

Masterarbeit

*zur Erlangung des akademischen Grades
Master of Science Medieninformatik*

*Vorgelegt dem Fachbereich Informationstechnik, Elektrotechnik,
Mechatronik (IEM) der Technischen Hochschule Mittelhessen*



Thema:

*Möglichkeiten zur Integration eines Echtzeit-Webanalyse-Systems
in bestehende Web-Applikationen*

Vorgelegt von:

Thomas Winkelmann

Referent: Prof. Dr. Stephan Euler

Korreferent: Prof. Dr. Michael Guckert

Studiengang: Medieninformatik (Master of Science)

Friedberg im September 2011

Zusammenfassung

Die aus dem Entwicklungsprojekt hervorgegangene Web-Analyse-Plattform zur Visualisierung des Besucherverhaltens auf einer Website in Echtzeit soll in dieser Masterarbeit darauf hin untersucht werden, ob sich ihre Einsatzfähigkeit unter praxisnahen Bedingungen bestätigt. Ebenso wird erörtert, wie sich die bestehenden Komponenten optimieren lassen, sodass der Nutzwert der Plattform für einen Website-Betreiber noch weiter steigt. Ebenso wird getestet, wie sich die Plattform in andere Systeme integrieren lässt oder mit diesen über Schnittstellen zusammenarbeiten kann.

Im ersten Schritt werden dabei die vorhandenen Module vor allem in Bezug auf die Art und Weise der Visualisierung der Daten als Diagramme optimiert, sodass diese einfacher und intuitiver zu verstehen sind und die Nutzer schnell die markanten Daten erkennen. Darüber hinaus werden zahlreiche Optimierungen an der Programmierung vorgenommen, sodass eine zuverlässigere Erfassung von Benutzerinteraktionen ermöglicht wird und alle aktuellen Browser unterstützt werden.

Die Integration des Systems in bestehende Web-Applikation wird an den Beispielen *Moodle* und *Typo3* gezeigt. In beide Web-Applikationen wird sowohl der Code zum Erfassen von Besucherinteraktionen, als auch die Visualisierung der damit erfassten Daten eingebunden.

Des Weiteren wird ein neues Modul für die Web-Analyse-Plattform erstellt, mit dem die im System vorhandenen Daten in eine weitere Web-Analyse-Plattform, hier am Beispiel *Piwik*, transferiert werden und somit auch längerfristig zur Verfügung stehen.

Ebenso wird mit der Erweiterung bestehender Module durch eine Web-App die Nutzung der Plattform auf mobilen Geräten wie Smartphones oder Tablets ermöglicht.

Inhaltsverzeichnis

Zusammenfassung	2
1 Einleitung	1
1.1 Motivation	1
1.2 Aufgabenstellung	2
2 Vorstellung des bestehenden Systems	4
2.1 Systemkomponenten	4
2.2 Integration in eine Website	7
3 Evaluierung des bestehenden Systems	8
3.1 Darstellung als Verlaufslinie	8
3.2 Dynamische Achsenskalierung	10
3.3 Zeitlicher Vergleich	11
3.4 Anzeige des aktuellen Zeitblocks	11
3.5 Sicherheitswarnung des JavaScript-Tracking-Codes	12
4 Optimierung des bestehenden Systems	14
4.1 Säulendiagramm und statische Achsenskalierung	14
4.2 Vergleich aktueller Werte mit historischen Daten	17
4.3 Anzeige von Messwerten als Ziffernausgabe	18
4.4 Einsatz des JavaScript-Tracking-Codes im Internet Explorer	19
5 Integrationsanforderungen	22
5.1 Allgemeine Anforderungen	22
5.2 Erfassung von Besucherdaten	23
5.3 Auswertung von Besucherdaten	25
5.4 Integrationstiefe	26
6 Integrationsbeispiel Typo3	27
6.1 Vorstellung des CMS Typo3	27
6.2 Erfüllung der Integrationsanforderungen	28
6.3 Integration des Trackingcodes	31
6.3.1 Grundsetup des Trackingcodes	31
6.3.2 Erweiterung des Tracking-Codes	35

6.3.3	Auslagerung des Tracking-Codes in eine eigene Extension.....	37
6.3.4	Anbindung an weitere Extensions	38
6.4	Integration der Auswertung.....	41
7	Integrationsbeispiel Moodle.....	47
7.1	Vorstellung der E-Learning-Plattform Moodle.....	47
7.2	Erfüllung der Integrationsanforderungen.....	48
7.3	Integration des Trackingcodes	51
7.4	Integration der Auswertung.....	54
8	Integrationsbeispiel Piwik.....	57
8.1	Vorstellung von Piwik.....	57
8.2	Integrationsmöglichkeiten	58
8.3	Erweiterung des Server-Moduls	61
8.4	Realisierung des Transfer-Moduls	63
8.5	Skalierbarkeit.....	67
9	Integrationsbeispiel Mobile-App.....	69
9.1	Motivation.....	69
9.2	Vorteile einer Web-App	70
9.3	Erweiterung des Präsentations-Moduls.....	70
9.4	Weitere Optimierungen der Web-App	75
10	Fazit und Ausblick.....	76
10.1	Aktueller Stand	76
10.2	Ausblick	77
11	Literatur- und Quellenverzeichnis.....	V
12	Diagramm-, Tabellen- und Abbildungsverzeichnis.....	VIII
13	Abkürzungsverzeichnis	XI
14	Anlagenverzeichnis.....	XII
14.1	CD-ROM.....	XII

1 Einleitung

1.1 Motivation

Im vergangenen Wintersemester 2010/2011 wurde im Rahmen des Entwicklungsprojektes aus dem Masterstudiengang Medieninformatik eine Web-Applikation erstellt, mit der es einem Website-Betreiber ermöglicht wird, den zeitlichen Verlauf von Seitenabrufen auf seiner Website in Echtzeit zu beobachten und auszuwerten.

Der Fokus bei der Entwicklung lag dabei weniger auf dem Funktionsumfang, welcher bei zahlreichen Produkten aus diesem Anwendungsbereich (z.B. *Google Analytics* oder *Piwik*) inzwischen nahezu unüberschaubar geworden ist, sondern vielmehr auf der Verarbeitung und Darstellung der gemessenen Besucherdaten mit nur einer sehr geringen zeitlichen Verzögerung. Ebenso wurde besonderes Augenmerk auf die Leistungsfähigkeit des Systems gelegt. So verarbeitet die Anwendung einen einzelnen Seitenaufruf sehr ressourcenschonend, um so auch auf stark besuchten Websites oder solchen mit großen kurzzeitigen Lastspitzen eingesetzt werden zu können.

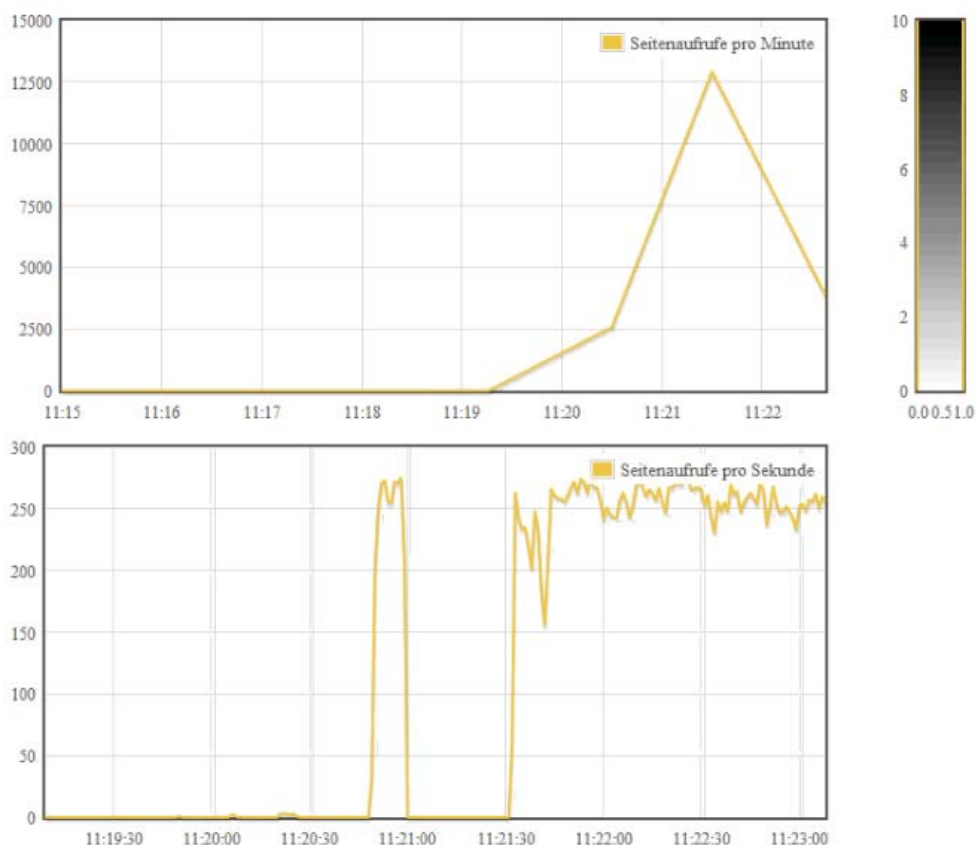


Abbildung 1 - Präsentations-Modul der entwickelten Plattform während eines Lasttests am Ende des Entwicklungsprojektes

Die Plattform wurde in ihrer Grundfunktionalität, dem Erfassen und Darstellen von Seitenabrufen, technisch nahezu vollständig implementiert. Während der abschließenden Tests hat sich immer wieder gezeigt, dass das System problemlos große Besucheranstürme verarbeiten kann, auch wenn es auf einem Server mit geringen Hardware-Ressourcen eingesetzt wird. Alle Testergebnisse bezogen sich jedoch immer auf Seitenabrufe, die mit diversen Benchmark-Tools automatisiert erzeugt wurden. Ein Einsatz unter Realbedingungen ist bis zu diesem Zeitpunkt noch nicht erfolgt und es ist eine spannende Frage, ob die entwickelte Plattform auch im Praxisbetrieb reibungslos arbeitet und in diesem auch einen Mehrwert für den Anwender mit sich bringt.

Da es bislang kein zur entwickelten Plattform vergleichbares Produkt gibt, das sich in einer ähnlichen Art und Weise auf die Echtzeitanzeige von Web-Analyse-Daten spezialisiert hat, erscheint es zudem sehr sinnvoll, dieses System weiter auszubauen und detailliert zu untersuchen, in welchen Umgebungen es genutzt und integriert werden kann.

Diese Masterarbeit soll daher das Thema aus dem vorangegangenen Entwicklungsprojekt aufgreifen und es im Hinblick auf die Anwendungsfälle und Integrationsmöglichkeiten der Plattform weiter behandeln. Die genaue Aufgabenstellung wird im nachfolgenden Abschnitt beschrieben.

1.2 Aufgabenstellung

Diese Masterarbeit gliedert sich in die folgenden Teilaufgaben:

- **Nutzungstest unter realen Bedingungen:**

Das System soll auf bestehenden Websites eingebunden werden. Dabei soll über einen längeren Zeitraum hinweg und mit Hilfe eines Vergleichssystems überprüft werden, ob alle Seitenaufrufe erfasst werden und ob markante Besucheranstürme in den vorhandenen Anzeigetypen des Präsentationsmoduls erkannt werden können. Nur wenn dieser Punkt erfüllt ist, bringt das System überhaupt einen Mehrwert für einen Website-Betreiber.

- **Optimierung der beiden vorhandenen Module:**

Aus den Ergebnissen sollen das Server- und Präsentationsmodul soweit optimiert werden, dass sie für einen Nutzer möglichst einfach und intuitiv zu verwenden sind. Die Optimierung soll sich dabei jedoch nur auf die im Entwicklungsprojekt geplanten und umgesetzten Funktionen beziehen und die beiden Module nicht um zahlreiche weitere und neue Funktionen erweitern.

- **Integration in bestehende Web-Applikationen:**

Als Hauptteil dieser Arbeit sollen die verschiedenen Möglichkeiten aufgezeigt werden, die Plattform in bestehende Web-Applikationen zu integrieren. Hierzu wird zunächst geklärt, welche Anforderungen bei den jeweiligen Systemen erfüllt sein müssen, damit eine Integration möglich wird.

Im Anschluss soll gezeigt werden, mit welchen Techniken die beiden Module in eine bestehende Web-Applikation integriert werden können, sowie auf welche Weise sie mit dem System zusammenarbeiten können. Als Beispiele sollen hier ein Content-Management-System und eine eLearning-Plattform zum Einsatz kommen.

Ziel der Integration der Plattform in bestehende Systeme ist es, die Nutzung für einen Anwender deutlich zu erleichtern, da er somit in der Lage ist, auf die Webanalysedaten in seiner gewohnten Arbeitsumgebung zuzugreifen.

- **Anbindung an bestehende Web-Applikationen:**

Neben der Integration soll außerdem untersucht werden, ob und wie die Plattform an bestehende Web-Applikationen angebunden werden kann. So soll eine Möglichkeit geschaffen werden, die erfassten Besucherdaten zu exportieren, in ein anderes System zu übertragen und dort zu importieren, um sie so zum Beispiel in einem Langzeit-Archiv zum Abruf bereit zu stellen.

2 Vorstellung des bestehenden Systems

2.1 Systemkomponenten

Während der Konzeptionsphase zur Erstellung der Webanalyse-Plattform wurde die Funktionalität und Programmierung in zwei wesentliche Module unterteilt. Diese werden im Folgenden mit den Begriffen **Server-Modul** und **Präsentations-Modul** bezeichnet:

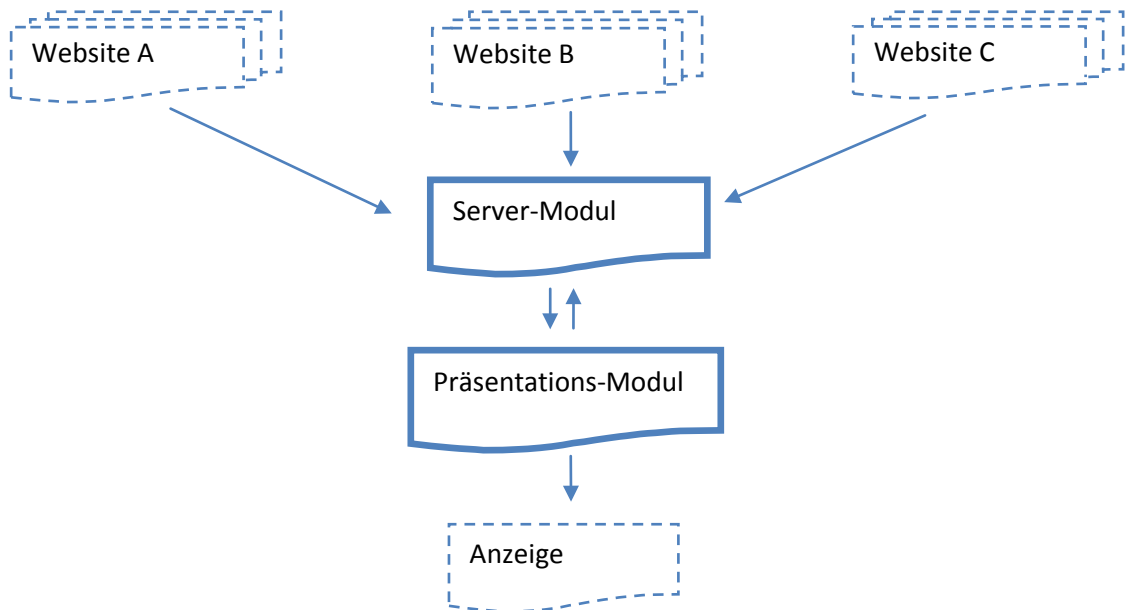


Abbildung 2 - Grundstruktur der Module

Das Server-Modul dient in erster Linie dazu, die von den einzelnen Websites erzeugten Trackinginformation zu speichern und anschließend auszuwerten, sowie zum Abruf durch andere Module oder Systeme bereitzustellen. Es stellt dazu unter anderem verschiedene Tracking-Codes zur Verfügung. Die Integration dieser Codes in eine oder mehrere Websites wird im nachfolgenden Abschnitt detailliert erläutert.

