indem man die Anzahl der Verbindungen zu einem Server zeitmäßig begrenzt bzw. zwischen jedem Verbindungsaufbau einige Sekunden wartet²³.

4.2.4 Differenzierung von Webressourcen

Webseiten werden über URLs identifiziert. Da diese Zuweisung aber nur surjektiv (eindeutig), aber nicht bijektiv (eindeutig) ist, kann dieselbe Webseite mehrmals gemessen werden. Um dies zu vermeiden, könnte der Inhalt der Webseite mit dem Inhalt aller Webseiten verglichen werden bzw. könnten Hashwerte über dem Inhalt gebildet werden, die einen Vergleich ermöglichen. Ein Cachen und Vergleichen von Webseiten wird bei der Web Measurement Suite nicht angewendet, da dies zwangsläufig zu Speicherproblemen auf dem messenden Computer führen würde. Hashwerte werden aufgrund der Laufzeitintensität ebenfalls nicht verwendet, wodurch das Messen von Duplikaten zugunsten der Performance hierbei in Kauf genommen wird.

²⁰z.B. Googlebot (Google), MSNBot (Microsoft) sowie „Web Measurement Suite“ bei diesem Messtool

²¹Squid ist ein Proxyserver.

²²Diese Änderung kann im Messtool im laufenden Betrieb geändert werden unter *Options > Set User Agent...*

²³Diese Änderung kann im Messtool im laufenden Betrieb geändert werden unter *Options > Set Spider Interval...*

Ein anderes Problem im Zusammenhang mit der Differenzierung von Webressourcen, ist die Unterscheidung von verschiedenen geschriebenen URLs, die jedoch auf dasselbe Webdokument referenzieren. Beispielsweise ist `http://www.Server.com:80/` mit der URL `www.server.com` identisch. Gelöst wird dieses Problem, indem jede URL in eine kanonische Form umgewandelt wird. Dieses Vorgehen wird in Abschnitt 6.2.3 auf Seite 83 beschrieben.

4.2.5 Statische Seiten versus Content-Management-Systeme

Statische Seiten sind Webseiten, die unverändert vom Webserver geladen und übertragen werden. Eine Änderung der statischen Seiten wird zumeist direkt im Seitenquelltext durchgeführt, wodurch der Wartungsaufwand steigt. Dagegen sinkt der Wartungsaufwand enorm, wenn es Administrationsoberflächen gibt, mit denen man die Seite bequem von beliebigen Rechnern ändern und ergänzen kann. Dies wird durch die Verwendung von *Content-Management-Systemen* (CMS) ermöglicht.

Im Rahmen der Messung von Webseiten-Eigenschaften wird eine Webseite anhand der Komplexität der HTML-Struktur bewertet. Beispielsweise erhöhen große, stark verschachtelte Tabellen die Komplexität. Während die daraus resultierende, qualitativ schlechte Bewertung bei statischen Seiten in diesem Fall berechtigt ist, ist der Fall bei CMS differenziert zu betrachten. Da hier die einzelnen Absätze, Strukturelemente und Tabellen separat gewartet werden können, ist die Bewertung des gesamten HTML-Quelltextes unter diesem Qualitätsaspekt nicht ausreichend.

Weil die Benutzung von CMS in den wenigsten Fällen von außerhalb des Webserver ersichtlich ist, kann dieses Problem nicht gelöst werden. Eine Überprüfung auf bekannte Muster von gängigen CMS ist nicht erstrebenswert, da diese sich weiterentwickeln und ein ständiges Ändern der Muster bewirken würden. Das Problem wird stattdessen verlagert. Ein Wartungsingenieur erhält die Aufgabe, die Messkriterien auf CMS anzupassen und somit korrektere Ergebnisse herzuleiten.

4.3 Zusammenfassung

„Nicht alles was zählt, kann gezählt werden, und nicht alles was gezählt werden kann zählt.“ (Albert Einstein)

Dieses Zitat kann auch für die Messung von Webseiten angewendet werden. Webseiten kann man über verschiedene Wege bewerten, aber viele Messwerte sind nicht verfügbar und viele besitzen keinen großen Aussagewert bezüglich der Qualität von Webseiten.

Im Folgenden wird noch einmal das Für und Wider von Messverfahren und Messwerten unter Betrachtung verschiedener Qualitätsaspekte zusammengefasst.

Messung der Nutzerakzeptanz und wirtschaftlicher Aspekte

Eine Messung der Nutzerakzeptanz durch Auswertung der Webserver-Logfiles ist nicht möglich, da kein Zugriff auf diese Logfiles besteht. Ebenso betrifft dies die Messung wirtschaftlicher Aspekte, weil diese Aspekte fast ausschließlich durch gezielte Untersuchung der Logfiles gewonnen werden.

Eine indirekte Auswertung der Akzeptanz durch den Verlinkungsgrad per PageRank-Algorithmus wäre theoretisch möglich. Praktisch ist dies jedoch ebenfalls nicht möglich. Um den PageRank eigenständig zu errechnen, müsste eine genügend große Datenbasis vorhanden sein. Diese Datenbasis müsste regelmäßig erneuert und erweitert werden. Dies ist jedoch nur mit großem Aufwand und hohen Kosten möglich.

Das Unternehmen Google bewertet Webseiten intern mit einem geheimen, abgewandelten PageRank-Algorithmus. Der PageRank einer Webseite kann von Google abgerufen werden, indem die Webseite `http://www.google.com/search?client=navclient-auto&ch=[Checksumme]&features=Rank&q=info:[URL]` aufgerufen wird. Hierzu ist eine URL-abhängige Checksumme gefordert. Für die URL `http://www.uni-magdeburg.de/` lautet die Checksumme zum Beispiel 6976286721. Google bewertet die Webseite durch die zurückgelieferte Zeichenkette `Rank.1:1:6` mit dem PageRank 6. Die Skala reicht von 1 bis 10, wobei Werte größer als 5 eher selten und somit als gut zu bewerten sind.

Der Algorithmus zur Berechnung der Checksumme wurde im Juni 2004 ohne das Einverständnis von Google im Internet verbreitet. Eine automatisierte Gewinnung des PageRanks über diesen Weg ist auch für Forschungszwecke verboten. Da zudem der Algorithmus zur Berechnung des PageRanks jederzeit geändert werden kann, wird dieser Ansatz zur Gewinnung des PageRanks nicht weiterverfolgt.

Messung performance-technischer Aspekte

Eine Bewertung der performanceabhängigen Eigenschaften aus technischer Sicht kann aufgrund der Netzwerktopologie nur eingeschränkt getätigt werden. Es kann keine direkte Messung, z.B. der Webdienst-Verkehrsdichte, am Webserver durchgeführt werden und alle Messwerte werden durch das verwendete Netzwerk (die Internet-Router-Struktur) beeinflusst.

In dieser Diplomarbeit wird die Zeit gemessen, die zum Empfangen einer Webressource benötigt wird. Diese Messung ist vom Messort und der Netzwerkauslastung abhängig und somit eher ungenau²⁴ ²⁵. Des Weiteren wird die Größe der Ressource gemessen und die Idealzeiten bei verschiedenen Internetverbindungen werden berechnet²⁶ ²⁷.

Messung des Wartungsaufwands

Eine Messung des Wartungsaufwands ist nur über Indikatoren möglich. Das *Model of Page Change* beobachtet Strukturen von Webseiten und deren Veränderungen. Dadurch kann das Änderungsintervall einer Webseite angegeben werden. Dieses Verfahren kann jedoch nichts über die Größe des Wartungsaufwands aussagen und wird somit nicht weiterverfolgt.

Indirekt kann Wartungsaufwand über Webseiteneigenschaften bewertet werden, zum Beispiel die Größe der Webseitenstruktur, Anzahl der Links, ...

Messung der Eigenschaften der Webseite

Eigenschaften von Webseiten können semantischer und syntaktischer Natur sein. Die Semantik kann nicht automatisiert bewertet werden. Dagegen kann der Quelltext der Webseite auf syntaktische Muster durchsucht werden. Durch Zählen bestimmter Muster kann eine Aussage über verwendete Technologien, Komplexität und damit indirekt Wartungsaufwand und Qualität getroffen werden.

Messung der Standardkonformität

Eine Webseite sollte sich an die offiziellen Standards halten, um somit eine Kompatibilität mit späteren Browsergenerationen und neuen Technologien (PDA, ...) zu gewährleisten. Auch spielen Gesetze eine Rolle, denn z.B. müssen Webprojekte, die Gelder aus öffentlicher Hand bekommen, in Deutschland bis zum 31. Dezember 2005 barrierefrei sein.

Probleme von Messprogrammen im WWW

Die mit Messungen im Internet einhergehenden Probleme sind bestmöglich zu lösen. Die genannten Lösungsvorschläge werden realisiert, indem die Laufzeiteigenschaften des Mess-tools entsprechend angepasst werden. Probleme äußern sich verschieden und erfordern einen regulierenden, manuellen Eingriff menschlicher Nutzer.

²⁴*msr_response* - Messung der Zeit, die zum Empfangen einer Webressource benötigt wird

²⁵*msr_response_all* - Messung der Zeit für Webressource und eingebettete Elemente

²⁶*MSR_LOAD_XX* - Idealzeit für die Ressource bei Geschwindigkeit XX (XX = 28k, 56k, ISDN und DSL)

²⁷*MSR_ELOAD_XX* - Idealzeit für die Ressource und eingebetteten Elemente bei XX

Webseiten bieten leider nicht die Möglichkeit alle Qualitätsaspekte direkt zu messen. Daher müssen Aussagen getroffen werden, indem konkret messbare Eigenschaften auf die Aspekte abgebildet werden.

Beispielsweise lässt sich die Anzahl von Hyperlinks sowohl auf Wartungsaufwand als auch auf Zuverlässigkeit abbilden. Referenzierte Webseiten müssen überprüft werden, um Fehlern vorzubeugen. Wird dieser Wartungsvorgang nicht regelmäßig durchgeführt, sinkt die Zuverlässigkeit und damit auch Benutzerfreundlichkeit und die Nutzerakzeptanz.

Anhang B ab Seite 125 bietet eine komplette Liste verfügbarer Webseiteneigenschaften und Qualitätsmaße. Die Daten werden in Kapitel 6.1 ab Seite 72 hergeleitet.

5 Agentensysteme

5.1 Grundlagen von Software-Agenten

Diese Diplomarbeit beschreibt ein Messtool zur Qualitätsbestimmung von Webseiten. Die Funktionalität der Ressourcenbeschaffung und Auswertung wird hierbei von sogenannten Software-Agenten ausgeführt. Software-Agenten (kurz: Agenten) sind selbständig agierende Objekte oder Komponenten, die ein bestimmtes Ziel anstreben. Agenten existieren in einer abgeschlossenen Umgebung, deren Teil sie sind.

Agenten können in wirtschaftlichem und privatem Umfeld genutzt werden. Ein Beispiel für Agenten sind Biet-Agenten bei Online-Auktionshäusern. Man liefert einen Maximalwert ab, bis zu dem gesteigert werden soll. Der Agent übernimmt nun automatisch das Bieten und erhöht das Gebot immer um die nötige Summe bis hin zum Maximalwert. Eine andere Art von Agent ist bei Suchmaschinen zu finden. Jede Suchmaschine sammelt ihre Informationen via Internet automatisiert mit Hilfe von Agenten - sogenannte Robots oder Webcrawler. Die Agenten beginnen bei einer URL, die sie aus dem WWW herunterladen und nach Informationen (z.B. weitere URLs) durchsuchen.

Agenten haben im Allgemeinen folgende Eigenschaften:

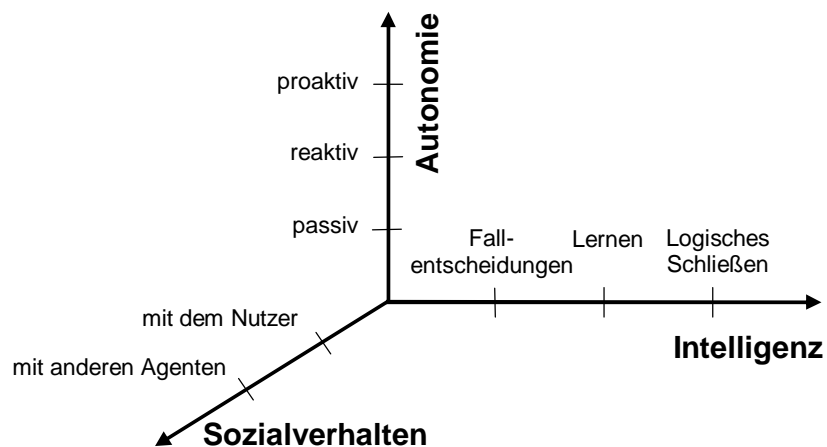


Abbildung 5.1: Eigenschaften von Software-Agenten [13]

In der Regel wird von Agenten gesprochen, wenn sie in irgendeiner Form autonom oder intelligent handeln. In Bezug auf Autonomie ist ein Agent *proaktiv*, wenn er selbständig die momentane Situation auswertet und seine Aktivität steuern kann. Ein *reaktiver* Agent wartet im Gegensatz dazu auf bestimmte Signale, um seine Tätigkeit zu beginnen. Diese Art wertet also die Situation nicht aus. Ist ein Agent *passiv*, muss er manuell von einem anderen Agenten oder vom Nutzer gestartet werden. Die Kommunikation beschreibt das Sozialverhalten des Agenten.

Wie ein Agent eine Lage einschätzt, ist von seiner Intelligenz abhängig. Entscheidet der Agent nach Fallunterscheidungen, befolgt er simpel die Anweisungen des Programmierers, der alle zu behandelnden Fälle fest implementiert und die Handlungsweisen an bestimmte Bedingungen geknüpft hat. Der Agent kann aber auch über eine eigene Logikeinheit verfügen. Optimal ist, wenn ein Agent aus vorhergehenden Entscheidungen lernt bzw. vom Nutzer angelernt werden kann.

Eine weitere Eigenschaft ist die Mobilität - also das Wandern des Agenten in einem Netzwerk von einem Knoten zu einem anderen. Das Gegenteil von mobilen Agenten sind stationäre Agenten. In Bild 5.1 (Seite 53) findet diese Eigenschaft allerdings keine Berücksichtigung, da der Aufenthaltsort der Agenten in dem Messtool allein vom Nutzer beeinflusst wird. Keiner der Agenten wechselt während des Einsatzes seinen Ort.

5.2 Multiagentensysteme

Ein einzelner Software-Agent ist wenig sinnvoll. Erst im Zusammenspiel mit anderen Agenten entfalten sich die Eigenschaften richtig. Die Aufgabengebiete sind in einem System mehreren Agenten zugeordnet. Um das Gesamtziel zu erreichen, müssen deshalb alle zusammenarbeiten. Ein System mit mehreren gekoppelten Software-Agenten nennt man Multiagentensystem (MAS).

Die Arbeit eines Agenten beginnt, wenn er Eingaben (*Inputs*) bekommt oder eine Änderung seiner Umwelt wahrnimmt (*Perceptions*). Diese Eingabe kann durch menschliche Nutzer oder durch andere Agenten vorgenommen werden. Auch ein zeitlicher Impuls, der regelmäßig ausgelöst wird, kann einen Agenten starten. Die Eingangsdaten werden vom Agenten anhand seiner Intentionen²⁸ einer ihm bekannten Zielstellung zugeordnet. Die Lösung der Zielstellung berechnet der Agent auf der Grundlage seines Wissens und unter Zuhilfenahme anderer Agenten und gibt diese dann aus (*Output*) [13, Kap.3.3].

²⁸Intentionen können beispielsweise Erfahrungen sein, wie bestimmte Daten am besten bearbeitet werden.

Ein optimal agierender Agent reagiert zusätzlich auf Einflüsse seiner Umwelt um seine Überlebensfähigkeit zu erhalten und zu verbessern. Ein mobiler Agent könnte zum Beispiel seinen Aufenthaltsort wechseln, wenn der Rechner, auf dem er sich befindet, heruntergefahren wird oder wenn sich die Qualität der Netzverbindung drastisch verschlechtert.

Bild 5.2 zeigt einen universellen Agenten, der durch Eingaben Aktionen oder Berechnungen durchführt und der auch auf Einflüsse aus seiner Umwelt reagiert.

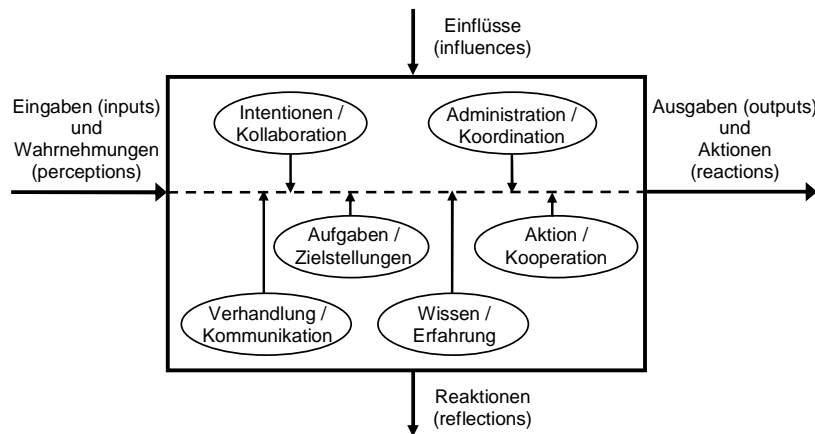


Abbildung 5.2: Komponenten eines universellen Software-Agenten [52]

Ein Agent durchlebt während seines Einsatzes einen Lebenszyklus, welcher in dem Zustandsdiagramm in Abbildung 5.3 dargestellt ist.

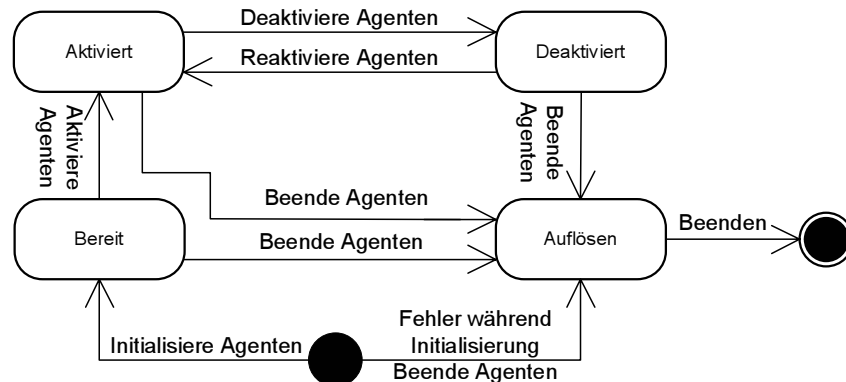


Abbildung 5.3: Lebenszyklus von Software-Agenten [37]

Ein neuer Agent kann vom Agentensystem akzeptiert werden, woraufhin er in den Status *Bereit* wechselt. Treten Fehler während der Initialisierung auf oder wird der Agent abgelehnt (weil zum Beispiel nur ein Agent des Types erlaubt ist), wird der Agent sofort terminiert. Dazu wechselt er in den Status *Auflösen* und beendet sich.

Im Status *Aktiviert* erwartet und bearbeitet der Agent Aufträge. In diesen Zustand wechselt er aus dem Zustand *Bereit* spätestens, sobald der erste Auftrag eintrifft. Um den Agenten kurzzeitig zu deaktivieren, wird eine *Deaktivieren*-Meldung abgegeben. Der Agent pausiert so lange, bis er wieder gestartet oder bis er aufgelöst wird. Dies kann zum Beispiel notwendig sein, wenn ein Konflikt mit einem anderen Agenten gelöst werden muss.

Nach [8] kann man ein Agentensystem in Sektoren einordnen, wie sie in der Abbildung 5.4 dargestellt sind. Die Agenteneigenschaften Autonomie und Sozialverhalten werden vernachlässigt, da hier das Agentensystem als Ganzes angesehen wird.

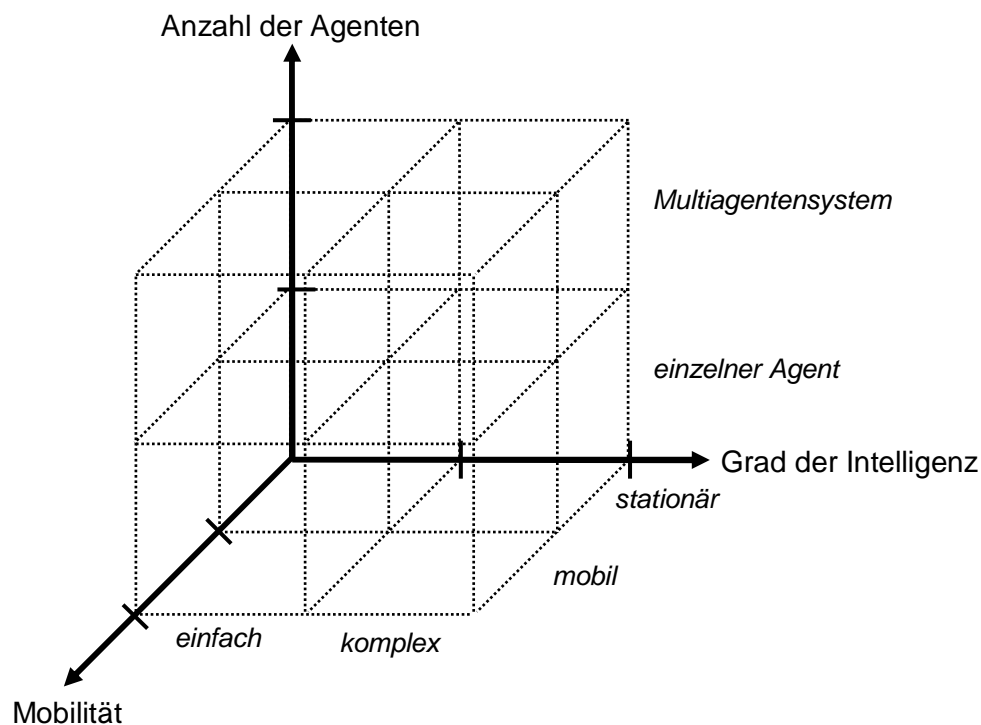


Abbildung 5.4: Die Klassifikationsmatrix eines Agentensystems [8, S.222]

5.3 Objektorientierung des Agentensystems

Die Terminologie der agentenorientierten Programmierung (AOP) definiert strukturelle Elemente und Relationen zwischen Agenten (siehe auch Bild 5.2). Sowohl die Agenten als auch die Daten, die vom Agentensystem bearbeitet werden, können mit einem objektorientierten Ansatz gelöst werden. Tabelle 5.1 stellt zu jeder strukturellen Eigenschaft und Relation in der AOP eine Verknüpfung mit der objektorientierten Programmierung her:

Charakteristik	AOP	OOP
Strukturelle Elemente	generische Rollen	abstrakte Klasse
	domainspezifische Rollen	Klasse
	Wissen, Glaube	Klassenvariable
	Fähigkeiten	Methoden
Relationen	Verhandlungsfähigkeit	Kollaborationen
	Agentengruppierung	Komposition
	Rollenvervielfachung	Vererbung
	domainspezifische Rolle + individuelles Wissen	Instanziierung
	Service-Kombinationen	Polymorphismus

Tabelle 5.1: Vergleich von AOP und OOP [51]

Objektorientierte Programmierung (OOP) ist ein Programmierparadigma. Es dient der Strukturierung von Software, indem Daten und die verarbeitenden Algorithmen und Programmstrukturen als Einheit behandelt werden. Im folgenden Verlauf des Abschnitts werden die Eigenschaften von OOP kurz erklärt und der Zusammenhang zu AOP hergestellt.

Eine *Klasse* ist ein abstrakter Oberbegriff für die Beschreibung der Struktur und des Verhaltens eines Types (Klassifizierung). Eine *abstrakte Klasse* stellt hierbei grundlegende Strukturmerkmale und Verhaltensweisen fest, die von Klassen geerbt und erweitert werden. Eine abstrakte Klasse ist somit immer eine Generalisierung von mehreren davon ererbenden Klassen. Eine Klasse enthält *Klassenvariablen* und *Methoden*, die die Variablen bearbeiten.

Aus Klassen erzeugte Objekte werden *Instanzen* oder Exemplare genannt. Es ist möglich, dass mehrere Instanzen einer Klasse existieren. Sie unterscheiden sich in ihren Klassenvariablen und können somit unterschiedliche Ergebnisse mit ihren Methoden erzielen. Durch Instanziierung ist es möglich, dass es immer mehrere Agenten eines Types geben kann.

Durch *Vererbung* ist es möglich, Agenten mit ähnlichen Eigenschaften zu abstrahieren. Ein abstrakter Agent stellt gemeinsame Fähigkeiten und Wissen bereit. Dieser Agent kann nicht instanziiert werden. Die eigentlichen Agenten spezialisieren nun diesen abstrakten Agenten. Ein Beispiel hierfür ist ein Agent zum Empfangen von Webressourcen. Die Spezialisierung könnte vom verwendeten URI-Schema abhängen - also `http`, `ftp` oder `file`.

Der Begriff *Komposition* stellt in der OOP eine starke Beziehung zwischen Objekten dar. Objekte sind Teile eines Gesamten und können nicht ohne das Ganze existieren. *Polymorphismus* erlaubt es, dass verschiedene Agenten dieselbe Botschaft verstehen, obwohl sich die technische Umsetzung der Datenverarbeitung unterscheidet. *Kollaborierende* Agentensysteme sind durch die Gemeinschaft effektiv in ihrer Aufgabenbewältigung.

Agentensysteme unterscheiden beim objektorientierten Ansatz zwischen agierenden Objekten und Datenobjekten. Die agierenden Objekte, auch Subjekte genannt, erhalten Datenobjekte als Eingabe, die sie getreu ihrer Zielstellung bearbeiten. Datenobjekte können Informationen über die angeforderte Webressource enthalten, beispielsweise die URL, der Inhalt der Webseite oder extrahierte Messwerte.

Abbildung 5.3 (Seite 55) stellt den Lebenszyklus eines allgemeinen, stationären Agenten dar. Daraus kann man nach [37, S.49] folgende Grundoperationen eines Agenten ableiten:

Create Erzeugen eines neuen Agenten und Einführen in die Umgebung

Activate Fortsetzen oder erstmaliges Starten der Agentenfunktion

Deactivate Anhalten der Agentenfunktion

Dispose Anhalten der Agentenfunktion und Entfernen des Agenten aus der Umgebung

Ein zentraler Agent - der Koordinator - verwaltet alle Agenten im System und koordiniert die Kommunikation zwischen den Agenten. Der Koordinator stellt also einen zentralen Punkt dar, durch den die Verwaltung vereinfacht wird. Die Alternative wäre, dass jeder Agent sich mit jedem Agenten verbindet und sich und seine Kommunikationspartner selbständig verwaltet.

5.4 Der Web-Tomograph WebTomix

Ein konkretes Beispiel für ein Agentensystem ist WebTomix, welches im Rahmen einer Diplomarbeit von Uwe Schäfer in [37] entwickelt wurde.

Tomographie ist eine schichtweise Untersuchung eines Objektes bzw. dessen gemessener Attribute. Im Bereich der Medizin ist die Tomographie unter anderem als Röntgenschnittverfahren bekannt, welches im Gegensatz zu herkömmlichen Summationsröntgenbildern eine detaillierte Bildgewinnung von einzelnen Sektionen zum Ziel hat. Das verwendete Werkzeug ist ein Tomograph [37].

Bei einer Web-Tomographie wird versucht, aus gewonnenen Querschnitten eine Diagnose über das WWW zu erstellen. Diese kann zu Hilfe genommen werden, um Entscheidungen zu treffen oder den aktuellen Marktdurchschnitt zu bewerten.

Der Web-Tomograph WebTomix sammelt im Internet Stichproben und trifft mit diesen Messproben eine Aussagen über verwendete Technologien. Die Stichproben werden durch die Eingabe von beliebigen WWW-Adressen sowie durch Hyperlinks auf der Webseite gesammelt. Durch diese rekursive Fortbewegung im WWW ist WebTomix ein *Webcrawler* bzw. *Webspider*.

5.4.1 Arten von Technologien und ihre Merkmale

Die von WebTomix gemessenen Technologien werden in Tabelle 5.2 nach verschiedenen Aspekten gruppiert. Diese Gruppierung richtet sich beispielsweise nach Form und Ort der Anwendung der Technologie. Die entstehende Unterteilung in den Gruppen ist nicht zwingend disjunkt, so dass konkrete Techniken mehreren Gruppen zugeteilt werden können.

Viele Webdokumente verwenden Technologien mehrerer Gruppierungen oder verwendete Technologien einer Gruppe bauen auf Techniken einer anderen Gruppe auf. Zum Beispiel die Technologie *HTTPS* ist eine Verknüpfung von *SSL* und *HTTP*, da als Verbindungsprotokoll weiterhin *HTTP* verwendet wird, deren Pakete aber durch *SSL* verschlüsselt und somit gesichert werden.

Technologie-Gruppe	Beispiele
Servertechnologien	SSI, PHP
Clienttechnologien	Flash, JavaScript
Beschreibungssprachen	HTML, SMIL, VRML
Dateiformate	PDF, GIF
Dynamik, Interaktion	Flash, CGI, SSI, PHP, Javascript
Kommunikation	HTTP, HTTPS, WAP, SSL
Datenschutz und Authentizität	SSL
Filtersysteme	PICS
Onsite Search Engines	HtDig
Entwicklungswerkzeuge	FrontPage

Tabelle 5.2: Einteilung der Webtechnologien [37, S.26]

Die Liste ist bei weitem nicht vollständig. Deshalb ist WebTomix so flexibel gestaltet, dass problemlos neue Technologien hinzugefügt werden können.

Technologien erkennt WebTomix über:

Muster im Inhalt

<html>, <!DOCTYPE HTML, <?xml, <!DOCTYPE wml, #VRML

Muster am Anfang der Datei

%PDF, %!PS-Adobe, GIF8, JFIF

Einbettung im Quelldokument

, <script language="Javascript">...</script>

Dateiendungen

htm, html, xml, wml, wrl, doc, pdf, ps, gif, wav, ...

Akzeptanz durch speziellen Parser

Trial-and-Error-Verfahren mit verschiedenen Parsern; z.B. XML-Parser

MIME-Typen

text/html, image/jpeg, image/gif, model/vrml

HTTP-Response-Header

HTTP Response Code 401 Unauthorized

... und weitere Verfahren²⁹ ...

5.4.2 Erkennungskette

Damit alle verwendeten Technologien in einem Webdokument erkannt werden, kombiniert WebTomix verschiedene Erkennungsverfahren. Die Ergebnisse eines ersten Erkennungsverfahrens werden durch die Ergebnisse eines zweiten bestätigt, erweitert oder sogar widerlegt, wenn das zweite Verfahren eine höhere Erkennungsgenauigkeit besitzt. Die Erkennungsgenauigkeit wird durch eine Dezimalzahl gekennzeichnet. Ist die Summe der Erkennungsfaktoren für eine Technologie positiv, wird diese der Webressource zugeordnet. Negative Zahlen schließen eine Technologie aus.

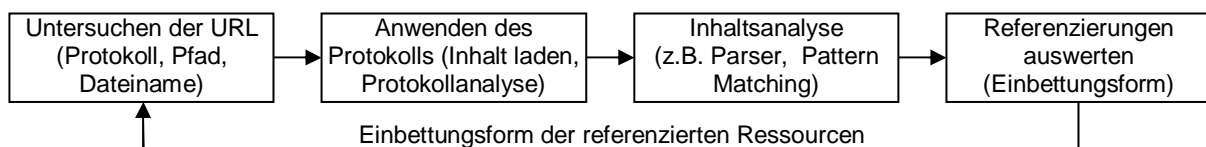


Abbildung 5.5: Erkennungskette von Webtechnologien in WebTomix [37, S.45]

²⁹Für detailliertere Informationen sei auf die Diplomarbeit von Uwe Schäfer [37] verwiesen.

Die Kette der Erkennungsverfahren ist auch von der Reihenfolge abhängig (siehe Abbildung 5.5). Weiter hinten liegende Verfahren können Ergebnisse revidieren und sind zumeist sichere Verfahren, die aber zugleich komplexer in Zeit und Aufwand sind (z.B. Parser). Die vorderen Verfahren stellen eine Vorauswahl in Bezug auf die Technologiegruppe her und beeinflussen somit den Verlauf der Messung. Durch die vorderen Verfahren wird entschieden, ob und wie eine Ressource ausgewertet wird (herunterladen, parsen und auswerten).

5.4.3 Das Agentensystem von WebTomix

Da die Web Measurement Suite in ihrem Agentensystem auf WebTomix basiert, ist ein Hauptaugenmerk auf dessen Agenten gerichtet.

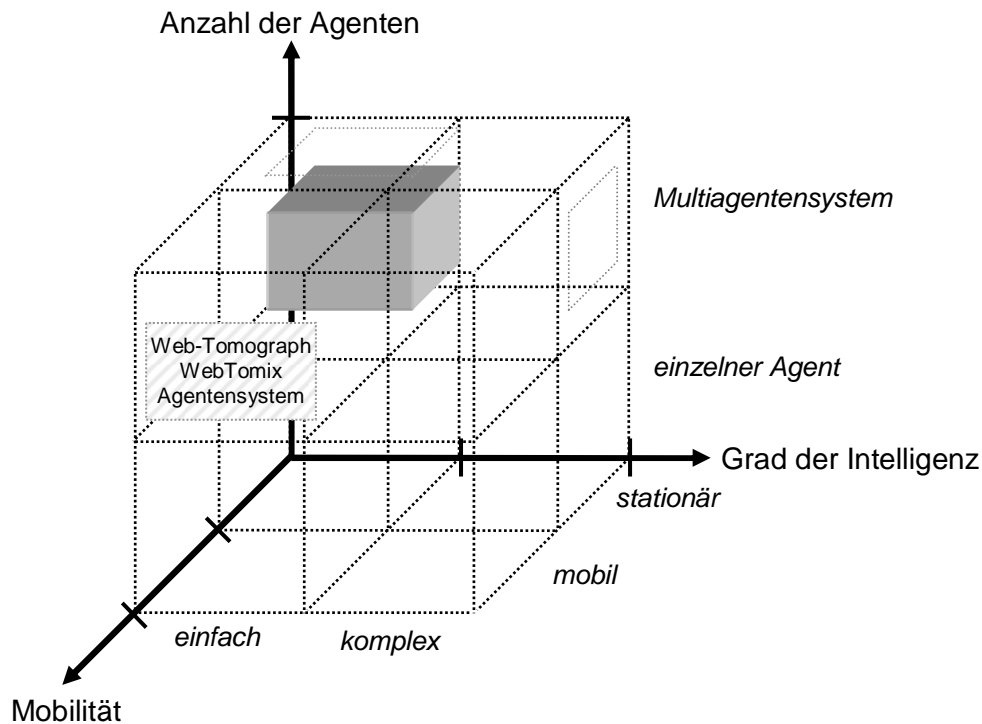


Abbildung 5.6: Klassifikation des Agentensystems von WebTomix

In die Klassifikationsmatrix von [8] eingeordnet (siehe Abbildung 5.4, Seite 56), kann das Agentensystem WebTomix als einfaches, stationäres Multiagentensystem klassifiziert werden.

Es besitzt nicht die Möglichkeit seine Agenten wandern zu lassen. Deshalb erfolgte die Einordnung des Systems in die Sektoren „stationär“. Der Grad der Intelligenz ist als „einfach“ klassifiziert, da WebTomix einzig Technologien zählt, ohne die Messergebnisse auszuwerten oder zu visualisieren. Diese Aufgabe wurde dem Menschen überlassen.