Das Modul umfasst außerdem eine webbasierte Administrationsoberfläche in der die verschiedenen Konfigurationsparameter zu den einzelnen Websites übersichtlich eingegeben werden können. Ebenso können in dieser Oberfläche alle anderen globalen Parameter des Systems administriert werden.

Intern arbeitet das Modul mit drei Zeiträumen. Diese sind:

- **Rohdaten-Speicherzeit:**

Innerhalb dieser Zeitspanne werden die Seitenabrufe mit allen dazugehörigen Parametern gespeichert. Hierzu gehören zu jedem Seitenaufruf Daten wie die genaue Adresse der aufgerufenen Seite, der Referrer des Seitenaufrufes, die

Bildschirmauflösung des Besuchers, usw. Je nach Frequentierung der Website können hier große Datenmengen anfallen.

- **Kurzzeit-Speicherzeit/Intervall:**

Dies ist die erste Komprimierungsstufe des Systems. Die angefallenen Daten werden innerhalb des angegebenen Kurzzeit-Intervalls zu einem Wert für das jeweilige Merkmal zusammengefasst.

- **Langzeit-Speicherzeit/Intervall:**

Dies ist die zweite Komprimierungsstufe, bei der die Daten aus den vorhandenen Kurzzeit-Intervallen erneut zusammengefasst und in nur noch einem Wert für ein Langzeit-Intervall zusammengefasst werden.

Projekte

Aktuell im System angelegte Projekte, für die Besucherdaten erfasst werden sollen.

Projekt-ID: 1

Projekt-Bezeichnung*:	Meine Website
Zeit für Besuch*:	3600
Rohdaten-Speicherzeit*:	240
Kurzzeit-Speicherzeit*:	7200
Kurzzeit-Intervall*:	60
Langzeit-Speicherzeit*:	172800
Langzeit-Intervall*:	3600

Entfernen

Projekt hinzufügen

Abbildung 3 - Konfiguration einer Website in der Administrationsoberfläche

Das **Server-Modul** lässt sich durch Plug-Ins erweitern. So können individuelle Storage-Backends und Export-Formate hinzugefügt werden, um die Besucherdaten beispielsweise in einem speziellen Datenbankmanagement-System zu speichern oder um diese Daten in einem individuellen Datenformat zu exportieren.

Das Modul speichert wie erwähnt im Rohdatenspeicher alle Seitenabrufe mit allen dazugehörigen Parametern. Eine Auswertung und die weitere Verarbeitung der Daten erfolgt momentan lediglich nach der Anzahl von Seitenabrufen innerhalb eines bestimmten Zeitraumes. Die Programmierung ist jedoch bereits so vorbereitet, dass auch eine Auswertung nach Anzahl der Besuche oder der Anzahl von Seitenabrufen innerhalb eines bestimmten Bereichs der Website leicht implementiert werden kann. Um diese Operationen möglichst performant durchzuführen, werden sie immer im Bereich des

jeweiligen Storage-Backend-Plug-Ins implementiert, sodass hier ein Algorithmus gefunden werden kann, der möglichst gut auf die verwendete Speichertechnologie zugeschnitten ist.

Auf der Gegenseite des Server-Moduls steht das **Präsentations-Modul**. Es dient zur Visualisierung der Daten mit Hilfe von Diagrammen. Es wurde speziell für den Einsatz auf einer Leinwand oder einem fest installierten Bildschirm konzipiert. Aus diesem Grund verfügt die Oberfläche nicht über Bedienelemente wie Buttons oder Optionsfelder, mit denen sich die Anzeige individuell anpassen ließe. Stattdessen werden alle darzustellenden Elemente einmalig von einem Administrator in einer Konfigurations-XML-Datei hinterlegt. Hier definiert er neben der Größe, Farbe und Position der Diagramme auch, welche Daten vom Server-Modul für dieses Diagramm abgefragt werden sollen, wie hoch die Aktualisierungsrate oder wie die Achsenskalierung der Diagramme erfolgen soll.

Derzeit ist das Präsentations-Modul in der Lage, zweidimensionale Diagramme mit jeweils einer Datenreihe anzuzeigen. Durch die verschiedenen Konfigurationsoptionen lassen sich zahlreiche Visualisierungsformen finden, die das Besucherverhalten auf der Website in den unterschiedlichsten Formen veranschaulichen können.

Beide Module wurden in der serverseitigen Skriptsprache PHP umgesetzt. Zur Speicherung der Daten wird derzeit im Server-Modul *Memcached* genutzt. Die Visualisierung der Informationen im Präsentations-Modul erfolgt clientseitig im Browser mit Hilfe von JavaScript. Die Kommunikation der beiden Module erfolgt mittels HTTP, als Datenformat wird dabei XML genutzt.

Aus den beiden beschriebenen Modulen wird erneut sichtbar, dass es sich bei dieser Plattform also nicht um ein klassisches Web-Analyse-System handelt, sondern um ein auf die Echtzeit-Darstellung von Besucherdaten in den drei genannten Zeiträumen spezialisiertes System. Detaillierte Recherchen nach mehreren Kriterien in der Form "Was ist der meistgenutzte Browser auf Unterseite X?" oder "Wie hoch ist der Anteil an Besuchern mit installiertem Flashplayer-Plug-In?" kann die Plattform in ihrer jetzigen Form nicht leisten.

Durch die Modularisierung der einzelnen Komponenten ist es problemlos möglich, die Plattform durch weitere Module zu ergänzen, die spezielle Funktionalitäten abdecken. Somit wäre es auch denkbar, ein Modul zu erstellen, mit dessen Hilfe dann die Durchführung komplizierterer Abfragen ermöglicht wird oder ein Modul zu entwickeln, das auf die Visualisierung der Daten auf einem Desktop-Rechner spezialisiert ist.

2.2 Integration in eine Website

Die Reihenfolge der minimal notwendigen Arbeiten, die ein Administrator erledigen muss, damit eine Seite mit Hilfe der Plattform in Echtzeit analysiert werden kann, ist dabei:

- **Anlegen des Projektes im Servermodul:**

Mit Hilfe des GUI des Server-Moduls kann ein Administrator mit wenigen Klicks ein neues Projekt anlegen. Dabei muss er neben dem Namen des Projektes auch die verschiedenen Speicherzeiträume und Intervalle festlegen. Typische Werte sind dabei: Rohdaten-Speicherzeit: 600 Sekunden für 5 Minuten. Kurzzeit-Speicherzeit/Intervall: 7200 Sekunden für zwei Stunden, sowie eine Zusammenfassung der Werte in Gruppen von 60 Sekunden, sodass sich Werte "pro Minute" ergeben. Langzeit-Speicherzeit/Intervall: 86400 Sekunden für einen Tag, sowie eine erneute Zusammenfassung der Werte in Gruppen von 3600 Sekunden, sodass sich Werte "pro Stunde" ergeben.

Die Werte müssen dabei jeweils unter Berücksichtigung des gesamten Systemumfeldes gewählt werden. Steht sehr viel Speicher zur Verfügung, kann zum Beispiel die Speicherzeit der Rohdaten deutlich erhöht werden.

- **Einbinden des Tracking-Codes:**

Durch den ersten Schritt erhält der Administrator eine Projekt-ID und kann diese in den Tracking-Code eintragen. So kann das System die Seitenabrufe dem richtigen Projekt zuordnen. Üblicherweise wird der JavaScript-Tracking-Code genutzt, da dieser die meisten Parameter bei einem Seitenabruf erfassen und leicht in eine bestehende Website eingebunden werden kann.

- **Konfiguration der Anzeige:**

Zum Abschluss der Konfiguration muss der Administrator in der Konfigurationsdatei des Präsentations-Moduls die gewünschten Diagramme anlegen und diese dem Projekt zuordnen. Es empfiehlt sich dabei, jeweils getrennte Diagramme für die drei Speicherzeiten anzulegen, damit die Daten in ihrer jeweiligen zeitlichen Auflösung gut dargestellt werden können. Die Konfiguration muss dabei Rücksicht auf die zuvor im Server-Modul eingetragenen Speicherzeiträume und Intervalle nehmen.

3 Evaluierung des bestehenden Systems

3.1 Darstellung als Verlaufslinie

In der ersten Zeit des Praxisbetriebes des Systems zeigten sich immer wieder einige Probleme und Einflussfaktoren, die bei der Konzeption oder der ersten Programmierung nicht bedacht wurden. So sollte bei der ersten Konfiguration des Präsentations-Moduls für alle Anzeigen ein Liniendiagramm genutzt werden. Erhofft wurden dabei Diagramme, wie sie in den Demos der zur Erstellung der Diagramme genutzten JavaScript-Bibliothek *Flot* zu finden sind:

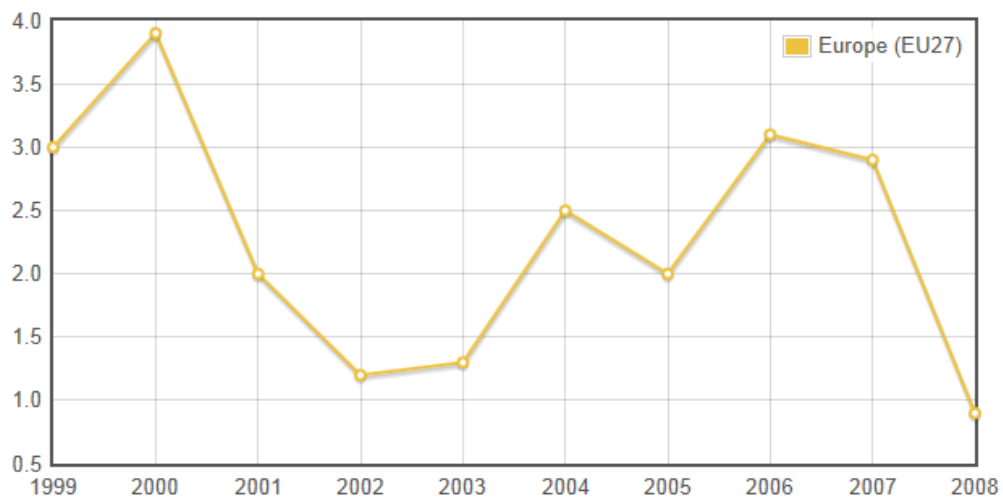


Abbildung 4 - Demo-Liniendiagramm von Flot (1)

In der Realität ergeben sich jedoch deutlich davon abweichende Diagramme. Vor allem bei der Visualisierung der Rohdaten, also der Seitenaufrufe pro Sekunde. Das Problem verstärkt sich besonders bei nicht extrem stark besuchten Seiten, bei denen es auch einzelne Sekunden gibt, in denen kein Seitenaufruf stattgefunden hat. Ebenso problematisch ist es, wenn die Differenz zwischen den einzelnen Sekundenwerten relativ groß in Bezug auf die Maximalwerte der Datenreihe ist. Statt einer einzigen Verlaufslinie ist dann in der Regel nur noch eine große Menge an Strichen zwischen den einzelnen Punkten zu sehen.

Die nachfolgende Grafik verdeutlicht dies. Die Werte variieren zwischen 0 und 3. Es entsteht jedoch der Eindruck, dass in den Bereichen mit einem Peak von 3 besonders viel Aktivität auf der Seite war. Schaut man jedoch genau hin, erkennt man, dass jeweils in der Sekunde davor und danach kein Seitenabruf stattfand. Somit ergibt sich für diese drei Sekunden nur ein Durchschnittswert von 1, was jedoch auf den ersten Blick überhaupt nicht erkennbar ist.

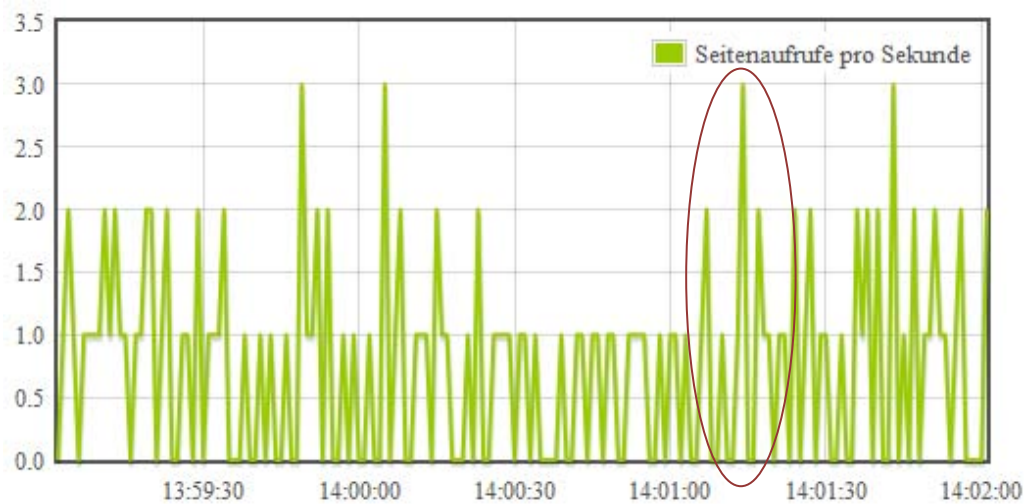


Abbildung 5 - Liniendiagramm zur Visualisierung der Seitenaufrufe pro Sekunde

Ein weiteres Problem bei dieser Anzeigeform: Die Länge der gesamten Linie im Diagramm steigt, je mehr Lücken es bei den Seitenaufrufen gibt. Das heißt, das Diagramm wird unter Umständen immer farbiger, je weniger Aktivität gerade auf der Website stattfindet. Dies könnte bei Betrachtern des Diagramms, die sich nicht über diesen Umstand bewusst sind, zu falschen Schlüssen führen, da intuitiv wohl eine höhere Farbigkeit auch einer höheren Aktivität zugeordnet wird.

Das Problem schwächt sich deutlich ab, sobald größere Zeitintervalle visualisiert werden. Die bei der Aufsummierung der Rohdaten entstehenden Werte unterscheiden sich in der Regel nicht derart sprunghaft von ihren Vorgängerwerten.

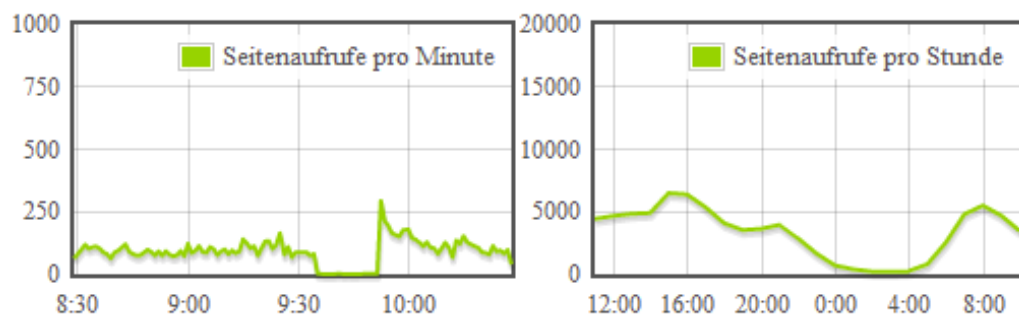


Abbildung 6 - Liniendiagramm zur Visualisierung der Seitenaufrufe pro Minute und Stunde

Auf dem obigen Diagramm ist selbst der Ausfall der Website durch eine technische Störung gut erkennbar. Auch der Verlauf der Seitenaufrufe, summiert in Stunden über einen ganzen Tag hinweg, ist problemlos erkennbar. Das heißt, das Liniendiagramm ist optimal für die Visualisierung größerer zeitlicher Intervalle geeignet, für die sekundengenaue Darstellung muss jedoch ein anderer Diagrammtyp gefunden werden.

3.2 Dynamische Achsenskalierung

Ein weiteres Problem betrifft die Achsenskalierung der Diagramme. Während die Skalierung der x-Achse durch den dargestellten Zeitraum fest vorgegeben ist, muss sich die y-Achse nach der Anzahl der jeweiligen Seitenaufrufe richten.

Dazu bietet *Flot* von Haus aus eine automatische Achsenskalierung. Der Maximalwert der y-Achse entspricht dabei dem im Daten-Set höchsten Wert + 1. Diese automatische Skalierung sorgt zwar dafür, dass die Diagrammfläche immer optimal ausgefüllt ist, bringt jedoch auch das Problem mit sich, dass die Diagramme schwer zu unterscheiden sind. Erhöht sich zum Beispiel die Anzahl der Seitenaufrufe bei gleichbleibender zeitlicher Verteilung genau um das Doppelte, ist diese Erhöhung im Diagramm nicht direkt erkennbar, da sich nur die Beschriftung der y-Achse ändert.

Ein weiteres Problem stellt in diesem Zusammenhang die automatisierte Aktualisierung der Diagramme dar. Verschwindet ein Peak aus dem dargestellten Bereich des Diagramms, passt sich daraufhin wieder die Achsenskalierung an. Das heißt der neue Maximalwert der y-Achse entspricht dem nun höchsten Wert + 1. Dadurch ändert sich die Optik des Diagramms im Moment des Verschwindens jedoch erheblich. Unter Umständen entsteht dabei sogar der völlig falsche Eindruck, dass sich die Aktivität auf der Seite erhöht hat. Der nachfolgende Vergleich verdeutlicht das Problem:

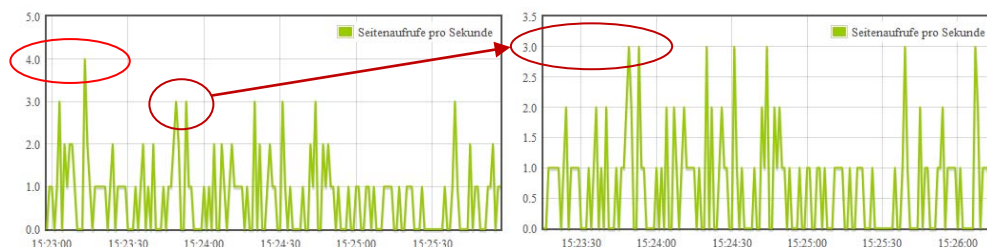


Abbildung 7 - Liniendiagramm im zeitlichen Verlauf

Es muss also eine Lösung gefunden werden, sodass auch aus der Form des Diagramms direkt auf den ersten Blick ein Rückschluss zur Aktivität auf der Website gezogen werden kann, ohne genau die Achsenbeschriftung zu beachten.

3.3 Zeitlicher Vergleich

Oftmals ist ein zeitlicher Vergleich der aktuellen Werte mit den Werten eines früheren Zeitpunkts gewünscht. Dies bietet sich besonders bei einer Auswertung nach Stunden an, wenn man die aktuelle Stunde mit der des Vortages vergleichen möchte:

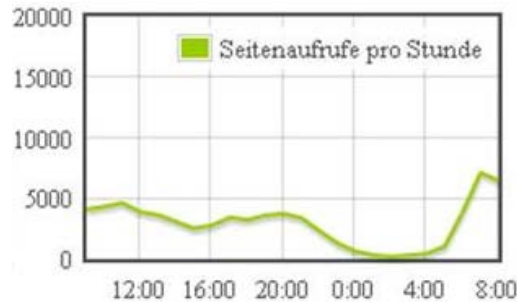


Abbildung 8 - Seitenaufrufe der letzten 24 Stunden.

Obwohl in den Projekteinstellungen eine Langzeit-Speicherzeit von 24 Stunden konfiguriert ist, fällt der Vergleich schwer, da der Verlauf der Werte vor der zu vergleichenden Stunde fehlt und auch die Beschriftung der x-Achse abgeschnitten wird.

3.4 Anzeige des aktuellen Zeitblocks

Ebenso verwirrend ist die Darstellung eines gerade begonnenen Zeitintervalls. Standardmäßig exportiert das Server-Modul bei der Abfrage der Kurzzeit- und Langzeit-Statistik auch immer das aktuelle Zeitintervall mit, das zum Zeitpunkt des Exports noch nicht vollständig abgelaufen ist und so noch mit Daten gefüllt wird. Dieses aktuelle Zeitintervall ist zugleich der letzte Wert auf der x-Achse des jeweiligen Diagramms.

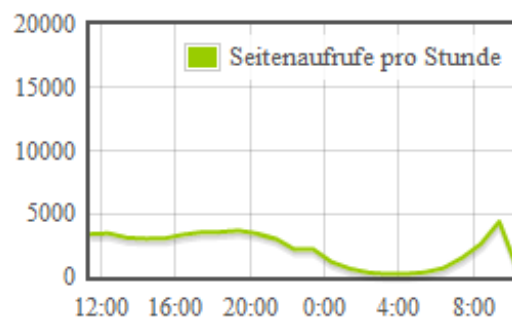


Abbildung 9 - Liniendiagramm im zeitlichen Verlauf

Es entsteht somit jedoch kurz nach Beginn eines neuen Zeitintervalls der Eindruck, dass der Besucheransturm abrupt abgerissen ist. Ändert sich an der Menge der Seitenabrufe im Vergleich zum vorangegangenen Zeitintervall nichts, hält dieser Eindruck bis zum Ende des Intervalls an, da erst dann ein gleiches Niveau erreicht wird.

3.5 Sicherheitswarnung des JavaScript-Tracking-Codes

Breibt man das Echtzeit-Webanalyse-Tool zusammen mit einem Referenz-Tool wie *Google Analytics* oder *Piwik* auf einer öffentlichen Website und sorgt dafür, dass beide Trackingcodes an allen notwendigen Stellen in den Code der Seite integriert sind, und vergleicht man über einen längeren Zeitraum hinweg die gemessenen Werte, ergeben sich dennoch deutliche Differenzen. Die Summe der gemessenen Seitenaufrufe mit dem Echtzeit-Webanalyse-Tool ist dabei immer auffällig geringer, wie die mit dem Referenztool gemessene Zahl.

Bei genauerer Untersuchung des Tracking-Codes in den am meisten verbreiteten Browsern fällt folgendes auf: Wird eine Website mit Hilfe des JavaScript-Tracking-Codes analysiert und wird diese Seite über den *Microsoft Internet Explorer* aufgerufen, kommt es zu einer Sicherheitswarnung, beziehungsweise zu einer Fehlermeldung, dass der Zugriff auf das serverseitige Tracking-Script verweigert wurde.

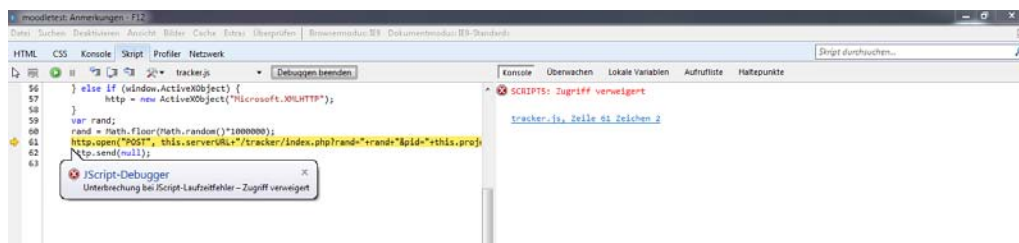


Abbildung 10 - Ansicht der Sicherheitsmeldung im IE-Script-Debugger

Dies verwundert zunächst, da der Aufruf in anderen Browsern ohne Probleme funktioniert.



Abbildung 11 - Aufruf des Tracking-Scripts

Neben der Warnung führt dieses Verhalten auch dazu, dass der komplette Seitenaufruf nicht mehr erfasst werden kann. Da der Marktanteil dieser Browserreihe bei rund 40 bis

50% liegt (2), geht somit ein erheblicher Teil an Seitenaufrufen verloren. Es muss also eine Lösung gefunden werden, die es ermöglicht, Seitenaufrufe über den JavaScript-Tracking-Code auch im *Microsoft Internet-Explorer* zu erfassen.

Dieses Problem ist während der Entwicklung nicht sichtbar geworden, da sowohl das Web-Analyse-System als auch die Testseite auf dem gleichen Server abgelegt waren und der Browser für diese Konstellation andere Sicherheitsrichtlinien nutzt.

4 Optimierung des bestehenden Systems

4.1 Säulendiagramm und statische Achsenskalierung

Um die sekundengenaue Darstellung der Seitenaufrufe übersichtlicher zu gestalten, kann auch auf ein Balkendiagramm zurückgegriffen werden. "Balkendiagramme stellen dabei Werte als messbare Längen dar. Die Balkenbreite bleibt dabei jeweils konstant. Zwei oder mehrere Balken nebeneinandergesetzt zeigen die absoluten Werte und ihre relative Differenz zu den anderen im Diagramm dargestellten Werte... In der Regel beschreibt jeder Balken denselben Gegenstand zu einem anderen Zeitpunkt" (3)

Da jedoch nach wie vor ein zeitlicher Verlauf von Werten präsentiert werden soll, eignet sich ein **Säulendiagramm** noch besser. Hier stehen die Säulen senkrecht zur x-Achse, auf der üblicherweise die Zeit abgetragen wird. (4)

Die Höhe einer Säule repräsentiert dabei die Menge an Seitenaufrufen in der jeweiligen Sekunde. Zeitgleich wird eine statische Achsen-Skalierung für die y-Achse eingeführt. Diese Anpassungen sorgen dafür, dass nun die mit Farbe gefüllte Fläche und die Anzahl an Seitenabrufe genau proportional zueinander sind.

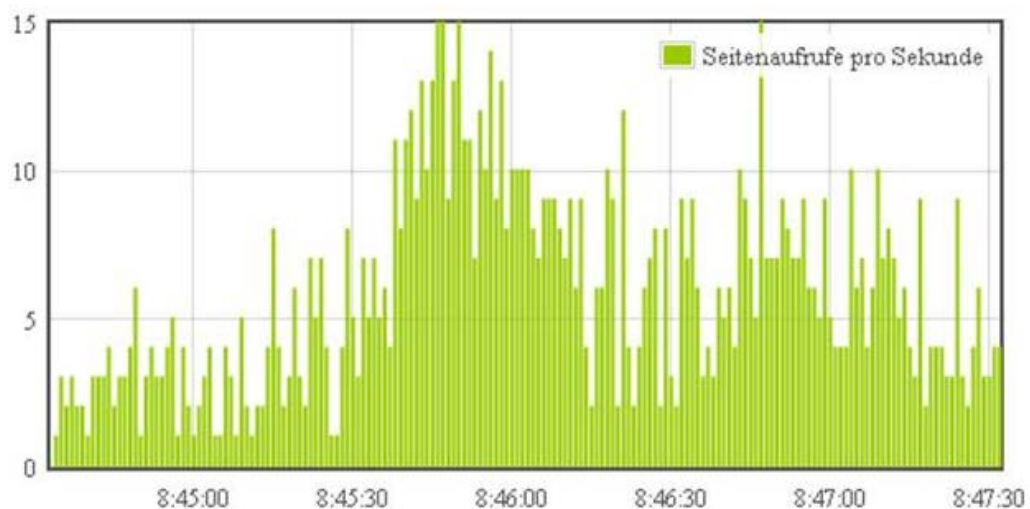


Abbildung 12 - Säulendiagramm im Präsentations-Modul

Wie das abgebildete Beispiel zeigt, sind Lastspitzen nun deutlich besser zu erkennen, obwohl bei einzelnen Sekunden nach wie vor große Differenzen zu den direkten Nachbarwerten vorhanden sind. Durch die **statische Skalierung der y-Achse** kann es nun jedoch vorkommen, dass einzelne Spitzenwerte abgeschnitten werden und so der absolute Wert für diesen Zeitabschnitt nicht ablesbar ist.

Als Lösung für dieses Problem wurden die folgenden Ansätze getestet:

- **Optimierte Ermittlung des Maximalwertes**

Die Skalierung der Achse wird nicht mehr durch das jQuery-Plugin *Flot* ermittelt. Stattdessen wird aus denen im Server-Modul vorhandenen Daten der Maximalauschlag ermittelt und dieser beim Zeichnen des Diagramms als fester Wert vorgegeben. Somit könnten auch Maximalwerte erkannt werden, die nicht mehr innerhalb des zeitlich dargestellten Intervalls liegen. Vorteil dieser Lösung ist, dass das häufige Umspringen der Achsenskalierung minimiert wird, wenn zum Beispiel der Höchstwert der letzten 24 Stunden genutzt wird.

Allerdings löst dieses Vorgehen das ursprüngliche Problem nicht: Nach wie vor kann eine Einschätzung der Werte nur durch genaue Betrachtung der Achsenbeschriftung erfolgen.

- **Logarithmische Skalierung**

Logarithmische Skalierungen bieten sich an, wenn sich die Werte stark unterscheiden. (3) Somit können extreme Spitzenwerte vollständig dargestellt werden, aber auch kleinere Werte noch gut sichtbar visualisiert werden. Um die y-Achse logarithmisch zu skalieren, bietet *Flot* in der Konfiguration für jede Achse die Möglichkeit, eine Transform-Funktion zu hinterlegen, die für alle in der Datenreihe vorkommenden Werte aufgerufen wird.

```
"yaxis": {
  "transform": function (v) {
    if (v == 0) return 0;
    if (v == 1) return 1;
    return Math.log(v);
  },
  "max": "1000.0",
  "ticks": [0, 10.00, 100.00, 1000.0]
}
```

Zusätzlich muss noch festgelegt werden, welche Intervalle als Achsenbeschriftung genutzt werden sollen. Es entsteht somit das folgende Diagramm:

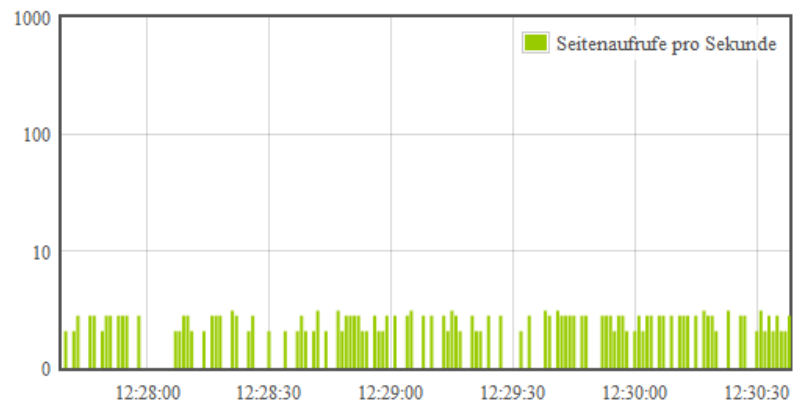


Abbildung 13 - Säulendiagramm mit logarithmischer y-Achse

Wie die Abbildung zeigt, erfolgt jedoch auch die Berechnung der Höhen der einzelnen Werte logarithmisch. So zeigt das abgebildete Diagramm nur Werte zwischen 1 und 3. Es ist damit zwar problemlos möglich, auch große Spitzenwerte noch innerhalb der Diagrammfläche darzustellen, die Menge der farbig gefüllten Fläche ist nun jedoch wieder nicht proportional zur Menge der Seitenaufrufe.

Optimal wäre hier also eine Achsenskalierung, die im Hauptbereich linear ist und nur in einem kleinen Bereich logarithmisch verläuft, damit ein Unterschied in den Spitzenwerten erkennbar wird.

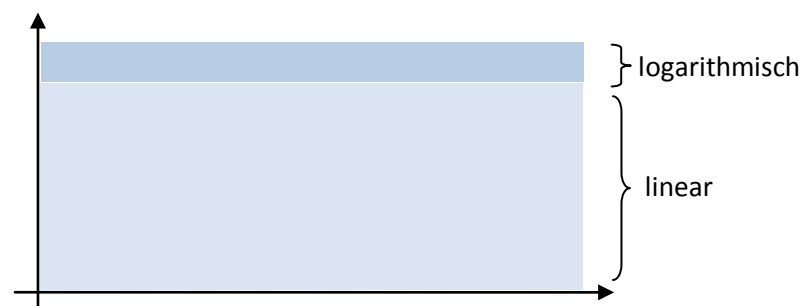


Abbildung 14 - Gleichzeitig linear und logarithmisch skalierte Achse

Obwohl *Flot*, wie erwähnt, Callback-Funktionen zur individuellen Skalierung der Achsen zulässt, gibt es hierbei wohl Einschränkungen. So ist zwar die in der Skizze gezeigte Aufteilung möglich, jedoch "verrutscht" dabei die komplette Achsenbeschriftung, sodass die Beschriftungen der Werte nicht mehr mit den im Diagramm dargestellten Werten übereinstimmen.