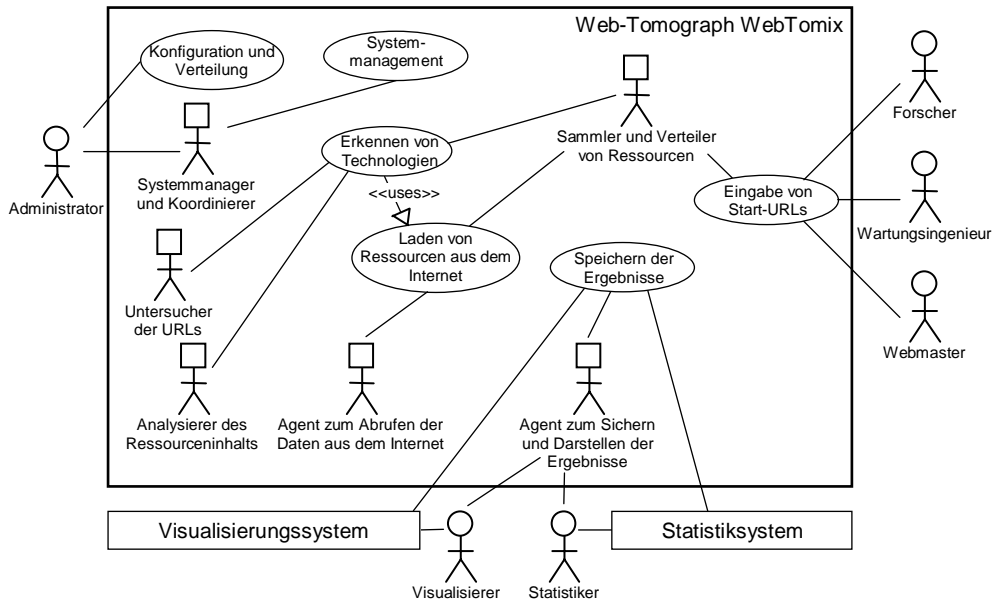


Abbildung 5.7: Das Agentensystem von WebTomix [37, S.54]

Abbildung 5.7 zeigt das Agentensystem des Web-Tomographen mit möglichen Anwendungsfällen. Man erkennt die Abhängigkeit von den Nutzern (Forscher, Wartungsingenieur und Webmaster) und die Auslagerung der Auswertung und Visualisierung auf externe Software.

5.4.4 Die Benutzerschnittstelle von WebTomix

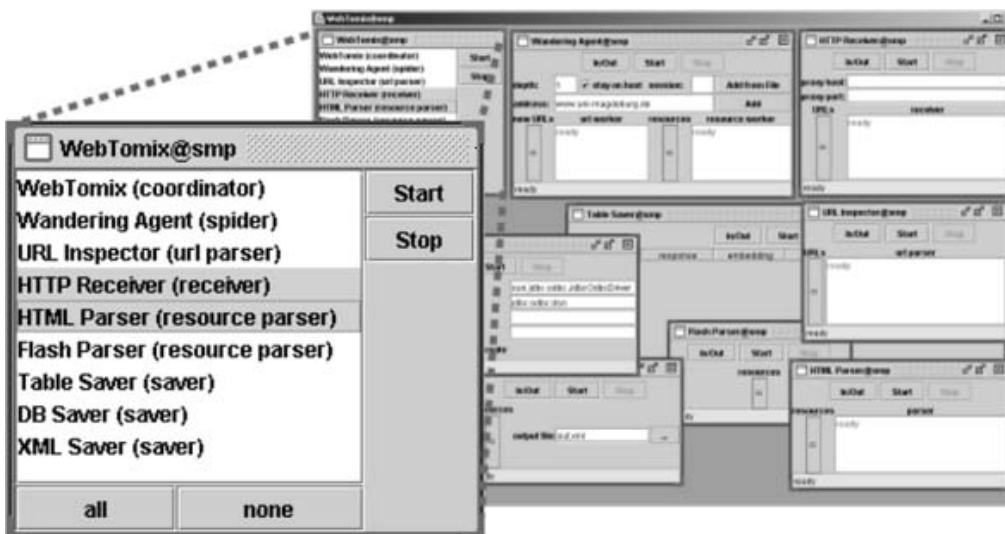


Abbildung 5.8: Die Benutzerschnittstelle von WebTomix [37, S.88]

WebTomix besteht aus einem Fenster mit mehreren internen Fenstern. Abbildung 5.8 zeigt das Hauptfenster von WebTomix. Hervorgehoben ist die aktuelle Agentenliste. Es ist möglich, gemäß Abbildung 5.3 (Seite 55) einzelne Agenten anzuhalten und wieder zu starten. Jeder Agent baut ein internes Fenster im Hauptfenster auf, über das der Agent gesteuert werden kann.

5.4.5 Anwendungsbeispiele von WebTomix

Ein Anwendungsbeispiel ist in Abbildung 5.9 zu sehen. Hier wurde die Domain der Universität Magdeburg <http://www.uni-magdeburg.de/> mit einer Suchtiefe von 3 auf Technologien untersucht. Das Ergebnis ist im Balkendiagramm zu erkennen.

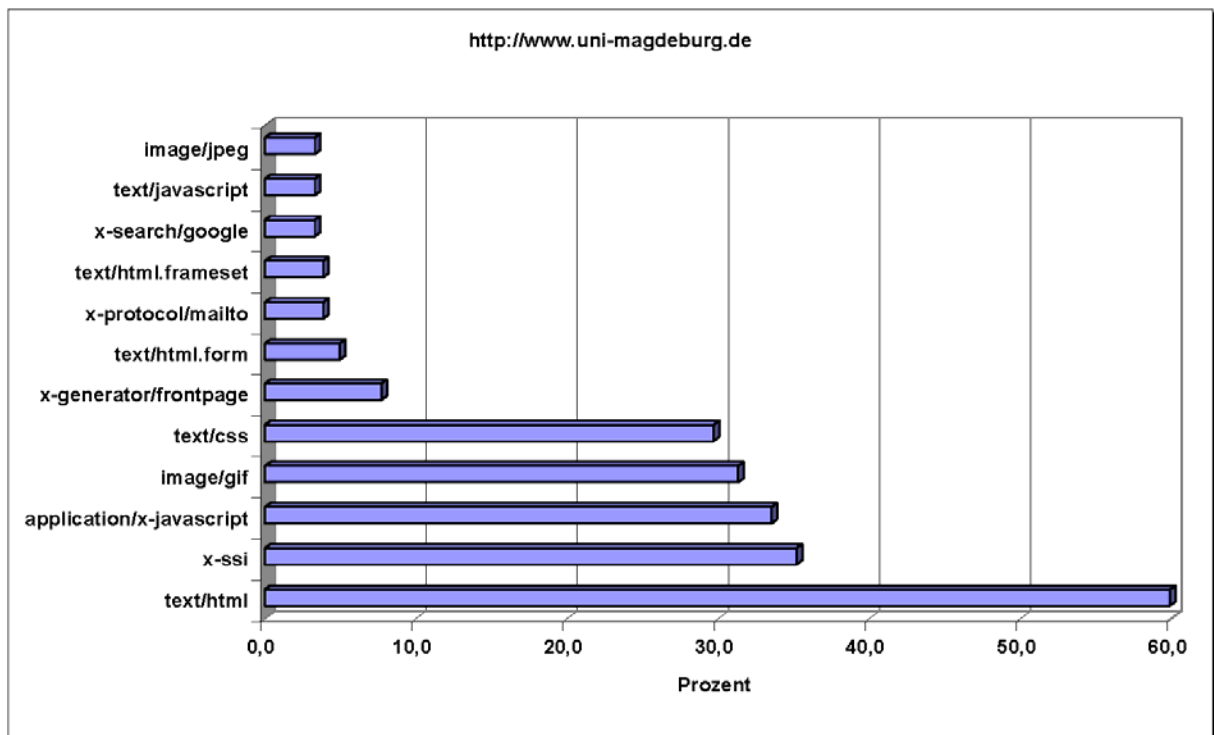


Abbildung 5.9: WebTomix: Ergebnis der Messung von www.uni-magdeburg.de [37, S.76]

Bei Verwendung des Web-Tomographen WebTomix sind Aussagen über den Einsatz verschiedener Technologien zu treffen. Die Universität Magdeburg verwendet primär HTML mit Server Side Includes zur Dynamisierung, Javascript und CSS, einige Formulare und die On-Site-Search-Engine von Google.

Erst richtig interessant wird WebTomix beim Vergleich verschiedener Webauftritte. Hierzu sei auf das Kapitel 7 in [37] verwiesen, welches verschiedene Stadtauftritte und Universitätsseiten vergleicht.

5.5 Das Agentensystem der Web-Qualitätssicherung

Ein Tool zur Bestimmung der Webseiten-Qualität muss die Fähigkeit besitzen, eine Webseitenstruktur zu erkennen und einen Messplan zu erstellen, wie diese Webstruktur optimal erfasst werden kann.

Messwerte sollen sowohl direkt aus dem Quelltext einer Webseite entnommen als auch während der Messung gesammelt werden, wie benötigte Zeit, eventuelle Fehler, ...

Konflikte beim Empfang der Webseiten sollen bewältigt werden und eine doppelte Bearbeitung derselben Webresource bei ähnlicher URL ist möglichst auszuschliessen.

5.5.1 Generische Rollen des Agentensystems

Generische Rollen und damit abstrakte Agenten-Klassen sind in Abbildung 5.10 zu finden.

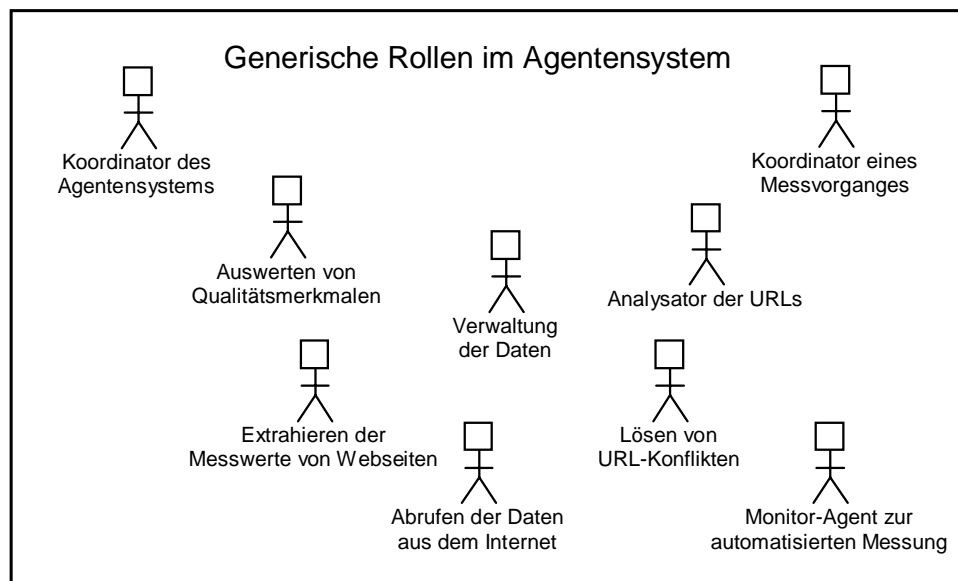


Abbildung 5.10: Generische Rollen im Messtool

Im Folgenden sind die einzelnen generischen Rollen mit den Eigenschaften der Agenten und einer kurzen Betätigungsbeschreibung aufgeführt. Der Name der generischen Rolle ist zweigeteilt. Der erste Teil ist eine beschreibende Rollenbezeichnung, wie sie in Abbildungen zum besseren Verständnis benutzt wird. Der zweite Teil ist der konkrete Name der abstrakten Klasse in der Implementation (basierend auf [37]).

Koordinator des Agentensystems

Name der generischen Rolle

Agent Coordinator / Coordinator

Agenteneigenschaften

reaktiv, kommuniziert mit anderen Agenten, Fallentscheidungen

Beschreibung

- zentraler Anlaufpunkt für alle Agenten
- Verwaltung der Verbindungen mit den Agenten
- Vermittlung zwischen den Agenten
- Schnittstelle des Agentensystems zum Nutzer,
keine direkte Kommunikation,
stattdessen Anforderung der kommunizierenden Agenten

Koordinator eines Messvorganges

Name der generischen Rolle

Measurement Coordinator / Wanderer

Agenteneigenschaften

proaktiv, kommuniziert mit anderen Agenten und dem Nutzer, logisches Schließen

Beschreibung

- Verwaltung der Messvorgänge
- Erkennen von Deadlocks (festgefahrenen Messvorgängen)
- Verteilung der Arbeit an spezialisierte Agenten
- Lastverteilung bei mehreren Agenten eines Types

Analysator der URLs

Name der generischen Rolle

URL Parser / URLInspector

Agenteneigenschaften

passiv, kein Sozialverhalten, Fallentscheidungen

Beschreibung

- Extraktion von Eigenschaften einer Webresource aus der URL

Lösen von URL-Konflikten

Diese generische Rolle ist später innerhalb eines Verbundes „intelligenter“ Objekte integriert. Sie hat keine Verbindung zum Koordinator des Agentensystems und besitzt auch nicht die für Agenten typischen Eigenschaften (siehe Abbildung 5.3, Seite 55). Der Name der generischen Rolle entfällt, da sie Teil des selbstverwaltenden Datenspeichers ist (siehe generische Rolle *Verwaltung der Daten*). Die Problematik ist in Kapitel 6 im Abschnitt 6.2.3 (Seite 81) erläutert.

Agenteneigenschaften

reaktiv, kein Sozialverhalten, Lernen

Beschreibung

- Duplikaterkennung zweier ähnlicher URLs
- Erkennen von bereits gemessenen Webressourcen

Abrufen der Daten aus dem Internet

Name der generischen Rolle

Receiver / Receiver

Agenteneigenschaften

passiv, kommuniziert unter Umständen mit dem Nutzer ansonsten kein Sozialverhalten, Fallentscheidungen

Beschreibung

- Verbindungsaufbau zum Webserver über HTTP oder HTTPS
- Anforderung der Webseite / eines Dokumentes
- Messung von beispielsweise Download-Zeit
- Abfrage der Authentifikationsdaten vom Nutzer wenn erforderlich

Extrahieren der Messwerte von Webseiten

Name der generischen Rolle

Data Extractor / ResourceParser

Agenteneigenschaften

passiv, kein Sozialverhalten, logisches Schließen

Beschreibung

- Parsen der Syntax einer Ressource
- Extrahieren von Messwerten aus dem Inhalt

Auswerten von Qualitätsmerkmalen

Diese Rolle benötigt ebenfalls nicht die für Agenten typischen Eigenschaften (siehe Abbildung 5.3 auf Seite 55). Sie greift direkt auf den Datenspeicher zu (siehe *Verwaltung der Daten*). Ein konkretes Sozialverhalten ist nicht vorhanden, da sich die Zugriffe nur auf Methodenaufrufe der Speichereinheiten beziehen.

Name der generischen Rolle

Result Analyser / ReportFactory

Agenteneigenschaften

passiv, kein Sozialverhalten, logisches Schließen

Beschreibung

- Erfassen der Menge aller gemessenen Webressourcen
- Bewertung der Messwerte durch Qualitätsmaße
- Einstufung der Qualität einer Webseite
- Erzeugung eines Qualitätsreports

Verwaltung der Daten

Diese generische Rolle ist später als Verbund „intelligenter“ Objekte realisiert. Miteinander verknüpfte Objekte verwalten alle Daten.

Einen konkreten Agenten wird es für diese Rolle nicht geben, deshalb entfällt auch der Name der generischen Rolle.

Agenteneigenschaften

passiv, kein Sozialverhalten, Fallentscheidungen

Beschreibung

- Verwaltung verschiedener Messprojekte
- Unterstützung von Messreihen (d.h. Wiederholung einer Messung)
- Verwaltung und Speicherung aller Daten
- Bereitstellung der Messwerte und Messdaten
- Lösen von Konflikten durch Duplikate (siehe *Lösen von URL-Konflikten*)

Monitor-Agent zur automatisierten Messung

Name der generischen Rolle

Monitoring Actuator / Monitor

Agenteneigenschaften

proaktiv, kommuniziert mit anderen Agenten und dem Nutzer, Lernen

Beschreibung

- Überwachung einer Webseite
- Periodische Messung von Webseiten
- Aufstellung von Messreihen
- Bewertung der Messung
- Benachrichtigung des Nutzers

Der Monitor-Agent nimmt eine eigenständige Position ein. Er ist kein Bestandteil des Kernagentensystems. Die Monitoring-Funktion erstellt ein eigenständiges Agentensystem, indem ein zweiter Agenten-Koordinator die benötigten Agenten verwaltet.

Dieser Agenten-Koordinator und der des Kernsystems kommunizieren miteinander und erlauben so eine Synchronisation auf der gemeinsamen Datenbasis.

5.5.2 Klassifikation des Agentensystems

Das gesamte Agentensystem kann, wie in Bild 5.11 dargestellt ist, als mittelmäßig komplexes, stationäres Multiagentensystem klassifiziert werden.

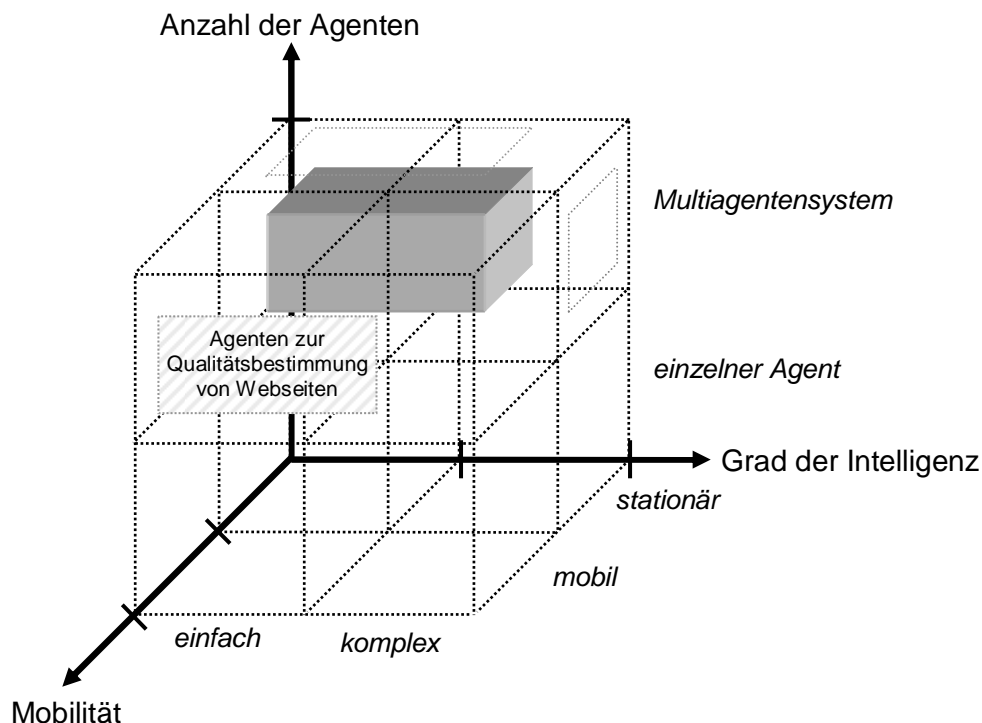


Abbildung 5.11: Die Klassifikationsmatrix des Messtool-Agentensystems

Die Web-Qualitätssicherung baut auf dem Agentensystem von WebTomix auf. Deshalb kann die Einordnung in die Sektoren „stationär“ und „Multiagentensystem“ unverändert übernommen werden.

Der Grad der Intelligenz wurde als ein wenig höher eingestuft, da im Gegensatz zu WebTomix eine mögliche Auswertungskomponente integriert ist, die zudem die Messergebnisse in Form eines Qualitätsreports visualisiert. Die Web-Qualitätssicherung verhält sich durch den Monitor-Agenten in einem gewissen Grad autonom, wodurch die Klassifizierung als „mittelmäßig komplexes“ System gerechtfertigt ist.

5.6 Zusammenfassung

Durch die Verwendung von Software-Agenten in der Web-Qualitätssicherung ist es möglich, die komplexen Vorgänge bei der Messung von Webseiten auf kleine Teilaufgaben aufzuteilen. Jeder einzelne Agent bearbeitet ein stark abgegrenztes Aufgabengebiet, wodurch das Multiagentensystem leicht erweitert werden kann. Die einzelnen Agenten sind wartbarer, da durch klar definierte Schnittstellenspezifikationen die internen Berechnungsmethoden problemlos ausgetauscht werden können. Ebenso ist eine räumliche Verteilung gewährleistet, wobei dies natürlich den Aufwand für den Transport der Daten zwischen den Agenten stark erhöht.

Mit WebTomix ist eine leistungsstarke und einfach zu erweiternde Agentenplattform gegeben, die es der Web-Qualitätssicherung erlaubt, durch Adaption und Erweiterung den nötigen Leistungsumfang zu erreichen.

Der Aspekt der mobilen Agenten ist sowohl bei WebTomix als auch bei der Web-Qualitätssicherung konzeptionell nicht eingebunden. Die Möglichkeit bestünde, dass Agenten im Internet zwischen Messplattformen hin- und herwandern, um ein und dieselbe Webseite aus verschiedenen Perspektiven (unterschiedliche Netze) zu betrachten. Die dazu erforderlichen technischen Bedingungen sind jedoch nicht gegeben, weshalb dieser Ansatz nicht weiterverfolgt wird.

6 Entwicklung einer Web-Qualitätssicherung

Dieses Kapitel beschäftigt sich mit der Entwicklung der Web-Qualitätssicherung *Web Measurement Suite* (WebMS). Abschnitt 6.1 ab Seite 72 leitet die verfügbaren Qualitätseigenschaften her und spezifiziert ein Qualitätsmodell, welches als Basis für eine Bewertung genommen wird.

In Abschnitt 6.2 ab Seite 75 werden Anforderungen an die Web Measurement Suite gestellt und modelliert. Dieser Abschnitt beschreibt demnach die Spezifikation und den Entwurf der Software.

Die dem System zugrunde liegende Agentenstruktur basiert auf der Diplomarbeit von Uwe Schäfer [37], der im Rahmen des Web-Tomographen *WebTomix*³⁰ ein in Java implementiertes, einfach erweiterbares Agentensystem entwickelt hat. Die Änderungen am Agentensystem von WebTomix behandelt Abschnitt 6.3 ab Seite 101. Hierzu werden zu den in Kapitel 5 definierten generischen Agentenrollen Anwendungsfälle konstruiert und das Agentensystem konzeptionell aufgebaut.

Die Web Measurement Suite erfordert neben WebTomix die Einbindung verschiedener Komponenten von Drittanbietern. Abschnitt 6.4 ab Seite 106 beschreibt fremde Komponenten innerhalb der Web-Qualitätssicherung und fasst sie in einer Abbildung zusammen, die die Software-Struktur des Messprogramms zeigt.

Die Nutzeroberfläche des Tools findet in dieser Diplomarbeit keine gesonderte Erwähnung. Sie basiert auf der Java Swing-Klassenbibliothek und ist modular aufgebaut. Jeder Bereich in der Oberfläche ist in einem eigenen Java-Paket zusammengefasst, wodurch die Deaktivierung und das Hinzufügen von Bereichen ermöglicht wird. Zentralisiert ist der Knotenpunkt, der alle Pakete aufruft und zu einer Nutzeroberfläche zusammenbaut. Diese Hauptklasse steuert den Zugriff der Bereiche untereinander.

³⁰WebTomix wurde in Abschnitt 5.4 erläutert, ohne jedoch ins Detail zu gehen.

6.1 Herleitung der Messwerte und Qualitätsmaße

Die Web Measurement Suite bietet in Anhang B für die Bestimmung der Qualität von Webseiten eine Liste von Messwerten und Qualitätsmaßen (*factors and metrics*). Per Definition werden Messwerte während des Messvorgangs einer Webseite gesammelt und gehen als Faktoren in die Berechnung der Qualitätsmaße ein. Eine Bewertung erfolgt ausschließlich auf Basis der Qualitätsmaße.

Sowohl Messwerte als auch Maße werden innerhalb der Diplomarbeit als mnemonischer Ausdruck der Form `[PRE]_NAME_OF_THE_METRIC` bezeichnet. Messwerte werden hierbei klein und Qualitätsmaße groß geschrieben. Der Präfix `[PRE]` ergibt sich aus Tabelle 6.1, wobei `msr` bzw. `MSR` im Allgemeinen dann gewählt wird, wenn kein anderer Präfix zutrifft.

Präfix	Beschreibung
cnt	Bildung des Messwertes durch Zählung einer Eigenschaft
max	Maximaler Wert einer Eigenschaft
msr	Allgemeiner Messwert
sum	Summe von messbaren Eigenschaftswerten
AVG	Durchschnitt von Messwerten
CNT	Zählbares Maß oder direkte Bewertung eines cnt-Messwertes
MAX	Direkte Bewertung eines max-Messwertes
MSR	Allgemeines Qualitätsmaß
RAT	Verhältnis zwischen Messwerten
SUM	Summe von Messwerten

Tabelle 6.1: Präfixe von Messwerten und Qualitätsmaßen

Das verwendete Qualitätsmodell in Abbildung 6.1 baut auf den in der ISO 9126 spezifizierten Qualitätsmerkmalen auf und erweitert diese. Aus der ISO 9126 wurden folgende Qualitätsmerkmale übernommen: **Zuverlässigkeit**, **Effizienz**, **Benutzbarkeit**, **Wartbarkeit**, **Portabilität**. Funktionalität ist nur über die Semantik einer Webseite messbar und wird somit ausgeblendet. Neben den in ISO 9126 definierten Qualitätsmerkmalen werden zusätzlich **Fremdinhalte** sowie **Nutzerakzeptanz** bewertet.

Fremdinhalte betrifft die Verwendung von rechtlich geschütztem Inhalt und die Verursachung von Kosten bei Dritten. Dies geschieht beispielsweise, wenn Bilder oder Dokumente von fremden Webservern eingebunden werden. Referenzen zu Bildern und Dokumenten auf fremden Webservern zählen die Messwerte `cnt_ext_img` und `cnt_ext_doc`, die in den Maßen `RAT_EXT_IMG` und `RAT_EXT_DOC` als Verhältnis zur Gesamtzahl gesetzt und bewertet werden.

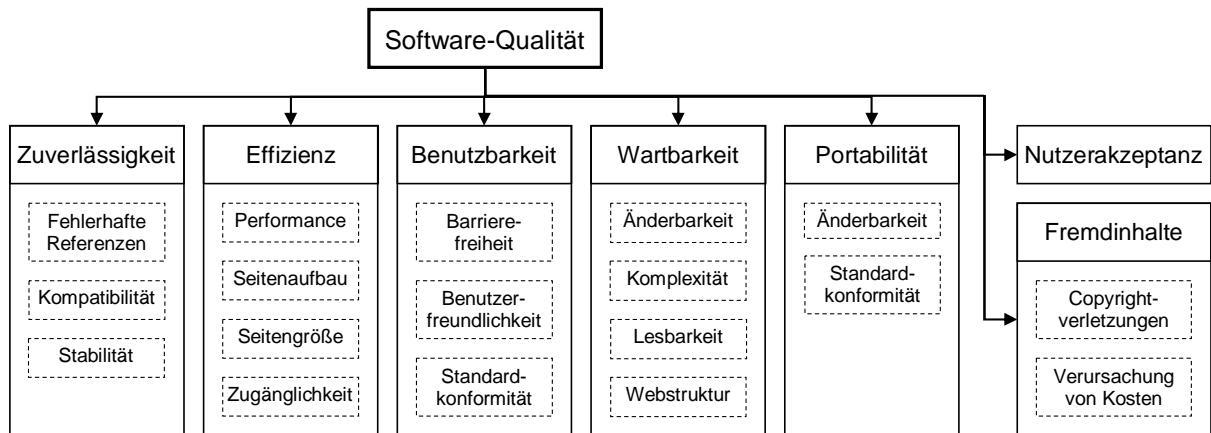


Abbildung 6.1: Qualitätsmodell der Web Measurement Suite

Nutzerakzeptanz ist laut Abschnitt 4.1.1 ab Seite 33 kaum messbar, da Nutzungslogfiles nicht vorliegen und eine Bewertung über PageRank nicht möglich ist. Es kann jedoch die Verwendung von Werbung nachgewiesen werden und damit einhergehende Qualitätseinbußen in der Nutzerakzeptanz unter der Annahme, dass übermäßige Werbung Nutzer verschreckt. Die URL eines Hyperlinks oder eines Bildes wird auf Werbe-Pattern aus Anhang C untersucht. Die Messwerte *cnt_adv_links* und *cnt_adv_img* zählen die Häufigkeit des Vorkommens und werden in den Maßen *RAT_ADV_LINKS* und *RAT_ADV_IMG* als Verhältnis zur Gesamtzahl externer Referenzen gesetzt und bewertet.

Messwerte und Maße zu **Zuverlässigkeit**, **Effizienz**, **Benutzbarkeit**, **Wartbarkeit** und **Portabilität** sind in vollem Umfang in Anhang B ab Seite 125 aufgeführt.

Messwerte, die im Zusammenhang mit der Größe der Webseite stehen, müssen normalisiert werden. Dies sind beispielsweise die Anzahl der Tabellen (*CNT_TABLES*), der Formulare (*CNT_FORMS*) und der eingebetteten Elemente (*CNT_INLINE*). Als Normalisierungsfaktor *MSR_NORM* wird das Verhältnis der Webseitengröße zu 10 KByte gebildet.

Der Normierungsfaktor wird nur dann verwendet, wenn eine Webseite größer als 10 KiloByte ist. Bei kleineren Dokumenten würde er beispielsweise die Anzahl der Tabellen vervielfachen und damit das Verhältnis zu Ungunsten der Webseiten verzerren.

Die Verwendung des Normierungsfaktors ist in Anhang B mit einem „normiert“ in der Beschreibung gekennzeichnet.

Die Messwerte, die den ISO 9126-Qualitätsmerkmalen zugeordnet sind, basieren zumeist auf syntaktischen Eigenschaften bzw. **Strukturmerkmalen**. In Abschnitt 4.1.5 auf Seite 43 wurden diese Strukturmerkmale in die Kategorien *Inhalt*, *Interaktion* und *Navigation* unterteilt. Tabelle 6.2 zeigt einige Messwerte und Qualitätsmaße, die durch Strukturanalyse gesammelt werden können.

Kategorie	Eigenschaften
Inhalt	<ul style="list-style-type: none"> - #Dokumente (<i>CNT_PAGES</i>) - Größe der HTML-Tags (<i>msr_html</i>) — Größe des Inhalts (<i>msr_content</i>) - Größe der Webseite (<i>SUM_SIZE</i>) — mit Bildern, ... (<i>MSR_SIZE</i>) - #unterschiedliche Dateitypen (<i>cnt_diff_doc</i>): Bilder, Applets, ... - #Bilder interner (<i>SUM_INT_IMG</i>) / externer Speicherort (<i>cnt_ext_img</i>) - #Werbung (<i>cnt_adv_img</i> und <i>cnt_adv_links</i>)
Interaktion	<ul style="list-style-type: none"> - #Formulare und Felder (<i>cnt_form</i> und <i>cnt_form_elem</i>) - #MailTo-Links total (<i>cnt_email</i>) / gruppiert (<i>cnt_email_g</i>)
Navigation	<ul style="list-style-type: none"> - #Frames (<i>cnt_frames</i>), IFrames (<i>cnt_iframes</i>), extern (<i>cnt_frames_el</i>) - #Interne Links absolut (<i>cnt_int_links_a</i>) / relativ (<i>cnt_int_links_r</i>) - #Externe Links (<i>cnt_ext_links</i>) - #Linkfehler (<i>cnt_fail_links</i>)

Tabelle 6.2: Messbare Strukturmerkmale einer Webseite

Die gesammelten Messwerte und Qualitätsmaße werden gruppiert zu Eigenschaften...

- des Webservers und performancetechnischer Aspekte (*Effizienz*),
- zu Links (*Zuverlässigkeit, Benutzbarkeit, Wartbarkeit, Portabilität, Nutzerakzeptanz*),
- zu Bildern und zu sonstigen Dokumenten (.js-Dateien, PDFs, ...) (*Zuverlässigkeit, Effizienz, Benutzbarkeit, Wartbarkeit, Portabilität, Nutzerakzeptanz, Fremdinhalte*),
- von eingebundenen E-Mailadressen (*Benutzbarkeit, Wartbarkeit*),
- von Frames (*Zuverlässigkeit, Effizienz, Benutzbarkeit*),
- der Webseite (*Zuverlässigkeit, Effizienz, Benutzbarkeit, Wartbarkeit, Portabilität, Nutzerakzeptanz*) und
- des HTML-Quellcodes (*Effizienz, Benutzbarkeit, Wartbarkeit, Portabilität*).

Anhang B ordnet alle Eigenschaften diesen Gruppierungen zu und bildet die Messwerte und Maße auf die Qualitätsmerkmale des Qualitätsmodells in Abbildung 6.1 ab. Die Wahl der Qualitätsmaße ergab sich aus Arbeiten von Luis Olsina [32] und aus eigener Erfahrung als Web-Qualitätssicherungsingenieur bei der Community <http://www.webuni.de>.

6.2 Spezifikation & Entwurf der Web-Qualitätssicherung

Mit Hilfe der Web Measurement Suite können Nutzer über das Medium Internet Verbünde von Webseiten anhand vordefinierter Qualitätsmerkmale untersuchen. Die Messwerte werden gespeichert, wodurch es ermöglicht wird, dass eine Messung wiederholt werden kann. In diesen Messreihen können so Entwicklungen und Änderungen verfolgt werden.

Zu den Anforderungen gehört das systematische Durchsuchen eben jener Webseitenverbünde ausgehend von mindestens einer Start-URL und der rekursive Durchlauf bis zu einer eingestellten Linktiefe. Innerhalb der Webseiten und während der Messung werden Messwerte aufgenommen und gespeichert, die anhand von Qualitätsmaßen das Treffen von Aussagen über verschiedene Aspekte der Qualität erlauben. Es soll eine Zusammenfassung der Qualität über den gesamten Webseitenverbund vorgenommen und ein Qualitätsreport generiert werden.

Dieser Abschnitt spezifiziert folgende Teilbereiche:

- Verwaltung der Messprojekte
- Erzeugung von Messreihen
- Aufbau des Suchbaumes während einer Messung
- Der Messvorgang
- Aufnahme der Messwerte und Qualitätsmaße
- Auswertung durch Report-Erzeugung
- Webseitenüberwachung mit dem Monitoring Service
- Datenhaltung

Die Web Measurement Suite soll auf dem Agentensystem des Web-Tomographen Web-Tomix aufbauen und ihn um Möglichkeiten der Web-Qualitätssicherung erweitern. Das angepasste Agentensystem wird in Abschnitt 6.3 behandelt.

Im Rahmen der Spezifikation und des Entwurfs basieren viele der folgenden Abbildungen auf UML (*Unified Modeling Language*) [13]. UML ist eine Beschreibungssprache und dient zur Darstellung von Strukturen und Abläufen in objektorientierter Software.

6.2.1 Verwaltung von Messprojekten

Alle Messungen von Webseiten werden in Messprojekten zusammengefasst. Hierbei bezeichnen Messprojekte Klassen, deren grundsätzliche Methoden und Eigenschaften in Abbildung 6.2 dargestellt sind. Die Abbildung zeigt einen Ausschnitt des Klassendiagramms. Die vollständigen Klassendiagramme sind auf der CD zu finden (siehe Anhang D).

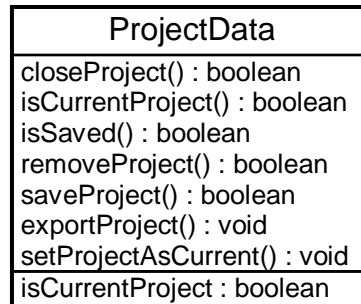


Abbildung 6.2: Klasse ProjectData (gekürzt)

Ein Messprojekt kann erst geöffnet werden, wenn das zuletzt geöffnete Projekt erfolgreich geschlossen wurde (`isCurrentProject()` liefert beim zuletzt geöffneten Projekt ein `FALSE` zurück). Das Projekt wird als geöffnet markiert, indem die Methode `setProjectAsCurrent()` aufgerufen wird. Nun kann das Messprojekt genutzt werden um Webseiten zu messen.

Sobald ein anderes Projekt geöffnet werden soll, muss das aktuelle Projekt erfolgreich geschlossen werden. Hierzu wird im aktuellen Projekt die Methode `closeProject()` aufgerufen, die bei Erfolg `TRUE` zurückliefert. Erfolgreich geschlossen wird es, wenn innerhalb des Projektes keine Messung läuft und wenn alle Daten des Projektes erfolgreich aus dem flüchtigen Speicher auf die Festplatte gespeichert wurden. Gespeichert wird das Projekt über den Aufruf der Methode `saveProject()`, welche `TRUE` zurückliefern muss, damit das alte Projekt geschlossen und das neue geöffnet werden kann.