Zusammenfassend lässt sich also sagen, dass wohl eine feste Achsenskalierung die einfachste und am intuitivsten zu verstehende Lösung ist. Da das System auf Grund der Zielgruppe vermutlich ohnehin durch einen Administrator betreut wird, ist bei speziellen Ereignissen ein Anpassen der Maximalwerte schnell möglich.

4.2 Vergleich aktueller Werte mit historischen Daten

Um den Eindruck der abfallenden Liniendiagramme zu vermeiden, wird die Exportschnittstelle, bzw. die Exportfunktionen des Servermoduls um einen Parameter ergänzt. Dieser kann einfach in die Konfiguration des Präsentations-Moduls eingefügt werden:

```
<data>
  <method> getMediumPageImpressions</method>
  <project>1</project>
  <params>
    <excludeCurrentTimeslot>true</excludeCurrentTimeslot>
  </params>
  <serie>
    <label>Seitenaufrufe pro Minute</label>
    <bars>
      <show>true</show>
    </bars>
  </serie>
</data>
```

Ist der Wert des Parameters auf *true* gesetzt, wird beim Export das letzte und somit immer das gerade aktuelle Zeitintervall nicht mit abgefragt und nicht angezeigt. Dieser Parameter ist jedoch nur bei den Methoden zur Abfrage des Kurzzeit- und Langzeit-Speichers nutzbar. Bei den Rohdaten erfolgt grundsätzlich keine Ausgabe des aktuellen Zeitintervalls.

Um einen besseren zeitlichen Vergleich der Besuchersituation zu ermöglichen, empfiehlt sich eine Anpassung der Konfiguration der Speicherzeiträume. Sollen zum Beispiel die letzten 24 Stunden visualisiert werden, sollte die Langzeit-Speicherzeit auf mind. 36 Stunden erhöht werden.

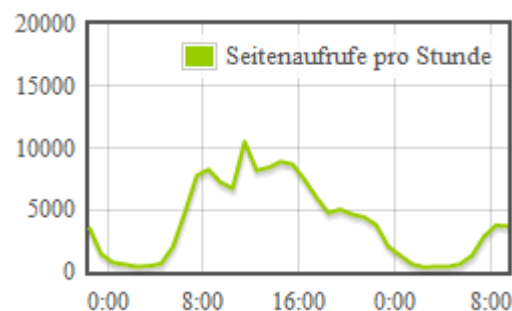


Abbildung 15 - Liniendiagramm mit Darstellung eines größeren Zeitraumes

Der Anstieg des aktuellen Tages (z.B. 8:00 Uhr) ist somit deutlich einfacher mit dem des Vortages vergleichbar. Zudem hat sich das Auffinden des Vergleichswertes vereinfacht, da für beide Tage die identischen Stundenwerte auf der x-Achse dargestellt werden.

4.3 Anzeige von Messwerten als Ziffernausgabe

Die im Präsentations-Modul vorhandenen Diagramme bieten somit nun einen schnellen Überblick über den zeitlichen Verlauf der Messwerte. Was jedoch auch noch fehlt, ist die Ausgabe der Werte als Ziffer, damit kein Diagramm abgelesen werden muss. Das Beispiel "Anzahl der Seitenabrufe am aktuellen Tag" verdeutlicht dies. Hier müsste man die einzelnen Stundenwerte aus dem Diagramm ablesen und addieren. Praktischer ist hier jedoch eine Ziffernanzeige.

Das Präsentations-Modul wird dazu so erweitert, dass in der XML-Konfigurations-Datei neben *diagram*-Elementen nun auch *counter*-Elemente angelegt und konfiguriert werden können. Die Konfiguration eines Zählers lehnt sich dabei stark an die eines Diagramms an:

```
<counter id="project1counter1">
  <layout>
    <top>10</top>
    <left>550</left>
  </layout>
  <font>
    <family>Arial</family>
    <size>20px</size>
    <weight>bold</weight>
  </font>
  <refreshTime>1500</refreshTime>
  <wrap>Heutige Seitenaufrufe: $</wrap>
  <data>
    <method>getPageImpressionCountForToday</method>
    <project>1</project>
  </data>
</counter>
```

Neu hinzugekommen ist dabei der Knoten *font* über den CSS-Eigenschaften zur Gestaltung des Textes beeinflusst werden können. Außerdem bietet der Knoten *wrap* die Möglichkeit, weiteren statischen Text der Ausgabe hinzuzufügen. Der Platzhalter \$ wird dabei durch den Wert der genutzten Export-Methode ersetzt. Es lässt sich somit zum Beispiel das folgende Bild umsetzen:

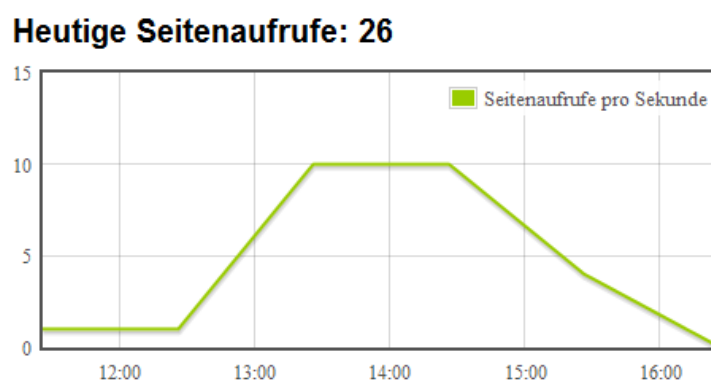


Abbildung 16 - Liniendiagramm mit zusätzlicher Zähler-Anzeige

4.4 Einsatz des JavaScript-Tracking-Codes im Internet Explorer

Das in Abschnitt 3.5 beschriebene Problem ist auf das Sicherheitsverhalten des Browsers zurückzuführen. Da die eigentliche Seite und das Tracking-Script über zwei verschiedene Domains aufzurufen sind, wird dieser Zugriff blockiert, um so eine mögliche Cross-Site-Scripting-Attacke, also das Einbinden und Ausführen von Code von einer anderen Seite, zu verhindern. (5)

Derzeit basiert der Aufruf des Tracking-Skripts auf folgender JavaScript-Programmierung:

```
ImpressionTracker.prototype.track = function() {
    var http;
    if (window.XMLHttpRequest) {
        http = new XMLHttpRequest();
    } else if (window.ActiveXObject) {
        http = new ActiveXObject("Microsoft.XMLHTTP");
    }
    var rand;
    rand = Math.floor(Math.random()*1000000);
    http.open("POST", this.serverURL+"/tracker/index.php?rand="+
        rand+"&pid="+this.projectId+"&pageURL="+escape(this.pageURL)+
        "&pageIdentifier="+escape(this.pageIdentifier)+"&pageTitle="+
        escape(this.pageTitle)+"&section="+escape(this.section)+
        "&userAgent="+escape(this.userAgent)+"&screenResolution="+
        this.screenResolution, true);
    http.send(null);
}
```

Im *Microsoft Internet Explorer* wird also versucht, über ein ActiveX-Objekt eine zusätzliche HTTP-Verbindung aufzubauen. Dieses erlaubt jedoch nur Verbindungen zur Domain der Seite, die gerade im Browser angezeigt wird. Der notwendige HTTP-Request muss also über eine andere Methode erzeugt werden.

Eine Alternative ist das Laden eines Bildes, wie es bereits im `<noscript>`-Bereichs des Tracking-Codes erfolgt. Es wird dabei ein 1x1 Pixel großes, transparentes GIF eingebunden, dessen Abruf so für den benötigten HTTP-Request sorgt. Um jedoch die über JavaScript ermittelten Werte wie Bildschirmauflösung oder Browser mit zu übermitteln, ist eine Kombination der beiden Methoden notwendig.

Hierzu wird zunächst der Tracking-Code um folgende Zeilen erweitert:

```
<script type="text/javascript">
    var httpHost = (("https:" == document.location.protocol) ? "https://" :
    "http://");
    document.write(unescape("%3Cscript
    src='"+httpHost+"serverurl/server/tracker/tracker.js'
    type='text/javascript'%3E%3C/script%3E"));
    document.write(unescape("%3Cimg src='' alt='tracker'
    id='livestatTrackingPixel'
    style='display:none;' \/%3E"));
</script>
<script type="text/javascript">
```

```

var impTrack = new ImpressionTracker();
impTrack.setServerURL(httpHost+"192.168.0.116/server");
impTrack.setProject("1");
impTrack.init();
impTrack.track();
</script>
<noscript>
  
</noscript>

```

Die Quelle des Bildes wird anschließend durch die Methode `track()` gesetzt. Als GET-Parameter werden dabei wie bisher alle Werte mit übertragen.

```

ImpressionTracker.prototype.track = function() {
  var rand;
  rand = Math.floor(Math.random()*1000000);
  var url;
  url = this.serverURL+"/tracker/index.php?rand="+rand+ "&pid="+
    this.projectId+ "&isImageTracker=true&pageURL="+
    escape(this.pageURL)+ "&pageIdentifier="+
    escape(this.pageIdentifier)+ "&pageTitle="+
    escape(this.pageTitle)+ "&section="+
    escape(this.section)+ "&userAgent="+
    escape(this.userAgent)+ "&screenResolution="+
    this.screenResolution;
  document.getElementById("livestatTrackingPixel").src = url;
}

```

Über diese Methode erfolgt der Aufruf des Tracking-Skriptes jetzt auch im *Microsoft Internet Explorer*. Der Grund dafür ist, dass die Browser das Laden von Bildern auch von externen IPs und Domains ohne Einschränkungen erlauben.

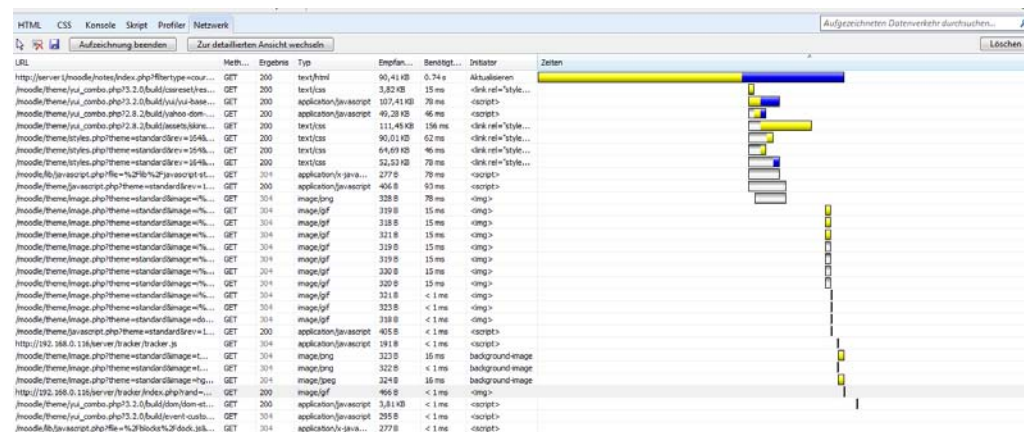


Abbildung 17 - Erfolgreich Aufruf des Tracking-Pixels im Internet-Explorer

Bei genauerer Betrachtung fällt auf, dass auch der JavaScript-Code von *Google Analytics* genau diese Methode nutzt.



Abbildung 18 - Google Analytics Pixel-Abruf

Bei einem erneuten Vergleich mit einem Referenz-Tool werden nun nahezu identische Werte erreicht. Dies liegt vor allem darin begründet, dass die Anforderungen an die Browser durch die Anpassung deutlich niedriger geworden sind. Eine erfolgreiche Messung hängt nun nicht mehr von komplexen asynchronen HTTP-Anfragen, die jeder Browser über unterschiedliche Schnittstellen durchführt, ab, sondern nur noch vom Ausführen einiger Zeichenkettenoperationen in JavaScript und dem Laden eines Bildes. Diese beiden grundlegenden Techniken sollten in jedem aktuell genutzten Browser zur Verfügung stehen, andernfalls würde auch die eigentliche Website vermutlich gar nicht korrekt angezeigt.

Die Implementierung des JavaScript-Tracking-Codes wurde unter den Browsern Microsoft Internet Explorer in Version 6,7,8 und 9, Mozilla Firefox 3.6, 4, 5 und 6, Chrome 12 und 13, sowie Safari 5.3 und Opera 11 unter Windows 7 getestet. Gemessen an den im Sommer 2011 erreichten Marktanteilen dieser Browser, sind damit bereits nahezu 99% abgedeckt. (2)

5 Integrationsanforderungen

5.1 Allgemeine Anforderungen

Bevor die Web-Analyse-Plattform in eine bestehende Web-Applikation integriert werden kann, muss geklärt werden, welche Anforderungen eine solche Applikation erfüllen muss, damit sie für die Zusammenarbeit mit der Webanalyse-Plattform geeignet ist.

- **Erweiterbarkeit:**

Grundvoraussetzung für die Integration der Webanalyse-Plattform in eine schon vorhandene Web-Applikation ist es, dass sich diese bestehende Applikation durch Software-Module individualisieren, bzw. erweitern lässt. Dabei spielt es prinzipiell keine Rolle, ob es sich um eine quelloffene oder geschlossene Applikation handelt. Voraussetzung ist lediglich, dass eine Schnittstelle zum Softwarekern der Applikation angeboten wird, die durch ein neu zu erstellendes Skript oder Programm-Modul angesprochen werden kann.

Professionelle Produkte stellen eine solche Programmierschnittstelle in der Regel in Form eines *Application Programming Interfaces (API)* (6) zur Verfügung. Die Programmierung zur Ansteuerung dieser Schnittstelle wird dabei üblicherweise getrennt von der Haupt-Applikation in *Plug-Ins* abgelegt.

Eine Integration ohne diese Voraussetzung der Erweiterbarkeit wäre bei quelloffenen Applikation auch gegeben, da hier die notwendige Programmierung direkt in der Software vorgenommen werden könnte. Ein solches Vorgehen ist jedoch nicht zu empfehlen, da es unter anderem dazu führt, dass eine Applikation ihre Updatefähigkeit verliert. (7) Wenn überhaupt, ist es nur für kleinere oder selbst entwickelte Applikationen geeignet, da hier auf individuelle Anpassungen Rücksicht genommen werden kann.

In Bezug auf die direkte Integration der Webanalyse-Plattform müssen die Anforderungen jedoch in zwei Bereiche unterteilt werden, da je nach gewünschter Integration nur ein Teil der Anforderungen erfüllt werden muss. Es kann durchaus gewünscht sein, in einer Applikation nur die Besucherinteraktionen zu erfassen und die Auswertung gar nicht in dieser, sondern in einer zweiten separaten Applikation durchzuführen. Ebenso kann es sinnvoll sein, nur eine ganz normale Website mit Hilfe der Webanalyse-Plattform zu untersuchen, dies jedoch in einer schon existierenden Applikation. Die jeweiligen Anforderungen werden daher in den beiden folgenden Abschnitten getrennt voneinander beschrieben.

5.2 Erfassung von Besucherdaten

Soll eine Web-Applikation dahingehend erweitert werden, dass die Interaktionen zwischen dieser Plattform und ihren Nutzern mit der Web-Analyse-Plattform erfasst und verarbeitet werden können, müssen die folgenden Anforderungen erfüllt sein:

- **Manuell ausgelöste Nutzerinteraktionen:**

Eine wichtige Voraussetzung für eine sinnvolle Nutzung der Web-Analyse-Plattform in einer bestehenden Applikation ist zunächst einmal, dass es in der zu analysierenden Applikation zu Interaktionen, bzw. Ereignissen kommt, die von den Nutzern manuell ausgelöst werden. Dies ist notwendig, da nur so eine realistische Abbildung der "Aktivität" der Benutzer in der Applikation erstellt werden kann. Alle anderen Möglichkeiten der Messung, z. B. die Speicherauslastung des Servers, genutzte Bandbreite, etc. stehen in keinem linearen Zusammenhang mit der Benutzeraktivität und sind daher ungeeignet.

- **Netzwerkverbindung zwischen beiden Systemen/Plattformen:**

Zwingend erforderlich ist es, dass die Web-Analyse-Plattform mittels HTTP(S) in der jeweiligen Netzwerkumgebung erreichbar ist. Je nach gewählter Integrationsmethode des Tracking-Codes (clientseitig oder serverseitig) muss also der Zugriff auf das Server-Modul der Web-Analyse-Plattform vom Client, also dem Nutzer der Applikation, aus oder vom Server der Applikation aus, möglich sein.

Da beide Systeme im Normalfall öffentlich im Internet betrieben werden, muss dieser Punkt nicht weiter beachtet werden. Besondere Beachtung muss er jedoch finden, wenn die Systeme in einem geschützten, nicht öffentlichen Bereich, zum Beispiel einem Firmen-Intranet, genutzt werden sollen.

Problematisch wird ein clientseitiges Tracking besonders dann, wenn die Kommunikation der Clients eingeschränkt ist. So wäre es zum Beispiel denkbar, dass in einem Firmennetzwerk nur auf Seiten mit bestimmten Domains oder auf Server mit bestimmten IPs zugegriffen werden darf.

Wird die Plattform dem Client zudem per HTTPS zur Verfügung gestellt, muss außerdem sichergestellt werden, dass auch das Server-Modul mittels HTTPS erreichbar ist. Wäre dies nicht der Fall, würden die Anfragen entweder gar nicht ausgeführt oder der Nutzer erhielt bei jedem Seitenaufruf störende Warnmeldungen mit der Frage, ob er auch "nicht sichereren Inhalt" laden möchte.

In allen Fällen erfolgt die Kommunikation jedoch immer nur unidirektional zum Server-Modul der Web-Analyse-Plattform hin. Dieses Modul muss nicht direkt auf einen Client oder den Server der Web-Applikation zugreifen können.

- **Einbindung des Tracking-Codes:**

Die im Abschnitt 5.1 angesprochene Schnittstelle muss es ermöglichen, den JavaScript-Trackingcode auf den Seiten der Web-Applikation einzubetten, so dass dieser mit den erzeugten Seiten an die Clients ausgeliefert, dort ausgeführt und die Benutzerinteraktion übertragen wird. Bedingt dadurch, dass es sich bei dem Trackingcode um ein in JavaScript realisiertes kleines Skript handelt, muss das Frontend der Web-Applikation als HTML-Seite in einem Web-Browser ausgeführt werden.

Sind diese Anforderungen nicht erfüllt, muss die Applikationsschnittstelle zumindest einen Weg zur Integration des serverseitigen Trackings anbieten. Dieses bietet zwar nicht einen solch großen Funktionsumfang wie ein clientseitiges Tracking, reicht jedoch aus, um ein grobes Bild der Benutzerinteraktionen zu erhalten. Wichtig hierbei ist jedoch, dass weitere Rahmenparameter eingehalten werden. Es muss zum Beispiel sichergestellt werden, dass bei einer Interaktion des Nutzers auch wirklich eine Kommunikation mit dem Server der Applikation stattfindet und er nicht auf gecachte Seiten zugreift und auch nicht zu viele Interaktionen erfasst werden, die zum Beispiel auf eine automatische Kommunikation zwischen Client und Server zurückzuführen sind (z.B. regelmäßige AJAX-Requests, etc.).

Um das serverseitige Tracking umzusetzen, muss die bestehende Web-Applikation lediglich dazu gebracht werden, an entsprechender Stelle eine HTTP(S)-Anfrage an das Server-Modul der Web-Analyse-Plattform zu stellen.

- **Zuordnung der Interaktionen in Bereiche:**

Ab einer gewissen Größe oder Komplexität der Anwendung sowie einer hohen Anzahl an gleichzeitigen Nutzern, sollten die Seitenabrufe, bzw. Nutzerinteraktionen identifizierbar sein, da andernfalls bei der Analyse keinerlei Rückschlüsse gezogen werden können, auf welcher Seite oder welchem Bereich der Web-Applikation gerade eine erhöhte Nutzeraktivität stattfindet.

Um dies zu realisieren muss die Web-Applikation, bzw. ihre Schnittstelle eine Möglichkeit bieten, den Trackingcode je nach Seite oder Bereich zu individualisieren, damit so die notwendigen Parameter an das Server-Modul der Web-Analyse-Plattform übertragen werden können.

5.3 Auswertung von Besucherdaten

Soll eine Auswertung der Web-Analyse-Daten in eine Web-Applikation integriert werden, muss die Web-Applikation über folgende Eigenschaften verfügen:

- **Erweiterbarkeit des GUI:**

Damit die Analyse-Oberfläche in die Web-Applikation integriert werden kann, muss die Möglichkeit bestehen, deren Oberfläche dahingehend zu erweitern, dass ein Zugriff auf die Analyse-Oberfläche ermöglicht wird. Das heißt, es muss zum Beispiel möglich sein, einen Menüpunkt hinzuzufügen oder einen anderen Zugriffspunkt auf die Oberfläche zu schaffen.

Des Weiteren sollte eine individuelle Oberfläche im Rahmen der Applikation erstellt werden können.

- **Netzwerkverbindung zwischen beiden Systemen/Plattformen:**

Wie bei der Integration des Tracking-Codes muss auch hier darauf geachtet werden, dass eine Netzwerkverbindung zwischen dem Server der Web-Applikation und dem Server-Modul der Web-Analyse-Plattform vorhanden ist.

- **Schutz vor unberechtigtem Zugriff:**

Da die Analyse-Daten in der Regel nicht öffentlich zugänglich sein sollten, muss auch hier die jeweilige Web-Applikation dafür sorgen, dass der Zugriff auf die integrierte Analyse-Plattform auf einen bestimmten Nutzerkreis eingeschränkt wird. Größere Systeme bieten in der Regel einen abgegrenzten Administrationsbereich, auf den ein Zugriff nur mit entsprechenden Berechtigungen möglich ist.

5.4 Integrationstiefe

Für beide Integrationsmöglichkeiten gilt jedoch, dass es sich nur um eine Verknüpfung der beiden Systeme handeln soll und nicht um den kompletten Einbau der Web-Analyse-Plattform in eine andere Web-Applikation. Beide Systeme sollen nach wie vor getrennt voneinander einsatzfähig bleiben. Um dies zu realisieren soll beim Zusammenschluss nur auf die von beiden Systemen/Applikationen bereitgestellten Schnittstellen zurückgegriffen werden.

Je nach eingesetzten Technologien wäre zwar auch eine komplette Integration technisch realisierbar, dies ist jedoch nicht zu empfehlen, da keinerlei Vorteile daraus resultieren.

6 Integrationsbeispiel Typo3

6.1 Vorstellung des CMS Typo3

Typo3 ist ein klassisches Content-Management-System. Es dient zur Pflege der Struktur und des Inhaltes einer Website. Grundlage ist dabei die Trennung von Design und Inhalten. Die Inhalte werden in einer Datenstruktur hinterlegt und dann mit Hilfe von hinterlegten Templates ausgegeben. Eine Typo3-Installation ist dabei in zwei wesentliche Bereiche unterteilt. Diese sind das Frontend, also die eigentliche öffentlich zugängliche Website sowie das Backend, über das die Inhalte gepflegt und die Website administriert wird. Beide Bereiche werden dabei immer zusammen auf dem gleichen System betrieben.

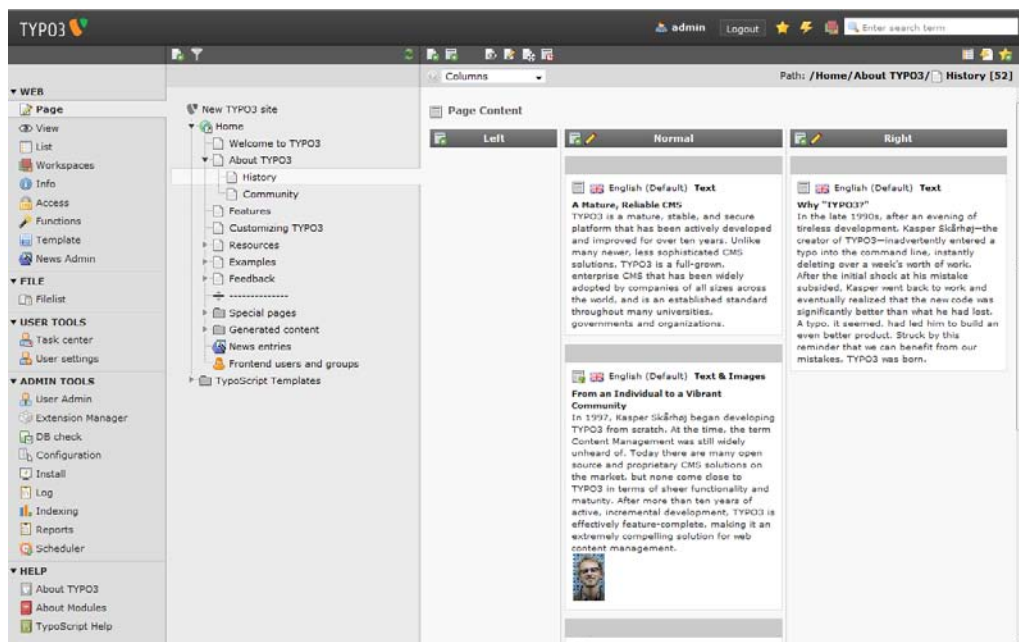


Abbildung 19 - Ansicht einer Seite im Seiten-Modul im Typo3-Backend

Die Entwicklung von Typo3 wurde von Kasper Skårhøj gestartet. Die erste öffentliche Version erschien im Jahr 2001. (8) Seitdem folgten zahlreiche Neuerungen und Verbesserungen. Aktuell liegt Typo3 in der Version 4.5 vor. Es beinhaltet Funktionen, die gerade für komplexe Websites und in großen Unternehmen benötigt werden. Dies sind zum Beispiel ein mächtiges Berechtigungssystem oder die Möglichkeit der Versionierung von Inhalten. Dies ist wohl mit einer der Hauptgründe, warum Typo3 bereits weltweit in mehreren Tausend Installationen und von großen Unternehmen und Organisationen, wie Cisco oder der UNESCO, genutzt wird. (9)

Das CMS ist in der Skriptsprache PHP geschrieben. Zur Ablage der Inhalte wird ein relationales Datenbank-System benötigt, typischerweise MySQL, aber auch andere

Produkte sind möglich. Das System stellt auf Grund der hohen Komplexität große Anforderungen an die auf dem Webserver vorhandene Software.

6.2 Erfüllung der Integrationsanforderungen

Da es sich bei Typo3 um ein quelloffenes und weit verbreitetes CMS handelt, existieren zahlreiche Bücher, Tutorials, Foren und Blogs, die sich mit den verschiedenen Problemstellungen innerhalb des Systems beschäftigen. Mit diesen Hilfsmitteln, sowie einer Testinstallation soll zunächst geklärt werden, ob alle in Abschnitt 5 aufgestellten Anforderungen erfüllt werden können, damit die Echtzeit-Web-Analyse-Plattform in Typo3 genutzt werden kann.

Alle Angaben beziehen sich dabei auf die zum Zeitpunkt dieser Arbeit aktuelle Version 4.5.3.

- **Erweiterbarkeit:**

Ein Grundprinzip von Typo3 ist die Organisation der Programmierung in einzelnen Paketen. Diese werden als Extensions bezeichnet. Der eigentlich Kern von Typo3 ist nicht besonders umfangreich, da bereits alle wesentlichen Funktionalitäten einer Grundinstallation in so genannten System-Extensions umgesetzt sind.

Neben diesen System-Extensions ist es für Entwickler auch möglich, eigene Extensions zu erstellen, die das System an drei wesentlichen Stellen erweitern können. Diese sind zum einen Frontend-Module, die auf der öffentlichen Website eine neue Funktionalität mit sich bringen (z.B. Gästebücher, Kontaktformulare,...). Im Gegensatz dazu stehen Backend-Module, die Benutzern im Backend weitere Funktionen zur Verfügung stellen können. Ein Beispiel hierfür ist eine Administrationsoberfläche für Nachrichten-Einträge. Die dritte Möglichkeit wäre die Bereitstellung eines Services, um zum Beispiel eine weitere Login-Methode in das Backend bereitzustellen.

Auf der offiziellen Typo3-Website existiert zudem eine Repository mit bestehenden Extensions. Dieses Verzeichnis umfasst derzeit rund 5.000 Erweiterungen. (10)

✚ Anforderung erfüllt!

- **Manuell ausgelöste Nutzerinteraktionen:**

Da es sich beim Typo3-Frontend in der Regel um eine klassische Website handelt, kann hier der einzelne Seitenabruf als manuell ausgelöste Nutzerinteraktion betrachtet werden. Auch wenn auf einer einzelnen Seite ein Frontend-Modul diverse Funktionen anbietet, führen diese in der Regel zu einem Neuladen der gesamten Seite, sodass auch diese Aktivität erfasst wird.

+ Anforderung erfüllt!

- **Netzwerkverbindung zwischen beiden Systemen/Plattformen:**

Im Regelfall wird das CMS dafür genutzt, eine öffentliche Website zu verwalten. Es kann also davon ausgegangen werden, dass sowohl das Frontend von Typo3 als auch die Web-Analyse-Plattform öffentlich im Internet zugänglich sind. Dieser Punkt muss daher nicht weiter beachtet werden.

+ Anforderung erfüllt!

- **Einbindung des Tracking-Codes:**

Üblicherweise handelt es sich bei den öffentlichen Frontend-Seiten von Typo3 um klassische (X)HTML-Seiten. Es kann also auf den JavaScript-Tracking-Code zurückgegriffen werden.

Die Einbindung eines solchen Codes in die auszugebenden Seiten ist dabei eine Anforderung, die bereits mit den Kernfunktionen von Typo3 erfüllt werden kann. Mit Hilfe der integrierten Skriptsprache *Typoscript* ist es problemlos möglich, bestimmte Code-Fragmente an beliebigen Stellen der Seiten einzubinden. Als Vorlage können hier zahlreiche Beispiel-Implementationen des *Google Analytics*-Codes dienen.

+ Anforderung erfüllt!

- **Zuordnung der Interaktionen in Bereiche:**

Da jede in Typo3 angelegte Seite über eine eindeutige ID verfügt, kann zunächst diese ID dazu genutzt werden, die Aktionen einer bestimmten Seite zuzuordnen.

Typo3 selbst ist ein seitenbaum-basiertes System. Das heißt, jede Seite befindet sich in einer hierarchischen Struktur in einem Seitenbaum. Somit ist es mit Hilfe von Conditions innerhalb von Typoscript möglich, zu überprüfen, ob eine bestimmte Seite im Pfad zur aktuellen Seite liegt, um damit mehrere Seiten in einen Bereich zusammenzufassen. (11)

+ Anforderung erfüllt!

- **Erweiterbarkeit der GUI:**

Innerhalb einer Extension ist es problemlos möglich, das GUI des Backends zu erweitern. So reicht die Registrierung eines Backend-Modul aus, um einen neuen Menüpunkt in das Hauptmenü auf der linken Seite einzufügen und individuelle Programmierungen zu hinterlegen, deren Ausgabe dann im Hauptfenster auf der rechten Seite angezeigt wird.

+ Anforderung erfüllt!

- **Schutz vor unberechtigtem Zugriff:**

Typo3 verfügt über ein mächtiges Berechtigungssystem, mit dem die Berechtigungen von Benutzern im Backend extrem detailliert festgelegt werden können. Für den konkreten Fall kann auf das Freigeben/Sperren von Backend-Modulen zurückgegriffen werden. Damit kann festgelegt werden, welchen Nutzern oder Nutzergruppen der Zugriff auf das Backend-Modul zur Anzeige der Echtzeit-Web-Analyse-Daten gewährt wird. Es muss also nur ein Eintrag in den ACL der berechtigten User und Gruppen für das neue Backend-Modul vorgenommen werden.