Soll das Messprojekt gelöscht werden (`removeProject()`), muss das Projekt ebenfalls zuerst geschlossen werden. Somit wird gewährleistet, dass keine Messung läuft, die auf ein gelöscht Messprojekt verweist.

Um das Messprojekt auf CD sichern oder auf einen anderen Computer übertragen zu können, müssen die Daten des Projektes in eine Archiv-Datei exportiert werden. Die Methode `exportProject()` fordert den Nutzer auf, ein Verzeichnis anzugeben, in dem die exportierte Archiv-Datei `.msz`³¹ gespeichert werden soll.

³¹ `.msz` ist die Dateierweiterung für *Measurement Suite Zip*.

Das Sequenzdiagramm in Abbildung 6.3 stellt den Vorgang dar, wie ein Projekt gewechselt wird.

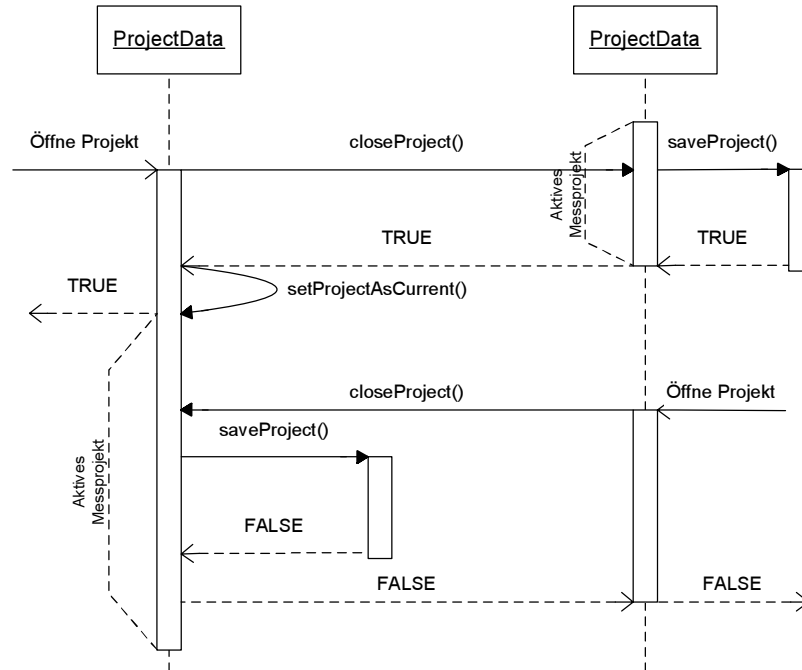


Abbildung 6.3: Öffnen von Messprojekten als Sequenzdiagramm

Abbildung 6.4 zeigt schematisch, dass das Agentensystem der Web Measurement Suite auf verschiedene Messprojekte zugreifen kann und diese dabei unterscheidet. Aus der Sicht des Nutzers kann immer nur ein Projekt geöffnet sein. Aus der Sicht des Agentensystems unterscheidet der Monitor-Agent nicht zwischen verschiedenen Projekten, weshalb er mehrere gleichzeitige Verbindungen zu unterschiedlichen Messprojekten aufbaut.

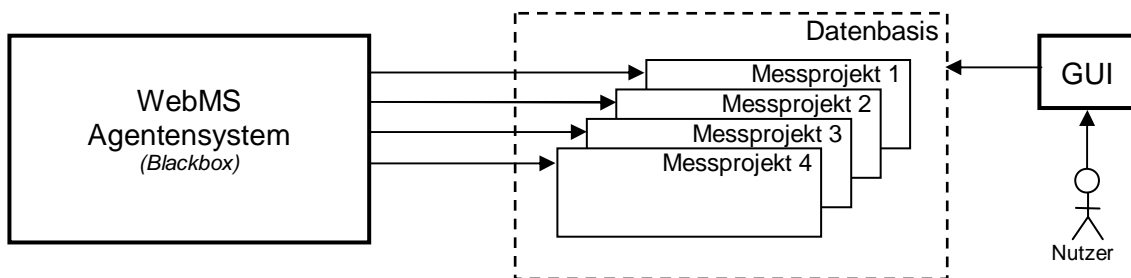


Abbildung 6.4: Messprojekte

In der Grafik ist das Agentensystem als *Blackbox* gekennzeichnet. Eine Blackbox bezeichnet im Allgemeinen ein geschlossenes System, dessen innerer Aufbau und Funktionsweise unbekannt oder als nicht weiter von Bedeutung erachtet wird. Von Interesse ist nur das äußere Verhalten der Blackbox. Hier wird der Zugriff auf die Messprojekte betrachtet.

6.2.2 Erzeugung von Messreihen

Ein Messprojekt ist das Rahmenwerk für Messungen. Eine Messung muss gemäß den Anforderungen rekursiv alle Webseiten ausgehend von einer oder einigen Start-URLs untersuchen, wobei einige Startbedingungen befolgt werden müssen. Diese werden bei der Konfiguration des Messprojektes festgelegt.

Eine Konfiguration kann entweder neu erstellt oder aus einer bestehenden Konfiguration geklont werden. Beim Klon werden die Startbedingungen der Original-Konfiguration übernommen. Zur Unterscheidung geklonter Konfigurationen ist dem Namen der Konfiguration eine Versionsnummer angehängt. Beispielsweise wird aus der Konfiguration `MyConfig [1]` der Klon `MyConfig [2]`. Mit Hilfe des Klonens von Konfigurationen können Messungen in Zeitabständen wiederholt und somit Änderungen verfolgt werden. Eine Messreihe definiert eine Reihe von Klonen derselben Konfiguration. Natürlich kann ein Messprojekt mehrere Konfigurationen besitzen und damit auch mehrere Messreihen.

Jede Konfiguration und jede Messreihe basiert auf denselben Startbedingungen:

Start-URL	eine oder mehrere Start-URLs, von der rekursiv verlinkte Webseiten untersucht werden
Linktiefe	Linktiefe bis zu der rekursiv getestet werden soll
404-Test	Untersuchung der nächsten Ebene auf Verbindungsfehler; neben dem Response Code 404 zählen auch alle Fehler, die einen Zugriff auf das Webdokument verwehren sowie Programmfehler (Linktiefe + 1)
StayOnHost	es sollen nur Ressourcen untersucht werden, die auf demselben Host gespeichert sind (im Rahmen des 404-Tests werden auch Verbindungen zu externen Dokumenten aufgebaut)
StayInDir	es sollen nur Ressourcen untersucht werden, die sowohl auf demselben Host als auch in dem Verzeichnis oder einem Unterverzeichnis des verlinkenden Dokumentes gespeichert sind (durch 404-Test ist trotzdem eine Verbindung möglich)

Tabelle 6.3: Startbedingungen einer Messkonfiguration

Mit Hilfe dieser Eigenschaften ist eine vollständige Suche in den gewünschten Webdokumenten möglich. Die Auswirkungen der Wahl der Startbedingungen ist beim Aufbau des Suchbaumes (siehe Abschnitt 6.2.3) zu erkennen. Die Startbedingungen sind in der Konfiguration individuell für jede Start-URL anzugeben. Verlinkte Dokumente der Start-URLs erben diese Bedingungen und verringern die Linktiefe um einen Zähler.

Der 404-Test kann optional deaktiviert werden, um die Zeit zu verringern, die zum Messen benötigt wird. Bei einer Suchtiefe von 3 mögen nur einige hundert Dokumente den Anforderungen entsprechen, aber die verlinkten Dokumente der vierten Ebene vervielfachen die Anzahl im Normalfall. Der Verbindungsaufbau zu einer Seite ist zumeist der zeitintensivste Kostenfaktor einer Messung und dieser ist für den 404-Test notwendig.

Da Konfigurationen ein Teil eines Messprojektes sind, muss die Klasse aus Abbildung 6.2 um die Konfigurationen betreffenden Attribute und Methoden erweitert werden.

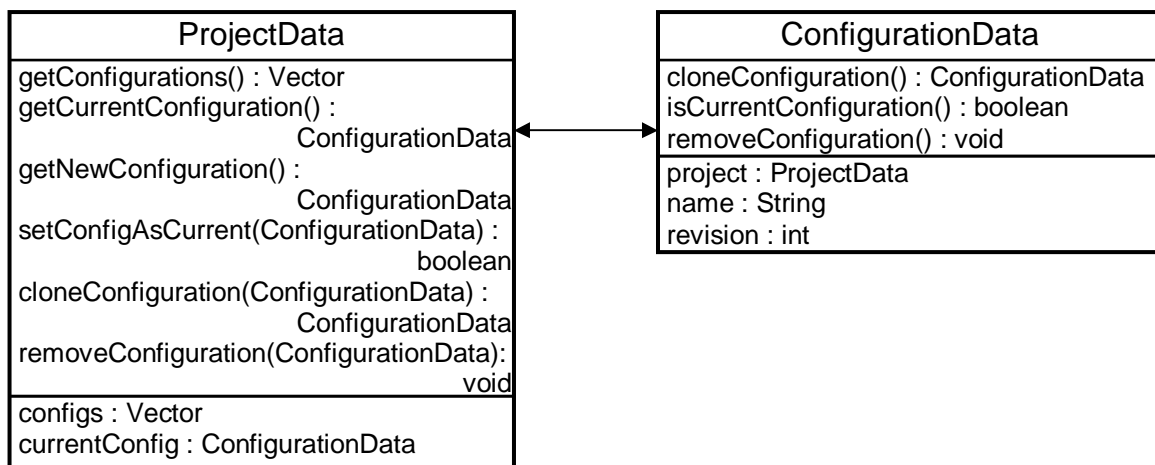


Abbildung 6.5: Verknüpfung zwischen ProjectData und ConfigurationData

Die Konfigurationen sind in dem Messprojekt `project` in einem Vektor³² gespeichert, der über die Methode `project.getConfigurations()` abgerufen werden kann. Eine neue Konfiguration wird über `project.getNewConfiguration()` erstellt. Um die Konfiguration `config` zu klonen, wird `config.cloneConfiguration()` aufgerufen. Die neue Konfiguration übernimmt den Namen, alle Startbedingungen und erhöht den Zähler `revision`. Existiert diese Revision für die Messreihe bereits, wird `revision` erneut erhöht, bis eine freie Zahl gefunden wird. Um die Konfiguration `config` zu löschen, wird die Methode `config.removeConfiguration()` aufgerufen (siehe auch Abbildung 6.6).

Die aktuelle Konfiguration kann über `project.getCurrentConfiguration()` angefordert werden. Der Nutzer kann innerhalb des Messprojektes mit mehreren Konfigurationen arbeiten, muss allerdings erst alle Arbeiten beenden (das Projekt speichern und alle laufenden Messvorgänge beenden), bevor er das Messprojekt wechseln kann. Abbildung 6.7 zeigt den erweiterten Zugriff auf ein Messprojekt und die integrierten Konfigurationen aus der Sicht des Nutzers und des Agenten.

³²Ein Vektor ist in Java eine Speicherklasse, welche Objekte unsortiert speichert. Der Zugriff wird über die Reihenfolge gesteuert.

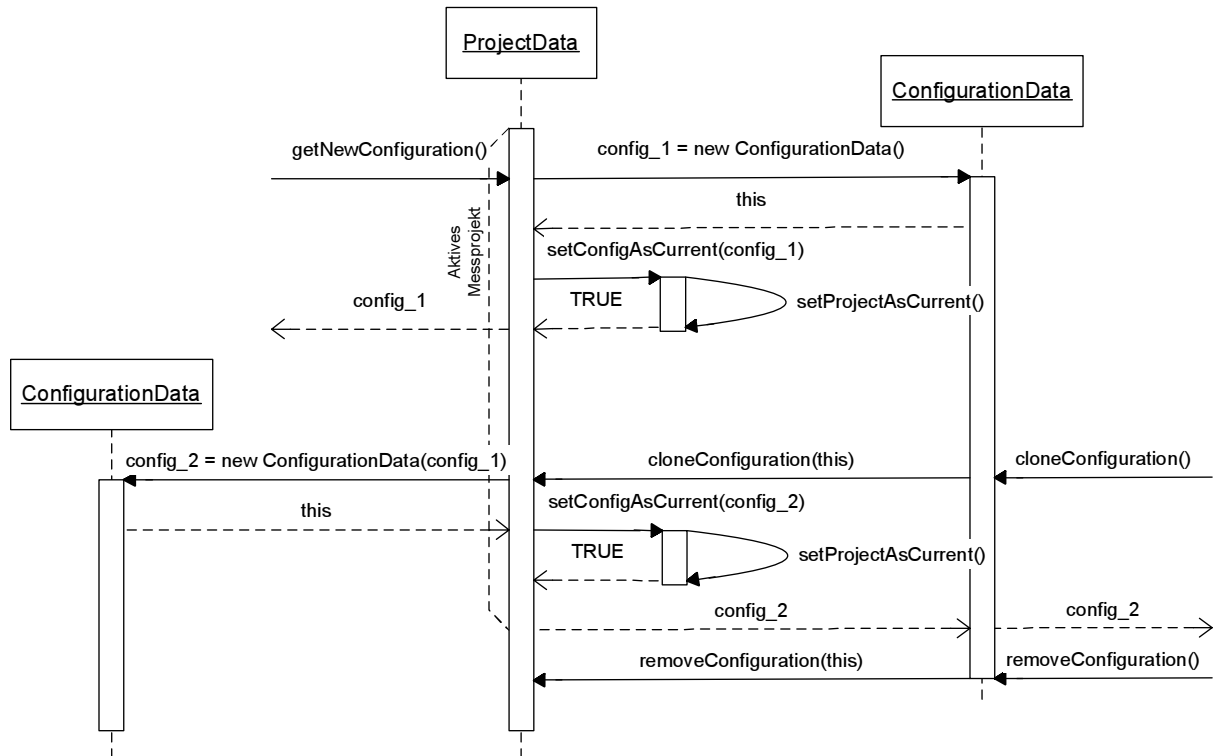


Abbildung 6.6: Verwaltung von Messkonfigurationen als Sequenzdiagramm

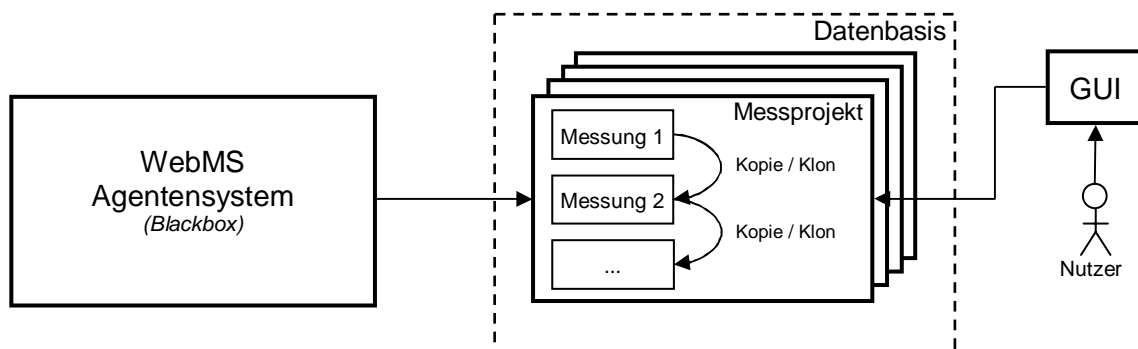


Abbildung 6.7: Messreihen

6.2.3 Aufbau des Suchbaumes

Nach Tabelle 6.3 auf Seite 78 beginnt eine Messung mit einer oder mehreren unterschiedlichen Start-URLs und verfolgt alle verlinkten Webdokumente bis hin zu einer vorgegebenen Linktiefe.

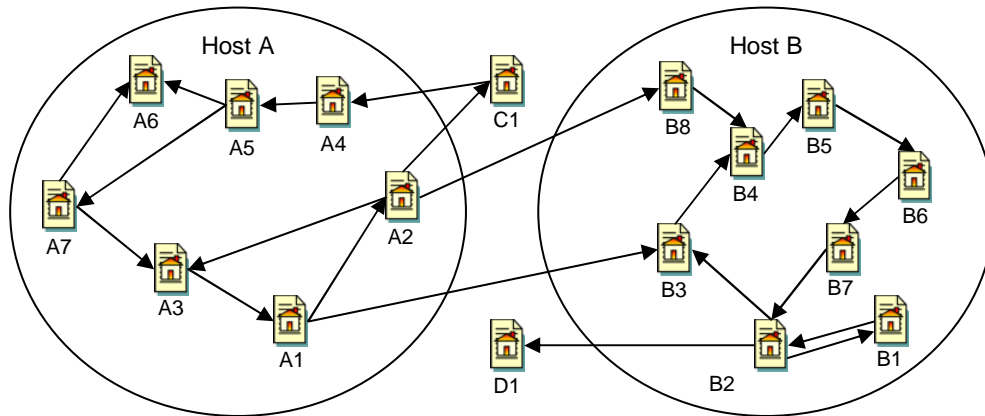


Abbildung 6.8: Webstruktur als Beispiel für den Aufbau des Suchbaumes

Bild 6.8 zeigt eine Webstruktur. Die Knoten des Graphen symbolisieren Webseiten, die Kanten die Hyperlinks untereinander. Alle Knoten innerhalb eines Kreises haben eine URL mit demselben Host.

Szenario 1

Startbedingungen: A1, B1 sind Start-URLs, die Linktiefe ist 3, es soll auf 404-Fehler getestet werden, StayOnHost, kein StayInDir

Tabelle 6.4 beschreibt den Aufbau des Suchbaums unter den gegebenen Bedingungen. Die erste Spalte # zeigt die Durchlaufnummer. Die zweite Spalte die Linktiefe (Start-URLs haben die 0) und die verbleibenden Ebenen.

Dokumente, die den Anforderungen nicht entsprechen, werden nicht gemessen (C1, D1). Dies gilt auch für Dokumente, die zwar den Bedingungen einer anderen Start-URL genügen würden (B3, B8), dies aber nicht beim verlinkenden Objekt tun. URLs, die bereits gemessen wurden, werden nicht noch einmal gemessen oder auf 404 getestet.

Wurde ein Dokument auf 404 getestet, da es bei einem vorherigen Durchlauf nicht den Bedingungen entsprach, und ist es in einem späteren Durchlauf gültig, so wird das Webdokument normal gemessen und der 404-Test überschrieben (B3).

Obwohl A4 den Anforderungen von A1 genügen würde (StayOnHost, Linktiefe 3), wird es nicht in der Messung erfasst, da es nur von C1 verlinkt ist.

#	Linktiefe	Webseiten	Aktion	Hyperlinks
1	0, 3 (+1)	A1	Messung	A2, B3
		B1	Messung	B2
2	1, 2 (+1)	A2	Messung	A3, B8, C1
		B2	Messung	B1, B3, D1
		B3	404-Test, da Hyperlink von A1	
3	2, 1 (+1)	A3	Messung	A1
		B1	bereits gemessen	
		B3	bereits 404-Test von A1; Hyperlink von B2, Messung	B4
		B8, C1	404-Test, da Hyperlink von A2	
		D1	404-Test, da Hyperlink von B2	
4	3, 0 (+1)	A1	bereits gemessen	
		B4	Messung	B5
5	4, 0	B5	obwohl StayOnHost wahr ist, nur 404-Test, da max. Linktiefe erreicht	

Tabelle 6.4: Aufbau des Suchbaumes in Szenario 1

Szenario 2

Startbedingungen: A1, B1 sind Start-URLs, die Linktiefe ist 3, keine weiteren Bedingungen, kein 404-Test

#	Linktiefe	Webseiten	Aktion	Hyperlinks
1	0, 3	A1	Messung	A2, B3
		B1	Messung	B2
2	1, 2	A2	Messung	A3, B8, C1
		B2	Messung	(B1), (B3), D1
		B3	Messung	B4
3	2, 1	A3	Messung	(A1)
		B4	Messung	B5
		B8	Messung	(B4)
		C1	Messung	A4
		D1	Messung	
4	3, 0	A4	Messung	A5
		B5	Messung	B6

Tabelle 6.5: Aufbau des Suchbaumes in Szenario 2

Hyperlinks zu bereits gemessenen Dokumenten sind in Klammern dargestellt.

Start-URLs sollen vom Nutzer manuell in ein Textfeld eingetragen werden können oder über einen integrierten Browser ausgewählt werden. Der integrierte Browser der Web Measurement Suite basiert auf der API (*Application Programming Interface*) des kommerziellen ICEbrowser SDK. Ein **Software Development Kit** (SDK) ist eine Sammlung von Programmen und Dokumentation zu einer Software, die das Erstellen von darauf basierenden Applikationen erleichtert.

Das ICEbrowser SDK ist ein optionaler Bestandteil der Web Measurement Suite. Sind die Pakete vorhanden, wird der Browser automatisch beim Programmstart eingebunden. Ansonsten sind alle Funktionen, die auf dem Browser basieren, deaktiviert. Der ICEbrowser ist unter <http://www.icesoft.com/> als Trial herunterladbar.

Die Verweise auf die Ressource B3 in Abbildung 6.8 von A1 und B2 können trotz desselben Ziels eine unterschiedliche URL haben. Es entsteht ein Problem der Identifikation von Duplikaten. Zwei URLs, die auf dasselbe Webdokument referenzieren, können durch Umwandlung in dieselbe kanonische Form überführt werden. Wenn die kanonische Form einer URL zu einer anderen identisch ist, kann die Identifikation als Duplikat erfolgen. Folgende Umwandlungen werden durchgeführt:

1. Entfernen von „../“ und des übergeordneten Verzeichnisses,
Beispiel: aus `/path/./main.asp` wird `/main.asp`
2. Entfernen von „./“,
Beispiel: aus `/path/./file.asp` wird `/path/file.asp`
3. Auflösung einer relativen URL durch Verwendung der URL des verweisenden Dokumentes,
Beispiel: `http://www.mycompany.org/path/file.asp` verweist auf `./main.asp`,
Umwandlung in `http://www.mycompany.org/main.asp`
4. Verwendung des HTTP-Protokolls, wenn keines angegeben ist,
Beispiel: aus `www.mycompany.org` wird `http://www.mycompany.org`
5. Umwandlung des Protokolls und des Hostnamens in Kleinbuchstaben,
Beispiel: aus `Http://www.MyCompany.org/` wird `http://www.mycompany.org/`
6. Entfernen einer unnötig angegebenen Standard-Portnummer,
Beispiel: `http://www.mycompany.org:80/` wird zu `http://www.mycompany.org/`

7. Entschlüsseln von URL, Ersetzen von hexadezimal-umschriebenen Zeichen aus der ASCII-Codetabelle,
 Beispiel: in `http://www.mycompany.org/%7Euser` wird aus `%7E` die Tilde:
`http://www.mycompany.org/~user`
8. Überprüfen, ob in der URL ein endender Slash fehlt, falls URL ein Verzeichnis bezeichnet (siehe Abbildung 6.9),
 Beispiel: `http://www.mycompany.org` wird zu `http://www.mycompany.org/`
9. Entfernen des Ankers, da dieser auf eine Position innerhalb des Dokumentes verweist,
 Beispiel: `http://www.mycompany.org/main.asp#top` wird umgewandelt in die URL `http://www.mycompany.org/main.asp`

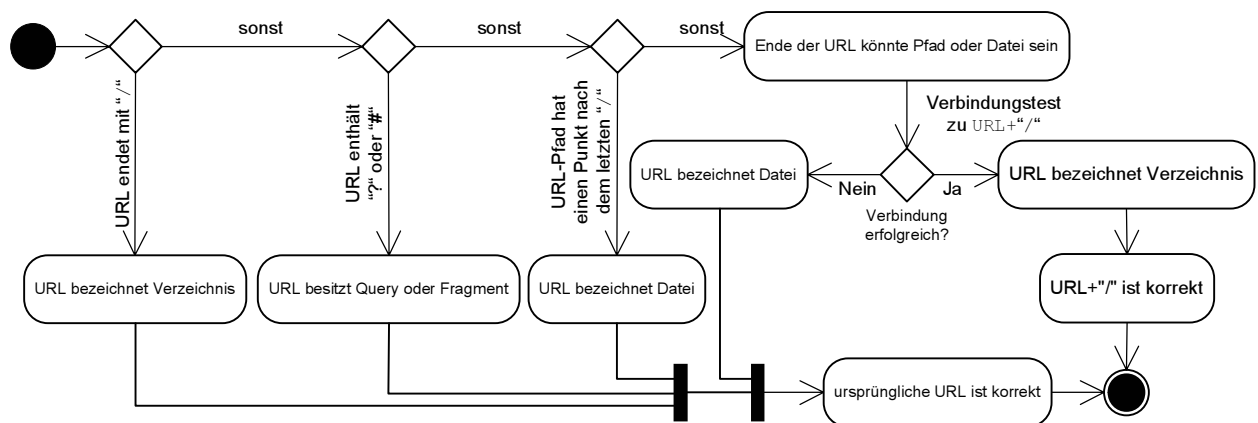


Abbildung 6.9: Test auf fehlenden Slash am Ende der URL

Des Weiteren müssen Umleitungen einer URL behandelt werden können. Zum Beispiel können Webserver durch die HTTP Response-Codes 3xx Anfragen an verschobene Dokumente mit der neuen Zieladresse beantworten. So könnte aus der ursprünglichen URL `http://www.mycompany.org/main.asp` die URL `http://www.mycompany.org/home.asp` werden. Da diese schon gemessen sein könnte, muss dieser Fall bei einer Weiterleitung erneut geprüft werden.

In Schritt 3 werden relative URLs anhand der URL des verlinkenden Dokumentes aufgelöst. Hierzu gibt es einen Sonderfall, der beachtet werden muss. Wird in einem HTML-Dokument `<base href="http://www.server.com/" />` verwendet, so beziehen sich alle relativen Links auf `http://www.server.com/` und nicht auf die URL des verlinkenden Dokumentes.

6.2.4 Der Messvorgang

Nachdem eine Konfiguration erstellt wurde, befindet sie sich im *Konfigurieren*-Status. In diesem Status können Start-URLs definiert und deren Startbedingungen eingestellt werden. Sobald die Messung beginnt, ist die Konfiguration vor Änderungen seitens des Nutzers geschützt. Hierbei ist es unerheblich, ob die Messung unterbrochen wurde oder nicht. Abbildung 6.10 zeigt mögliche Status einer Konfiguration.

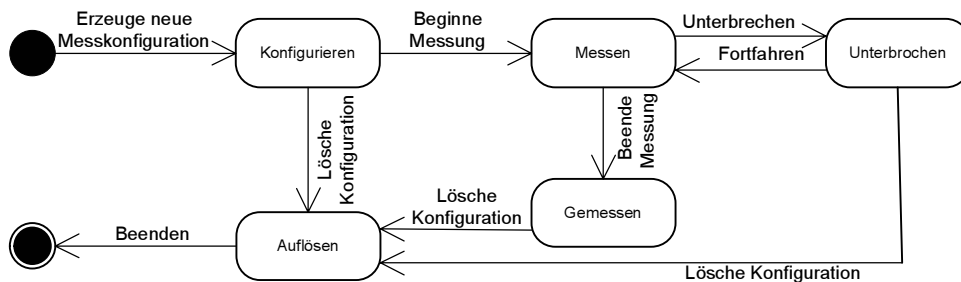


Abbildung 6.10: Lebenszyklus von Konfigurationen eines Messprojektes

Die Webdokumente der Start-URLs, und alle verlinkten URLs werden auf *Resource*-Klassen abgebildet. Abbildung 6.11 zeigt einen Ausschnitt aus den Klassendiagrammen der betreffenden Klassen. Die Diagramme sind auch hier der Übersichtlichkeit halber nur gekürzt dargestellt.

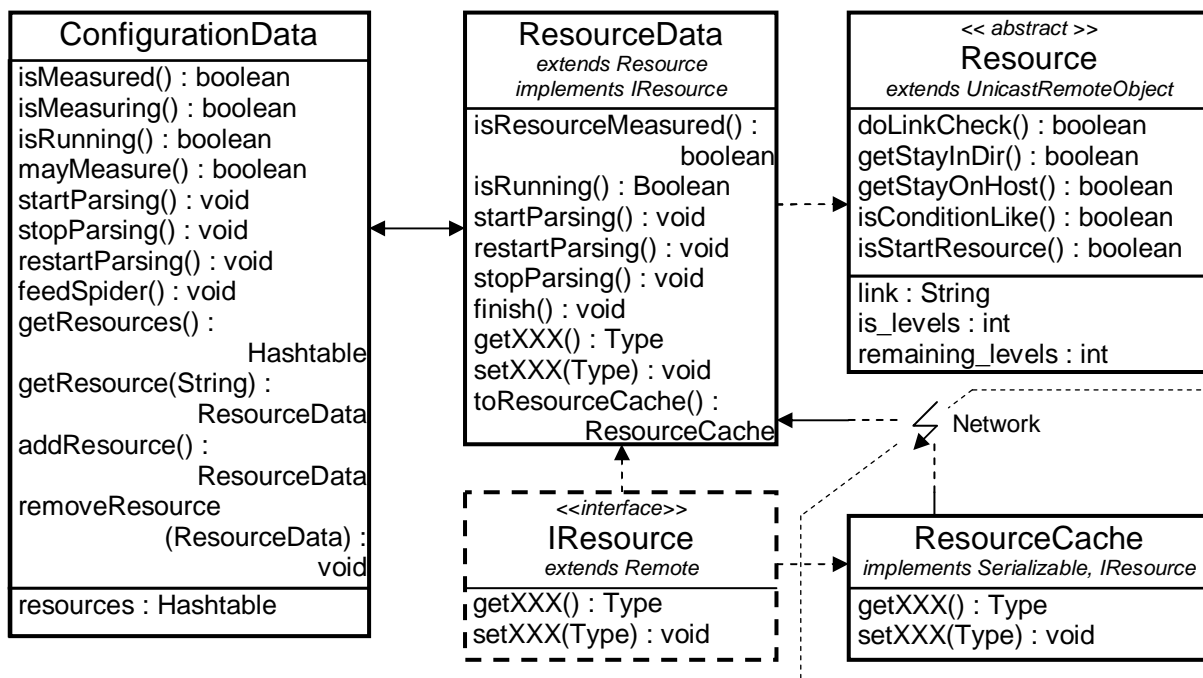


Abbildung 6.11: Webressourcen im Zusammenhang mit dem Messvorgang I

Konfigurationen halten Webdokumente in der Hashtable³³ `resources`. Webdokumente werden in Objekten gehalten, die von der abstrakten Klasse `Resource` erben. Diese Klasse hält die Startbedingungen aus Tabelle 6.3 (Seite 78), welche vom Webdokument zu allen Folge-URLs weitervererbt werden. Die Variablen `is_levels` und `remaining_levels` speichern die momentane Linktiefe und wie viele Ebenen noch durchsucht werden sollen.

Der allgemeine Ressourcentyp wird durch die Klasse `ResourceData` abgebildet, welcher alle Ressourcentypen speichert. Bei Bedarf können typspezifische Klassen ergänzt werden, die beispielsweise speziell auf Bilder zugeschnitten sind. Momentan genügt jedoch der allgemeine Ressourcentyp.

Objekte des Typs `ResourceData` werden in der Startphase direkt in der Konfiguration mittels `addResource()` erschaffen und mit `removeResource()` aus der Hashtable `resources` gelöscht. Als Schlüssel der Hashtable dient der Link `link`, der in der Klasse `Resource` gespeichert wird.

Die Messung einer Konfiguration wird mit `startParsing()` begonnen und kann gemäß Abbildung 6.10 mittels `stopParsing()` gestoppt und `restartParsing()` fortgesetzt werden. Wurde die Messung innerhalb der Konfiguration gestartet, können keine weiteren Start-URLs hinzugefügt werden. Die Methode `addResource()` ändert ab diesem Zeitpunkt nichts mehr an der Konfiguration. In diesem Fall verweist `startParsing()` auch direkt auf `restartParsing()`.

Die zu messenden Ressourcen werden über die Methode `feedSpider()` an den „Spider“ `Measurement Coordinator` gesendet. Nicht alle Agenten müssen zwingend auf demselben Computer laufen. Deshalb ist es erforderlich das Objekt zwischen den Agenten zu transferieren, welches die zu messende Webressource abbildet. Hierzu gibt es die serialisierbare Cache-Klasse namens `ResourceCache`. Diese wird via `toResourceCache()` aus einem `ResourceData`-Objekt erstellt und hält eine Verbindung zu diesem Objekt. Die Serialisierbarkeit ermöglicht den Transport innerhalb des Agentensystems. Die gecachten Informationen innerhalb des Cache-Objektes beschränken sich auf ein Minimum. Alle weiteren Daten werden über das Netzwerk vom Original angefordert. Das Interface `IResource` definiert einige Getter- und Setter-Methoden, die sowohl in der Ressource-Klasse als auch in der Cache-Klasse implementiert sein müssen. Ändert das Agentensystem etwas an der Ressource, werden die Informationen an das Original weitergeleitet. Die Cache-Klasse wird aufgelöst, sobald die Messung auf diesem Webdokument beendet ist.

³³Eine Hashtable ist in Java eine Speicherklasse, welche Objekte dem Hashwert eines Strings (Schlüssel) zuordnet. Der Zugriff erfolgt über die Schlüssel.

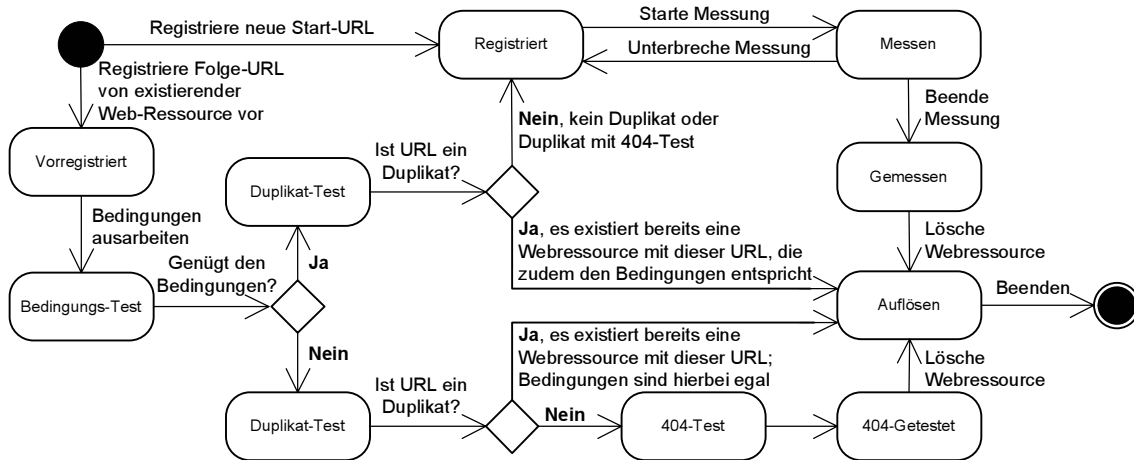


Abbildung 6.12: Lebenszyklus eines eine Webressource abbildenden Objektes

Abbildung 6.12 zeigt den Lebenszyklus eines `ResourceData`-Objektes. Hier wird der Unterschied zwischen Start- und Folge-URLs erklärt. Eine URL, die über `addResource()` vor dem erstmaligen Starten des Messvorgangs der Konfiguration hinzugefügt wird, wechselt sofort in den Status *Registriert* und wird danach gemessen. Folge-URLs werden zunächst einmal vorregistriert, da vor dem Messen einige Bedingungen geprüft werden müssen.

Damit die Einhaltung der Bedingungen³⁴ geprüft werden kann, muss die Folge-URL in die kanonische Form aus Abschnitt 6.2.3 (Seite 83) umgewandelt werden um beispielsweise relative Links aufzulösen. Mit der kanonischen Form der URL ist es möglich zu testen, ob die URL ein Duplikat ist.

Es gibt fünf mögliche Fälle, die sich aus der Kombination der beiden Tests ergeben:

1. Ressource A stimmt nicht mit den Bedingungen überein und wurde bereits gemessen oder auf 404-Fehler getestet (Ressource B). Das Ressourcen-Objekt A wird sofort aufgelöst und die Referenzen umgeschrieben auf das Objekt B
2. Der Bedingungstest von Ressource A war negativ, aber die URL ist bisher unbekannt. Wenn gewünscht wird die Ressource nun auf 404-Fehler getestet³⁵.
3. Ressource A hält die Bedingungen ein und die URL ist bisher nicht gemessen worden. Ressource A wechselt in den Status *Registriert* und wird danach gemessen.