+ Anforderung erfüllt!

6.3 Integration des Trackingcodes

6.3.1 Grundsetup des Trackingcodes

Typo3 wird in verschiedenen Paketen zum Download angeboten. Neben dem Software-Kern wird auch ein Paket angeboten, in dem bereits eine komplett konfigurierte kleine Beispielwebsite enthalten ist, die die wichtigsten Funktionen und Konzepte des CMS präsentiert. Dieses *Introduktion Package* soll hier genutzt werden, um die Möglichkeiten der Integration des Tracking-Codes zu zeigen.

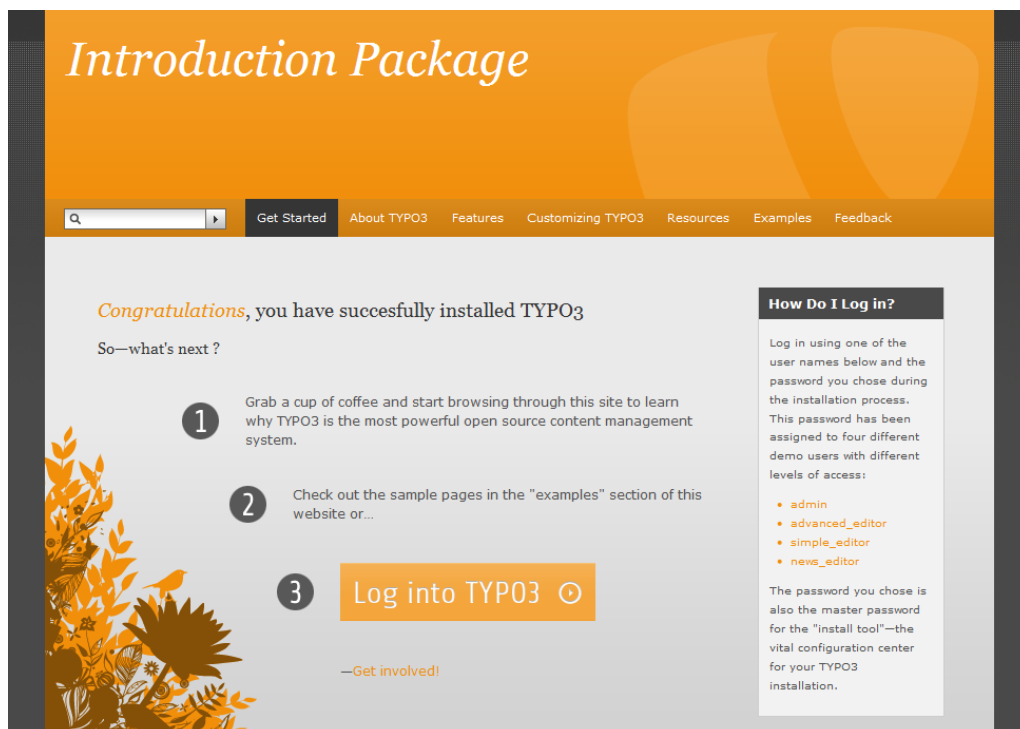


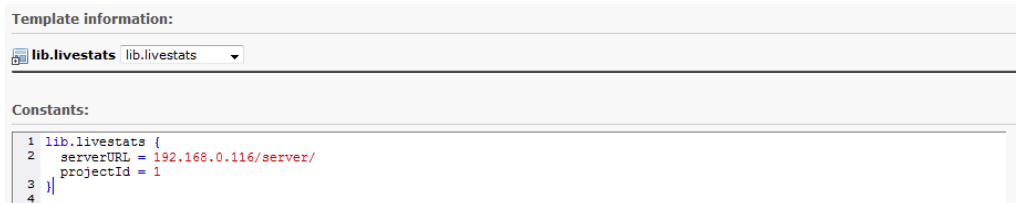
Abbildung 20 - Screenshot der im Introduction-Package enthaltenen Website

Um den Tracking-Code einzubinden, wird zunächst ein neues Typoscript-Template im Storage Folder *page_blocks_configuration* angelegt, da es sich bei dem einzubettenden Code letztlich auch nur um einen HTML-Block handelt. Die Bezeichnung des Templates ist frei wählbar, muss jedoch mit *lib.* beginnen.



Abbildung 21 - Neu angelegtes Typoscript-Template *lib.livestats*

Da der Tracking-Code aus zahlreichen variablen Werten besteht (URLs des Servermoduls, Projekt-ID, usw.) macht es Sinn, hierfür Konstanten innerhalb des Typoscript-Templates anzulegen.



```

Template information:
lib.livestats lib.livestats

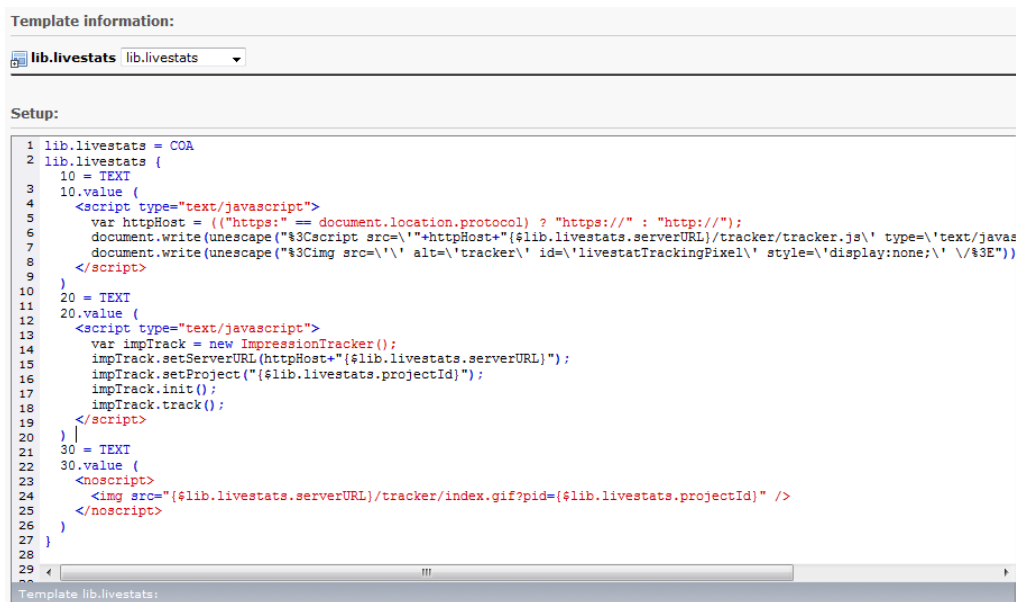
Constants:
1 lib.livestats {
2   serverURL = 192.168.0.116/server/
3 }
4

```

Abbildung 22 - Konstanten des Typoscript-Templates *lib.livestats*

Im nächsten Schritt kann nun im Setup-Bereich dieses Typoscript-Templates der eigentliche Tracking-Code hinterlegt werden. Da dieser in einzelne Bereiche unterteilt ist, kann dabei sehr gut auf den Einsatz des Typoscript-Content-Objects *COA* zurückgegriffen werden. *COA* steht dabei für *Content-Object-Array* und ist also nichts anderes, als eine Zusammenfassung weiterer Content-Objects. (12) Da im konkreten Fall nur Textinformationen zusammengesetzt werden sollen, wird lediglich *TEXT* als weiterer Content-Object-Typ benötigt.

Das Grundsetup sieht daher wie folgt aus:



```

Template information:
lib.livestats lib.livestats

Setup:
1 lib.livestats = COA
2 lib.livestats {
3   10 = TEXT
4   10.value {
5     <script type="text/javascript">
6       var httpHost = (("https:" == document.location.protocol) ? "https://" : "http://");
7       document.write(unescape("%3Cscript src='"+httpHost+"{lib.livestats.serverURL}/tracker/tracker.js\" type='text/javas
8       document.write(unescape("%3Cimg src='\" alt='tracker\" id='livestatTrackingPixel\" style='display:none;\" \\/%3E\"))
9     </script>
10  }
11  20 = TEXT
12  20.value {
13    <script type="text/javascript">
14      var impTrack = new ImpressionTracker();
15      impTrack.setServerURL(httpHost+"{lib.livestats.serverURL}");
16      impTrack.setProject("{lib.livestats.projectId}");
17      impTrack.init();
18      impTrack.track();
19    </script>
20  }
21  30 = TEXT
22  30.value {
23    <noscript>
24      
25    </noscript>
26  }
27 }
28
29
Template lib.livestats:

```

Abbildung 23 - Setup des Typoscript-Templates *lib.livestats*

Die zuvor festgelegten Konstanten werden dabei über Platzhalter *{\$path.to.constant}* eingefügt und bei der Ausgabe dann automatisch ersetzt.

Damit nun eine Ausgabe von *lib.livestats* in der Seite erfolgt, muss das Typoscript zunächst in die bestehende Template-Struktur eingebunden werden. Im *Introduction-Package* bietet sich das im Template *root_blocks* an.

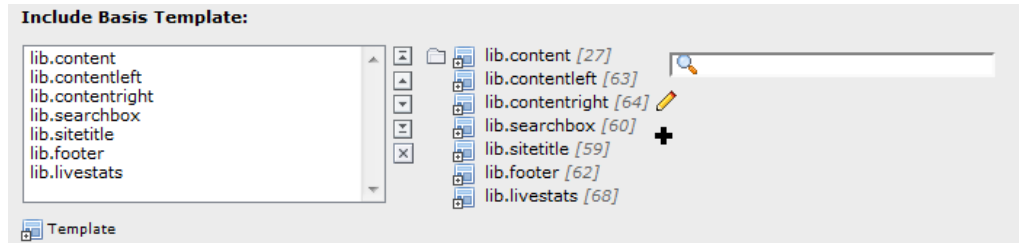


Abbildung 24 - Einbinden von *lib.livestats* in *root_blocks*

Das Ergebnis ist anschließend direkt im *Template-Analyser* sichtbar:

lib.content					11
lib.contentleft					11
lib.contentright					11
lib.searchbox					11
lib.sitetitle					11
lib.footer					11
lib.livestats					11
root_blocks					11
page					14
page.config					14
page.meta					14
page.headerData					14
page.includeCSS					14
root_page					14
ROOT	✓	✓	✓		8
Introduction Package	✓	✓	✓		1 0

Abbildung 25 - Template-Struktur des *Introduction-Packages* im *Template-Analyser*

Als letzter Schritt muss nun im Haupt-Template (hier *page*) ein Verweis auf den neu angelegten Typoscript-Object-Pfad gesetzt werden, damit die Ausgabe an der entsprechenden Stelle stattfindet.

```

Template information:
-----
page page

Setup:
-----
1 # Make the PAGE object
2 page = PAGE
3 page {
4   # Regular pages always have typeNum = 0
5   typeNum = 0
6
7   # Add the icon that will appear in front of the url in the browser
8   # This icon will also be used for the bookmark menu in browsers
9   shortcutIcon = {$filepaths.images}favicon.ico
10
11 # Add a TEMPLATE object to the page
12 # We use the template autoparser extension to easily replace parts of the HTML template by dynamic TypoScript objects
13 #
14 #
15 # Use the HTML template from the automake template plugin
16 template =< plugin.tx_automaketemplate_pi1
17
18 # Use the <body> subpart
19 workOnSubpart = DOCUMENT_BODY
20
21 # Link content and page blocks to id's that have been enabled in the
22 # automaketemplate template in the extension_configuration sysfolder
23 subparts {
24   #####
25 }
26
27
28 20 < lib.livestats
29 }
30
31
Template page:

```

Abbildung 26 - Template-Struktur des Introduction-Packages im Template-Analyser

Wie die bestehende Programmierung zeigt, wird an der Stelle 10 des Page-Objects die komplette bestehende Seite ausgegeben. Um also den Tracking-Code unten an die Seite anzufügen, genügt es, einen Verweis auf *lib.livestats* an einer Stelle größer 10 (hier zum Beispiel 20) zu setzen.

Der schließlich generierte Quellcode der Frontendseite hat die folgende Form:

```

<a rel="license" href="http://creativecommons.org/licenses/by-nc-nd/2.5/ch/deed.en_GB">
  
</a>
</div></div></div>
</div>
<script type="text/javascript">
var httpHost = (("https:" == document.location.protocol) ? "https://" : "http://");
document.write(unescape("%3Cscript src='"+httpHost+"192.168.0.116/serverz/tracker/tracker.js' type='text/javascript'%3E%3C/script%3E"));
document.write(unescape("%3Cimg src='\" alt='\" tracker\" id='livestatTrackingPixel' style='display:none;' /%3E"));
</script> <script type="text/javascript">
var impTrack = new ImpressionTracker();
impTrack.setServerURL(httpHost+"192.168.0.116/serverz");
impTrack.setProject("1");
impTrack.init();
impTrack.track();
</script> <noscript>

</noscript>

</body>
</html>

```

Abbildung 27 - Finale HTML-Ausgabe des Tracking-Codes

Die hier gezeigte Konfiguration reicht aus, um alle Seitenaufrufe zu erfassen. Wie weitere Informationen in den Tracking-Code eingebaut werden können, wird im nachfolgenden Abschnitt gezeigt.

Vorteilhaft an dieser Lösung mit reinem Typoscript ist es, dass keinerlei Kollisionen mit anderen Anpassungen auftreten und dass das gezeigte Setup auch problemlos in einer Multi-Domain-Umgebung, also einer Typo3-Installation, bei der mehrere Websites

gleichzeitig verwaltet werden, funktionieren würde. Um die Seitenaufrufe getrennt zu erfassen, müssten nur die Konstanten entsprechend angepasst werden.

Die Lösung ist außerdem unabhängig davon, welches Template-Verfahren genutzt wird. Hier im Beispiel kam das klassische Templating von Typo3 zum Einsatz, aber in eine Seite, die mit Hilfe von *Templavoila* erstellt wurde, ließe sich der Code ebenfalls auf diese Art und Weise einbetten.

6.3.2 Erweiterung des Tracking-Codes

Um nun Informationen wie die Seiten-ID oder den Bereich mit in den Tracking-Code einzufügen, ist ein etwas komplizierteres Typoscript-Setup notwendig.

Um zunächst die Seiten-ID in die Funktion *setPageIdentifizier()* des Tracking-Codes einzufügen, wird Gebrauch von den weiteren Eigenschaften und Funktionen des TEXT-Content-Objects in Typo3 gemacht. Mit diesem ist es unter anderem möglich, ein beliebiges Datenbankfeld aus der Tabelle *pages*, also auch die Seiten-ID, auszugeben. Das Typoscript-Setup wird also im mittleren Teil wie folgt erweitert:

```

10 /
11 20 = TEXT
12 20.value (
13 <script type="text/javascript">
14     var impTrack = new ImpressionTracker();
15     impTrack.setServerURL(httpHost+"{lib.livestats.serverURL}");
16     impTrack.setProject("{lib.livestats.projectId}");
17 )
18 21 = TEXT
19 21 {
20     field = uid
21     wrap = impTrack.setPageIdentifizier("|");
22 }
23 29 = TEXT
24 29.value (
25     impTrack.init();
26     impTrack.track();
27 </script>
28 )

```

Abbildung 28 - Typoscript-Code zum Auslesen der Seiten-ID

Es wird so der Wert des Feldes *uid* (Seiten-ID) abgefragt. Mit Hilfe der Eigenschaft *wrap* des *TEXT*-Objects wird der notwendige JavaScript-Code anschließend um diesen Wert gefügt. Das Ergebnis ist in der HTML-Ausgabe zu sehen:

```

<script type="text/javascript">
var httpHost = (("https:" == document.location.protocol) ? "https://" : "http://");
document.write(unescape("%3Cscript src='"+httpHost+"192.168.0.116/server/tracker/tracker.js' type='text/javascript'%3E%3C/script%3E"));
document.write(unescape("%3Cimg src='' alt='tracker' id='livestatTrackingPixel' style='display:none;' /%3E"));
</script> <script type="text/javascript">
var impTrack = new ImpressionTracker();
impTrack.setServerURL(httpHost+"192.168.0.116/server");
impTrack.setProject("1");impTrack.setPageIdentifizier("51");    impTrack.init();
impTrack.track();
</script> <noscript>

</noscript>

```

Abbildung 29 - Ausgabe der Seiten-ID

Die Zahl 51 ist dabei die Seiten-ID der Seite *About Typo3* des Introduction-Packages. Die etwas unübersichtlichen Zeilenumbrüche sind der Aufteilung in mehrere *TEXT*-Objects geschuldet. Da nach diesen kein automatischer Umbruch erfolgt, fehlt dieser hier teilweise.