³⁴StayOnHost: der Host ist derselbe; StayInDir: das Webdokument befindet sich in dem Verzeichnis oder darunter; Test auf Linktiefe

³⁵In Abbildung 6.12 wurde auf den Status verzichtet, den das Objekt wählt, wenn kein 404-Test durchgeführt werden soll. In diesem Fall spielt die Ressource keine Rolle mehr.

4. Ressource A hält die Bedingungen ein, aber die URL wurde bereits gemessen in Ressource B. Ressource B hat ebenfalls den Bedingungstest bestanden. Von einer erneuten Messung kann abgesehen werden. Die Referenzen auf Ressource A werden auf das bereits registrierte Objekt der Ressource B umgeschrieben und A wird aufgelöst.
5. Ressource A hält die Bedingungen ein, aber die URL wurde bereits gemessen in Ressource B. Ressource B hat allerdings nicht den Bedingungstest bestanden und wurde somit maximal auf 404-Fehler getestet. A wechselt in den Status *Registriert* und wird gemessen. Alle Referenzen zu B werden auf A umgeschrieben.

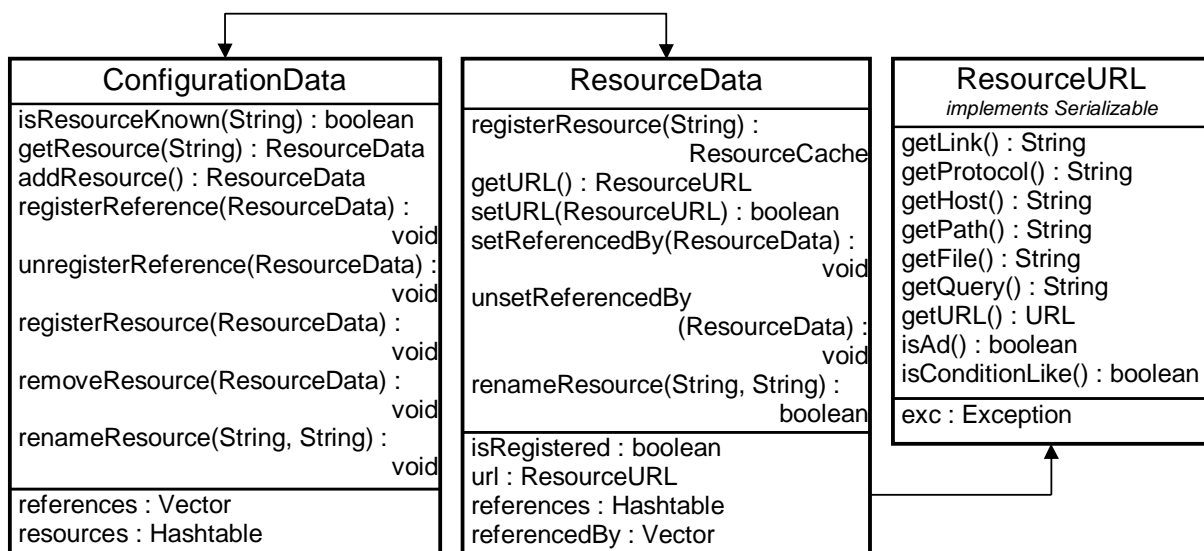


Abbildung 6.13: Webressourcen im Zusammenhang mit dem Messvorgang II

Erweiterte Klassendiagramme sind in Bild 6.13 zu finden. Die neue Klasse `ResourceURL` wandelt einen Link in seine kanonische Form um und bietet einige Getter-Methoden an, um Teile der URL zu erhalten. Mit der Methode `isAd()` wird zurückgeliefert, ob diese URL eine Übereinstimmung mit Werbe-Pattern hat (siehe Anhang C auf Seite 145). Diese Methode wird für einige Qualitätsmaße benötigt. Zusätzlich ist ein `ResourceURL`-Objekt serialisierbar und kann innerhalb des Agentensystems transportiert werden.

Da neue Start-URLs über `addResource()` der Konfiguration hinzugefügt werden und diese Methode nach dem Beginn der Messung nicht mehr verfügbar ist, werden weitere Methoden in der Konfigurationsklasse `ConfigurationData` definiert.

Die Methoden `registerReference()` und `unregisterReference()` registrieren Ressourcen vor und entfernen die Vorregistrierung. Über `registerResource()` wird eine vorregistrierte Ressource endgültig registriert und zur Messung vorbereitet. Registrierte Ressourcen werden in der Hashtable `resources` gehalten und vorregistrierte Folge-URLs im Vektor `references`.

Um den Link eines Webdokumentes zu ändern (beispielsweise durch HTTP-Header-Relinks), kann die URL jederzeit mittels `setURL()` geändert werden, indem ein neues `ResourceURL`-Objekt übergeben wird. `renameResource()` wird automatisch aufgerufen.

6.2.5 Aufnahme der Messwerte und Qualitätsmaße

Messwerte, die innerhalb des Agentensystems transportiert werden können, werden ebenfalls wie `ResourceData`-Objekte in serialisierbaren Objekten gehalten. Abbildung 6.14 zeigt die Speicherung von Messwerten innerhalb der Datenbasis des Messtools.

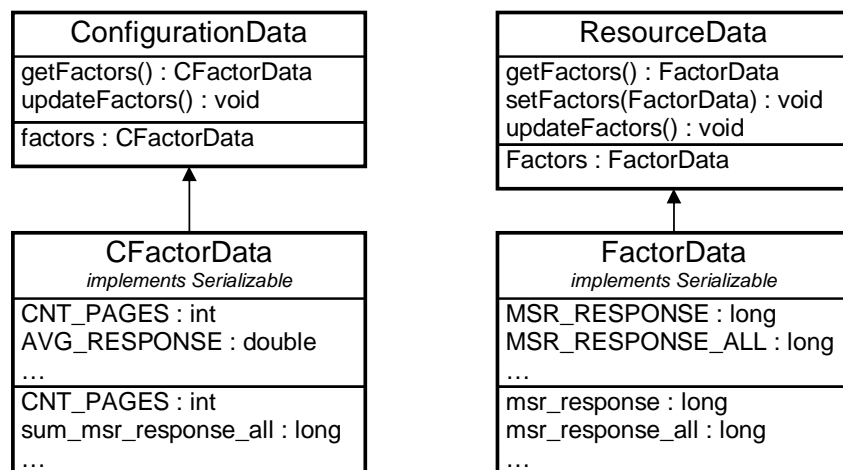


Abbildung 6.14: Speicherung von Messwerten

Jede Webressource (`ResourceData`) besitzt ein Objekt, welches die Messwerte aufnimmt. Die zu den Objekten gehörende Klasse `FactorData` berechnet die Qualitätsmaße. Als Faktoren für die Berechnungen verwendet es die intern gespeicherten Messwerte. In der Grafik sind als Beispiel die beiden Messwerte `msr_response` und `msr_response_all` im Klassendiagramm aufgeführt. Diese Messwerte werden anhand der Berechnungsvorschrift in zugehörige Qualitätsmaße umgerechnet. Die beiden oben erwähnten Messwerte fließen in die Maße `MSR_RESPONSE` und `MSR_RESPONSE_ALL` ein. Die Berechnungsvorschriften sind in Anhang B.2 auf Seite 132 angegeben.

In Anhang B.2 werden alle Qualitätsmaße in ihren Gültigkeitsbereich unterschieden. Qualitätsmaße mit *SP* beziehen sich auf eine einzelne Webseite und können deshalb in `FactorData` berechnet werden, Maße mit *EM* beziehen sich auf die gesamte Messung. Qualitätsmaße, die sich auf die gesamte Messung beziehen³⁶, können nicht in einem `FactorData`-Objekt berechnet werden, da dieses einer Ressource zugeordnet ist. Stattdessen wird `CFactorData` verwendet, welches der Konfiguration zugeordnet ist.

Viele Messwerte ergeben sich aus Summen. Beispielsweise summiert *msr_response_all* die Zeit, die benötigt wird, um die zugehörige Webressource zu empfangen, sowie alle Zeiten der eingebetteten Bilder und Dokumente. Deshalb ist es nötig, dass mit der Berechnung des Messwertes gewartet wird, bis alle *msr_response*-Werte vorhanden sind. Zu diesem Zweck wird am Ende der Messung `updateFactors()` in allen Ressourcen aufgerufen, wodurch Messwerte zusammengefasst werden und die Messung abgeschlossen wird.

Dasselbe gilt für die Konfiguration. Auch hier können die Informationen erst am Ende der Messung gesammelt werden. *CNT_PAGES* zählt alle `ResourceData`-Objekte in der Hashtable `resources`, die ein HTML-Dokument sind und den Anforderungen *StayOn-Host*, *StayInDir* und der Linktiefe genügen.

Beide Objekte (`FactorData` und `CFactorData`) sind serialisierbar. Somit können sie durch `ResourceCache` an Agenten geliefert werden, die direkt lokal darauf zugreifen. Am Ende werden die Objekte wieder zurückgeliefert.

6.2.6 Auswertung durch Report-Erzeugung

Die Messung einer Konfiguration ist beendet. Das Agentensystem hat die Messwerte gesammelt und die Werte der Qualitätsmaße können berechnet werden. Messwerte und Ergebnisse der Qualitätsmaße lassen sich direkt in der Nutzer-Oberfläche anschauen und ad-hoc vergleichen. Um aus den Ergebnissen der Qualitätsmaße automatisiert eine Aussage zu treffen, wird eine Auswertungskomponente eingeführt.

Ein Report fasst alle Werte und Einstellungen übersichtlich zusammen. Er wird automatisch generiert und kann direkt angezeigt oder als gepackte ZIP-Datei per E-Mail versendet bzw. direkt auf der Festplatte gespeichert werden. Der Report besteht aus mehreren miteinander verknüpften HTML-Seiten, so dass das Ergebnis als Qualitätszeugnis auf Wunsch direkt ins WWW gestellt werden kann.

³⁶Gesamtzahl der gemessenen Webseiten (*CNT_PAGES*), Antwortzeiten aller Webseiten (*AVG_RESPONSE* und *RAT_RESPONSE*) sowie die Navigations-Linkdichte (*MSR_LINK_DENS*)

Durch die Unterstützung von Messreihen kann ein Report über mehrere Konfigurationen eines Messprojektes erstellt werden. Die Bewertung eines Maßes basiert auf dem Durchschnitt aller Messungen eines Webdokumentes.

Die Bewertung eines Qualitätsmaßes erfolgt durch die Abbildung des Wertes auf eine Skala von 1 bis 10. Die Werte 1 bis 3 sind gut, 4 bis 7 normal und 8 bis 10 schlecht. Zur Abbildung auf diese Skala werden 2 Werte, ein positiver und ein negativer Grenzwert, ausgewählt, die für jedes Qualitätsmaß definiert sind. Übersteigt oder unterbietet ein Wert den positiven Grenzwert befindet er sich im Bereich 1 bis 3; unterbietet bzw. übersteigt er den negativen Grenzwert wird er mit 8 bis 10 bewertet und wenn er dazwischenliegt, wird er mit 4 bis 7 bewertet. Die standardmäßigen Grenzwerte sind durch empirische Beobachtungen anhand der Messung von Webseiten mit Vorbildcharakter gewählt (siehe Kapitel 7).

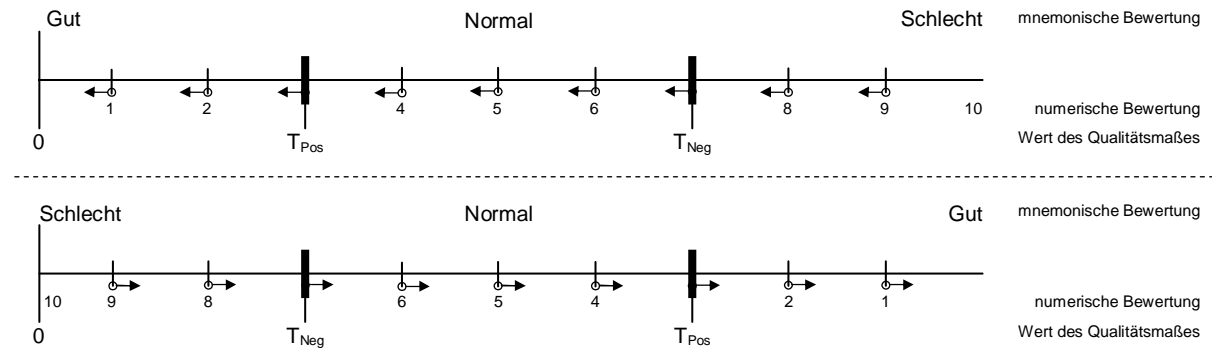


Abbildung 6.15: Allgemeine Bewertungsskalen von Qualitätsmaßen

In Bild 6.15 sind die Bewertungen auf der Skala eingezeichnet. Der obere Graph zeigt ein Qualitätsmaß, dessen Wert möglichst gering sein muss um gut zu sein (Beispiel: *MSR_RESPONSE*). Der untere Graph zeigt beispielsweise *MSR_LINK_DENS*, da eine hohe Navigations-Linkdichte im Sinne der Usability erwünscht ist und deshalb die Werte möglichst groß sein sollten.

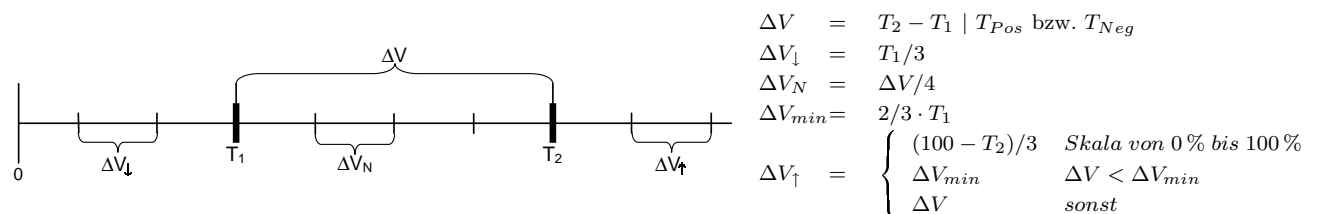


Abbildung 6.16: Abstände der Grenzen

Die Grenzwerte T_{Pos} und T_{Neg} (bzw. T_1 und T_2) sind für alle Qualitätsmaße standardmäßig vordefiniert, können jedoch an persönliche Präferenzen angepasst werden.

Die Abstände zwischen den Bewertungsschritten berechnen sich anhand der in Abbildung 6.16 angegebenen Formeln. ΔV_{\uparrow} wird je nach Skalentyp unterschiedlich berechnet. Bei einer nach oben offenen Skala wird der höhere Wert gewählt (entweder ΔV oder ΔV_{min}), damit der Abstand zwischen den Bewertungseinheiten genügend groß ist.

Aus den Bewertungen der einzelnen Qualitätsmaße berechnet sich die Qualität einer Webseite. Da die Qualitätsmaße in ihrer empirischen Bedeutung nicht alle gleichgesetzt werden können, werden sie von 0 bis 1 gewichtet. Wichtige und sehr genaue Maße bekommen eine hohe Wichtung (nahe 1) und ungenaue oder unwichtigere Maße eine niedrige Wichtung (nahe 0). Die Qualität Q_i^{res} einer Webressource i berechnet sich also aus

$$\begin{aligned} q_{sum}^{res} &= \sum_{\forall i | M_i \in M^{res}} \omega_i^{res} \cdot \epsilon_i^{res} \cdot v_i^{res} \\ \omega_{sum}^{res} &= \sum_{\forall i | M_i \in M^{res}} \omega_i^{res} \cdot \epsilon_i^{res} \\ Q_i^{res} &= q_{sum}^{res} / \omega_{sum}^{res} \end{aligned}$$

wobei ω_i^{res} die Wichtung des Qualitätsmaßes v_i^{res} darstellt. M^{res} ist die Gesamtmenge aller Qualitätsmaße M_i , die für die Webressource berechnet wurden.

Die Wichtung ω_i^{res} eines Qualitätsmaßes wird zusätzlich mit dem Faktor ϵ_i^{res} versehen. ϵ_i^{res} ist abhängig von der mnemonischen Bewertung des Maßes. Ein als gut bewertetes Maß geht in der Standardeinstellung fünffach in die Wertung ein und ein schlecht bewertetes Maß zwanzigfach. Somit ist gewährleistet, dass einzelne schlechte Bewertungen die Qualität erheblich verringern. Hiermit wird der Sachverhalt nachgebildet, dass ein menschlicher Nutzer eine Webseite anhand einzelner schlechter Merkmale bewertet ohne sich die Zeit zu nehmen, die Gesamtheit zu betrachten.

Zusammen mit den Grenzwerten T_{Pos} und T_{Neg} der einzelnen Qualitätsmaße, können die Faktoren für gute und schlechte Bewertungen ϵ frei an die eigenen Bedürfnisse angepasst werden (Qualitätszielbestimmung). Angepasste Einstellungen werden in nutzerspezifischen Report-Typen (siehe Seite 94) gespeichert.

Aus den Qualitätsbewertungen Q_i^{res} der Webseiten i in der Menge aller Ressourcen Q^{res} wird der Durchschnitt Q_{avg}^{res} gebildet.

$$Q_{avg}^{res} = \left(\sum_{\forall i | Q_i^{res} \in Q^{res}} Q_i^{res} \right) / \sum_{\forall i | Q_i^{res} \in Q^{res}}$$

Q^{conf} berechnet den Durchschnitt der Qualitätsmaße, deren Gültigkeitsbereich die gesamte Messung bzw. Messkonfiguration ist. Die Maße sind in der Menge M^{conf} zusammengefasst. Auch diese Maße werden gewichtet und je nach mnemonischer Bewertung mit einem Faktor versehen.

$$\begin{aligned}
 q_{sum}^{conf} &= \sum_{\forall j | M_j \in M^{conf}} \omega_j^{conf} \cdot \epsilon_j^{conf} \cdot v_j^{conf} \\
 \omega_{sum}^{conf} &= \sum_{\forall j | M_j \in M^{conf}} \omega_j^{conf} \cdot \epsilon_j^{conf} \\
 Q^{conf} &= q_{sum}^{conf} / \omega_{sum}^{conf}
 \end{aligned}$$

Zusammenfassend enthält Q_{avg}^{res} die durchschnittliche Qualität aller Webseiten in einer Messung auf Basis der seitenabhängigen Qualitätsmaße. Q^{conf} berechnet den Durchschnitt der Qualitätsmaße, die über die gesamte Messung gebildet werden. Diese beiden Werte müssen nun zusammengefasst werden, um die absolute Qualität Q_k^{meas} einer Messung k zu erhalten. Hierzu muss ein Verhältnis zwischen den beiden Werten gefunden werden, welches erlaubt einen gewichteten Durchschnitt zu bilden.

Als Basis für den gewichteten Durchschnitt werden die Summen der Wichtungen ω_{sum}^{conf} und ω_{sum}^{res} herangezogen. ω_{sum}^{conf} wird mit der Anzahl der gemessenen Webseiten normalisiert zu ω_{norm}^{conf} . Nun kann eine Aussage getroffen werden, in welchem Verhältnis Q^{conf} zu Q_{avg}^{res} steht und der gewichtete Durchschnitt Q_k^{meas} der Messung k kann gebildet werden.

$$\begin{aligned}
 \omega_{norm}^{conf} &= \omega_{sum}^{conf} \cdot \sum_{\forall i | Q_i^{res} \in Q^{res}} \\
 Q_k^{meas} &= \frac{Q_{avg}^{res} \cdot \omega_{sum}^{res} + Q^{conf} \cdot \omega_{norm}^{conf}}{\omega_{sum}^{res} + \omega_{norm}^{conf}}
 \end{aligned}$$

Wurden mehrere Messkonfigurationen bei der Reporterzeugung ausgewählt, so bildet sich der endgültige Qualitätswert des Reports Q^{rep} aus dem Durchschnitt der Qualitätswerte Q_k^{meas} aller Konfigurationen k in der Messreihe Q^{meas} .

$$Q^{rep} = \left(\sum_{\forall k | Q_k^{meas} \in Q^{meas}} Q_k^{meas} \right) / \sum_{\forall k | Q_k^{meas} \in Q^{meas}}$$

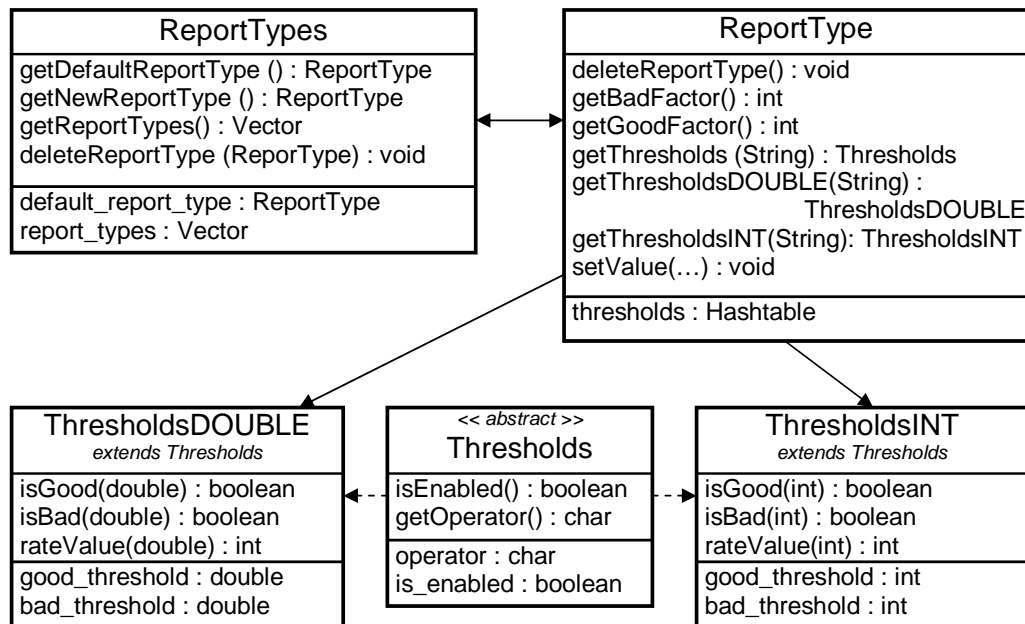


Abbildung 6.17: Report-Typen und Evaluation von Qualitätsmaßen

Abbildung 6.17 zeigt die Klassen, die im Zusammenhang mit Report-Typen definiert sind. In der Klasse `ReportTypes` werden im Vektor `report_types` verschiedene Report-Typen gehalten, die bis auf den Standard-Report-Typen geändert und gelöscht werden können. Ein neuer Report-Typ wird über `getNewReportType()` angefordert.

Report-Typen speichern die einzelnen Grenzwerte in Form von `Thresholds`-Objekten in einer Hashtable. Die Werte werden über die beiden `setValue()`-Methoden gesetzt, die folgende Parameter erwartet:

```

setValue(String field, boolean is_enabled, char operator,
          double good_threshold, double bad_threshold)
setValue(String field, boolean is_enabled, char operator,
          int good_threshold, int bad_threshold)
  
```

`field` ist der Name des Qualitätsmaßes und damit der Schlüssel in der Hashtable. Per `is_enabled` kann dieses Maß einzeln deaktiviert werden. Der `operator` kennzeichnet die Richtung der Skala - also ob der Wert möglichst hoch oder niedrig sein soll um gut zu sein. Die beiden letzten Werte sind die Grenzwerte, die je nach Art der Skala (ganzzahlig oder nicht ganzzahlig) entweder `ThresholdsINT` oder `ThresholdsDOUBLE`-Objekte initiieren.

Die beiden von der `Thresholds`-Klasse erbinden Klassen übernehmen die Einordnung eines Wertes und die Bewertung desselben auf der Bewertungsskala von 1 bis 10.

Um folglich den Wert eines Qualitätsmaßes zu evaluieren, muss man im gewählten Report-Typ per `getThresholdsINT()` oder `getThresholdsDOUBLE()` das entsprechende Objekt anfordern und kann dann den Wert des Maßes mit `rateValue()` evaluieren. Je nach Art der Grenzwerte (ganzzahlig oder nicht ganzzahlig) unterscheiden sich die Berechnungen, weshalb in zwei `Thresholds`-Objekte unterschieden wird.

Die Faktoren der Wichtungen von guten oder schlechten Bewertungen sind über die Methoden `getGoodFactor()` und `getBadFactor()` abrufbar.

Die Generierung des Reports wird durch ein Templatesystem durchgeführt, welches auf der freien *JavaBY Template Engine* von Alexey Popov basiert³⁷. Mit Hilfe des Templatesystems ist es möglich das Layout der erzeugten HTML-Seiten vom Rest des Systems abzukoppeln (Details in Abschnitt 6.4.3 auf Seite 108).

6.2.7 Webseitenüberwachung mit dem Monitoring Service

Der *Web Measurement Monitoring Service* (WebMMS) gehört nicht konkret zur Web Measurement Suite sondern baut auf dem System auf und benutzt dieselben Ressourcen und Datenquellen. Der Monitoring Service läuft als eigenständige Applikation und verfolgt eigene Ziele. Die Ziele sind die automatisierte Messung von Webseiten in einem regelmäßigen Zeitrahmen und die Überwachung von Webseiten auf sich ändernde Qualitätswerte. Hierzu können bestehende Technologien der Web Measurement Suite wie Agentensystem und Datenstruktur benutzt werden.

Die Verwendung des Monitors ist nicht vom Messprojekt abhängig. Es können gleichzeitig Konfigurationen aus verschiedenen Projekten bearbeitet werden. Deshalb wurden in Abbildung 6.4 auf Seite 77 mehrere Pfeile vom Agentensystem zur Datenbasis gezeichnet, während der Nutzer nur auf einem Messprojekt gleichzeitig arbeiten kann. Nutzer könnten ansonsten zu exzessiv mit den zur Verfügung stehenden Ressourcen umgehen.

Der Nutzer legt fest, welche Messvorgänge gestartet werden sollen³⁸ und leistet somit die konzeptionelle Arbeit. Die Entscheidung, wann eine Messung genau gestartet werden soll, wird durch den Monitor-Agenten getroffen, wodurch ein effektiver Ressourcenverbrauch gewährleistet wird.

³⁷<http://javaby.sf.net>

³⁸Zuordnung von Konfigurationen zum Monitor-Prozess; ist eine Konfiguration bereits gemessen, so wird sie geklont und der Klon wird gemessen.

Der Messvorgang unterscheidet sich nicht vom normalen Messbetrieb. Liegt eine Messkonfiguration im *Gemessen*-Status vor, wird automatisch ein Report generiert. Je nach Ergebnis der Auswertung können bestimmte Ereignisse ausgelöst werden:

- Der Messvorgang für diese Konfiguration kann gestoppt werden, wenn das Ergebnis der Messung negativ ausfiel.
- Der Nutzer wird per E-Mail informiert.
- Ein Popup-Fenster erscheint, welches den Report anzeigt.

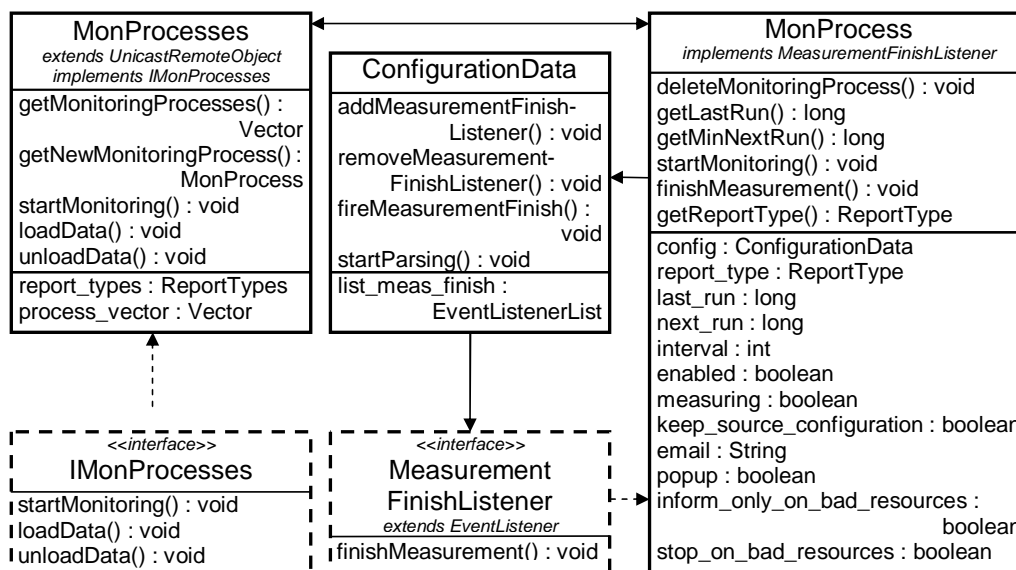


Abbildung 6.18: Verknüpfung zwischen Monitor-Objekten und deren Messkonfiguration

Abbildung 6.18 stellt die Klassendiagramme der Monitor-Objekte *MonProcess* dar und wie diese mit dem Rest des Systems verknüpft sind. *MonProcesses* verwaltet die einzelnen Monitor-Prozesse im Vektor *process_vector*.

Das Interface *IMonProcesses* dient als Schnittstelle zwischen dem Monitor-Agenten und der Klasse *MonProcesses*. Die Funktion *startMonitoring()* veranlasst *MonProcesses* dazu, eine zu messende *MonProcess*-Instanz zu ermitteln. Ein Prozess erfüllt die Kriterien, wenn der Prozess aktiviert ist (*enabled*) und *getMinNextRun()* kleiner als der momentane Zeitstempel ist. *loadData()* lädt die Daten neu aus dem Datenspeicher und synchronisiert durch den Nutzer über die Web Measurement Suite durchgeführte Änderungen. *unloadData()* startet einen Synchronisationsvorgang. Die momentan geladenen Daten werden als unaktuell gekennzeichnet. Somit muss jede Messung unterbrochen und es dürfen keine weiteren Messungen gestartet werden. Durch *loadData()* wird das System wieder gestartet und eine eventuell unterbrochene Messung wiederholt.

Ein Monitor-Prozess besteht aus der Zuordnung zu einer Messkonfiguration und einem Report-Typ zur Auswertung. Dem Objekt `MonProcess` wird durch `startMonitoring()` signalisiert, dass die Messung beginnen soll. Das Objekt leitet die Order an die Methode `startParsing()` in `ConfigurationData` weiter, die den weiteren Verlauf steuert. Sollte die Konfiguration bereits gemessen sein, so wird diese geklont. Auf Wunsch des Nutzers kann die alte Konfiguration bestehen bleiben (`keep_source_configuration`) oder gelöscht werden.

Um das Ende der Messung mitgeteilt zu bekommen, registriert sich `MonProcess` über die Interface-Klasse `MeasurementFinishListener` in der Konfiguration. Diese sendet beim Ende des Messvorgangs einen *Event* an alle bei sich registrierten *Listener* über die Methode `finishMeasurement()`.

Durch den Aufruf `finishMeasurement()` kann das Prozess-Objekt mittels des zugeordneten Report-Typs die Report-Generierung beginnen. Der Nutzer wird informiert, wenn ein Bewertungskriterium im Report-Typ schlecht abgeschnitten hat (z.B. die Anzahl der Fehllinks `RAT_FAIL_LINKS`) oder `inform_only_on_bad_resources` einen negativen Wert hat. Ist das Feld `email` gefüllt, versendet das System eine E-Mail. `popup` öffnet ein Popup-Fenster, in dem direkt der Report angezeigt wird.

Beim negativen Ausgang der Messung kann durch `stop_on_bad_resources` der Prozess deaktiviert werden (`enabled`). Des Weiteren wird das Feld `last_run` aktualisiert und der früheste nächste Messdurchgang (`next_run`) durch die Anzahl der Stunden in `interval` ermittelt.

6.2.8 Datenhaltung

Der letzte Gesichtspunkt in diesem Abschnitt ist die Datenhaltung. Da alle Werte und Einstellungen nicht nur temporär sondern auch nach einem Neustart des Programms noch erhalten sein sollen, müssen diese permanent gespeichert werden.

Es könnte traditionell ein relationales Datenbanksystem genutzt werden, das sich um das Management der Datenspeicherung und Anfragen kümmern würde. Bei der Web Measurement Suite wurde von der Verwendung relationaler Datenbanken abgesehen. Somit wird eine einfache und sofortige Verwendung des Messtools ermöglicht, denn das Datenbankmanagementsystem müsste erst installiert werden. Dadurch wird allerdings auf die ausgereifte Technik verzichtet, die konventionelle Datenbanken mit sich bringen.

Die Web Measurement Suite bevorzugt die semistrukturierte Datenhaltung in XML-Dateien³⁹, die es ermöglicht die bisher beschriebenen Datenobjekte direkt auf eine Speicherstruktur abzubilden.

XML hält Daten hierarchisch in einer Baumstruktur. Ein Knoten kann beliebig viele andere Knoten beinhalten. Die Struktur wird hierbei von einer DTD (*Document Type Definition*) vorgegeben.

Ein Knoten kann entweder andere Knoten (Element-Knoten) oder Inhalt in Form von freiem Text (Text-Knoten) beinhalten. Der Inhalt kann auch als Attribut dem Knoten angehängt sein.

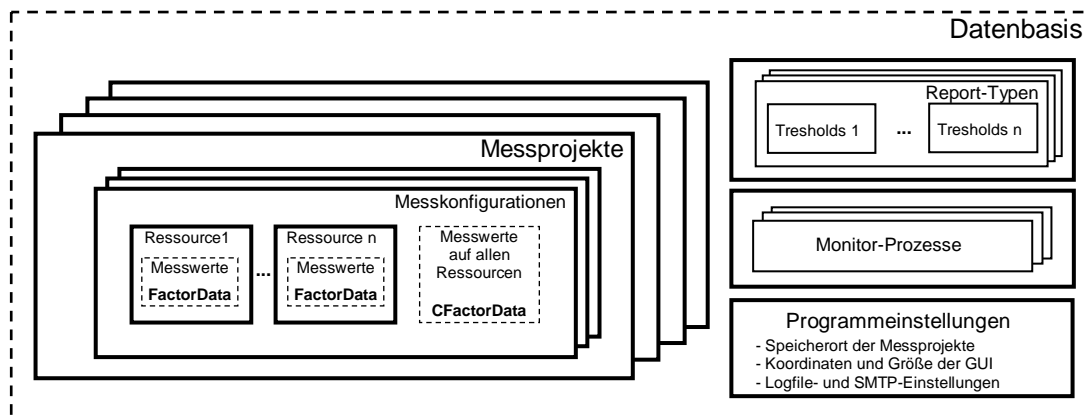


Abbildung 6.19: Datenstruktur des Meßtools

Die verwendete Datenstruktur ist in Abbildung 6.19 dargestellt. Die dicken Rahmen markieren jeweils eine eigene XML-Datei.

Die Programmeinstellungen sind in der Datei `wms_settings.xml` gespeichert; Monitor-Prozesse beherbergt `wms_monitor.xml`; Report-Typen werden in `wms_types.xml` gehalten.

Messprojekte sowie deren Konfiguration und Ressourcen werden durch verschiedene XML-Dateien in einer Verzeichnisstruktur gespeichert. In der Verzeichnisstruktur besitzt jedes Messprojekt ein eigenes Verzeichnis. In diesem ist eine Datei `project.xml` gespeichert, die projektspezifische Daten enthält und Namen von Unterverzeichnissen. In diesen Unterverzeichnissen sind die einzelnen Konfigurationen in den Dateien `conf.xml` zu finden. Diese wiederum verweisen auf die Ressourcen, die in den Dateien der Art `res.1.xml`, `res.2.xml`, ... gespeichert sind.

³⁹Extensible Markup Language

Das folgende Beispiel zeigt den grundlegenden Aufbau eines Messprojektes:

project.xml

```
<project>
  <pname>Name des Projektes</pname>
  <configs>
    <config name="Name der Konfiguration" rev="Revisionsnummer">Pfad</config>
    <config>...</config>...
  </configs>
</project>
```

Pfad/config.xml

```
<config>
  <cname>Name der Konfiguration</cname><rev>Revisionsnummer</rev>
  <resources>
    <resource>res_1.xml</resource>
    <resource>...</resource>...
  </resources>
</config>
```

Pfad/res_1.xml

```
<resource>
  <link depth="2">http://www.server.com</link>
  <mime><e m="text/javascript" s="3.0" /><e m="text/html" s="1.0" /></mime>
  <factors msrresponse="6" extlinks="2" intlinksa="0" intlinksr="1" />
  <references>
    <reference>
      <l>http://www.mycompany.org/main.asp?show=team</l>
      <includings>
        <e r="link" t="a-tag">www.mycompany.org/main.asp?show=team#team</e>
      </includings>
    </reference><reference>...</reference>
  </references>
</resource>
```

Alle Speicherobjekte (`ProjectData`, `ConfigurationData`, `ResourceData`, ...) besitzen einen Konstruktor, der ein XML-Dokument annimmt und dieses auswerten kann, sowie eine Methode, die ein XML-Dokument generiert (`getXMLNode()`).

Wird ein Speicherobjekt `ProjectData` durch den Konstruktor erzeugt, der ein XML-Dokument annimmt, so werden rekursiv alle zu dem Messprojekt gehörigen XML-Dateien in den Speicher geladen. `ProjectData` liest den Projektnamen aus (`pname`) und verarbeitet die Konfigurationsliste `configs`. Für jeden Knoten `config` in der Konfigurationsliste initiiert das Projekt-Objekt ein neues `ConfigurationData`-Objekt.

Konfigurationen erhalten als Argument im Konstruktor den Verzeichnisnamen. Aus diesem Verzeichnis laden sie die Datei `conf.xml` und werten die enthaltenen Datenknoten aus. Den Namen und die Revisionsnummer der Konfiguration erhält das Objekt aus den Knoten `cname` und `rev`. Die einzelnen Links `resource` in der Ressourcenliste `resources` werden an `ResourceData`-Objekte delegiert, die wiederum die XML-Datei laden und die für sich interessanten Daten auswerten. Der Knoten `factors` wird an das `FactorData`-Objekt weitergereicht.

Beim Speichern der Objekte in eine XML-Datei ist der Vorgang genau umgekehrt. Das `ProjectData`-Objekt fordert alle Konfigurationen auf, ihre Daten auf die Festplatte zu schreiben und den Pfadnamen zurückzugeben. Die Konfigurationen wiederum leiten die Speicherforderung an alle Ressourcen-Objekte weiter und bekommen die Dateinamen. Die Daten aus dem `FactorData`-Objekt werden direkt in den Ressourcendateien mitgespeichert.

Damit Daten nicht unnötigen Speicher belegen, wird regelmäßig eine Nachricht an alle Messprojekte gesendet, die die Aufforderung zur Freigabe von Speicher enthält. Ein Messprojekt gibt Speicher frei, wenn es aktuell nicht ausgewählt (`isCurrentProject()` ist falsch) und gespeichert ist sowie keine Messung in diesem Projekt läuft. Es werden alle Konfigurationen und Ressourcen im Speicher freigegeben. Erst bei einer Anforderung dieser oder einer Aktivierung des Projektes werden die fehlenden Daten von der Festplatte nachgeladen und sind somit wieder verfügbar.

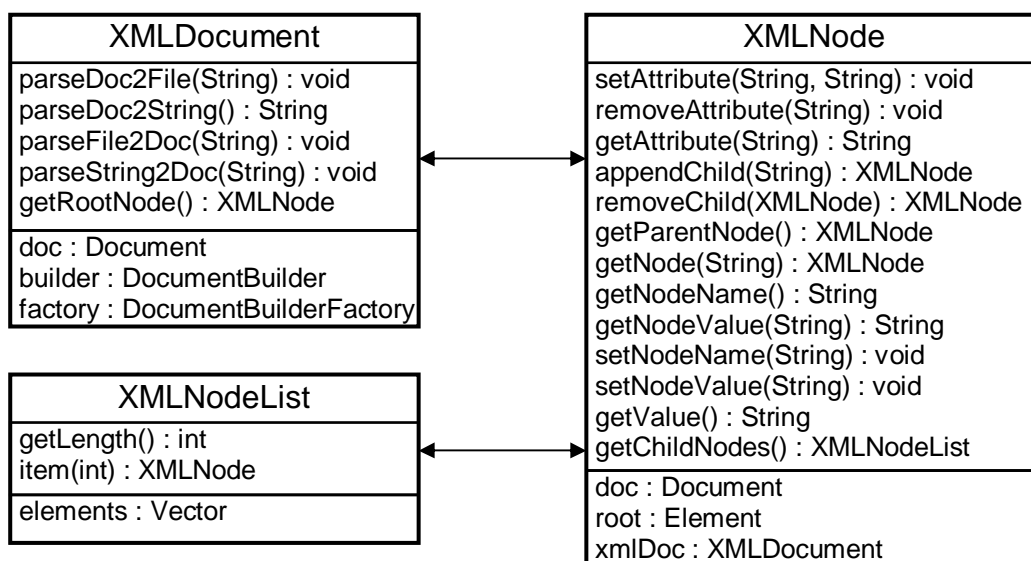


Abbildung 6.20: Klassendiagramme aus dem XML-Paket

Die für die Verarbeitung der XML-Dokumente verwendeten Klassen (siehe Abbildung 6.20) basieren auf dem *Document Object Model* (DOM) aus der Java Klassenbibliothek.

Ein XML-Dokument kann aus einer Zeichenkette erstellt werden oder die Daten aus einer Datei laden. Per `getRootNode()` wird der Wurzelknoten als `XMLNode`-Objekt zurückgegeben. `XMLNode` beinhaltet Methoden, um Attribute und Knoten hinzuzufügen, abzufragen und zu löschen. Per `getChildNodes()` wird ein `XMLNodeList`-Objekt kreiert, welches alle Unterknoten beinhaltet.

6.3 Agenten und Nutzer im System

Mit WebTomix basiert die Web Measurement Suite auf einem vielseitigen, anpassbaren Agentensystem. Aufgrund des geänderten Messbereichs wurde das Agentensystem an die neuen Anforderungen adaptiert und ergänzt. Die Nutzeroberfläche von WebTomix, welche zur Steuerung der einzelnen Agenten benutzt wurde, ist komplett in die GUI der Web Measurement Suite integriert.

Aufgrund der Änderungen an den Agenten und der Integration von WebTomix ist es nicht möglich, dieses Messtool auf eine laufende WebTomix-Instanz aufzusetzen. WebTomix und die Web Measurement Suite können nicht gleichzeitig auf einem Rechner laufen. Die integrierte und abgeänderte WebTomix-Version konkurriert mit einem originalen WebTomix-System, da beide Systeme dieselben Kommunikationsports belegen.

Im Gegensatz zu WebTomix baut die Web Measurement Suite auf Messdaten, die aus früheren Messvorgängen bekannt sind, auf und ermöglicht die Speicherung dieser. WebTomix besitzt zwar die Möglichkeit, die Ergebnisse in eine XML-Datei und eine Datenbank zu schreiben, kann diese Daten aber weder lesen noch auswerten. Die Auswertungskomponente, die in der Web Measurement Suite integriert ist, fehlt ebenso bei WebTomix. Dort werden Balkendiagramme (siehe Abbildung 5.9 auf Seite 63) durch Drittprogramme aus Textdateien generiert.

Nachdem bisher im System vor allem die Datenhaltung beleuchtet wurde (Abbildungen 6.4 auf Seite 77, 6.7 auf Seite 80 sowie 6.19 auf Seite 98), werden nun die Struktur des Systems und die Anwendungsfälle näher betrachtet. Auf grundlegende Implementationsdetails der einzelnen Agenten wird aufgrund der Ähnlichkeit zu WebTomix verzichtet und bei Interesse sei auf die Diplomarbeit von Uwe Schäfer verwiesen[37].

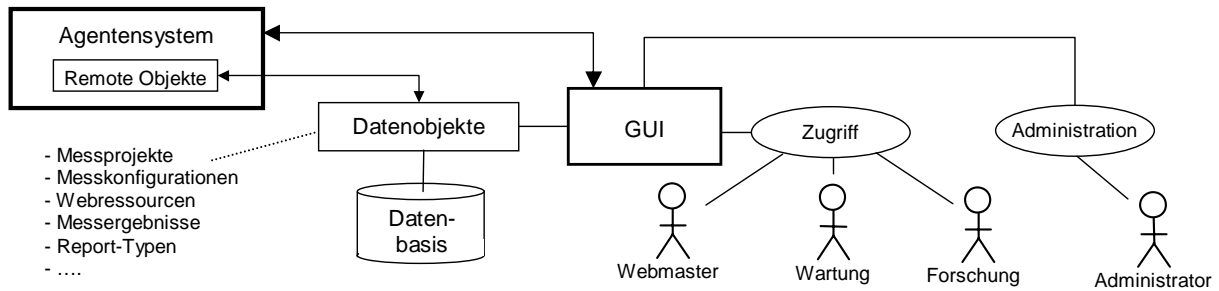


Abbildung 6.21: Zugriff der Nutzer auf das System

Abbildung 6.21 erweitert das bisherige Bild des Systems um mehrere Nutzerrollen. Administrator und Nutzer greifen auf das Agentensystem und die Datenobjekte über eine Benutzeroberfläche (*GUI*) zu.

Der *Administrator* stellt die Software ein. Er richtet die Proxy-Einstellungen, das HTTP-Header-Feld *User-Agent*, die SMTP-Einstellungen und die Logfile-Rahmenrichtlinien ein.

Der *Webmaster* will die ihm anvertraute Webseite auf technische und konzeptionelle Fehler überprüfen und sich informieren lassen, sobald sich die Qualität drastisch verschlechtert.

Der *Wartungsingenieur* durchsucht einen Webseitenverbund nach potentiellen Problemen oder Teilen im Quellcode, die unnötig komplex sind.

Der *Forscher* will anhand des Messtools Trends und Vergleiche zwischen verschiedenen Webauftritten erstellen und den Fortschritt des WWW dokumentieren.

Die Datenbasis in Abbildung 6.21 ist abstrahiert. Das Agentensystem greift nicht direkt auf die Daten zu. Stattdessen bekommen die Agenten Zugriff auf Remote Objekte der echten Datenobjekte. Die Kommunikation wird folglich nicht vom Agentensystem gesteuert sondern von den Remote Objekten selbst. Die Datenobjekte beziehen wiederum ihre Daten aus einer Datenbasis, die bei der Web Measurement Suite durch einen Verbund aus XML-Dateien gelöst wird.

Abbildung 6.22 stellt den Teil des Agentensystems dar, der für das Sammeln der Rohdaten (Messdaten, Fehlermeldungen, Statusinformationen, ...) zuständig ist. Hierbei werden die generischen Agentenrollen aus Abbildung 5.10 (Seite 64) konkretisiert und anhand ihrer Aufgabengebiete miteinander verknüpft.

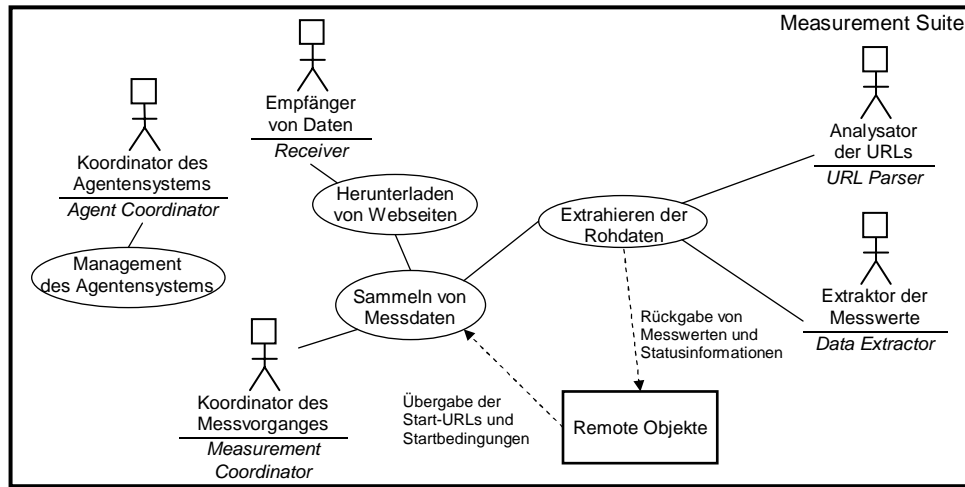


Abbildung 6.22: Agentenstruktur der Messagenten

Der *Agent Coordinator* stellt den zentralen Anlaufpunkt für alle Agenten des Systems dar und verwaltet die Kommunikation. Die Kommunikation der Agenten untereinander wird über *Remote Method Invocation* (RMI) realisiert. RMI ist der Aufruf einer Methode eines entfernten Objektes. „Entfernt“ ist dabei nicht zwingend räumlich sondern unter dem Aspekt verschiedener Laufzeitumgebungen zu sehen.

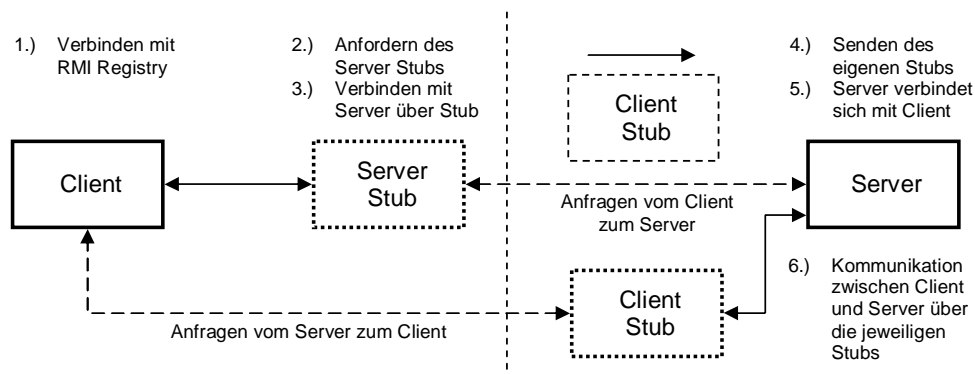


Abbildung 6.23: Kommunikation der Agenten über RMI

Wenn ein Agent A über RMI mit Agent B kommunizieren will, interagiert A mit einer entfernten Schnittstelle von B (*Remote Interface* von B). Dieses Remote Interface definiert Methoden von B, die von nicht lokalen Objekten aufgerufen werden können. Durch den Namensdienst *RMI Registry* des RMI-Dienstes erhält A eine Referenz auf B (den *Stub*). Wenn A eine Methode auf dem Remote Interface von B aufruft, erhält der Stub den Aufruf und leitet ihn über das Netzwerk zum entfernten Objekt weiter. Der Stub erwartet auch das Rückgabergebnis. Der Kommunikationsaufbau ist in Abbildung 6.23 erläutert.

Der *Agent Coordinator* hat die Aufgabe (falls noch nicht existent) die RMI Registry aufzubauen und seinen Namen in dieser zu binden. Die anderen Agenten fragen in der RMI Registry die Referenz zum Koordinator ab und registrieren sich bei diesem. Der Koordinator verwaltet alle verbundenen Agenten. Der Vorteil ist, dass bei der Suche nach speziellen Agenten nicht jedesmal eine Namensabfrage in der RMI Registry vorgenommen und die Referenz gebunden werden muss. Der suchende Agent fordert einfach mittels der bestehenden Verbindung zum *Agent Coordinator* gebundene Referenzen zu den gewünschten Agenten ab.

Der *Measurement Coordinator* verwaltet alle Messvorgänge. Er sammelt alle offenen Vorgänge und beauftragt die „Arbeiter“-Agenten die verschiedenen Ressourcen herunterzuladen oder Messwerte zu extrahieren. Da gleichzeitig verschiedene Konfigurationen gemessen werden können, ist der *Measurement Coordinator* in der Lage, die Arbeit an verschiedene Agenten desselben Typs zu verteilen, so dass die Arbeitslast pro Agent möglichst gering ist. Deadlocks (festgefahrene Messvorgänge) werden durch Timeout-Verfahren erkannt und dann abgebrochen.

Agenten des Typs *Receiver* haben die Aufgabe Webseiten und Dokumente aus dem Internet anzufordern. Hierzu bauen sie eine Verbindung zu einer URL auf und fordern über Verbindungsprotokolle die benötigten Ressourcen an. Die Web Measurement Suite unterstützt z.B. mit dem *HTTPReceiver*-Agenten HTTP. Um Daten von einem FTP-Server zu messen, müsste das Agentensystem um einen *Receiver*-Agenten erweitert werden, der FTP unterstützt.

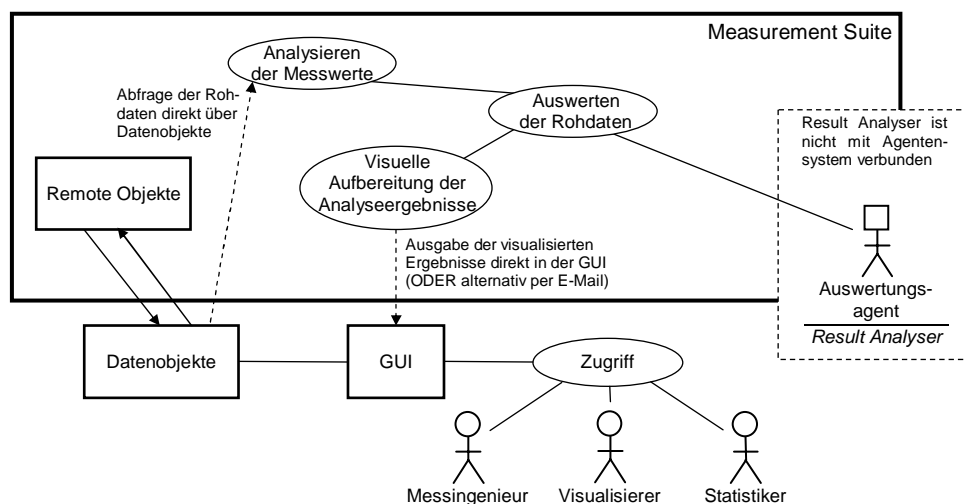


Abbildung 6.24: Auswertungskomponente

URL Parser- und **Data Extractor**-Agenten übernehmen die Extraktion der Rohdaten. Die gesammelten Messwerte werden durch die Remote Objekte zur Datenbasis zurückgeschickt.

Abbildung 6.24 erweitert das Agentensystem um die Auswertungskomponente. Der **Result Analyser** fasst die Funktionalität der *ReportFactory* zusammen, die in Abschnitt 6.2.6 ab Seite 90 eingeführt wurde.

Der *Result Analyser* ist kein direkter Teil des Agentensystems, da er keine Verbindung zum *Agent Coordinator* aufbaut. Für jede Report-Generierung wird das Paket neu initiiert. Die Messwerte und sonstigen Rohdaten können direkt über die Datenobjekte abgerufen werden. Der generierte Report wird in der GUI angezeigt und kann dort von Messingenieuren, Visualisierern und Statistikern ausgewertet werden. Darauf aufbauend können benutzer-spezifische Report-Typen erstellt werden, die die Report-Generierung beeinflussen.

Abbildung 6.25 setzt alle bisherigen Komponenten des Agentensystems zusammen und erweitert es um den Monitoring Service.

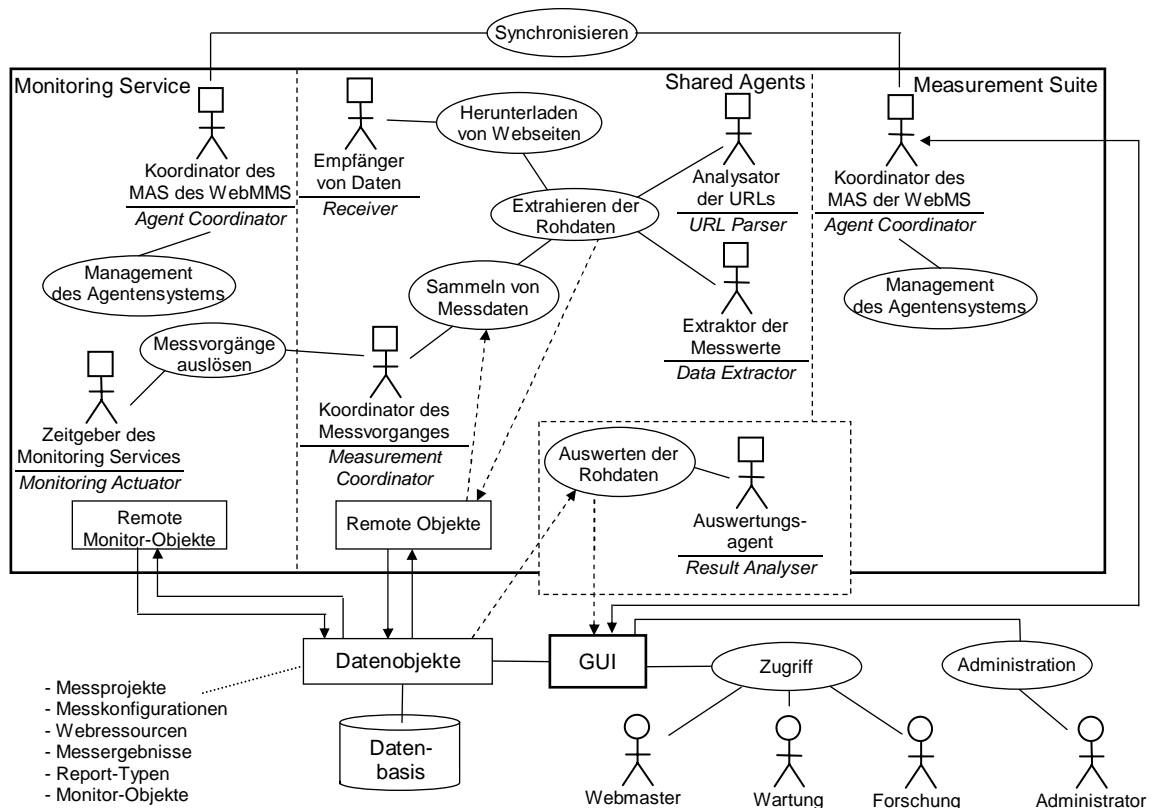


Abbildung 6.25: Der Monitoring Service im vollständigen Agentensystem

Der Monitoring Service startet neben der Measurement Suite ein zweites Agentensystem. Die gemeinsam genutzten Agenten sind hier als *Shared Agents* deklariert. Tatsächlich existiert pro System allerdings jeder Agent separat, um eine Unabhängigkeit zu gewährleisten.

Neben den *Shared Agents* verbindet der gemeinsame Datenspeicher die beiden Systeme. Die Daten werden von beiden Systemen aus der Datenbasis in den flüchtigen Speicher geladen. Da kein *Shared Memory* Konzept eingebunden ist, können sich die Daten in den beiden Systemen unterscheiden und somit unaktuell sein.

Die Lösung ist ein einfaches Synchronisationsprinzip: Es darf nur ein System gleichzeitig auf die Daten zugreifen. Es dürfen zwar beide Systeme gleichzeitig laufen, allerdings hat die Measurement Suite aufgrund der Nutzernähe den Vorrang.

Beim Start und bei der Beendigung der beiden Systeme kontaktieren sie den jeweils anderen und melden den eigenen Status. Der Monitor-Agent entscheidet daraufhin, ob über das Interface `IMonProcesses` die Methoden `loadData()` und `unloadData()` aufgerufen werden (siehe Abschnitt 6.2.7 auf Seite 96).

6.4 Implementationsbeschreibung

Durch die Verwendung von WebTomix wurde die Wahl der Programmiersprache auf Java begrenzt. Java verfolgt vier Hauptziele: Objektorientierung, Plattformunabhängigkeit, Vernetzbarkeit und sichere Ausführung auf fernen Computern⁴⁰.

Objektorientierung erlaubt es Dinge aus der realen Welt auf Software abzubilden, wodurch große Softwareprojekte einfacher zu verwalten sind. Ein weiteres Ziel der Objektorientierung ist eine hohe Wiederverwendbarkeit der Klassen.

Der Aspekt der **Plattformunabhängigkeit** erlaubt es, dass die GUI der Software auf einem Windows-Rechner läuft, während einzelne Agenten auf Zweitrechnern mit Linux oder Solaris laufen können.

Vernetzbarkeit wird durch die große Unterstützung aller Aspekte von Netzwerken garantiert. Der im Agentensystem verwendete Transportmechanismus RMI ist ebenso wie HTTP, URL, IP integraler Bestandteil der Programmiersprache Java.

⁴⁰Die **sichere Ausführung** wird durch die Ausführung des Programmcodes in einer abgeschotteten Laufzeitumgebung gewährleistet. Dies ist jedoch hier nicht der Fall, da die Applikation lokal läuft.

Durch die hohe Wiederverwendbarkeit von Java ist es möglich, externe Softwarepakete in die Web Measurement Suite einzubinden und deren Funktionalität zu nutzen.

6.4.1 ICEBrowser SDK

Der integrierte Browser basiert auf dem ICEBrowser SDK (<http://www.icesoft.com/>). Abbildung 6.26 stellt die Komponentenstruktur des Browsers dar. Ob das kommerzielle Paket in das System eingebunden ist, wird über die Fähigkeit von Java Klassen nachzuladen getestet:

```
try {
    Class.forName("ice.pilots.html4.ThePilot");
    Class.forName("ice.pilots.html4.DDocument");
    Class.forName("ice.storm.StormBase");
    browser_enabled = true;
} catch (Exception exc) {
    browser_enabled = false;
}
```

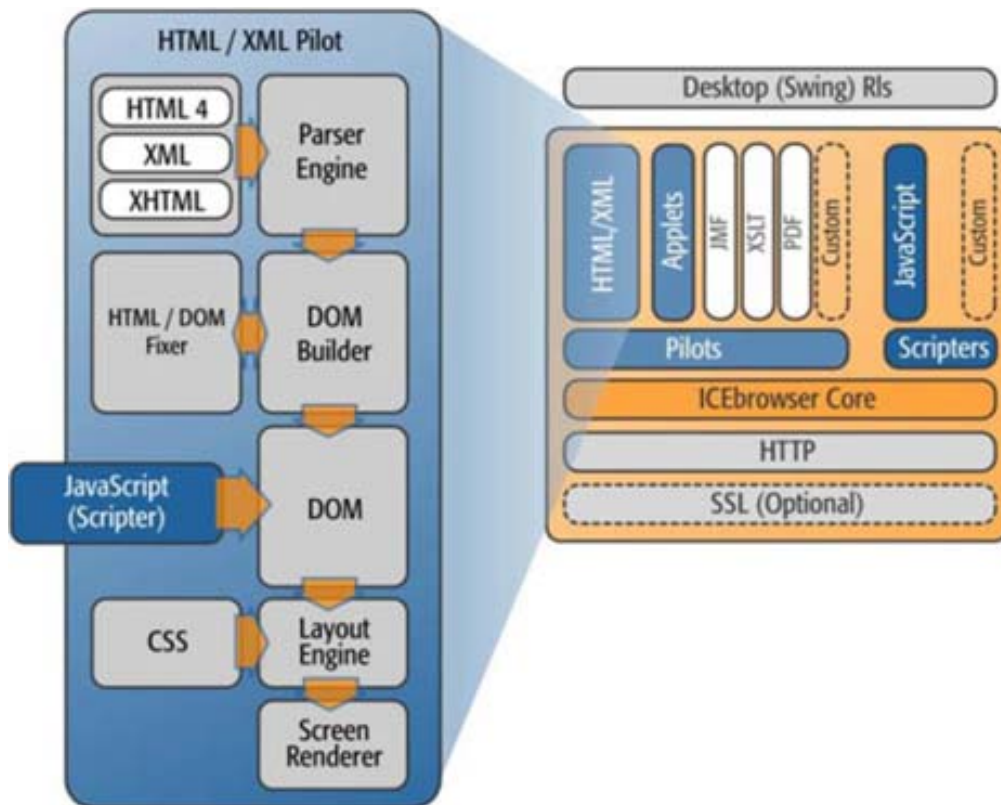


Abbildung 6.26: Komponenten des ICEBrowser SDK [21, S.22]

Der integrierte Browser wird nur für die alternative Auswahl der Start-URLs verwendet. Für die integrierte Hilfe sowie die Anzeige der Reports wird eine in der Java-Klassenbibliothek enthaltene Klasse benutzt, die in der Lage ist, einfaches HTML anzuzeigen (`javax.swing.JEditorPane`). Die Qualität und der Funktionsumfang reichen allerdings nicht an Qualität und Funktion des ICEbrowsers heran.

6.4.2 JavaHelp

Unter <http://java.sun.com/products/javahelp/> bietet Sun Microsystems ein freies Framework an, um Applikationen mit einem Hilfesystem zu ergänzen. Die Hilfeseiten sind als HTML auf der Festplatte abgelegt. Die Indizes und das Inhaltsverzeichnis der Hilfe wird in XML-Dateien gespeichert [43].

Die Web Measurement Suite unterstützt Mehrsprachigkeit. Alle sprachsensitiven Daten sind im Verzeichnis `language/` abgelegt. Dieses enthält Verzeichnisse, die die Sprachdateien enthalten.

In der Datei `language/english/lang.xml` sind alle Texte der GUI gespeichert. Das Verzeichnis `language/english/help/` beinhaltet die HTML-Dateien und die XML-Dateien, die die Indizes und das Inhaltsverzeichnis bilden. `language/english/report/` beinhaltet die Template-Dateien, die für die Generierung der Reports benötigt werden.

Die Sprachen können zur Laufzeit unter *Options > Languages > English* gewechselt werden. Die möglichen Sprachen werden beim Programmstart ermittelt, indem alle Unterverzeichnisse von `language/` erfasst werden.

6.4.3 JBYTE - JavaBY Template Engine

Abschnitt 6.2.6 (Seite 95) hat bereits das Template System JBYTE eingeführt, das zur Generierung der Qualitätsreports benutzt wird.

Spezialisierte Template-Klassen erzeugen HTML-Seiten. Diese Klassen behandeln beispielsweise die einzelnen Webressourcen-Ergebnisseiten, die Zusammenfassung und den Qualitätsreport. Die Template-Klassen greifen bei der Generierung auf spezielle Template-Dateien mit der Dateierdung `tpl` zu. Template-Dateien enthalten Platzhalter, die beim Generieren mit beliebigem Inhalt gefüllt werden können. Im Beispiel der Abbildung 6.27 wird aus der Template-Datei im Punkt 1. mittels des Java-Codes im Punkt 2. die Ausgabe generiert, die im Punkt 3. dargestellt ist.

1. Template-Datei *template.tpl*:

```
<table>
  <tr><th><br /></th><th>{v:title}</th></tr>
<t:entry>
  <tr><td>{v:name}</td><td>{v:value}</td></tr>
</t:entry>
</table>
```

3. Ausgabe:

```
<table>
  <tr><th><br /></th><th>Messwerte</th></tr>
  <tr><td>msr_response</td><td>3.8</td></tr>
  <tr><td>msr_response_all</td><td>7.2</td></tr>
</table>
```

2. Java-Code:

```
Template table = new Template("template.tpl"),
table.set("title","Messwerte");
Template entry = table.get("entry");

entry.set("name", "msr_response");
entry.set("value", "3.8");
table.append("entry", entry);

entry.set("name", "msr_response_all");
entry.set("value", "7.2");
table.append("entry", entry);

System.out.println(table);
```

Abbildung 6.27: Beispielcode für die Template-Engine JBYTE

Integriert sind die einzelnen Template-Klassen in die `ReportFactory`, die den Ablauf der Generierung steuert. Die `ReportFactory` wird für jeden Report neu initiiert. Nach der Generierung wird sie als Zugriffsstruktur für die Ergebnisse der Qualitätsbestimmung verwendet.

Die JBYTE-Engine ist frei unter <http://javaby.sf.net> zu beziehen.

6.4.4 JavaMail

Die Ergebnisse eines Messvorgangs können als Report per E-Mail versendet werden. Als Mail-Engine wurde die ebenfalls von Sun Microsystems angebotene JavaMail API gewählt (<http://java.sun.com/products/javamail/>), die u.a. das Versenden von E-Mails über SMTP erlaubt [44].

Die Klasse `SMTPSend` definiert vier statische Methoden:

- `sendPlainTextEmail(String to, String subject, String message);`
- `sendPlainTextEmail(String name, String from, String user, String password, String mailhost, boolean auth, String to, String subject, String message);`
- `sendMultiMIMEEmail(String to, String subject, String message, File attachment);`
- `sendMultiMIMEEmail(String name, String from, String user, String password, String mailhost, boolean auth, String to, String subject, String message, File attachment);`

Die ersten beiden Methoden (`sendPlainTextEmail`) versenden Textnachrichten und die letzten beiden (`sendMultiMIMEEmail`) E-Mails mit einem Attachment (dem gezippten Report). Unter *Options > Set SMTP Settings...* muss der Administrator SMTP-Einstellungen vornehmen. Diese werden in der ersten und der dritten Methode verwendet, um über den eingestellten SMTP-Server eine E-Mail zu versenden. Bei den anderen beiden Methoden werden die SMTP-Einstellungen manuell übergeben.

6.4.5 Software-Struktur der Web Measurement Suite

Abbildung 6.28 zeigt die Struktur des Messtools anhand der definierten Komponenten. Die Software ist in drei Teile aufgeteilt: die GUI, das Agentensystem und den Datenspeicher.

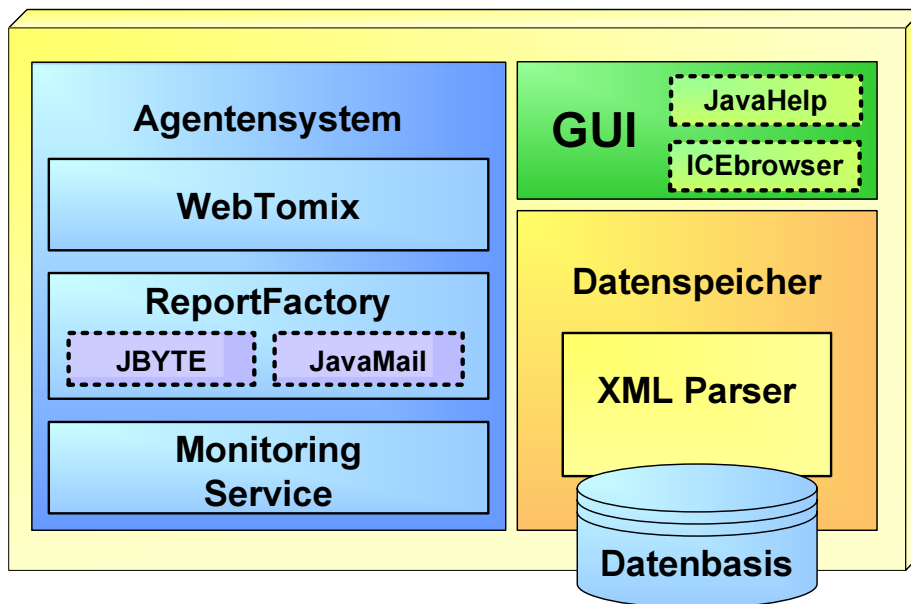


Abbildung 6.28: Komponenten der Web Measurement Suite

Die Eigenleistungen in dieser Diplomarbeit sind

- die in dieser Arbeit nicht näher beschriebene GUI, die eine komfortable und benutzerfreundliche Zugriffsschnittstelle zum System darstellt,
- die dauerhafte Speicherung der Daten in einer semistrukturierten Datenstruktur auf XML-Basis,
- die Erweiterung des Agentensystems von WebTomix um neue Konzepte und Anforderungen und
- die Integration von Drittanbietern zum Erfüllen der Anforderungen.

7 Beispielhafte Anwendung der Web-Qualitätssicherung

Um die gewählten Qualitätsmaße zu evaluieren, werden einige Webseiten aus den Bereichen *Business-to-Consumer* und *Content* mit der Web Measurement Suite gemessen.

Die B2C-Webseiten sind Marktführer in ihrem Bereich. Sie sind in allen Aspekten als qualitativ hochwertig zu betrachten, da die Nutzerakzeptanz über Jahre hinweg unter Beweis gestellt wurde. Gewählt wurden **Amazon** als Vertreter eines Online-Handels für Bücher, CDs, DVD & Videos, u.a.m., und **eBay** als das führende Online-Auktionshaus.