Um nun einen Bereich in den Tracking-Code einzufügen, kann wie folgt vorgegangen werden: Zunächst wird eine weitere Konstante *section* angelegt, damit ein Seitenaufruf auf jeden Fall immer einem Bereich zugeordnet werden kann. Ein möglicher Wert für diese Konstante ist *default* oder *unknown*. Der Wert dieser Konstante kann im Anschluss an beliebiger Stelle überschrieben werden. Um bestimmte Werte je nach Position der Seite im Seitenbaum auszuwählen, können Typoscript-Conditions genutzt werden (11):



```

Template information:
lib.livestats lib.livestats

Constants:
1 lib.livestats {
2   serverURL = 192.168.0.116/server
3   projectId = 1
4   section = default
5 }
6 [PIDinRootline = 6]
7 lib.livestats.section = welcome
8 [global]
9
10
11 [PIDinRootline = 51]
12 lib.livestats.section = about
13 [global]
14
15
16 [PIDinRootline = 57]
17 lib.livestats.section = features
18 [global]
19

```

Abbildung 30 - Conditions zur Abfrage der Position im Seitenbaum

Die Einbindung der Funktion *setSection()* des Tracking-Codes erfolgt wie üblich im Typoscript-Setup:

```

18 21 = TEXT
19 21 {
20   field = uid
21   wrap = impTrack.setPageIdentifier("|");
22 }
23
24 22 = TEXT
25 22.value = impTrack.setSection("#{lib.livestats.section}");
26
27 29 = TEXT
28 29.value (
29   impTrack.init();
30   impTrack.track();
31 </script>
-- )

```

Abbildung 31 - Typoscript-Code zum Einbinden des Seitenbaumbereiches

So werden nun die Seite *About Typo3* und alle ihre Unterseiten dem Bereich *about* zugeordnet.

```

<script type="text/javascript">
var httpHost = (("https:" == document.location.protocol) ? "https://" : "http://");
document.write(unescape("%3Cscript src='"+httpHost+"192.168.0.116/server/tracker/tracker.js' type='text/javascript'%3E%3C/script%3E"));
document.write(unescape("%3Cimg src='\" alt='\" tracker' id='livestatTrackingPixel' style='display:none;' %3E"));
</script> <script type="text/javascript">
var impTrack = new ImpressionTracker();
impTrack.setServerURL(httpHost+"192.168.0.116/server");
impTrack.setProject("1");impTrack.setPageIdentifier("51");impTrack.setSection("about"); impTrack.init();
impTrack.track();
</script> <noscript>

</noscript>

```

Abbildung 32 - Ausgabe des Seitenbereichs

Auf Grund der Komplexität und Mächtigkeit von Typo3 beziehungsweise Typoscript gibt es noch zahlreiche weitere Möglichkeiten, den Tracking-Code mit Seiten-ID und Bereichszuordnung einzubinden. Das gezeigte Vorgehen ist daher nur eine Beispiellösung.

6.3.3 Auslagerung des Tracking-Codes in eine eigene Extension

Bislang wurde die gesamte Programmierung direkt in der Typo3-Installation vorgenommen. Damit diese jedoch leichter wiederverwendet werden kann, bietet es sich an, eine eigene Extension anzulegen.

Da die Struktur einer Typo3-Extension recht aufwendig ist, soll speziell hierauf nicht weiter eingegangen werden. Um ein Extension-Grundgerüst anzulegen, empfiehlt sich die Nutzung des Extension-Kickstartes. Mit Hilfe dieses Assistenten können die einzelnen Elemente, die die Extension beinhalten soll, ausgewählt und konfiguriert werden. (13) Die einzige Aufgabe, die die zu erstellende Extension hier erfüllen muss, ist das Bereitstellen von statischem Typoscript-Code.



Abbildung 33 - Anlegen einer Extension im Kickstarter

Die Extension steht nach Abschluss des Assistenten im Ordner `/typo3conf/ext/livestats` zur Verfügung. Per Hand müssen nun noch die beiden Text-Dateien, die den statischen Typoscript-Code enthalten, bearbeitet werden. Diese befinden sich im Verzeichnis `static/livestat-tracking-code` und sind mit `constants.txt` und `setup.txt` bezeichnet.

Die Programmierung kann im Wesentlichen übernommen werden. Dabei muss lediglich der Typoscript-Pfad `lib.livestats` in `plugin.tx_livestats` abgeändert werden. Die Typoscript-Programmierung steht nach der Installation der Extension als statischer Include zur Verfügung.

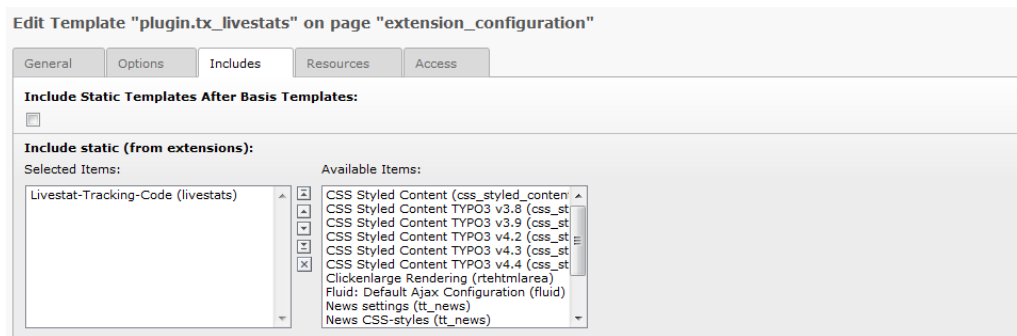


Abbildung 34 - Anlegen einer Extension im Kickstarter

In der eigentlichen Typoscript-Programmierung der Seite müssen nun noch die Konstanten angepasst werden, da diese von System zu System variabel sind und daher nicht fest in der Extension hinterlegt werden können. Das Einbinden der statischen Typoscript-Programmierung wird im Template-Analyser wie folgt sichtbar:



Abbildung 35 - Anlegen einer Extension im Kickstarter

Zu beachten ist jedoch, dass im Page-Object der Tracking-Code nach wie vor per Hand an eine Position größer 10 eingehängt werden muss. Dies könnte zwar auch in das statische Typoscript ausgelagert werden, allerdings besteht dann die Gefahr einer Kollision mit einer anderen Extension, sollte diese zufällig die gleiche Position nutzen.

Auf den ersten Blick ergeben sich aus diesem Vorgehen keine offensichtlichen Vorteile gegenüber dem zuerst geschilderten Ablauf. Es ist jedoch der eher übliche Weg, um eine externe Anwendung in Typo3 zu integrieren und bietet darüber hinaus bessere Möglichkeiten zur späteren Erweiterung und Austausch der Extension.

6.3.4 Anbindung an weitere Extensions

Mit der in den beiden vorigen Abschnitten beschriebenen Technik werden alle Aktionen der Besucher erfasst, die einen kompletten Seitenabruf hervorrufen. Bietet ein Frontend-Modul jedoch weitere Funktionen an, die jedoch keinen kompletten Seitenabruf erzeugen, weil die Daten beispielsweise mittels AJAX nachgeladen werden, ist es nicht möglich, diese zu erfassen.

Um auch diese Aktionen zu erfassen, bestehen prinzipiell zwei Lösungsansätze:

- **Aufruf der track()-Funktion durch die Extension:**

Die jeweilige Extension kann in ihrer Javascript-Programmierung dafür sorgen, dass an den entsprechenden Stellen, an denen eine Aktion verarbeitet wird, die *track()*-Funktion des Javascript-Tracking-Codes aufgerufen wird. Es würde dann ein weiterer Seitenabruf mit exakt den gleichen Parameterwerten wie beim Hauptaufruf der Seite erfasst werden.

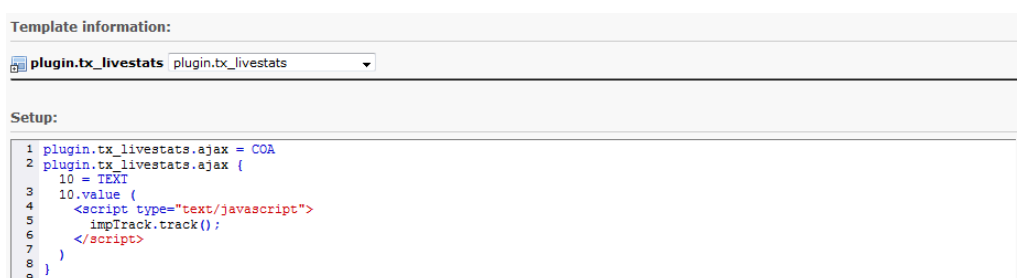
Nachteilig an dieser Lösung ist, dass sie sehr stark von der Programmierung des Javascript-Tracking-Codes abhängig ist. Ändert sich dort etwas, müssen alle Extensions, die davon abhängig sind, angepasst werden.

- **Einbinden eines Typoscript-Object-Pfads:**

Eine weitere Möglichkeit besteht darin, die bestehende Typoscript-Programmierung um ein weiteres Objekt zu erweitern, das genau den passenden Code zurückliefert, der für die Erfassung einer Nutzeraktion notwendig ist. Der Inhalt dieses Typoscript-Objects kann dann von anderen Extensions an der entsprechenden Stelle ausgegeben werden.

Der Vorteil dieser Lösung ist, dass die Programmierungen sauber getrennt bleiben und so Probleme, beziehungsweise Anpassungen bei Updates vermieden werden.

Der zuletzt beschriebene Lösungsansatz soll etwas detaillierter erläutert werden. Die notwendige Erweiterung des Typoscript-Setups hat dabei die folgende Form:



```

Template information:
plugin.tx_livestats plugin.tx_livestats

Setup:
1 plugin.tx_livestats.ajax = COA
2 plugin.tx_livestats.ajax {
3     10 = TEXT
4     10.value {
5         <script type="text/javascript">
6             impTrack.track();
7         </script>
8     }
9 }

```

Abbildung 36 - Erweiterung des Typoscript-Setups von plugin.tx_livestats I

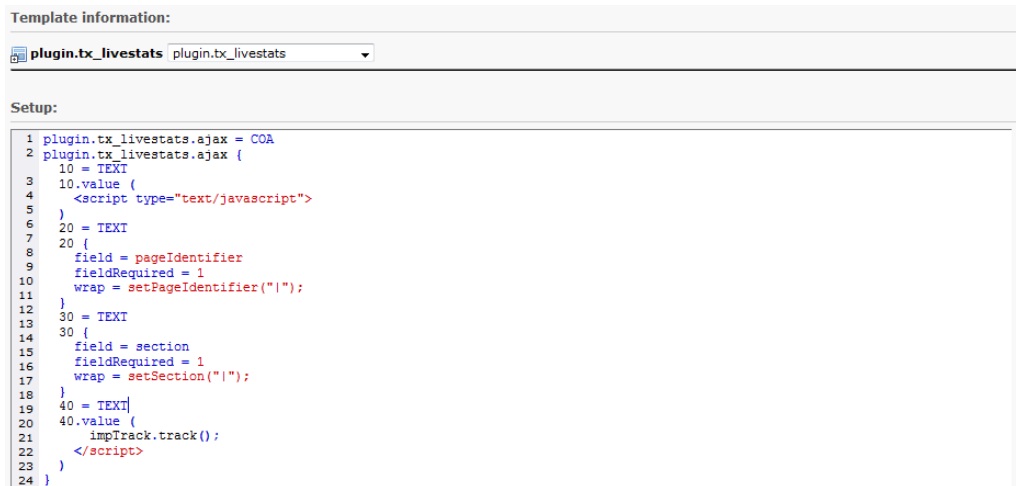
Die Ausgabe kann zum Beispiel in einer auf Extbase/Fluid basierten Extension einfach mit Hilfe des folgenden Befehls ausgegeben werden:

```
<f:cObject typoscriptObjectPath="plugin.tx_livestats.ajax" />
```

Besonders interessant ist dabei die Möglichkeit, Daten aus der Extension heraus an das Typoscript-Object zu übergeben (14).

```
<f:cObject typoscriptObjectPath="plugin.tx_livestats.ajax"
  data="{section: 'myPluginName'}"/>
```

Im Typoscript-Setup können diese Daten nun ausgewertet und der Code entsprechend angepasst werden.



```

Template information:
plugin.tx_livestats plugin.tx_livestats

Setup:
1 plugin.tx_livestats.ajax = COA
2 plugin.tx_livestats.ajax {
3     10 = TEXT
4     10.value {
5         <script type="text/javascript">
6     }
7     20 = TEXT
8     20 {
9         field = pageIdentifier
10        fieldRequired = 1
11        wrap = setPageIdentifier("|");
12    }
13    30 = TEXT
14    30 {
15        field = section
16        fieldRequired = 1
17        wrap = setSection("|");
18    }
19    40 = TEXT
20    40.value {
21        impTrack.track();
22    }
23 }
24 }
```

Abbildung 37 - Erweiterung des Typoscript-Setups von plugin.tx_livestats II

Je nachdem, welche Daten geliefert werden, wird nun immer ein korrekter Javascript-Code zurückgeliefert.

Neben dem Erfassen von Aktionen aus Extensions heraus, wäre es auch denkbar, dass eine Extension den Tracking-Code der Hauptseite manipulieren soll. Dies ist auf Grund der gewählten Umsetzung in Typoscript problemlos möglich. Einzige Voraussetzung dabei ist, dass die Extension ihre Anpassungen nach dem Einbinden des ursprünglichen Codes vornimmt.

6.4 Integration der Auswertung

Neben dem reinen Erfassen von Benutzeraktionen, soll auch eine Auswertung der erfassten Daten in das Typo3-Backend integriert werden. Hierzu wäre es möglich der bestehenden Extension *livestats* ein Backend-Modul hinzuzufügen. Auf Grund der Übersichtlichkeit soll hier jedoch eine eigene Extension *livestats-backend* angelegt werden, die nur dieses Backend-Modul beinhaltet. Hierzu wird wieder der Kickstarter genutzt.

The screenshot shows the 'Kickstart' tool interface for creating a new extension. On the left, there are various categories like 'General info', 'Setup languages', 'New Database Tables', etc. The main area displays a table of files to be included in the extension:

Filename:	Size:	Overwrite:
ChangeLog	102	<input checked="" type="checkbox"/>
README.txt	80	View <input checked="" type="checkbox"/>
ext_icon.gif	124	<input checked="" type="checkbox"/>
ext_tables.php	302	View <input checked="" type="checkbox"/>
doc/wizard_form.dat	603	
doc/wizard_form.html	8.0 K	
mod1/conf.php	307	View <input checked="" type="checkbox"/>
mod1/index.php	7.2 K	View <input checked="" type="checkbox"/>
mod1/locallang.xml	508	View <input checked="" type="checkbox"/>
mod1/locallang_mod.xml	453	View <input checked="" type="checkbox"/>
mod1/mod_template.html	1.1 K	<input checked="" type="checkbox"/>
mod1/moduleicon.gif	82	<input checked="" type="checkbox"/>

Below the table, there is a form to enter the extension key: 'livestats_backend'. There are buttons for 'Update result', 'Update...', 'Total form', 'View result', 'D/L as file', and 'Print WOP comments'. On the right, the author information is displayed: 'Author name: Thomas Winkelmann', 'Author email: info@thomas-winkelmann.de', and the 'Write to location' is set to 'Local: livestats_backend/ (empty)' with a 'WRITE' button.

Abbildung 38 - Anlegen der Extension mit Backend-Moduls im Kickstarter

Nach der Installation der Extension ist in der linken Spalte ein neuer Menüpunkt *Livestats* sichtbar, über den das Modul aufgerufen werden kann.