Eine Content-Webseite mit Vorbildcharakter ist **cssZenGarden**. Sie wird immer dann als Beispiel genannt, wenn die korrekte Verwendung von CSS erläutert wird. Standardkonformität wird auch bei der Messung der **W3C**-Webseite vorausgesetzt. Weitere Content-Webseiten sind Communities (**Wikipedia** und **WebUni**), Universitätsportale (**Universität Magdeburg**, **Universität Halle**, **Universität Stanford** und **Fernuniversität Hagen**) sowie die Webseite der **Bundesregierung**.

Die Messungen wurden auf einem Einzelplatzrechner mit T1-Anschluss unter der Javaversion 1.5 durchgeführt und in separaten Messprojekten gespeichert. Je nach Umfang einer Webseite wurden verschiedene Startbedingungen gewählt. Die Projekte wurden jeweils in einem Durchgang gemessen, da sich qualitative Eigenschaften bei mehrmaliger Wiederholung in Messreihen nicht verändern. Eine regelmäßige Beobachtung einer Webseite mit dem Monitor-Agenten wird in der Community WebUni.de durchgeführt.

Alle Projekte werden im Folgenden mit dem Standard-Report-Typ ausgewertet, wodurch keine der Webseiten übermäßig gut ist. Je nach Anwendungsfeld fallen Qualitätsmerkmale bei einem Messprojekt weg oder werden stärker betont. Eine Evaluation mit durch Qualitätszielbestimmung angepassten Report-Typen würde die vom Webmaster der Seite gewünschte Qualität daher besser darstellen. Der Standard-Report-Typ lässt hingegen einen Vergleich zu. Durch die Messungen mit den Beispielen werden zudem die standardmäßigen Grenzwerte justiert und der Standard für dieses Messverfahren festgelegt.

World Wide Web Consortium

Das W3C⁴¹ ist das Gremium für alle Standards im WWW. Durch die Vielfalt der vom W3C spezifizierten Technologien ist eine rekursive Messung der Seite des W3C nicht möglich. Die in diesem Messprojekt ausgewählten Startressourcen wurden über den internen Browser hinzugefügt, weshalb in Tabelle 7.1 die maximale Linktiefe null ist.

Start-URL	Linktiefe	404-Test	StayOnHost	StayInDir
http://www.w3.org/	0	ja	ja	ja
Barrierefreiheit (WAI/)	0	ja	ja	ja
CSS (Style/CSS/)	0	ja	ja	ja
DOM (DOM/)	0	ja	ja	ja
HTML (Markup/)	0	ja	ja	ja
HTTP (Protocols/)	0	ja	ja	ja
RDF (RDF/)	0	ja	ja	ja
XML (XML/)	0	ja	ja	ja

Tabelle 7.1: Startbedingungen Messprojekt W3C

Bei der Bewertung wurden 73,4% der Qualitätsmaße als gut, 13,3% als schlecht und 13,3% als normal bewertet. Von den 8 gemessenen Webseiten schnitten zwei gerundet mit gut (1x **2**, 1x **3**) und der Rest mit normal (4x **4**, 1x **5**, 1x **6**) ab. Aus den ungerundeten Bewertungen ergibt sich ein Durchschnitt von **4,02** (*Fair/Good*).

Die Webseite des W3C (<http://www.w3.org/>) fällt trotz der großen Anzahl an Hyperlinks (150) positiv auf, da nur 12,7% (19) externe Referenzen haben (*RAT_EXT_LINKS*). Nur 0,8% der restlichen 131 Links sind relativ referenziert (*RAT_INT_LINKS_A*). Bei einem zugegebenermaßen sehr unwahrscheinlichem Domainwechsel müssen die Referenzen nicht separat angepasst werden, wodurch der Wartungsaufwand verringert wird.

Das W3C verhält sich selbstverständlich standardkonform. Laut HTML-Spezifikation sind alle HTML-Tag-Identifikatoren (ID-Attribut) eindeutig, so dass die drei Qualitätsmaße ****_NON_UNIQUE_ID* nicht gewertet werden. Zudem sind im Rahmen der Barrierefreiheit 100% der Bilder mit Alternativtexten angegeben *RAT_IMG_AOT*, so dass spezielle Textbrowser Blinden den Zweck des Bildes vorlesen können. Eher benutzerfreundlich ist der Fakt, dass keines der Bilder dessen Alternativtexte per ALT und TITLE-Attribut anbieten. Nicht alle Browser stellen den Text des ALT-Attributes dar, wenn der Mauszeiger über dem Bild stehen bleibt. Dies ist jedoch nicht vom Standard gefordert und somit vom Webmaster während der Qualitätszielbestimmung anzupassen. <http://www.w3.org/> ist mit **3,27** die qualitativ zweitbeste Webseite im Projekt.

⁴¹<http://www.w3.org/>

Die schlechteste Webseite im Messprojekt ist mit **5,65** <http://www.w3.org/Style/CSS/>. Diese Seite verlinkt insgesamt 296 Seiten. Mit 73,6% (218) sind extrem viele externe Referenzen verlinkt. *RAT_EXT_LINKS* wurde dementsprechend mit **9** schlecht bewertet, da viele externe Hyperlinks viel Wartungsaufwand oder mangelnde Aktualität resultieren. In diesem Fall weist die Kennzahl *RAT_FAIL_LINKS* mit 8,8% auf den Missstand hin, dass 26 verlinkte Webdokumente nicht erreicht werden konnten.

Obwohl <http://www.w3.org/Style/CSS/> die Spezifikationen des W3C zu CSS darstellt, fällt die schlechte Bewertung des Maßes *MSR_CSS* auf, die mit einem normalisierten Wert von 2,67 CSS-Klassen pro 10 KByte Quellcode mit **8** schlecht bewertet wurde. Das Qualitätsmaß zählt die Anzahl der in HTML-Tags verwendeten CLASS-Attribute mit separat definierten Stylesheetklassen. Das Beispiel zeigt den Nachteil des Maßes. Da die Seite einen sehr einfachen Aufbau hat, sind alle Stylesheetinformationen direkt den einzelnen HTML-Tags zugeordnet (*< H1 >*, *< H2 >*, *< P >*, ...). Eine für ein HTML-Tag über das CLASS-Attribut separat definierte Stylesheetklasse wird selten benutzt. Daraus ist zu schlussfolgern, dass dieses Qualitätsmaß bei großen, einfachen Dokumenten ungenau ist.

cssZenGarden: The Beauty in CSS Design

ZenGarden zeigt Webautoren beispielhaft, wie man Inhalt und Layout strikt trennen kann. Die Idee hinter ZenGarden besteht darin, dieselbe Seite mit unterschiedlichen CSS-Dateien vollkommen unterschiedlich darzustellen. Mit knapp 450 verschiedenen Oberflächen, bestehend aus einer CSS-Datei und verschiedenen Bildern, stellt sich eine graphische Vielfalt dar, die die Verwendung von Stylesheets eindrucksvoll bestätigt.

Start-URL	Linktiefe	404-Test	StayOnHost	StayInDir
http://www.csszengarden.com/	3	ja	ja	ja

Tabelle 7.2: Startbedingungen Messprojekt cssZenGarden

Mit den in Tabelle 7.2 definierten Startbedingungen ergibt sich eine Gesamtqualität von **2,71** (*Good*). Bei der Bewertung wurden 79,2% der Qualitätsmaße als gut, 3,1% als schlecht und 17,7% als normal bewertet. Von den 62 gemessenen Webseiten schnitten alle gut ab (59x **2**, 3x **3**).

Erwartungsgemäß befinden sich alle Messungen in demselben Bereich um **2.4** mit Schwankungen, die aus den Maßen *MSR_RESPONSE* und *MSR_RESPONSE_ALL* resultieren.

Wikipedia: Die freie Enzyklopädie

Wikipedia⁴² ist ein Projekt zur Erstellung einer Enzyklopädie. Es ist gleichzeitig der Name für im Rahmen dieses Projekts im Aufbau befindliche Enzyklopädien in zahlreichen Sprachen. Wikipedia verwendet die Wiki-Technik als Werkzeug für die Zusammenarbeit zwischen Autoren. Wikis sind Websysteme, die es jedem Internetnutzer erlauben, ohne weitere Anmeldung mitzuarbeiten. Jeder kann darin neue Artikel schreiben oder bestehende verbessern [49].

Start-URL	Linktiefe	404-Test	StayOnHost	StayInDir
http://www.wikipedia.de/	1	ja	ja	ja

Tabelle 7.3: Startbedingungen Messprojekt Wikipedia

Da Wikipedia als Enzyklopädie bereits in der zweiten Suchtiefe weit verzweigt, wurden die Startbedingungen in Tabelle 7.3 auf die Linktiefe 2 mit 404-Test begrenzt. 80,8 % der Qualitätsmaße sind als gut, 5,4 % als schlecht und 13,8 % als normal bewertet. Von den 17 gemessenen Webseiten schnitten 15 gut (9x **2**, 6x **3**) und 2 normal (1x **5**, 1x **7**) ab. Insgesamt wurde Wikipedia mit **2,61** (*Good*) gewertet.

`/w/wiki.phtml?title=Diskussion:Hauptseite&action=edit` ist die am schlechtesten bewertete Seite. Es ist die Diskussion zur Startseite. Die Teilnahme an der Diskussion ist sehr stark und somit ist die Webseite mit 140 KByte HTML-Quelltext sehr groß. Dies zeigt sich in den negativen Bewertungen aller Performancemaße.

Da auf dieser Seite sehr viel Freitext vorhanden ist, steigt auch die Anzahl der nicht-standardkonformen Zeichen. Die Webseite wird durch *MSR_SPEC_CHARS* mit insgesamt 2856 gefundenen Sonderzeichen als schlecht (**10**) bewertet.

WebUni: Studenten Community

WebUni.de⁴³ ist ein Studentenportal aus der Region Magdeburg mit dem Schwerpunkt Community. Es wurde im Herbst 2003 als ein Gemeinschaftsprojekt der Landeshauptstadt Magdeburg, der Otto-von-Guericke-Universität Magdeburg, der Hochschule Magdeburg-Stendal, des Studentenrates der Otto-von-Guericke-Universität Magdeburg und des Studentenrates der Hochschule Magdeburg-Stendal gestartet. WebUni stellt Diskussionsforen, eine Materialienbörse zum Verteilen von Studiumsunterlagen sowie ein Schwarzes Brett zur Verfügung und bindet das universitätsinterne Informationssystem UnivIS ein.

⁴²<http://www.wikipedia.de/> bzw. dessen Weiterleitung <http://de.wikipedia.org/wiki/Hauptseite/>

⁴³<http://www.webuni.de/>

Start-URL	Linktiefe	404-Test	StayOnHost	StayInDir
http://www.webuni.de/	1	nein	ja	ja

Tabelle 7.4: Startbedingungen Messprojekt WebUni

WebUni hat durch den sehr hohen Verlinkungsgrad extrem viele Referenzen, weshalb die Linktiefe in Tabelle 7.4 sehr niedrig eingestellt ist und keine 404-Tests durchgeführt werden. 404-Tests sind allerdings auch nicht notwendig, da die Webseite so aufgebaut ist, dass die verschiedenen Inhalte des Content-Management-Systems über Query-Argumente in der URL unterschieden werden. Bei falsch übermittelten Query-Argumenten werden statt Fehlermeldungen alternative Seiten angezeigt.

Insgesamt wurde WebUni mit **5,53** (*Fair*) eher durchwachsen bewertet. 63,7% der Qualitätsmaße wurden als gut, 23,4% als schlecht und 12,9% als normal bewertet. Von den 124 gemessenen Webseiten schnitt nur eine gerundet mit gut (1x **3**), 27 mit schlecht (27x **8**) und der Rest mit normal (64x **4**, 4x **6**, 28x **7**) ab.

Positiv fiel das Maß *MSR_LINK_DENS* auf, welches die interne Linkdichte misst. 99,5% der gefundenen Hyperlinks referenzieren auf eine interne Webseite von WebUni. Dies belegt eine durchgängige und konsistente Navigation im Sinne der Benutzerfreundlichkeit.

Ein großes Problem von WebUni ist die Größe des HTML-Quelltextes, wodurch die Bewertungen aller Performancemaße beeinflusst werden. Schuld ist die Verwendung von HTML-Tabellen als Positionierungs- und Layoutmittel. Rund 50 Tabellen mit einer maximalen Verschachtelungstiefe 6 erhöhen die Komplexität des HTML-Quelltextes und verschlechtern das Verhältnis zwischen HTML und Inhalt auf Kosten der Seitengröße.

WebUni wird aufgrund des schlechten Ergebnisses der Messung derzeit XHTML-konform überarbeitet. Statt der Positionierung mit HTML-Tabellen wird das Layout ähnlich zu cssZenGarden über CSS realisiert. Durch diese Maßnahme verringert sich die Seitengröße um über 40% und mit einer Erhöhung der Qualität der Webseite ist zu rechnen.

Die Messung von WebUni zeigt den Nachteil des Qualitätsmaßes *RAT_INT_LINKS_A*, welches ein Verhältnis von Hyperlinks mit internen Referenzen darstellt. Bei diesem Maß werden die Referenzen mit absoluter und relativer Pfadangabe verglichen. WebUni gibt alle Pfadangaben absolut an, ersetzt jedoch den Host in der URL dynamisch mit der derzeit verwendeten Domain. Das Messtool bewertet die absolute Pfadangabe jedoch als schlecht, da durch eine Domainänderung Wartungsaufwand entstehen würde.

Bundesregierung

Auf dieser Webseite berichtet die deutsche Bundesregierung über Neuigkeiten, Beschlüsse und Gesetzesänderungen. Die Folge sind viele, große Webdokumente. Mit den in Tabelle 7.5 definierten Startbedingungen ergibt sich eine mit **5,62** (*Fair*) bewertete Webseite.

Start-URL	Linktiefe	404-Test	StayOnHost	StayInDir
http://www.bundesregierung.de/	1	ja	ja	ja

Tabelle 7.5: Startbedingungen Messprojekt Bundesregierung

55,4% der Qualitätsmaße wurden als gut, 17,5% als schlecht und 27,1% als normal bewertet. Von den 50 gemessenen Webseiten schnitten zwei gerundet mit gut (2x **3**), eine mit schlecht (1x **8**) und der Rest mit normal (10x **5**, 26x **6**, 11x **7**) ab.

Das Negativbeispiel in diesem Messprojekt ist die Seite `/aktuell.html` mit einer Bewertung von **8,19**. Die Bewertung resultiert aus über 2000 Hyperlinks.

Universitätsseiten

Gemessen wurden die Webauftritte der Universitäten Magdeburg, Halle, Stanford sowie der Fernuniversität Hagen. Tabelle 7.6 listet die jeweiligen Startbedingungen auf. Die große Linktiefe der Fernuniversität Hagen resultiert aus der Frameset-Struktur.

Start-URL	Linktiefe	404-Test	StayOnHost	StayInDir
http://www.uni-magdeburg.de/	2	ja	ja	ja
http://www.uni-halle.de/	2	ja	ja	ja
http://www.stanford.edu/	2	ja	ja	ja
http://www.fernuni-hagen.de/	4	ja	ja	ja

Tabelle 7.6: Startbedingungen Messprojekte von Universitäten

Aus den gegebenen Startbedingungen erfolgen die in Tabelle 7.7 ausgewiesenen Ergebnisse.

Stadt	Bewertung	Qualitätsmaße			Webseiten			
		gut	schlecht	normal	gut	schlecht	normal	Fehler
Magdeburg	3,61 (<i>Fair/Good</i>)	79,3%	9,6%	11,1%	26,6%	0%	69,8%	3,6%
Halle	3,26 (<i>Good</i>)	73,7%	8,6%	17,7%	66,5%	0%	33,5%	0%
Stanford	6,31 (<i>Fair</i>)	61,4%	18,8%	19,8%	2%	0%	98%	0%
Hagen	2,66 (<i>Good</i>)	85,7%	8,9%	5,4%	92%	0%	8%	0%

Tabelle 7.7: Messergebnisse von Universitäten

Die Universitätsseite von Stanford ist geprägt durch teils fehlende Standardkonformität und Barrierefreiheit sowie große HTML-Quelltexte und viele Hyperlinks zu anderen Universitäten.

Magdeburg hat sowohl qualitativ gut ausgeprägte als auch qualitativ schlecht bewertete Seiten. Ein einheitliches Bild kann nicht gebildet werden. Auffallend ist die Häufigkeit an fehlerhaften Referenzen.

Sowohl Halle als auch Hagen schneiden bei der Bewertung gut ab. Allgemein besitzen Universitätsseiten jedoch kein ausgeprägtes Qualitätsbild. Da viele verschiedene Bereiche auf dem Universitätsportal zusammengesetzt werden, beeinflussen die Arbeiten unterschiedlicher Webmaster die Bewertung.

Sinnvoll ist die Messung allerdings erst, wenn gezielt Qualitätsmerkmale auf dem Universitätsportal gemessen werden. Eine Bewertung des Gesamtportals ist durch die dezentrale Wartung desselben nicht möglich.

Ein einheitliches Gesamtkonzept mit einem Content-Management-System als Basis würde grundsätzlich bei jeder Webseite mit mehreren Webmastern sowohl Wartungsaufwand und Kosten als auch Benutzerfreundlichkeit und Wiedererkennungswerte verbessern.

Amazon und eBay

Die Startbedingungen von Amazon und eBay sind in Tabelle 7.8 aufgeführt. Da eBay sehr stark verzweigt ist, wurden vier Beispielseiten ausgesucht: die Startseite, eine Artikelrubrik, eine Artikelansicht und der Bereich My eBay.

Start-URL	Linktiefe	404-Test	StayOnHost	StayInDir
http://www.amazon.de/	1	ja	ja	ja
http://www.ebay.de/	0	ja	nein	nein
Rubrik Computer	0	ja	nein	nein
Artikelansicht	0	ja	nein	nein
My eBay	0	ja	nein	nein

Tabelle 7.8: Startbedingungen Messprojekte von B2C-Seiten

B2C-Webseiten basieren im Regelfall auf Content-Management-Systemen. Artikel werden aus Datenbanken ausgelesen und performance- und berechnungsoptimal in HTML-Seiten generiert. Deshalb wird oftmals auf Overhead in Form von Standardkonformität und Barrierefreiheit verzichtet. Die Messergebnisse zeigt Tabelle 7.9.

Anbieter	Bewertung	Qualitätsmaße			Webseiten			
		gut	schlecht	normal	gut	schlecht	normal	Fehler
Amazon	6,77 (<i>Fair/Poor</i>)	54,9 %	26,6 %	18,5 %	0 %	0 %	100 %	0 %
eBay	6,55 (<i>Fair/Poor</i>)	54,1 %	29,1 %	16,8 %	0 %	66,7 %	33,3 %	0 %

Tabelle 7.9: Messergebnisse von B2C-Seiten

Zusammenfassung

Fasst man alle Messergebnisse in einer Tabelle zusammen ergibt sich folgende Auflistung:

Webseite	Bewertung	Qualitätsmaße			Webseiten			
		gut	schlecht	normal	gut	schlecht	normal	Fehler
Wikipedia	2,61 (<i>Good</i>)	80,8 %	5,4 %	13,8 %	88,2 %	0 %	11,8 %	0 %
Uni Hagen	2,66 (<i>Good</i>)	85,7 %	8,9 %	5,4 %	92 %	0 %	8 %	0 %
cssZenGarden	2,71 (<i>Good</i>)	79,2 %	3,1 %	17,7 %	100 %	0 %	0 %	0 %
Uni Halle	3,26 (<i>Good</i>)	73,7 %	8,6 %	17,7 %	66,5 %	0 %	33,5 %	0 %
Uni Magdeburg	3,61 (<i>Fair/Good</i>)	79,3 %	9,6 %	11,1 %	26,6 %	0 %	69,8 %	3,6 %
W3C	4,02 (<i>Fair/Good</i>)	73,4 %	13,3 %	13,3 %	25 %	0 %	75 %	0 %
WebUni	5,53 (<i>Fair</i>)	63,7 %	23,4 %	12,9 %	0,8 %	21,8 %	77,4 %	0 %
Bundesregierung	5,62 (<i>Fair</i>)	55,4 %	17,5 %	27,1 %	4 %	2 %	96 %	0 %
Uni Stanford	6,31 (<i>Fair</i>)	61,4 %	18,8 %	19,8 %	2 %	0 %	98 %	0 %
eBay	6,55 (<i>Fair/Poor</i>)	54,1 %	29,1 %	16,8 %	0 %	66,7 %	33,3 %	0 %
Amazon	6,77 (<i>Fair/Poor</i>)	54,9 %	26,6 %	18,5 %	0 %	0 %	100 %	0 %

Tabelle 7.10: Zusammenfassung der Messergebnisse

Es lässt sich der Schluss ziehen, dass optimierte moderne Seiten, die sich an Standards halten, die Liste anführen. Universitäten befinden sich infolge der heterogenen Struktur im Mittelfeld. Content-Management-Systeme bilden das Schlusslicht, da die interne Optimierung nicht erkennbar ist und somit keine Aussagen über die verschiedenen Qualitätsaspekte getroffen werden können.

8 Zusammenfassung und Ausblick

Das Internet ist als Standard für den weltweiten Datenaustausch etabliert und stellt mit dem World Wide Web eine Informationsplattform bereit, mit der die Daten von jedem Computer mit Internetanschluss abgerufen werden können. Die Daten werden hierbei über Webseiten ausgeliefert und präsentiert.

Um den Umgang mit Webseiten zu verbessern, stellt die *Web Measurement Suite* ein System zur Verfügung, welches die Bewertung und Überwachung der Qualität von Webseiten ermöglicht.

Ergebnis

Die in dieser Arbeit entwickelte Web-Qualitätssicherung wurde durch die folgenden Schritte realisiert:

- Ermittlung möglicher Qualitätsaspekte von Webseiten
- Aufstellen eines Qualitätsmodell basierend auf ISO 9126
- Abbildung von messbaren Eigenschaften auf die Merkmale des Qualitätsmodells
- Konzeptionierung einer Bewertungsformel zur Bestimmung der Qualität
- Adaption des Agentensystems des Web-Tomographen WebTomix sowie Modellierung weiterer Agentenrollen
- Entwurf und Implementation der Web-Qualitätssicherung

Eine benutzerfreundliche graphische Oberfläche erlaubt es den Anwendern der Web-Qualitätssicherung ohne Vorkenntnisse Websysteme zu messen und qualitativ zu bewerten. Als technische Grundlage für eine Messung wird das Agentensystem von WebTomix verwendet, welches vollständig integriert und um weitere Aspekte und Fähigkeiten erweitert wurde. Durch die Integration von WebTomix ist es möglich, das Agentensystem dynamisch zusammenzustellen und es um Erkennungsverfahren, Crawlingstrategien sowie Verarbeitungsmechanismen zu erweitern.

Die Web Measurement Suite erwies sich als geeignet, um Webseiten qualitativ zu bewerten. Ein allgemein gültiger Messansatz für jeden Webseitentyp ist nicht praktikabel, da die qualitativen Präferenzen sich bei Verantwortlichen sehr unterscheiden. Aus diesem Grunde zeigte sich der Ansatz als zweckmäßig, eine Anpassung des Qualitätsmaß-Kataloges anzubieten. Die Web Measurement Suite bietet dies im Rahmen einer Qualitätszielbestimmung mit sogenannten Report-Typen an, die die Anpassung der Ausprägungen von Qualitätseigenschaften zulässt.

Im Rahmen einer beispielhaften Anwendung des Messprogramms zeigte sich eine grobe Zuordnung verschiedener Webseitentypen auf die Bewertungsskala. So schnitten moderne, standardkonforme Webseiten am besten ab. Universitätsportale wurden durch ihre heterogene Struktur und durch unterschiedliche Verantwortungsbereiche im Mittelfeld eingeordnet, mit abweichenden Tendenzen, die durch den Grad der Heterogenität bestimmt sind. Eher schlecht wurden große kommerzielle Websysteme sowie Communities bewertet, deren qualitative Optimierung durch Content-Management-Systeme aus dem WWW nicht ersichtlich ist.

Ausblick

Ein großes Problem der Web Measurement Suite ist die Gebundenheit an das WWW und damit den Schwankungen des weltweiten Netzwerkes. In Bezug auf das Agentensystem besteht die Möglichkeit der Optimierung zugunsten der Mobilität der Agenten. Hierzu wird die Einbettung von Messplattformen in Webservern angeregt, die den Lebensraum für mobile Agenten bereitstellen.

Die Web Measurement Suite kann weiterhin in ihrer Fähigkeit der Auswertung erweitert werden. Eine konkrete Bewertung einer Webseite nach dem Qualitätsmodell mit der Darstellung der Ausprägung verschiedener Qualitätseigenschaften könnte einen tieferen Einblick in den Optimierungsbedarf einer Webseite bieten. Durch diese Änderung könnte gezielt nach nutzerunfreundlichen oder wartungsintensiven Teilen des Websystems geforscht und die Schwachstellen somit frühzeitig gefunden werden. Auf der Basis der Diplomarbeit von Sören Heider und Raik Pauer [18] ist zusätzlich ein Export der Messdaten in ein XML-Format möglich, um eine 3D-Visualisierung der Messergebnisse als drehbaren Graph zu erhalten.

Im Rahmen der Erweiterung des Messkatalogs sollten Validatoren des W3C fest integriert und somit die Standardkonformität der gemessenen Webseiten gefördert werden. Eine weitere Anpassung und Skalierung des Messkataloges ist durch Messungen wünschenswert.

A Abkürzungen

AOP	Agent-oriented programming
API	Application Programming Interface
ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
ASP	Active Server Pages
ATM	Asynchronous Transfer Mode
BITV	Barrierefreie Informationstechnik Verordnung
CFML	ColdFusion Markup Language
CGI	Common Gateway Interface
CMS	Content-Management-System
CSS	Cascading Style Sheets
DARPA	Defense Advanced Research Projects Agency
DNS	Domain Name System
DOM	Document Object Model
DTD	Document Type Definition
ECMD	European Computer Manufacturers Association
EP	Entire Measurement Project
FDDI	Fiber Distributed Data Interface
FTP	File Transfer Protocol
GIF	Graphics Interchange Format
GQM	Goal Question Metric
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IANA	Internet Assigned Numbers Authority
ICANN	Internet Corporation for Assigned Names and Numbers
IP	Internet Protocol (<i>IPv4</i> und <i>IPv6</i> sind Version 4 und Version 6 von IP)
IRC	Internet Relay Chat
IRI	Internationalized Resource Identifiers

ISBN	International Standard Book Number
ISO	International Organization for Standardization
IT	Information Technology
J2EE	Java 2 Platform Enterprise Edition
J2SE	Java 2 Standard Edition
JAR	Java Archive
JFIF	JPEG File Interchange Format
JPEG	Joint Photographic Experts Group
JRE	Java Runtime Edition
JSP	Java Server Pages
LOC	Lines of Code
MAS	Multi Agent System
MathML	Mathematical Markup Language
MIME	Multipurpose Internet Mail Extensions
OOP	Object-oriented programming
ORSN	Open Root Server Network
PDF	Portable Document Format
Perl	Practical Extraction and Report Language
PHP	rekursives Akronym für PHP: Hypertext Preprocessor
PICS	Platform for Internet Content Selection
RDF	Resource Description Framework
RFC	Request for Comments
RMI	Remote Method Invocation
RTT	Round Trip Time
sdk	Software Development Kit
SGML	Standard Generalized Markup Language
SMIL	Synchronized Multimedia Integration Language
SMTP	Simple Mail Transfer Protocol
SP	Single Web Page
SSH	Secure Shell
SSI	Server Side Includes
SSL	Secure Sockets Layer
SSL/TLS	Secure Sockets Layer / Transport Layer Security
SVG	Scalable Vector Graphics
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
TLD	Top Level Domain
UDP	User Datagram Protocol

UML	Unified Modeling Language
URI	Uniform Resource Identifier
URN	Uniform Resource Name
URL	Uniform Resource Locator
Usenet	User Network
VBScript	Visual Basic Script
VRML	Virtual Reality Modeling Language
W3C	World Wide Web Consortium
WAI	Web Accessibility Initiative
WAP	Wireless Application Protocol
WaSP	Web Standards Project
WCAG	Web Content Accessibility Guidelines
WebMMS	Web Measurement Monitoring Service
WebMS	Web Measurement Suite
WWW	World Wide Web
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language

B Messwerte und Qualitätsmaße

Die Messwerte und Qualitätsmaße werden innerhalb der *Web Measurement Suite* den folgenden Rubriken zugeteilt:

Sonstiges (falls vorhanden) — Server — Links — Bilder — sonstige Dokumente
— E-Mails — Frames — Webseite — HTML

Der Qualitätsaspekt bezieht sich auf das Qualitätsmodell aus Abbildung 6.1 auf Seite 73.

B.1 Messwerte

Die folgenden Präfixe wurden bei den Messwerten verwendet:

cnt — Bildung des Messwertes durch Zählung einer Eigenschaft

max — Maximaler Wert einer Eigenschaft

msr — Allgemeiner Messwert

sum — Summe von messbaren Eigenschaftswerten

Index

cnt			
cnt_adv_img	127	cnt_frames	128
cnt_adv_links	127	cnt_frames_el	129
cnt_comments	130	cnt_iframes	129
cnt_colspan	130	cnt_img_alt	127
cnt_css	131	cnt_img_alt_title	128
cnt_css_style	131	cnt_img_title	127
cnt_diff_doc	128	cnt_inline	129
cnt_email	128	cnt_int_doc_a	128
cnt_email_g	128	cnt_int_doc_r	128
cnt_ext_doc	128	cnt_int_img_a	127
cnt_ext_img	127	cnt_int_img_r	127
cnt_ext_links	126	cnt_int_links_a	126
cnt_fail_doc	128	cnt_int_links_r	126
cnt_fail_img	127	cnt_non_unique_id	131
cnt_fail_links	126	cnt_rowspan	130
cnt_form	130	cnt_spec_chars	129
cnt_form_elem	130	cnt_style	130
		cnt_tables	130

cnt_td	130	msr_non_unique_id	131
max		msr_query_size	129
max_nested	130	msr_reponse	126
msr		msr_response_all	126
msr_age	129	sum	
msr_comments	130	sum_css	131
msr_content	129	sum_doc_size	128
msr_html	129	sum_img_size	127
msr_inline_size	129	sum_nested	130
msr_loc	129	sum_style_el	130

Messwert	Rubrik	Qualitätsaspekt	Beschreibung
msr_response	Server	Performance, Seitenaufbau	Antwortzeit beim Anfordern der Webseite, verstrichene Zeit beim Downloaden der Webseite (nur der HTML-Teil)
msr_response_all	Server	Performance, Seitenaufbau	Antwortzeit beim Anfordern der Webseite, verstrichene Zeit beim Downloaden der Seite und aller Inline-Elemente (Bilder, CSS, ...)
cnt_ext_links	Links	Benutzerfreundlichkeit, Webstruktur, Stabilität	Anzahl der Hyperlinks ⁴⁴ zu externen Dokumenten ⁴⁵
cnt_int_links_a	Links	Benutzerfreundlichkeit, Webstruktur, Änderbarkeit	Anzahl der Hyperlinks zu internen Dokumenten mit absoluter Pfadangabe ⁴⁶
cnt_int_links_r	Links	Benutzerfreundlichkeit, Webstruktur, Änderbarkeit	Anzahl der Hyperlinks zu internen Dokumenten mit relativer Pfadangabe
cnt_fail_links	Links	fehlerh. Referenzen	Anzahl der Hyperlinks zu Dokumenten, deren Download fehlgeschlagen ist (z.B. wegen 404-Fehlern)

⁴⁴Ein Hyperlink wird hier als ein Link verstanden, der erst nach Anklicken durch den Nutzer geladen wird. Beispiel: ``

⁴⁵Das Webdokument eines externen Hyperlinks ist auf einem anderen Webserver gespeichert als die verlinkende Webseite.

⁴⁶Das Webdokument eines internen Hyperlinks ist auf demselben Webserver gespeichert wie die verlinkende Webseite.

Messwert	Rubrik	Qualitätsaspekt	Beschreibung
cnt_adv_links	Links	Nutzerakzeptanz	Anzahl der Hyperlinks, deren URLs eine Übereinstimmung mit den Werbe-Pattern aus Anhang C ergaben (nur externe Links)
cnt_ext_img	Bilder	Webstruktur, Fremdinhalte, Stabilität	Anzahl der eingebundenen Bilder ⁴⁷ von einem externen Webserver
cnt_int_img_a	Bilder	Webstruktur, Änderbarkeit	Anzahl der mit absoluter Pfadangabe eingebundenen Bilder, die auf dem eigenen Webserver gespeichert sind
cnt_int_img_r	Bilder	Webstruktur, Änderbarkeit	Anzahl der mit relativer Pfadangabe eingebundenen Bilder, die auf dem eigenen Webserver gespeichert sind
cnt_fail_img	Bilder	fehlerh. Referenzen	Anzahl der eingebundenen Bilder, die nicht empfangen werden konnten (z.B. 404-Fehler)
cnt_adv_img	Bilder	Nutzerakzeptanz	Anzahl der eingebundenen Bilder, deren URLs eine Übereinstimmung mit den Werbe-Pattern aus Anhang C ergaben (nur externe URLs)
sum_img_size	Bilder	Seitengröße	Größe aller eingebundenen Bilder in Bytes
cnt_img_alt	Bilder	Barrierefreiheit, Benutzerfreundlichkeit, Standardkonformität	Anzahl aller eingebundenen Bilder mit ALT-Attribute (siehe Barrierefreiheit)
cnt_img_title	Bilder	Kompatibilität, Benutzerfreundlichkeit, Standardkonformität	Anzahl aller eingebundenen Bilder mit TITLE-Attribute (Kompatibilität zum Webbrowser Opera)

⁴⁷Eingebundene Bilder zählen zu den Inline-Elementen einer Seite und werden beim Download der Seite mit angefordert. Beispiel: `` oder Hintergründe

Messwert	Rubrik	Qualitätsaspekt	Beschreibung
cnt_img_alt_title	Bilder	Kompatibilität, Benutzerfreundlichkeit, Standardkonformität	Anzahl aller eingebundenen Bilder mit ALT- und TITLE-Attribute
cnt_ext_doc	sonstige Dokumente	Webstruktur, Fremdinhalte, Stabilität	Anzahl der eingebundenen sonstigen Dokumente ⁴⁸ von einem externen Webserver
cnt_int_doc_a	sonstige Dokumente	Webstruktur, Änderbarkeit	Anzahl der mit absoluter Pfadangabe eingebundenen sonstigen Dokumente, die auf dem eigenen Webserver gespeichert sind
cnt_int_doc_r	sonstige Dokumente	Webstruktur, Änderbarkeit	Anzahl der mit relativer Pfadangabe eingebundenen sonstigen Dokumente, die auf dem eigenen Webserver gespeichert sind
cnt_fail_doc	sonstige Dokumente	fehlerh. Referenzen	Anzahl der eingebundenen sonstigen Dokumente, die nicht empfangen werden konnten (z.B. 404-Fehler)
sum_doc_size	sonstige Dokumente	Seitengröße, Nutzerakzeptanz	Größe aller eingebundenen sonstigen Dokumente in Bytes
cnt_diff_doc	sonstige Dokumente	Komplexität	Anzahl verschiedener MIME-Typen in den eingebundenen sonstigen Dokumenten (siehe WebTomix [37, S.7ff])
cnt_email	E-Mails	Benutzerfreundlichkeit, Webstruktur	Anzahl der verwendeten E-Mailadressen (per mailto:)
cnt_email_g	E-Mails	Benutzerfreundlichkeit, Webstruktur	Anzahl verschiedener Mailhosts in den E-Mailadressen (user@host)
cnt_frames	Frames	Barrierefreiheit, Seitenaufbau, Zugänglichkeit	Anzahl der verwendeten Frames ⁴⁹

⁴⁸Eingebundene Dokumente zählen zu den Inline-Elementen einer Seite dazu und werden beim Download der Seite mitangefordert. Da Bilder bereits separat behandelt werden, sind folgende Dokumente gemeint: `<applet src="..." />`, `<link href="..." rel="stylesheet" />`; `<script src="..." language="Javascript" type="text/javascript" />` und andere...

⁴⁹Beispiel: `<frameset rows="150,*"><frame name="oben" src="..." />...</frameset>`

Messwert	Rubrik	Qualitätsaspekt	Beschreibung
cnt_iframes	Frames	Barrierefreiheit, Kompatibilität, Seitenaufbau, Zugänglichkeit	Anzahl der verwendeten internen Frames (<code><iframe ... /></code>)
cnt_frames_el	Frames	Webstruktur, Fremdinhalte, Stabilität	Anzahl der verwendeten Frames mit Link zu einem externen Webserver
mnr_age	Webseite	Standardkonformität, Stabilität	Alter des Webdokumentes (LastModified-Attribut im HTTP-Response-Header)
mnr_content	Webseite	Seitengröße, Lesbarkeit	Anzahl der Zeichen im HTML-Quellcode, die für Text und Inhalt verwendet werden ⁵⁰
mnr_html	Webseite	Seitengröße, Komplexität, Lesbarkeit	Anzahl der Zeichen im HTML-Quellcode, die für HTML-Tags und Attribute verwendet werden
mnr_loc	Webseite	Lesbarkeit	Anzahl der Zeilen im HTML-Quellcode (Lines of Code)
cnt_inline	Webseite	Komplexität, Seitenaufbau	Anzahl der eingebetteten Elemente ⁵¹
mnr_inline_size	Webseite	Seitenaufbau, Seitengröße	Größe aller eingebetteten Elemente in Bytes
mnr_query_size	Webseite	Kompatibilität, Komplexität, Lesbarkeit, Standardkonformität	Anzahl der Argumente im URL-Query (<code>?var1=val&var2=val</code>)
cnt_spec_chars	Webseite	Kompatibilität, Standardkonformität	Anzahl nicht korrekt kodierter Zeichen ⁵²

⁵⁰Inhalte sind Texte zwischen den HTML-Tags und die Werte der Attribute `alt` und `title`.

⁵¹Bilder, Javascript-Dateien, Stylesheet-Dateien, Multimedia-Objekte, Java Applets, ...

⁵²Beispiel: ä statt `ä`; , asiatische Zeichen, ...

Gezählt werden alle Zeichen mit Ausnahme von:

- Buchstaben (A-Z, a-z) und Zahlen (0-9),
- Klammern ([] () { }) und Satzzeichen (. ! ? , : ;),
- mathematische Zeichen (+ - * / ^ % = ~),
- sonstige Zeichen (_ " § \$ & \ ' # < > | ° @ und das Leerzeichen) und
- Steuerzeichen (\t \n \r \f (Tabulator, Zeilenvorschub, Schreibkopfrücklauf und Seitenvorschub))

Messwert	Rubrik	Qualitätsaspekt	Beschreibung
cnt_comments	HTML	Lesbarkeit	Anzahl der HTML-Kommentare (<!-- ... -->))
msr_comments	HTML	Lesbarkeit, Seitengröße	Länge der HTML-Kommentare (<!-- ... -->))
cnt_form	HTML	Benutzerfreundlichkeit, Komplexität	Anzahl der HTML-Formulare
cnt_form_elem	HTML	Benutzerfreundlichkeit, Komplexität	Anzahl der HTML-Formularelemente in allen Formularen (Textfelder, Radio- und Checkboxes, Buttons, Auswahlliste, ...)
cnt_tables	HTML	Komplexität, Seitenaufbau, Standardkonformität	Anzahl der HTML-Tabellen
sum_nested	HTML	Komplexität, Seitenaufbau	Summe der Tabellenverschachtelungen ⁵³
max_nested	HTML	Komplexität, Seitenaufbau	Tiefste Verschachtelung einer Tabelle
cnt_td	HTML	Komplexität, Seitenaufbau	Anzahl der <td>- und <th>-Tags
cnt_colspan	HTML	Komplexität, Seitenaufbau	Anzahl der <td>- und <th>-Tags, die das colspan-Attribut besitzen (horizontaler Zellenzusammenschluss)
cnt_rowspan	HTML	Komplexität, Seitenaufbau	Anzahl der <td>- und <th>-Tags, die das rowspan-Attribut besitzen (vertikaler Zellenzusammenschluss)
cnt_style	HTML	Standardkonformität, Komplexität, Zugänglichkeit	Anzahl von style-Attributen in HTML-Tags Verwendung von positionsspezifischen CSS-Eigenschaften, die nicht separat in einer CSS-Klasse definiert sind
sum_style_el	HTML	Standardkonformität, Komplexität, Zugänglichkeit	Anzahl aller CSS-Eigenschaften in allen style-Attributen

⁵³Beispiel: <table == 1 Punkt><tr><td><table == 2 Punkte>...</table></td></tr></table>

Messwert	Rubrik	Qualitätsaspekt	Beschreibung
cnt_css	HTML	Standardkonformität, Komplexität, Zugänglichkeit	Anzahl der verwendeten CSS-Klassen (über <code>class</code> -Attribut in HTML-Tags), CSS-Klassen sind global im Dokument definiert, Jede Klasse wird nur einmal gezählt
sum_css	HTML	Standardkonformität, Komplexität, Zugänglichkeit	Anzahl, wie oft CSS-Klassen verwendet wurden (über <code>class</code> -Attribut in HTML-Tags)
cnt_css_style	HTML	Standardkonformität, Komplexität, Zugänglichkeit	Anzahl der HTML-Tags, die eine globale CSS-Klasse (<code>class</code> -Attribut) über positionsspezifische CSS-Eigenschaften ergänzen (<code>style</code> -Attribut)
msr_non_unique_id	HTML	Standardkonformität	Anzahl der unterschiedlichen ID-Attributwerte von HTML-Tags, die nicht eindeutig sind (<code>id</code> -Attribut)
cnt_non_unique_id	HTML	Standardkonformität	Anzahl der HTML-Tags, die nicht eindeutige ID-Attributwerte verwenden

B.2 Qualitätsmaße

Die folgenden Präfixe wurden bei den Qualitätsmaßen verwendet:

- AVG** — Durchschnitt von Messwerten
- CNT** — Zählbares Maß oder direkte Bewertung eines `cnt`-Messwertes
- MAX** — Direkte Bewertung eines `max`-Messwertes
- MSR** — Allgemeines Qualitätsmaß
- RAT** — Verhältnis zwischen Messwerten
- SUM** — Summe von Messwerten

In der folgenden Liste beschreibt die erste Zeile ein Qualitätsmaß und die zweite Zeile beinhaltet die berechnende Formel, die verwendete Maßskala sowie die Wichtung und durch empirische Versuche gewonnene Standardgrenzwerte ($Threshold_{Pos}$ ist der positive Grenzwert und $Threshold_{Neg}$ der negative). Die Werte zwischen den Grenzwerten sind Standardwerte (also weder gut noch schlecht).

Die Spalte Rubrik beinhaltet neben dem Rubriknamen noch ein Kürzel:

- EM** (*Entire Measurement*) —
Das Qualitätsmaß bezieht sich auf die gesamte Messung.
- SP** (*Single Web Page*) —
Das Qualitätsmaß bezieht sich auf eine einzelne Webseite.

In der Liste wird die Überprüfung auf 0 im Divisor verzichtet. Das folgende Beispiel zeigt das Maß `RAT_INT_LINKS`.

$$\text{RAT_INT_LINKS} = \begin{cases} -1 & \text{SUM_LINKS} = 0 \\ 100 \cdot \frac{\text{cnt_int_links_a}}{\text{SUM_INT_LINKS}} & \text{sonst} \end{cases} \quad (\text{B.1})$$

$$\text{RAT_INT_LINKS} = 100 \cdot \text{cnt_int_links_a} / \text{SUM_INT_LINKS} \quad (\text{B.2})$$

Formel B.1 zeigt die vollständige und korrekte Formel, wie sie auch im Messtool verwendet wird⁵⁴. Formel B.2 zeigt die abgekürzte Form, wie sie hier in der folgenden Liste verwendet wird.

⁵⁴Der Wert -1 bedeutet hierbei soviel wie „nicht vorhanden“. In diesem Fall geht das Qualitätsmaß nicht in die Wertung der Messung ein.

Index

AVG			
AVG_CSS	143	MSR_SPEC_CHARS	141
AVG_DOC_SIZE	139	MSR_STYLE	143
AVG_FORM_ELEM	142	RAT	
AVG_IMG_SIZE	137	RAT_ADV_IMG	137
AVG_LINE_LENGTH	140	RAT_ADV_IMG_E	137
AVG_NESTED	142	RAT_ADV_LINKS	135
AVG_NON_UNIQUE_ID	144	RAT_ADV_LINKS_E	??
AVG_RESPONSE	134	RAT_COLSPAN	142
AVG_STYLE	143	RAT_COMMENT	142
CNT		RAT_COMMENT_S	142
CNT_DIFF_DOC	139	RAT_EMAIL_G	139
CNT_EMAIL	139	RAT_EXT_DOC	138
CNT_EMAIL_G	139	RAT_EXT_IMG	136
CNT_FORM	142	RAT_EXT_LINKS	135
CNT_FRAMES	139	RAT_FAIL_DOC	139
CNT_IFRAMES	139	RAT_FAIL_IMG	136
CNT_INLINE	140	RAT_FAIL_LINKS	135
CNT_NON_UNIQUE_ID	143	RAT_FRAMES_EL	140
CNT_PAGES	133	RAT_HTML	140
CNT_TABLES	142	RAT_IMG_AAT	138
MAX		RAT_IMG_AOT	137
MAX_NESTED	142	RAT_INT_DOC_A	138
MSR		RAT_INT_IMG_A	136
MSR_AGE	140	RAT_INT_LINKS_A	135
MSR_CSS	143	RAT_RESPONSE	134
MSR_ELOAD_28k	141	RAT_RESPONSE_28k	134
MSR_ELOAD_56k	141	RAT_RESPONSE_56k	134
MSR_ELOAD_ISDN	141	RAT_RESPONSE_ISDN	134
MSR_ELOAD_DSL	141	RAT_RESPONSE_DSL	134
MSR_LINK_DENS	136	RAT_ROWSPAN	143
MSR_LOAD_28k	141	SUM	
MSR_LOAD_56k	141	SUM_DOC	138
MSR_LOAD_ISDN	141	SUM_FRAMES	140
MSR_LOAD_DSL	141	SUM_IMG	136
MSR_NON_UNIQUE_ID	143	SUM_IMG_AOT	137
MSR_NORM	140	SUM_INT_DOC	138
MSR_QUERY_SIZE	141	SUM_INT_IMG	136
MSR_RESPONSE	134	SUM_INT_LINKS	134
MSR_RESPONSE_ALL	134	SUM_LINKS	134
MSR_SIZE	140	SUM_SIZE	140

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
CNT_PAGES	Sonstiges (EM)	Webstruktur	Anzahl der gemessenen Webseiten im Messprojekt		
= SUM(Pages.*)		= $\{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.6	< 100	> 500

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
AVG_RESPONSE	Server (EM)	Performance, Seitenaufbau	Durchschnittliche Antwortzeit aller angeforderten Webseiten im Messprojekt, Zeit, die zum Herunterladen der Webseiten und aller Inline-Elemente benötigt wurde		
= AVG(msr_response_all)		= $\{x x \in \mathbb{R} \wedge x \geq 0\}$	0.75	< 5 s	> 10 s
RAT_RESPONSE RAT_RESPONSE_28k RAT_RESPONSE_56k RAT_RESPONSE_ISDN RAT_RESPONSE_DSL	Server (EM)	Performance, Seitenaufbau	Anteil der gemessenen Webseiten in Prozent, die eine schlechte Antwortzeit haben		
= $100 \cdot [0.5 \cdot \text{SUM}(\text{Pages.} * \text{ with } \text{msr_response_all}^{20 < T \leq 40 \text{ s}}) + \text{SUM}(\text{Pages.} * \text{ with } \text{msr_response_all}^{40 \text{ s} < T})] / \text{CNT_PAGES}$		= $\{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.75 0.75 0.6 0.5	< 35 % < 25 % < 15 % < 5 %	> 50 % - 28k > 40 % - 56k > 30 % - ISDN > 20 % - DSL
MSR_RESPONSE	Server (SP)	Performance, Seitenaufbau	Antwortzeit beim Anfordern der Webseite, Zeit die zum Downloaden der Webseite benötigt wird (nur der HTML-Teil)		
= msr_response		= $\{x x \in \mathbb{R} \wedge x \geq 0\}$	0.8	< 2 s	> 4 s
MSR_RESPONSE_ALL	Server (SP)	Performance, Seitenaufbau	Antwortzeit beim Anfordern der Webseite Zeit die zum Downloaden der Seite und aller Inline-Elemente (Bilder, Includedateien, ...) benötigt wird		
= msr_response_all		= $\{x x \in \mathbb{R} \wedge x \geq 0\}$	0.8	< 5 s	> 10 s
SUM_INT_LINKS	Links (SP)	Benutzerfreundlichkeit, Webstruktur	Anzahl der Hyperlinks zu internen Dokumenten		
= cnt_int_links_a + cnt_int_links_r		= $\{x x \in \mathbb{Z} \wedge x \geq 0\}$	keine Bewertung		
SUM_LINKS	Links (SP)	Benutzerfreundlichkeit, Webstruktur	Anzahl aller Hyperlinks auf einer Webseite		
= SUM_INT_LINKS + cnt_ext_links		= $\{x x \in \mathbb{Z} \wedge x \geq 0\}$	keine Bewertung		

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
RAT_EXT_LINKS	Links (SP)	Webstruktur, Stabilität	Verhältnis zwischen Anzahl der Hyperlinks zu externen Dokumenten und der Gesamtzahl der Links in Prozent		
$= 100 \cdot \text{cnt_ext_links}/\text{SUM_LINKS}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.6	< 20 %	> 40 %
RAT_INT_LINKS _A	Links (SP)	Webstruktur, Änderbarkeit	Verhältnis zwischen Anzahl der Hyperlinks zu internen Dokumenten mit absoluter Pfadangabe und der Gesamtzahl interner Hyperlinks in Prozent		
$= 100 \cdot \text{cnt_int_links_a}/\text{SUM_INT_LINKS}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.6	< 10 %	> 20 %
RAT_FAIL_LINKS	Links (SP)	fehlerh. Referenzen	Verhältnis zwischen Anzahl der Hyperlinks zu Dokumenten, die nicht herunter geladen werden konnten (z.B. wegen 404-Fehlern) und der Gesamtzahl der Hyperlinks in Prozent		
$= 100 \cdot \text{cnt_fail_links}/\text{SUM_LINKS}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.8	= 0 %	> 5 %
RAT_ADV_LINKS	Links (SP)	Nutzerakzeptanz	Verhältnis zwischen Anzahl der Hyperlinks, deren URLs eine Übereinstimmung mit den Werbe-Pattern aus Anhang C ergaben (nur bei externen Links) und der Gesamtzahl der Links in Prozent		
$= 100 \cdot \text{cnt_adv_links}/\text{SUM_LINKS}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.4	< 5 %	> 10 %
RAT_ADV_LINKS _E	Links (SP)	Nutzerakzeptanz, Stabilität	Verhältnis zwischen Anzahl der Hyperlinks, deren URLs eine Übereinstimmung mit den Werbe-Pattern aus Anhang C ergaben (nur bei externen Links) und der Anzahl der Links zu externen Dokumenten in Prozent		
$= 100 \cdot \text{cnt_adv_links}/\text{cnt_ext_links}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.4	< 30 %	> 70 %

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
MSR_LINK_DENS	Links (EM)	Benutzerfreundlichkeit, Webstruktur	Navigations-Linkdichte, Verhältnis zwischen der Summe aller internen Hyperlinks aus allen Webdokumenten und der Gesamtzahl der Hyperlinks in allen Webdokumenten im Messprojekt in Prozent		
$= 100 \cdot \text{cnt_adv_links}/\text{SUM_LINKS}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.6	> 80 %	> 10 %
SUM_INT_IMG	Bilder (SP)	Webstruktur	Anzahl der eingebundenen Bilder, die auf dem eigenen Webserver gespeichert sind		
$= \text{cnt_int_img_a} + \text{cnt_int_img_r}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	keine Bewertung		
SUM_IMG	Bilder (SP)	Webstruktur	Anzahl der eingebundenen Bilder, die auf dem eigenen und auf fremden Webservern gespeichert sind		
$= \text{SUM_INT_IMG} + \text{cnt_ext_img}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	keine Bewertung		
RAT_EXT_IMG	Bilder (SP)	Webstruktur, Fremdinhalte, Stabilität	Verhältnis zwischen Anzahl der eingebundenen Bilder, die auf einem externen Webserver gespeichert sind und der Gesamtzahl der eingebundenen Bilder in Prozent		
$= 100 \cdot \text{cnt_ext_img}/\text{SUM_IMG}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.9	= 0 %	> 5 %
RAT_INT_IMG_A	Bilder (SP)	Webstruktur, Änderbarkeit	Verhältnis zwischen Anzahl der eingebundenen internen Bilder mit absoluter Pfadangabe und der Gesamtzahl der eingebundenen internen Bilder in Prozent		
$= 100 \cdot \text{cnt_int_img_a}/\text{SUM_INT_IMG}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.6	< 10 %	> 20 %
RAT_FAIL_IMG	Bilder (SP)	fehlerh. Referenzen	Verhältnis zwischen Anzahl der eingebundenen Bilder, die nicht herunter geladen werden konnten (z.B. wegen 404-Fehlern) und der Gesamtzahl der eingebundenen Bilder in Prozent		
$= 100 \cdot \text{cnt_fail_img}/\text{SUM_IMG}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.75	= 0 %	> 5 %

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
RAT_ADV_IMG	Bilder (SP)	Nutzerakzeptanz	Verhältnis zwischen Anzahl der eingebundenen Bilder, deren URLs eine Übereinstimmung mit den Werbe-Pattern aus Anhang C ergaben (gilt nur für externe URLs) und der Gesamtzahl der eingebundenen Bilder in Prozent		
$= 100 \cdot \text{cnt_adv_img}/\text{SUM_IMG}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.3	< 5 %	> 20 %
RAT_ADV_IMG_E	Bilder (SP)	Nutzerakzeptanz, Stabilität	Verhältnis zwischen Anzahl der Hyperlinks, deren URLs eine Übereinstimmung mit den Werbe-Pattern aus Anhang C ergaben (gilt nur für externe Links) und der Anzahl der eingebundenen Bilder, die auf einem externen Webserver gespeichert sind		
$= 100 \cdot \text{cnt_adv_img}/\text{cnt_ext_img}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.4	< 30 %	> 50 %
AVG_IMG_SIZE	Bilder (SP)	Seitengröße	Durchschnittliche Größe aller eingebundenen Bilder in Bytes		
$= \text{sum_img_size}/\text{SUM_IMG}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.6	< 20 <i>KBytes</i>	> 50 <i>KBytes</i>
SUM_IMG_AOT	Bilder (SP)	Barrierefreiheit, Benutzerfreundlichkeit, Standardkonformität	Anzahl aller eingebundenen Bilder mit ALT- (Barrierefreiheit) oder TITLE-Attribute (Kompatibilität zum Webbrowser Opera)		
$= \text{sum_img_alt} + \text{sum_img_title} - \text{sum_img_alt_title}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	keine Bewertung		
RAT_IMG_AOT	Bilder (SP)	Barrierefreiheit, Benutzerfreundlichkeit, Standardkonformität	Verhältnis zwischen Anzahl aller eingebundenen Bilder mit ALT- (Barrierefreiheit) oder TITLE-Attribute (Kompatibilität zum Webbrowser Opera) und der Gesamtzahl der eingebundenen Bilder in Prozent		
$= 100 \cdot \text{SUM_IMG_AOT}/\text{SUM_IMG}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.8	> 60 %	< 40 %

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
RAT_IMG_AAT	Bilder (SP)	Kompatibilität, Benutzerfreundlichkeit, Standardkonformität	Verhältnis zwischen Anzahl aller eingebundenen Bilder mit ALT- (Barrierefreiheit) und TITLE-Attribute (Kompatibilität zum Webbrowser Opera) und der Gesamtzahl der eingebundenen Bilder in Prozent		
$= 100 \cdot cnt_img_alt_title / SUM_IMG$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.3	> 60 %	< 40 %
SUM_INT_DOC	sonstige Dokumente (SP)	Webstruktur	Anzahl der eingebundenen sonstigen Dokumente, die auf dem eigenen Webserver gespeichert sind		
$= cnt_int_doc_a + cnt_int_doc_r$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	keine Bewertung		
SUM_DOC	sonstige Dokumente (SP)	Webstruktur	Anzahl der eingebundenen sonstigen Dokumente, die auf dem eigenen und auf fremden Webservern gespeichert sind		
$= SUM_INT_DOC + cnt_ext_doc$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	keine Bewertung		
RAT_EXT_DOC	sonstige Dokumente (SP)	Webstruktur, Fremdinhalte, Stabilität	Verhältnis zwischen Anzahl der eingebundenen sonstigen Dokumente, die auf einem externen Webserver gespeichert sind und der Gesamtzahl der eingebundenen sonstigen Dokumente in Prozent		
$= 100 \cdot cnt_ext_doc / SUM_DOC$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.9	= 0 %	> 5 %
RAT_INT_DOC_A	sonstige Dokumente (SP)	Webstruktur, Änderbarkeit	Verhältnis zwischen Anzahl der eingebundenen internen Dokumenten mit absoluter Pfadangabe und der Gesamtzahl der eingebundenen internen Dokumenten in Prozent		
$= 100 \cdot cnt_int_doc_a / SUM_INT_DOC$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.6	< 10 %	> 20 %

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
RAT_FAIL_DOC	sonstige Dokumente (SP)	fehlerh. Referenzen	Verhältnis zwischen Anzahl der eingebundenen sonstigen Dokumente, die nicht herunter geladen werden konnten (z.B. wegen 404-Fehlern) und der Gesamtzahl der eingebundenen sonstigen Dokumente in Prozent		
$= 100 \cdot \text{cnt_fail_doc} / \text{SUM_DOC}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.75	$= 0\%$	$> 5\%$
AVG_DOC_SIZE	sonstige Dokumente (SP)	Seitengröße, Nutzerakzeptanz	Durchschnittliche Größe aller eingebundenen sonstigen Dokumente in Bytes		
$= \text{sum_doc_size} / \text{SUM_DOC}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.7	$< 20 \text{ KBytes}$	$> 50 \text{ KBytes}$
CNT_DIFF_DOC	sonstige Dokumente (SP)	Komplexität	Anzahl verschiedener MIME-Typen in den eingebundenen sonstigen Dokumenten (siehe WebTomix [37, S.7ff])		
$= \text{cnt_diff_doc}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.4	< 4	> 8
CNT_EMAIL	E-Mails (SP)	Benutzerfreundlichkeit, Webstruktur	Anzahl der verwendeten E-Mailadressen (per <code>mailto:</code>)		
$= \text{cnt_email}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.1	< 3	> 10
CNT_EMAIL_G	E-Mails (SP)	Benutzerfreundlichkeit, Webstruktur	Anzahl verschiedener Mailhosts in den E-Mailadressen (<code>user@host</code>)		
$= \text{cnt_email_g}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.1	< 2	> 5
RAT_EMAIL_G	E-Mails (SP)	Benutzerfreundlichkeit, Webstruktur	Verhältnis zwischen verschiedenen Mailhosts in den E-Mailadressen und der Gesamtzahl der E-Mailadressen in Prozent		
$= 100 \cdot \text{cnt_email_g} / \text{cnt_email}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.2	$< 33\%$	$> 50\%$
CNT_FRAMES	FRAMES (SP)	Barrierefreiheit, Seitenaufbau, Zugänglichkeit	Anzahl der verwendeten Frames		
$= \text{cnt_frames}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.1	< 3	> 10
CNT_IFRAMES	FRAMES (SP)	Barrierefreiheit, Kompatibilität, Seitenaufbau, Zugänglichkeit	Anzahl der verwendeten internen Frames		
$= \text{cnt_iframes}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.1	< 2	> 5

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
SUM_FRAMES	FRAMES (SP)	Barrierefreiheit, Seitenaufbau, Zugänglichkeit	Anzahl der Summe der Frames und internen Frames		
$= cnt_frames + cnt_iframes$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.1	< 2	> 5
RAT_FRAMES _EL	FRAMES (SP)	Webstruktur, Fremdinhalte, Stabilität	Verhältnis zwischen Frames zu Webseiten auf externen Webservern und der Gesamt- zahl der Frames in Prozent		
$= 100 \cdot cnt_frames_el/SUM_FRAMES$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.2	< 33 %	> 50 %
MSR_AGE	Webseite (SP)	Standardkonfor- mität, Stabilität	Alter des Webdokumentes (LastModified-Attribut im HTTP-Response-Header)		
$= NOW() - msr_age$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.1	< 7 d	> 60 d
MSR_SIZE	Webseite (SP)	Lesbarkeit, Seitengröße	Größe des Webdokumentes in KBytes		
$= msr_html + msr_content$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.5	< 30 KBytes	> 70 KBytes
MSR_NORM	Webseite (SP)	Normierungsfaktor, der verschieden große Webseiten ver- gleichbar werden lässt, Messwerte werden so umgerechnet, als wäre die Webseite 10 KByte groß			
$= MSR_SIZE/10\text{ KBytes}$			keine Bewertung		
AVG_LINE _LENGTH	Webseite (SP)	Lesbarkeit	Durchschnittliche Länge einer Zei- le im HTML-Quellcode (beein- flusst die Lesbarkeit des Quellco- des)		
$= MSR_SIZE/msr_loc$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.8	< 75 Zeichen	> 125 Zeichen
RAT_HTML	Webseite (SP)	Komplexität, Lesbarkeit	Anteil der Zeichen im HTML- Quellcode, die für HTML-Tags und Attribute verwendet werden		
$= 100 \cdot msr_html/MSR_SIZE$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.8	< 70 %	> 90 %
CNT_INLINE	Webseite (SP)	Komplexität, Seitenaufbau	Anzahl der eingebetteten Elemen- te (normiert)		
$= \begin{cases} cnt_inline & MSR_NORM < 1 \\ \frac{cnt_inline}{MSR_NORM} & sonst \end{cases}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.6	< 15	> 30
SUM_SIZE	Webseite (SP)	Seitenaufbau, Seitengröße	Größe der HTML-Seite und aller eingebetteten Elemente in KBytes		
$= msr_inline_size + MSR_SIZE$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.6	< 55 KBytes	> 100 KBytes

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
MSR_LOAD_28k MSR_LOAD_56k MSR_LOAD_ISDN MSR_LOAD_DSL	Webseite (SP)	Nutzerakzeptanz, Performance, Seitenaufbau	Zeit in Sekunden die benötigt wird, um die Webseite (nur den HTML-Teil) mit einem 28k-, 56k-Modem, ISDN (ein Kanal) oder DSL (768k) zu empfangen, Die benötigte Verwaltungszeit der HTTP-Verbindung (Verbindungsaufbau, ...) sowie Netzschwankungen werden hierbei vernachlässigt		
= MSR_SIZE · 8/28800 = MSR_SIZE · 8/56000 = MSR_SIZE · 8/128000 = MSR_SIZE · 8/768000		= $\{x x \in \mathbb{R} \wedge x \geq 0\}$	0.9 0.9 0.9 0.9	< 5 s < 3 s < 2 s < 1 s	> 10 s - 28k > 6 s - 56k > 4 s - ISDN > 2 s - DSL
MSR_ELOAD_28k MSR_ELOAD_56k MSR_ELOAD_ISDN MSR_ELOAD_DSL	Webseite (SP)	Nutzerakzeptanz, Performance, Seitenaufbau	Zeit in Sekunden die benötigt wird, um die Webseite mit allen eingebundenen Elementen mit einem 28k-, 56k-Modem, ISDN (ein Kanal) oder DSL (768k) zu empfangen, Die benötigte Verwaltungszeit der HTTP-Verbindung (Verbindungsaufbau, ...) sowie Netzschwankungen werden hierbei vernachlässigt		
= SUM_SIZE · 8/28800 = SUM_SIZE · 8/56000 = SUM_SIZE · 8/128000 = SUM_SIZE · 8/768000		= $\{x x \in \mathbb{R} \wedge x \geq 0\}$	0.9 0.9 0.9 0.9	< 8 s < 6 s < 4 s < 2 s	> 20 s - 28k > 10 s - 56k > 7 s - ISDN > 4 s - DSL
MSR_QUERY_SIZE	Webseite (SP)	Komplexität, Lesbarkeit, Standardkonformität	Anzahl der Argumente im URL-Query (?var1=val&var2=val)		
= msr_query_size		= $\{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.2	< 3	> 5
MSR_SPEC_CHARS	Webseite (SP)	Kompatibilität, Standardkonformität	Anzahl nicht korrekt kodierter Zeichen (normiert)		
= $\begin{cases} \frac{msr_query_size}{MSR_NORM} & MSR_NORM < 1 \\ msr_query_size & sonst \end{cases}$		= $\{x x \in \mathbb{R} \wedge x \geq 0\}$	0.6	< 2	> 20

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
RAT_COMMENT	HTML (SP)	Lesbarkeit	Verhältnis zwischen Anzahl der HTML-Kommentare und der Anzahl der Zeilen im Quellcode in Prozent		
$= 100 \cdot \text{cnt_comments}/\text{msr_loc}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.2	> 20 %	= 0 %
RAT_COMMENT_S	HTML (SP)	Lesbarkeit, Seitengröße	Verhältnis zwischen Größe der HTML-Kommentare und der Größe des gesamten HTML-Quellcodes in Prozent		
$= 100 \cdot \text{msr_comments}/\text{MSR_SIZE}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.2	> 10 %	= 0 %
CNT_FORM	HTML (SP)	Benutzerfreundlichkeit, Komplexität	Anzahl der HTML-Formulare (normiert)		
$= \begin{cases} \frac{\text{cnt_form}}{\text{MSR_NORM}} & \text{MSR_NORM} < 1 \\ \text{sonst} & \text{sonst} \end{cases}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.5	< 2	> 4
AVG_FORM_ELEM	HTML (SP)	Benutzerfreundlichkeit, Komplexität	Durchschnittliche Anzahl der HTML-Formularelemente in den Formularen (Textfelder, Radio- und Checkboxes, Buttons, Auswahlliste, ...)		
$= \text{cnt_form_elem}/\text{cnt_form}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.2	< 5	> 20
CNT_TABLES	HTML (SP)	Komplexität, Seitenaufbau, Standardkonformität	Anzahl der HTML-Tabellen (normiert)		
$= \begin{cases} \frac{\text{cnt_tables}}{\text{MSR_NORM}} & \text{MSR_NORM} < 1 \\ \text{sonst} & \text{sonst} \end{cases}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.7	< 2	> 4
AVG_NESTED	HTML (SP)	Komplexität, Seitenaufbau	Durchschnittliche Tabellenverschachtelung		
$= \text{sum_nested}/\text{cnt_tables}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.7	< 3	> 4
MAX_NESTED	HTML (SP)	Komplexität, Seitenaufbau	Tiefste Verschachtelung einer Tabelle		
$= \text{max_nested}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.9	< 4	> 6
RAT_COLSPAN	HTML (SP)	Komplexität, Seitenaufbau	Verhältnis zwischen Anzahl der Zellen (<td>- und <th>-Tags) mit dem colspan-Attribut (horizontaler Zellenzusammenschluss) und der Gesamtzahl der Zellen in Prozent		
$= 100 \cdot \text{cnt_colspan}/\text{cnt_td}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.5	< 10 %	> 30 %

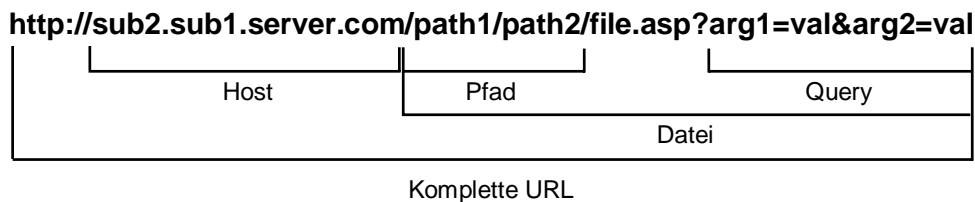
Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
RAT_ROWSPAN	HTML (SP)	Komplexität, Seitenaufbau	Verhältnis zwischen Anzahl der Zellen (<td>- und <th>-Tags) mit dem rowspan -Attribut (vertikaler Zellenzusammenschluss) und der Gesamtzahl der Zellen in Prozent		
$= 100 \cdot \text{cnt_rowspan} / \text{cnt_td}$		$= \{x x \in \mathbb{R} \wedge 0 \leq x \leq 100\}$	0.6	< 10 %	> 30 %
MSR_STYLE	HTML (SP)	Standardkonformität, Komplexität, Zugänglichkeit	Anzahl von HTML-Tags mit style -Attributen aber ohne class -Attribute (normiert) Verwendung von positionsspezifischen CSS-Eigenschaften, die nicht separat in einer CSS-Klasse definiert sind; die Verwendung von CSS-Klassen wäre optimaler		
$tmp = \text{cnt_style} - \text{cnt_css_style}$ $= \begin{cases} tmp & \text{MSR_NORM} < 1 \\ \frac{tmp}{\text{MSR_NORM}} & \text{sonst} \end{cases}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.8	< 3	> 5
AVG_STYLE	HTML (SP)	Standardkonformität, Komplexität, Zugänglichkeit	Durchschnittliche Anzahl von CSS-Eigenschaften in den style -Attributen		
$= \text{sum_style_el} / \text{cnt_style}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.8	< 3	> 4
MSR_CSS	HTML (SP)	Standardkonformität, Komplexität, Zugänglichkeit	Anzahl der verwendeten CSS-Klassen (normiert)		
$= \text{cnt_css} / \text{MSR_NORM}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.7	> 6	< 3
AVG_CSS	HTML (SP)	Standardkonformität, Komplexität, Zugänglichkeit	Durchschnittliche Anzahl, wie oft CSS-Klassen verwendet werden		
$= \text{sum_css} / \text{cnt_css}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.8	> 4	< 2
MSR_NON_UNIQUE_ID	HTML (SP)	Standardkonformität	Anzahl der unterschiedlichen ID-Attributwerte von HTML-Tags, die nicht eindeutig sind (id -Attribut)		
$= \text{msr_non_unique_id}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.75	= 0	> 3
CNT_NON_UNIQUE_ID	HTML (SP)	Standardkonformität	Anzahl der HTML-Tags, die nicht eindeutige ID-Attributwerte verwenden		
$= \text{cnt_non_unique_id}$		$= \{x x \in \mathbb{Z} \wedge x \geq 0\}$	0.75	= 0	> 9

Messwert	Rubrik	Qualitätsaspekt	Beschreibung		
Formel		Wertebereich	Wichtung	$Threshold_{Pos}$	$Threshold_{Neg}$
AVG_NON_UNIQUE_ID	HTML (SP)	Standardkonformität	Durchschnittliche Anzahl, wie oft nicht eindeutige Identifier verwendet werden		
$= \text{cnt_non_unique_id} / \text{msr_non_unique_id}$		$= \{x x \in \mathbb{R} \wedge x \geq 0\}$	0.75	= 0	> 3

C Werbe-Pattern in URLs

Die URL einer Webseite oder eines Bildes wird auf Werbe-Pattern untersucht, wodurch eine Klassifizierung als Werbung erfolgt.

Die Werbe-Pattern untersuchen verschiedene Elemente der URL (siehe Bild C.1).



Komplette URL	http://sub2.sub1.server.com/path1/path2/file.asp?arg1=val&arg2=val
Host	sub2.sub1.server.com
Pfad	/path1/path2/
Datei	/path1/path2/file.asp?arg1=val&arg2=val
Query	arg1=val&arg2=val

Abbildung C.1: Elemente einer URL für Werbe-Pattern

Kompletter Link

doubleclick.net/click
 ads.lycosasia.com.sg/RealMedia/ads/
 alladvantage.com/go.asp?refid=
 bfast.com/booklink
 crosswalk.com/click.ng/transactionID=
 impartner.de/cgi-bin/
 java.yahoo.com/a/1-/java
 movielink.com/media/imagelinks/MF.ad
 movielink.com/media/imagelinks/MF.sponsor
 rd.yahoo.com/M=2
 rd.yahoo.com/M=5
 telecom-pros.com/images
 webunion.com/cgi-localbin/click.cgi?
 www.anonymizer.com/cm/door.cgi?
 www.epaper.com.tw/cgi-bin/adm/ad_red
 www.modchip.com/clickcgi/click.cgi?
 www.warehouse.com/netbuyer/ticker/
 yahoo.com/adv/
 partners.webmasterplan.com/click.asp?ref=
 altfarm.mediaplex.com/ad

ad.tw.doubleclick.net/ad/
 advertisements
 banner_ad
 clk_thru
 fastclick.net/w/click.here
 infoseek.com/redirect
 java.yahoo.com/a/a-/flash
 mediahits.com/click.fcg
 pathfinder.com/r0/marketing
 rd.yahoo.com/M=3
 rd.yahoo.com/M=6
 texchange.com/cgi-bin/
 websponsors.com/cgi-bin/
 www.clickxchange.com/fr.phtml
 www.link4link.com/cgi-bin/
 www.nj.com/adverts
 www.whispa.com/tracking/
 yahoo.com/CategoryID=0
 werbung

adoptimizer
 allpolitics.com/ads/
 bannerad
 exchange-it.cin/click.go?
 findcommerce.com/tracking
 java.yahoo.com/a/1-/flash
 java.yahoo.com/a/a-/java
 netbanner.com/cgi-bin/
 pathfinder.com/sponsors
 rd.yahoo.com/M=4
 submit-it.com/images
 us.a1.yimg.com/us.yimg.com/a
 ww3.cybercity.com.tw/adexe/
 www.halee.com/advert/
 www.netvigator.com.tw/popad/
 www.search.com/Banners
 www.wishing.com/webaudit/
 yimg.com/images/compliance/
 adfarm.mediaplex.com/ad

Vergleich mit Host

.ad.	.ads.	.net-on.com	.webconnect.net
1-2-free.com	a32.g.a.yimg.com	ad.infoseek.com	ad.linkexchange.com
ad.preferences.com	adbanner	adbot.com	adcenter.in2.com
adcount.hollywood.com	adlink.preferences.com	adman.mediust.net	ads.adsmart.net
ads.clickagents.com	ads.imagine-inc.com	ads.imdb.com	ads.infospace.com
ads.narrowline.com	ads.realmedia.com	ads.softbank.net	ads.usatoday.com
ads.washingtonpost.com	ads.web21.com	advertising.com	badservant.guj.de
bannerexchange.com	bannerpower.com	bannerswap.com	click1.wisewire.com
click100.genesis.com	click2.wisewire.com	click2net.com	commonwealth.riddler.com
count4all.com	dirtycash.com	ds.cybereps.com	flycast.com
focalink.com	globaltrack.com	globaltrak.net	hitbox.com
imgis.com	infoback.net	linkexchange.com	marketing.com
mediaserv.247.media.com	mirror.qking.net	netads.hotwired.com	nrsite.com
pagecount.com	pegasoweb.com	pennyweb.com	qksrv.net
register-it.netscape.com	rmbclick.com	safe-audit.com	smartclicks.com
tracker.clicktrade.com	valueclick.com	www.247.media.com.tw	www.admax.com
www.ads.warnerbros.com	www.asiad.net	www.banner.com.tw	www.bannerwomen.com
www.clickadhere.com	www.cyberone.com.tw	www.marketspace	www.nrsite.com

Vergleich mit Pfad

/ad-	/ad/	/ad_control	/ad_image	/ad_track/	/adcouncil/
/adgifs/	/adgraph/	/adimages/	/adinfo	/adjuggler	/adnet/
/adpics/	/adpopup	/adproof/	/adredirect	/ads-	/ads/
/adsales	/adserv	/adspace	/adsrc	/adv/	/advertentie
/advert/	/advertise/	/advertiser/	/advertisers/	/advertising/	/adverts/
/banner/	/banner=	/banner_images/	/banners/	/cyberfirst	/graphics/advert
/htmlad/	/jbanner/	/liveads/	/promos/	/promote/	/promotions/
/sponsor/	/sponsors/	/videobanners/	/viewad/	_ads/	images/mainad
progcgi/ads					

Vergleich mit Datei

_ad.	-ads/	adlink.htm	/ad.	/adbot.	/adclient.
/adcontent.	/adlog.	/adman.	/adnet.	/ads.	/adview.
/banner.	/clickover.	/generate_ad.	/sponsor.	/sponsors.	/by.banclk?
/follow_ad?	/nph-bounce?	/nph-load?	/nph-redir?	/rankem.cgi?action=	adredir.asp
click.php?ref=	geoad?	maxcash.cgi?	pagead/adclick?	showad.	spinbox.

Vergleich mit Query

%2Fads%2E	%3Fad%2E	&ad_	&banner_	&acb=acb	&ad.cgi	&ad.	&ad=	&adlink
&adserv	&adv=	&adid=	&adclick					

D Endnutzerdokumentation der Web Measurement Suite

D.1 Inhalt der CD

Die folgende Auflistung enthält alle Dateien auf der CD, die für den Zugriff bzw. Aufruf von Verzeichnissen und Dateien wichtig sind.

<code>/_doc/_class/index.html</code>	die Übersichtsdatei der Klassendiagramme aller Java-Klassen im HTML-Format
<code>/_doc/_help/deutsch/index.html</code>	die Übersichtsdatei der deutschen Hilfeseiten zur Web Measurement Suite
<code>/_doc/_help/english/index.html</code>	die Übersichtsdatei der englischen Hilfeseiten zur Web Measurement Suite
<code>/_doc/_javadoc/index.html</code>	die Übersichtsdatei der Dokumentation aller Java-Klassen im HTML-Format
<code>/_example/_projects/</code>	beispielhafte .msz-Projektarchive zum Import
<code>/_example/_reports/</code>	generierte Qualitätsreports mit dem Standard-Report-Typ der Beispielprojekte
<code>/_jre/</code>	JRE 1.5.0 Installationssätze
<code>/_src/_build/</code>	Build-Skripten für Sun ONE Studio 5u1 SE zur Erstellung der JAR-Archive, eine Kompilation der Quelldateien ist manuell durchzuführen
<code>/_src/_third/</code>	Dokumentation und Quellen der Drittanbieter
<code>/_src/src.zip</code>	dieses Archiv enthält die Quelldateien der Web Measurement Suite sowie benötigte JAR-Pakete von Drittanbietern
<code>/diplomarbeit.pdf</code>	die Diplomarbeit als PDF-Dokument
<code>/diplomarbeit.ps</code>	die Diplomarbeit als PostScript-Dokument
<code>/wmsinstall.jar</code>	Das JAR-Archiv beinhaltet einen Installationssatz der kompilierten Web Measurement Suite; Quelldateien sind nicht enthalten

D.2 Systemvoraussetzungen

Die Web Measurement Suite wurde für die Java 2 Plattform programmiert. Zum Betrieb wird die Laufzeitumgebung (JRE) der Standardedition (J2SE) in der Version 1.4 oder höher benötigt. Installationssätze für die JRE 1.5.0 sind auf der CD im Verzeichnis `/__jre/` zu finden.

Zum Betrieb der Web Measurement Suite werden mindestens 10 MByte freier Festplattenspeicher für die Programmdateien, temporäre Daten und Messprojekte vorausgesetzt. Es werden mindestens 256 MByte RAM vorausgesetzt. Da die Kommunikation des Agentensystems vollständig auf Fernaufrufen beruht, wird ein funktionsfähiger TCP/IP-Stack gefordert.

D.3 Installation

Die Installation erfolgt durch das Kopieren der Dateien `wmsinstall.jar` sowie `install.sh` für Unixsysteme, die Shellskripten unterstützen, oder `install.bat` für auf Microsoft DOS oder Windows basierende Systeme in ein neues Verzeichnis auf der Festplatte.

Um die Installation zu starten genügt ein Doppelklick auf die `wmsinstall.jar` bei einer funktionsfähigen JRE-Installation oder auf die jeweilige `install.***`-Datei. Für die Installation auf anderen Systemen muss in der Kommandozeile der folgende Befehl `java -jar wmsinstall.jar` eingetragen werden. Eine Anpassung des Pfades zur JRE bzw. zur `wmsinstall.jar` ist je nach Bedarf vorzunehmen.

Durch die Ausführung der `wmsinstall.jar` werden alle nötigen Programmdateien im aktuellen Verzeichnis entpackt. Bitte sorgen Sie dafür, dass zuvor ein neues Verzeichnis erstellt wird und sich die `wmsinstall.jar` in diesem Verzeichnis befindet. Um die Installation rückgängig zu machen können die entpackten Programmdateien jederzeit gelöscht werden. Ein Eintrag in die Windows-Registry o.ä. wird nicht vorgenommen. Nach der Installation können die Dateien `wmsinstall.jar` und `install.***` gelöscht werden.

Vor dem eigentlichen Betrieb des Agentensystems sollte die RMI-Registry der Java-Umgebung auf dem Rechner gestartet sein. Die Registry kann wahlweise mit dem Aufruf „`rmiregistry`“ gestartet werden, oder ist je nach Systeminstallation schon in der Startumgebung des Betriebssystems eingetragen. Der explizite Start einer RMI-Registry ist jedoch nicht zwingend notwendig. Die Web Measurement Suite startet bei Nichtvorhandensein von sich aus eine Registry, was für die meisten Anwendungsfälle ausreichend ist.

D.4 Nach der Installation

Nach der Installation der Web Measurement Suite befinden sich folgende Daten im Verzeichnis:

/agents/	das Verzeichnis enthält Softwareagenten, die separat gestartet werden können
/lib/	das Verzeichnis enthält benötigte JAR-Archive
/lib/ice/	dieses Verzeichnis ist leer (siehe readme.txt), in dieses Verzeichnis müssen die JAR-Archive des ICEbrowser SDK
/debug.bat & /debug.sh	startet die Web Measurement Suite im Debug-Modus (eine Konsole für Ausgaben wird angezeigt, ein Schließen der Konsole beendet auch die Web Measurement Suite)
/run.bat und /debug.sh	startet die Web Measurement Suite
/wmms_debug.bat &	startet den Web Measurement Monitoring Service
/wmms_debug.sh	im Debug-Modus (eine Konsole für Ausgaben wird angezeigt, ein Schließen der Konsole beendet auch den Web Measurement Monitoring Service)
/wmms_start.bat &	startet den Web Measurement Monitoring Service
/wmms_start.sh	im Hintergrund, zum Beenden desselben muss „wmms_stop.bat“ bzw. „.sh“ gestartet oder ein Doppelklick auf die „wmmsstop.jar“ ausgeführt werden
/wmms_stop.bat &	beendet den Web Measurement Monitoring Service
/wmms_stop.sh	vice
/wmmsstart.jar	JAR-Archiv, welches zum Starten des Web Measurement Monitoring Service benötigt wird
/wmmsstop.jar	JAR-Archiv, welches zum Beenden des Web Measurement Monitoring Service benötigt wird
/wmsrun.jar	JAR-Archiv, welches zum Starten der Web Measurement Suite benötigt wird

Beim erstmaligen Start der Web Measurement Suite werden zusätzliche Dateien und Verzeichnisse erzeugt. Gestartet wird die Software durch einen Doppelklick auf „wmsrun.jar“ bzw. durch „run.bat“ bzw. „.sh“ oder „debug.bat“ bzw. „.sh“.

Zusätzlich werden folgende Verzeichnisse und Dateien erzeugt:

- /language/ das Verzeichnis enthält Sprach- und Hilfedateien
- /log/ das Verzeichnis enthält Logdateien, bei Fehlern bitte in die wms_error.log und wms_error.log schauen
- /temp/ das Verzeichnis wird für das Speichern temporärer Dateien sowie der Projektdaten benötigt, bitte nicht löschen
- /wms_monitor.xml XML-Datei, die Monitor-Prozesse enthält
- /wms_settings.xml XML-Datei, die alle Einstellungen enthält
- /wms_types.xml XML-Datei, die Report-Typen enthält

D.5 Bedienung

Nach dem Starten der Web Measurement Suite erscheint die Nutzeroberfläche wie in Abbildung D.1. Links ist der Projektbaum abgebildet, der alle bestehenden Messprojekte bereitstellt. In der Abbildung wird momentan das Projekt *W3C* geladen.

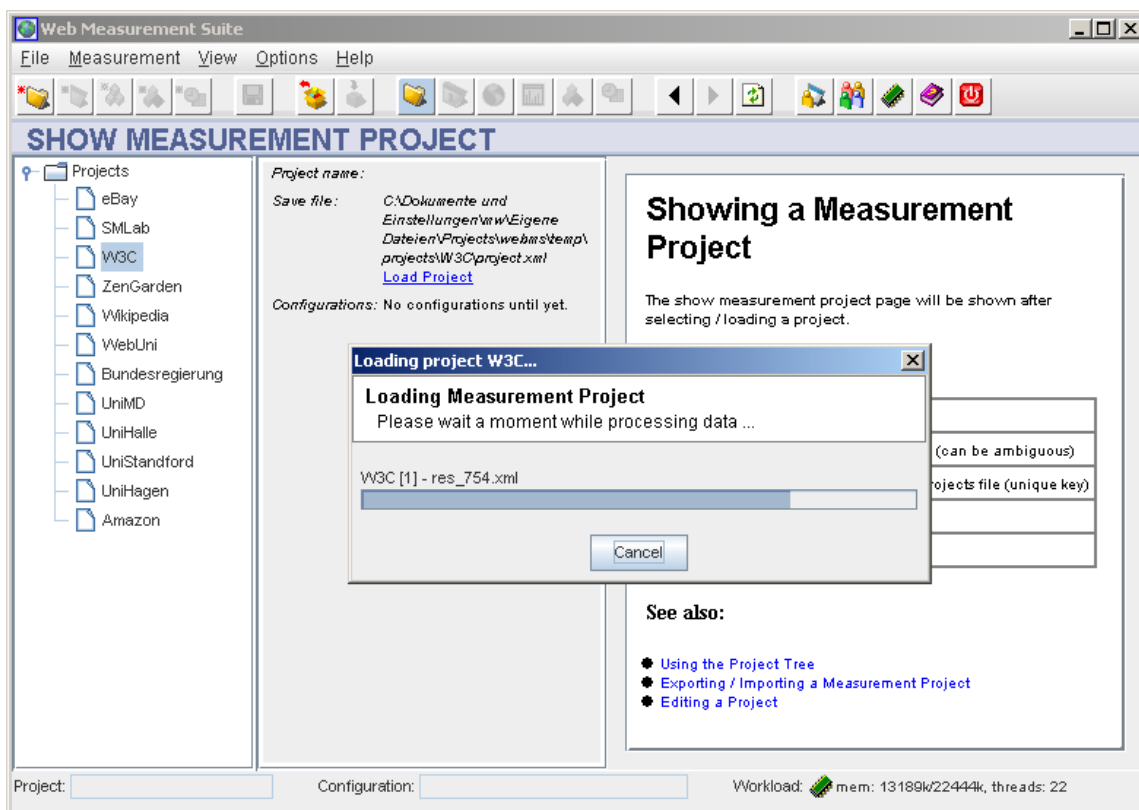


Abbildung D.1: Web Measurement Suite: Laden eines Messprojektes

In der Mitte sind Projektdaten dargestellt und rechts eine themenbezogene Hilfeseite. Die Hilfe wird durchgängig an wichtigen Stellen der Web Measurement Suite angezeigt und verhilft dem Nutzer so zu einem einfachen Einstieg. Um die gesamte Hilfe in einem separaten Fenster anzuschauen, ist im Menü *Hilfe* der Eintrag *Hilfe*, der auch über F1 erreicht werden kann.

Ein zweites Beispiel der Nutzeroberfläche in Abbildung D.2 zeigt einen generierten Qualitätsreport. Der Report kann direkt in der Web Measurement Suite angeschaut werden. Um den Report per E-Mail zu versenden, muss im Textfeld in der zweiten Symbolleiste eine E-Mailadresse eingegeben werden und der E-Mail-Knopf gedrückt werden.

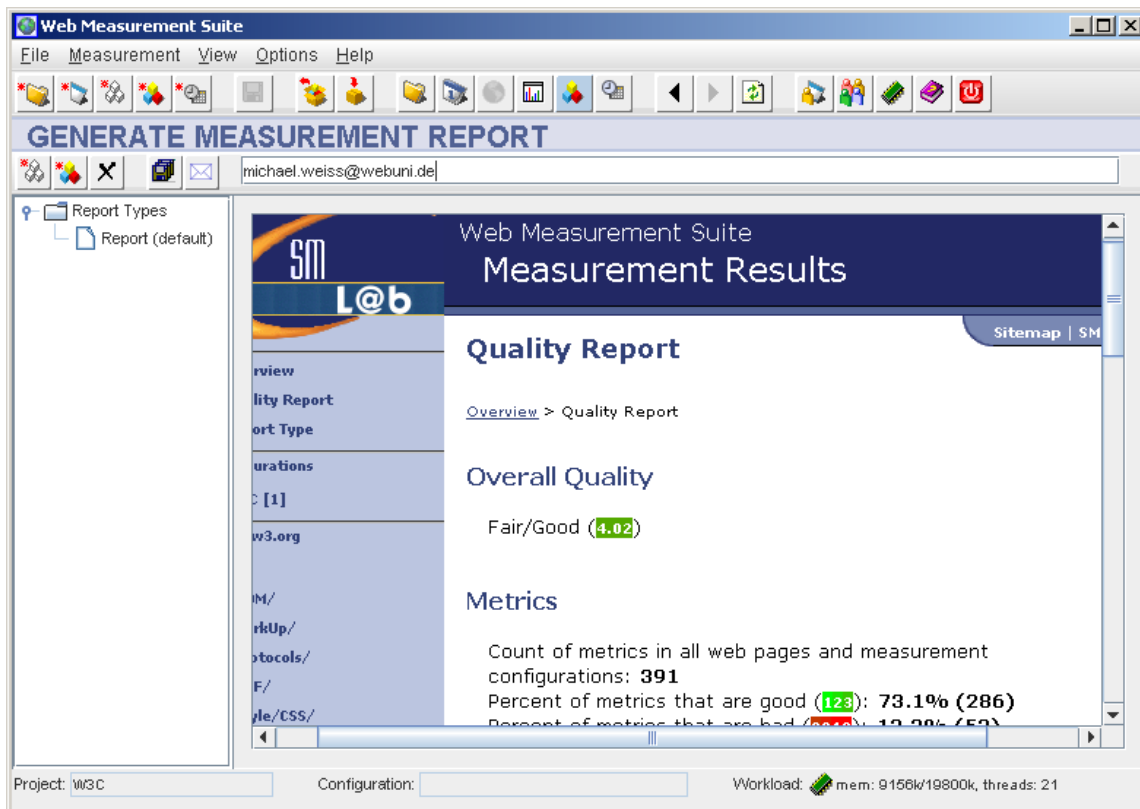


Abbildung D.2: Web Measurement Suite: Einen Qualitätsreport anschauen

Über der zweiten Symbolleiste ist die Titelzeile, die den momentanen Standort innerhalb der Web Measurement Suite darstellt. Die erste Symbolleiste bietet die Möglichkeit, zwischen den Bereichen zu wechseln und neue Messprojekte zu erstellen.

In der Statuszeile am unteren Ende der Abbildung sind das momentan ausgewählte Messprojekt⁵⁵ und die selektierte Konfiguration dargestellt.

⁵⁵In der GUI kann gleichzeitig nur ein Messprojekt geöffnet sein (siehe Abbildung 6.4 auf Seite 77).

Abbildung D.3 stellt WebUni.de im integrierten Browser dar. Dieser Browser ist nur verfügbar, wenn das ICEbrowser SDK im Verzeichnis `/__lib/__ice/` eingebunden ist. Ist das SDK nicht verfügbar, ist der interne Browser nicht anwählbar. Die integrierten Hilfeseiten und die Reportansicht in Abbildung D.2 sind davon nicht betroffen, da sie auf dem nicht so leistungsfähigen Java-Browser von Sun basieren.



Abbildung D.3: Web Measurement Suite: WebUni.de im integrierten Browser

Weitere Hilfen zur Nutzeroberfläche sind der integrierten Hilfe zu entnehmen bzw. der Hilfe auf der CD im Verzeichnis `/__doc/__help/`.

D.6 Die Web Measurement Suite als Verteiltes System

Das Verzeichnis `/agents/` enthält das JAR-Paket `wmsremote.jar` um Softwareagenten verteilt auf anderen Rechnern zu starten und diese in der Web Measurement Suite zu integrieren.

Die Agenten können entweder auf demselben Computer wie die Web Measurement Suite gestartet werden oder von einem in einem Netzwerk verbundenen Rechner.

Die Startdatei „help.bat“ bzw. „help.sh“ zeigt eine Hilfe an, wie das Paket bedient wird. Die möglichen Parameter lauten:

```
java -jar wmsremote.jar [-?] [-a {AGENT} [-h IP|HOST [-localgui]]]
./run_agent.bat [-?] [-a {AGENT} [-h IP|HOST [-localgui]]]
./run_agent.sh [-?] [-a {AGENT} [-h IP|HOST [-localgui]]]
```

-? zeigt die Hilfe an.

-a Agent startet den Agenten namens Agent, mögliche Werte sind *wanderer*, *urlparser*, *httpreceiver* und *htmlparser*

-h IP|HOST ist optional, mit diesem Parameter kann ein Verbindungsversuch mit einem entfernten Rechner unternommen werden

-localgui ist ebenfalls optional, bei einem Verbindungsaufbau mit einem entfernten Rechner, wird bei Angabe dieses Parameters die GUI des Agenten lokal angezeigt und nicht innerhalb der Web Measurement Suite.

Wenn z.B. `./run__agent.sh -a htmlparser -h 192.168.100.150 -localgui` eingegeben wird, startet der HTMLParser und verbindet sich auf IP 192.168.100.150 mit der dortigen Web Measurement Suite. Die GUI wird allerdings lokal angezeigt.

Diesen Sachverhalt zeigen die Abbildungen D.4 und D.5 auf Seite 154.

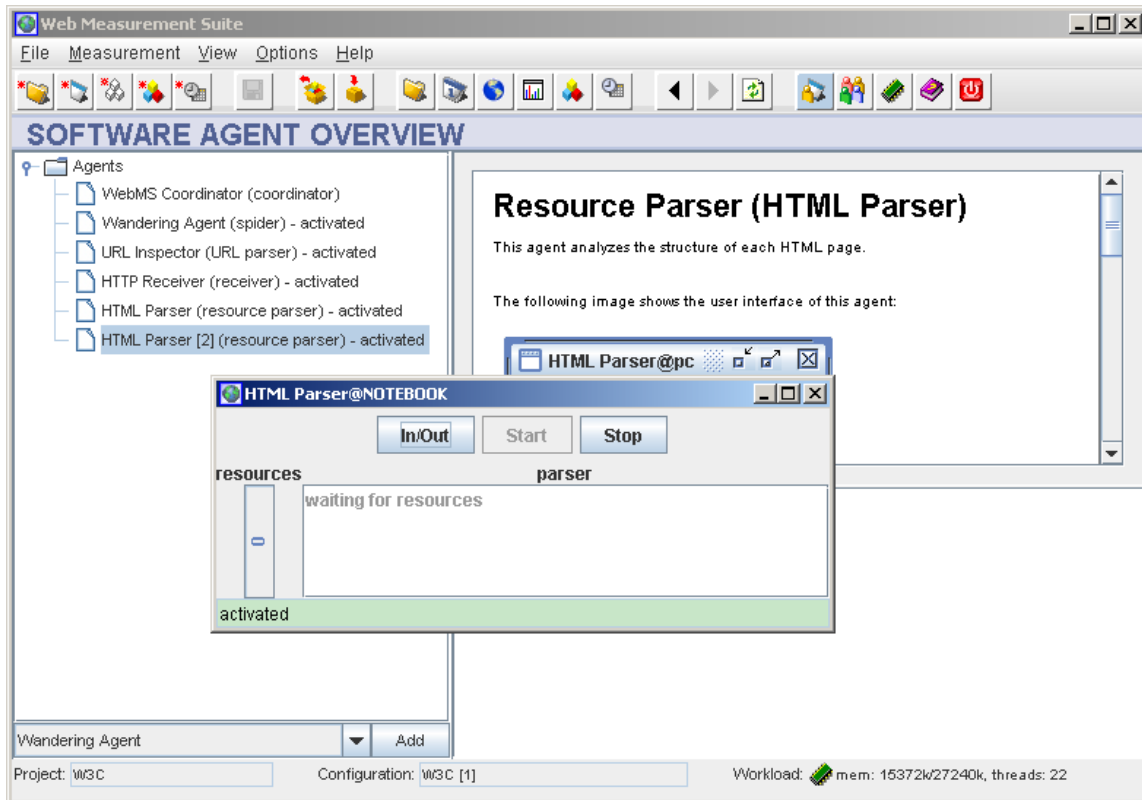


Abbildung D.4: Web Measurement Suite: lokaler HTMLParser auf Notebook



Abbildung D.5: Web Measurement Suite: entfernter HTMLParser auf PC

Literaturverzeichnis

- [1] BARABÁSI, Albert-László ; ALBERT, Réka: Emergence of Scaling in Random Networks. In: *Science* Vol. 286 (Oktober 1999), S. 509–512
- [2] BERLIN-BRANDENBURGISCHE AKADEMIE DER WISSENSCHAFTEN: DWDS - das Digitale Wörterbuch der Deutschen Sprache des 20. Jahrhunderts. – <http://www.dwds.de/> - Besucht am 08.09.2004.
- [3] BERNERS-LEE, Tim ; FIELDING, Roy T. ; GETTYS, James ; MOGUL, Jeffrey C. ; NIELSEN, Henrik F. ; MASINTER, Larry ; LEACH, Paul J.: Hypertext Transfer Protocol – HTTP/1.1 / Network Working Group. June 1999. – RF2616.
- [4] BERNERS-LEE, Tim ; FIELDING, Roy T. ; MASINTER, Larry: Uniform Resource Identifier (URI): Generic Syntax / Network Working Group. August 1998. – RF2396.
- [5] BERNERS-LEE, Tim ; FIELDING, Roy T. ; MASINTER, Larry: Uniform Resource Identifier (URI): Generic Syntax / Network Working Group. Juli 2004. – <http://gbiv.com/protocols/uri/rev-2002/rfc2396bis.html> - Besucht am 04.09.2004.
- [6] BIENE - BARRIEREFREIES INTERNET ERÖFFNET NEUE EINSICHTEN: Kriterien zum BIENE-Award 2004. – <http://www.biene-award.de/award/kriterien/krit.cfm> - Besucht am 15.11.2004.
- [7] BRADEN, Robert: Requirements for Internet Hosts – Communication Layers / Network Working Group. Oktober 1989. – RFC1122.
- [8] BRENNER, Walter ; ZARNEKOW, Rüdiger ; WITTIG, Hartmut: *Intelligente Softwareagenten. Grundlagen und Anwendungen.* . Lehrstuhl für Wirtschaftsinformatik, TU Bergakademie Freiberg : Springer-Verlag, Oktober 1997. – ISBN 3–540–63431–2
- [9] BRIN, Sergey ; PAGE, Lawrence: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: *7th Int. World Wide Web Conf.* (1998)

- [10] CHO, Junghoo ; GARCIA-MOLINA, Hector: The Evolution of the Web and Implications for an Incremental Crawler. In: *Proceedings of the 26th International Conference on Very Large Databases*, 2000, S. 200–209
- [11] DEERING, Stephen E. ; HINDEN, Robert M.: Internet Protocol, Version 6 (IPv6) Specification / Network Working Group. Dezember 1995. – RFC1883.
- [12] DENIC: DENIC startet Internationalized Domain Names (IDNs) unter .de... – <http://www.denic.de/de/domains/idns/index.html> - Besucht am 05.09.2004.
- [13] DUMKE, Reiner R.: *Software Engineering*. 4. Auflage. Fakultät für Informatik, Universität Magdeburg : Vieweg, Dezember 2003. – ISBN 3–528–35355–4
- [14] DUMKE, Reiner R. ; LOTHER, Mathias ; WILLE, Cornelius ; ZBROG, Fritz: *Web Engineering*. . Pearson Studium, September 2003. – ISBN 3–8273–7080–9
- [15] DUMKE, Reiner R. ; SCHÄFER, Uwe ; WILLE, Cornelius ; ZBROG, Fritz: Agentenbasierte Web-Technologiebewertung für das Performance Engineering. In: *5. Workshop Performance Engineering in der Softwareentwicklung, Tagungsband, Siemens München* (Mai 2004), S. 37–66
- [16] EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION (ECMA): ECMAScript Language Specification. – Standard ECMA-262. 3rd Edition - Dezember 1999.
- [17] GEBHARDT, Klaus: quality-Datenbank - Das QM-Lexikon. – <http://www.quality.de/lexikon.htm> - Besucht am 08.09.2004.
- [18] HEIDER sören ; PAUER, Raik: *Konzeption und prototypische Implementierung einer Visualisierungsinfrastruktur für tomographbasiertes Web-Measurement*, Otto-von-Guericke-Universität Magdeburg, Diplomarbeit, Juni 2004
- [19] HEISE ONLINE: VeriSign nimmt Site Finder vom Netz. – <http://www.heise.de/newsticker/meldung/40815> - Besucht am 18.10.2004.
- [20] HORN, Torsten: *Internet, Intranet, Extranet - Potentiale in Unternehmen*. . Oldenbourg Verlag, Mai 1999. – ISBN 3–486–25129–5
- [21] ICESOFTECHNOLOGIES INC.: *ICEbrowser SDK 5.4 Programmer's Guide*. . ICE-Soft Technologies Inc., Dezember 2003
- [22] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO): ISO 3166 Country Codes. – <http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/index.html> - Besucht am 04.09.2004.

-
- [23] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO): ISO 9000 Quality Management. – <http://www.iso.org/iso/en/iso9000-14000/iso9000/iso9000index.html> - Besucht am 08.09.2004.
- [24] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION (ISO): ISO/IEC 9126: 1991 Software Product Evaluation, Quality, Characteristics, and Guidelines for their Use. – Specification.
- [25] INTERNET ASSIGNED NUMBERS AUTHORITY (IANA): Uniform Resource Identifier (URI) SCHEMES. – <http://www.iana.org/assignments/uri-schemes> - Besucht am 04.09.2004.
- [26] INTERNET ASSIGNED NUMBERS AUTHORITY (IANA): Well Known Port Numbers. – <http://www.iana.org/assignments/port-numbers> - Besucht am 03.09.2004.
- [27] JENDRYSCHIK, Miuchael: Pinselführung - Wie Browser über die Darstellung von (X)HTML entscheiden. In: *iX 3/2004* (März 2004)
- [28] LEVENE, Mark ; POULOVASSILIS, Alexandra ; FLAKE, Gary W. ; TSIOUTSIOULIKLIS, Kostas ; ZHUKOV, Leonid: *Web Dynamics - Adapting to Change in Content, Size, Topology and Use*. . Springer-Verlag, Mai 2004. – ISBN 3-540-40676-X
- [29] LIGGESMEYER, Peter: *Software-Qualität - Testen, Analysieren und Verifizieren von Software*. . Hasso-Plattner-Institut, Universität Potsdam : Spektrum, Akadem. Verlag, August 2002. – ISBN 3-8274-1118-1
- [30] LOSHIN, Pete: *IPv6 Theory, Protocol, and Practice*. 2nd Edition. Morgan Kaufmann Publishers, April 2004. – ISBN 1-558-60810-9
- [31] MENASCÉ, Daniel A. ; ALMEIDA, Virgilio A.: *Capacity Planning for Web Performance - Metrics, Models, and Methods*. . Prentice Hall, Mai 1998. – ISBN 0-13-693822-1
- [32] OLSINA, Luis ; LAFUENTE, Guillermo ; ROSSI, Gustavo: Specifying Quality Characteristics and Attributes for Websites. In: *Lecture Notes in Computer Science* 2016 (2001), S. 266
- [33] PATTON, Susannah: Web Metrics That Matter. In: *CIO Magazine* (November 2002)
- [34] POSTEL, John: Internet Protocol / Defense Advanced Research Projects Agency (DARPA). September 1981. – RFC791.
- [35] POSTEL, John: Transmission Control Protocol / Defense Advanced Research Projects Agency (DARPA). September 1981. – RFC793.

- [36] SCHARL, Arno: *Evolutionary web development*. . Springer-Verlag, Oktober 2000. – ISBN 1–85233–310–3
- [37] SCHÄFER, Uwe: *Analyse und Auswertung von Webpräsentationen auf der Grundlage eines implementierten Web-Tomographen*, Otto-von-Guericke-Universität Magdeburg, Diplomarbeit, Februar 2004
- [38] SCHMIETENDORF, Andreas: Modellbezogene Notationen, Methoden und Tools für ein Software Performance Engineering. In: *Fakultät für Informatik, O-v-G-Universität Magdeburg* Preprint Nr. 15 (Dezember 2000)
- [39] SELFHTML E.V.: SelfHTML. – <http://de.selfhtml.org/> - Besucht am 06.09.2004.
- [40] SOMMERVILLE, Ian: *Software Engineering*. 6. Auflage. Addison-Wesley, Pearson Studium, Juni 2001. – ISBN 3–8273–7001–9
- [41] STATISTISCHES BUNDESAMT DEUTSCHLAND: Ausstattung mit Gebrauchsgütern und Wohnsituation privater Haushalte. In: *Wirtschaft und Statistik* (Februar 2004), Nr. 2, S. 213–214
- [42] STERNE, Jim: *Web Metrics - Proven Methods for Measuring Web Site Success*. . John Wiley and Sons, Juni 2002. – ISBN 0–471–22072–8
- [43] SUN MICROSYSTEMS, INC.: *JavaHelp 2.0 System User's Guide*. . Sun Microsystems, Inc, Oktober 2003
- [44] SUN MICROSYSTEMS, INC.: *JavaMail API 1.2 Design Specification*. . Sun Microsystems, Inc, September 2000
- [45] TANENBAUM, Andrew S.: *Computernetzwerke*. 3. Auflage. Vrije Universiteit, Amsterdam, The Netherlands : Prentice Hall, Pearson Studium, November 2000. – ISBN 3–8273–7011–6
- [46] UNIVERSITÄT DES SAARLANDES - INFORMATIONSWISSENSCHAFT: Browserkrieg - Microsoft gegen Netscape. – <http://server02.is.uni-sb.de/courses/ident/kontroverses/browserkrieg/> - Besucht am 07.09.2004.
- [47] WEAKLEY, Russ ; MÜLLER, Mario: A web standards checklist. August 2004. – <http://www.lingo4u.de/article/checklist/> - Besucht am 19.09.2004.
- [48] WEB STANDARDS PROJECT (WASP): Standards in the WWW. – <http://www.webstandards.org/> - Besucht am 19.09.2004.

- [49] WIKIMEDIA FOUNDATION INC.: Wikipedia - The Free Encyclopedia - Allgemeine Informationen. – <http://www.wikipedia.org/> - Besucht am 18.10.2004.
- [50] WILDE, Erik: *Wilde's WWW - Technical Foundations of the World Wide Web.* . International Computer Science Institute, Berkeley CA, USA : Springer-Verlag, Juni 1999. – ISBN 3-540-64285-4
- [51] WILLE, Cornelius: Framework zur Messung und Bewertung der Entwicklung von Multi-Agenten-Systemen. In: *Tage der Doktoranden an der Fakultät für Informatik der Otto-von-Guericke Universität Magdeburg*, Juli 2004
- [52] WILLE, Cornelius ; DUMKE, Reiner ; STOJANOV, Stanimir: Softwaremessung und -Bewertung für agentenbasierte Systementwicklung und Anwendung. In: *Workshop der GI-Fachgruppe 2.1.10. „Software-Messung und Bewertung“*, September 2001
- [53] WORLD WIDE WEB CONSORTIUM (W3C): Allgemeine Informationen. – <http://www.w3.org/> - Besucht am 07.09.2004.
- [54] WORLD WIDE WEB CONSORTIUM (W3C): Cascading Style Sheets Home Page. – <http://www.w3.org/Style/CSS/> - Besucht am 06.09.2004.
- [55] WORLD WIDE WEB CONSORTIUM (W3C): HTML 4.01 Specification. – <http://www.w3.org/TR/html401/> - Besucht am 06.09.2004.
- [56] ZUSE, Horst: *Software Complexity.* . de Gruyter, 1991. – ISBN 3-1101-2226-X
- [57] ZUSE, Horst: *A Framework of Software Measurement.* . de Gruyter, 1998. – ISBN 3-1101-5587-7