The screenshot shows the Typo3 Backend interface. The left sidebar contains a menu with categories like 'WEB', 'FILE', 'USER TOOLS', and 'ADMIN TOOLS'. The 'Livestats' module is highlighted under the 'WEB' category. The main content area shows the 'History' view for the 'Livestats' module. The path is '/Home/About TYPO3/History [52]'. The message content includes a 'Hello World!' greeting and a note about the 'Kickstarter' tool. Below this, there is a section for 'GET/POST vars sent to the script' with the following data:

```

GET:
Mweb_txlivestatsbackendM1
id52

POST:
EMPTY!

```

Abbildung 39 - Geöffnetes Backend-Modul im Typo3-Backend

Das Modul ist mit einer kleinen Beispielprogrammierung gefüllt. Wird nun eine Seite aus dem Seitenbaum aufgerufen, wird die entsprechende Seiten-ID als GET-Parameter mit an die Hauptklasse des Backend-Moduls übergeben.

Um nun die Diagramm-Anzeige in dieses Backend-Modul zu integrieren, müsste zunächst dafür gesorgt werden, dass *JQuery* im Typo3-Backend zur Verfügung steht. Dies ist zwar prinzipiell möglich, jedoch nicht besonders sinnvoll, da das Typo3-Backend größtenteils auf *Prototype* und *ExtJS* basiert. Hinzu kommt, dass die gesamte Programmierung des Server-Moduls auch fest in die Extension eingebunden werden müsste, da die gesamte Datenverarbeitung statt im Servermodul nun innerhalb von Typo3 ablaufen müsste. Somit würden zwei völlig eigenständige Systeme entstehen, was den Aufwand zur Pflege und Wartung deutlich erhöht. Zudem wäre es keine Integration des bestehenden Systems mehr.

Als Alternative zu dieser Lösung steht die Nutzung von *Inlineframes*, sogenannten *Iframes*. Hiermit läuft das Präsentations-Modul in einem komplett gekapselten Bereich und es muss keine Programmierung des Moduls in die neue Extension übernommen werden. Ein erster Test mit einem einfachen Einbinden einer bestehenden Ansicht des Präsentations-Moduls verdeutlicht das Vorgehen:

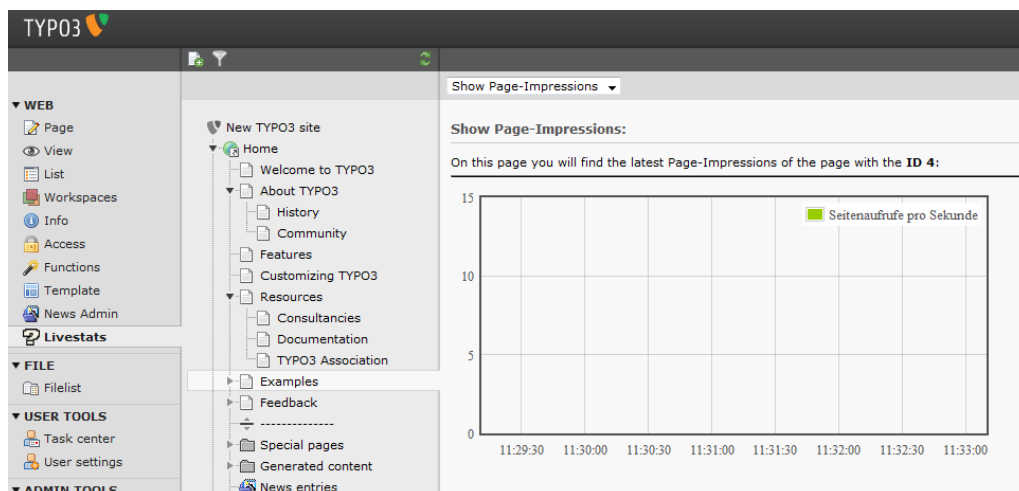


Abbildung 40 - Mittels Iframe eingebettetes Präsentations-Modul

Aus dem Kontext der Integration in Typo3 erwartet ein Nutzer jedoch nun, dass das Diagramm immer nur die Werte der Seite anzeigt, die gerade im Seitenbaum ausgewählt ist. Die Konfiguration befindet sich jedoch derzeit noch komplett im Präsentationsmodul. Um nun eine Verknüpfung zwischen beiden System zu erreichen, müsste die Typo3-Extension in der Lage sein, die Konfiguration des Präsentations-Moduls zu beeinflussen. Dies betrifft in erster Linie den Knoten `<params>` in der XML-Konfigurationsdatei. Hier werden die Parameter hinterlegt, die vom Präsentations-Modul an die Export-Schnittstelle

des Server-Moduls weitergegeben werden. Da hier auch die jeweiligen Filter-Parameter zu setzen sind, muss dieser Bereich der Konfiguration also von "Außen" beeinflussbar sein.

Um dies zu ermöglichen, sind die folgenden Anpassungen am Präsentations-Modul nötig:

- **Ergänzung der Konfiguration über GET-Parameter:**

Nach dem Einlesen der Konfiguration wird die Sammlung an Parametern zur Steuerung der Export-Schnittstelle aus der statischen XML-Konfigurationsdatei durch die Werte ergänzt, die dem Haupt-Script des Präsentations-Moduls über einen GET-Parameter übermittelt werden.

```
// Elemente aus Konfiguration einlesen
$elements = $configManager->getElements();
if(isset($_GET['additionalParamsConfig'])) {
    foreach ($elements['diagram'] as $key => $diagram) {
        // Get config from GET
        $additionalConfig =
            unserialize(urldecode($_GET['additionalParamsConfig']));
        if(is_array($additionalConfig)) {
            // Werte in bestehendes Array einfügen
            $elements['diagram'][$key]['data']['params'] =
                array_merge($diagram['data']['params'], $additionalConfig);
        }
    }
}
// Elemente an Template übergeben
$template->setElements($configManager->getElements());
```

- **Weitergabe der Sonderkonfiguration an das AJAX-Update-Script:**

Die Konfiguration der Diagramme wird im Präsentations-Modul an einer zweiten Stelle ausgelesen. Diese befindet sich in dem Skript, das zur Aktualisierung der Diagramme mittels *AJAX* genutzt wird. Es stellt die Anfrage an die Export-Schnittstelle, verarbeitet die zurückgelieferten Daten und stellt somit die Daten für das neu zu zeichnende Diagramm zur Verfügung.

Da dieses Script jedoch vom Client aus aufgerufen wird, muss ihm erneut die Sonderkonfiguration mitgeteilt werden. Dies geschieht in der Konfiguration des AJAX-Requests:

```
$code .= 'function updateDiagram_.$diagram["@attributes"]["id"].'.'(){
    $.ajax({
        url: \'ajax/get_graph_data.php\',
        data: {\`id\`: \'`.$diagram["@attributes"]["id"].'\`,
            \'additionalParamsConfig\':
                \'`rawurlencode(serialize($diagram["data"]["params"]))\`},
        method: \'GET\',
        dataType: \'json\',
        cache: false,
        success: onDataReceived
    });';
```

Somit kann nun im eigentlichen Script die Anpassung der Parameter nach einem ähnlichen Muster erfolgen:

```
// Diagrammeinstellungen laden
$diagramm = $configManager->getDiagramById($_GET['id']);
// Params überschreiben, falls weitere per GET übermittelt wurden
if(isset($_GET['additionalParamsConfig'])) {
    $params = unserialize(rawurldecode(
        $_GET['additionalParamsConfig']));
    $diagramm['data']['params']=array_merge
        ($diagramm['data']['params'], $params);
}
```

Unter diesen Voraussetzungen kann der Aufruf des Iframes im Typo3-Backend-Modul angepasst werden. Hier wird nun ein serialisiertes Array als GET-Parameter mit den notwendigen Parameterwerten an das Präsentations-Modul übergeben.

```
function moduleContent() {
    switch((string)$this->MOD_SETTINGS['function']) {
        case 1:
            $content='On this page you will find the latest Page-Impressions of
                the page with the <b>ID '.$_GET['id'].'</b>:<hr />';

            $paramsArray = array();
            $paramsArray['pageIdentifier'] = $_GET['id'];

            $content.='<iframe src="http://server1/screen/
                ?additionalParamsConfig='.rawurlencode(serialize($paramsArray))
                ." style="width:600px; height: 400px; border: 0px;"
                frameborder="0" scrolling="yes"></iframe>';

            $this->content.=$this->doc->section('Show Page-
                Impressions:', $content, 0, 1);
            break;
        }
    }
}
```

Ein erster Test zeigt das erhoffte Verhalten: Es wird jeweils zwei Mal die Seite mit der ID 64 aufgerufen, sowie jeweils einmal die Seite mit der ID 65. Der Aufruf der Statistiken liefert im Anschluss die folgenden Bilder:

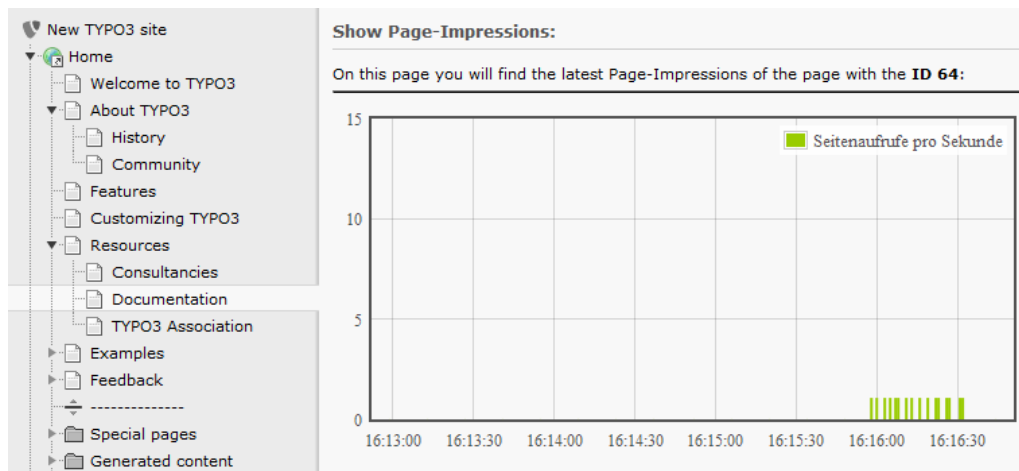


Abbildung 41 - Statistik der Seite mit der ID 64

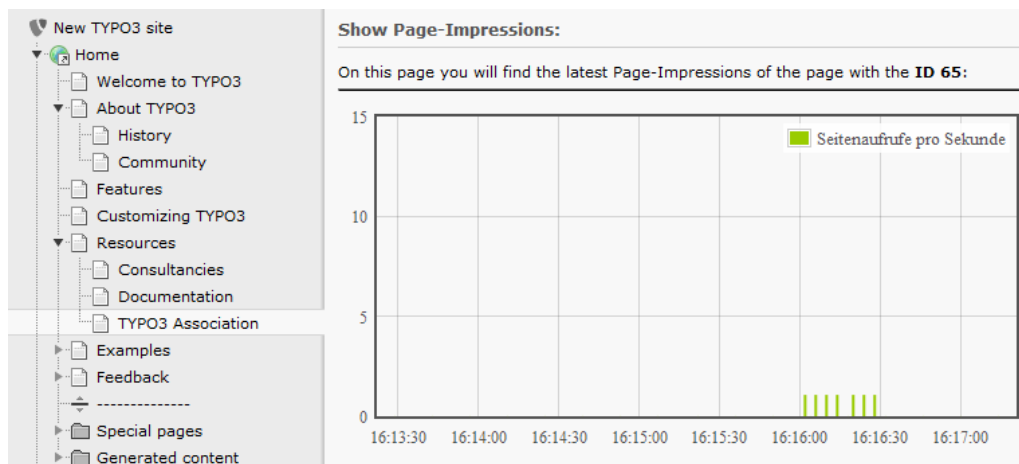


Abbildung 42 - Statistik der Seite mit der ID 65

Im Gegensatz zur Extension, die zur Integration des Tracking-Codes erstellt wurde, kann hier kein Typoscript zur Konfiguration des Backend-Moduls genutzt werden. Typoscript steht für Extensions nur im Frontend zur Verfügung. Um jedoch Einstellungen wie die Adresse der Export-Schnittstelle des Server-Moduls zu konfigurieren, kann auf die Extension-Konfiguration in Typo3 zurückgegriffen werden (15). Hierzu muss lediglich eine Datei mit der Bezeichnung *ext_conf_template.txt* im Stammverzeichnis der Extension hinterlegt werden, in der die benötigten Felder konfiguriert werden:

```
# cat=basic/export/1; type=input; label=Server URL
serverUrl = http://
```

```
# cat=basic/export/2; type=input; label=Project Id
projectId = 1
```

Im Typo3 Extension-Manager erscheint daraufhin die folgende Ausgabe:

Extension Manager
Extension: **Livestats Backend** (livestats_backend)

Current status:
The extension is installed (loaded and running)!
Click here to remove the extension:

Configuration:
(Notice: You may need to clear the cache after the configuration of the extension. This is required if the extension adds TypeScript depending on these settings.)

Server URL [serverUrl]

Project Id [projectId]

Details:

Title	Livestats Backend
Description	
Author	Thomas Winkelmann <info@thomas-winkelmann.de>
Version	0.0.0
Category	Backend Modules
State	Alpha
Shy	No

Abbildung 43 - Konfiguration der Extension `livestats_backend`

Die passenden Werte können darin gesetzt werden und anschließend in der Programmierung des Backend-Moduls wie folgt ausgelesen und verwendet werden:

```
$this->extConf = unserialize($GLOBALS['TYPO3_CONF_VARS']
    ['EXT']['extConf']['livestats_backend']);

$serverUrl = $this->extConf['serverUrl'];
$projectId = $this->extConf['projectId'];
```

7 Integrationsbeispiel Moodle

7.1 Vorstellung der E-Learning-Plattform Moodle

Bei Moodle handelt es sich um ein Open-Source Software-Projekt, das speziell für den Aufbau und Betrieb einer E-Learning-Umgebung ausgelegt ist. Der Name Moodle war ursprünglich ein Akronym für *Modular Object-Oriented Dynamic Learning Environment*. (16)

Eine Moodle-Installation ist unterteilt in einzelne Kurse. Innerhalb dieser Kurse werden von Lehrenden, bei Moodle in der Regel mit *Trainer* bezeichnet, Arbeitsmaterialien bereitgestellt, die von den Lernenden, bei Moodle mit *Teilnehmer* bezeichnet, dann online bearbeitet werden. In diesen Kursen werden zudem zahlreiche weitere interaktive Module bereitgestellt, die eine Zusammenarbeit der Teilnehmer/innen erlauben. Dies sind unter anderem Foren, Wikis oder Umfragen.

The screenshot shows the Moodle course interface for 'THM Infokurs Master MI'. At the top, the user is logged in as 'Thomas Winkelmann'. The left sidebar contains several navigation menus: 'Personen' (Teilnehmer/innen), 'Aktivitäten' (Arbeitsmaterialien, Foren, Wikis), 'Suche in Foren' (with a search box), 'Administration' (Bewertungen, Kursteilnahme, IMMI09 beenden, Profil), and 'Meine Kurse' (Bachelorarbeit/Master-Infos, Infokurs Master MI, MIM16 Modellierung). The main content area is titled 'Themen dieses Kurses' and lists various course activities: 'Nachrichtenforum der Studiengangsleiter an Alle', 'Infoforum der Studierenden', 'Formulare', and a numbered list of topics. Topic 1 is 'Gibt es Probleme? Dann hier eintragen!' and Topic 2 is 'Infos zu den Fächern', which includes sub-items like 'Infos zu Fachangeboten SS2011', 'Infos zu trmd-Fächern', 'Einführung Master MI', 'Wer hat was an Fächern/Infos?', 'Terminkalender der ICE-Veranstaltungen', 'Material IP', 'Notenrechner MIM Excel-Sheet (PO bis SS11)', and 'Formular Zuordnungsvorschlag WPF'. The right sidebar shows 'Neue Nachrichten', 'Bald aktuell ...' (no further dates), and 'Neue Aktivitäten' (last activity on May 31, 2011). Below that, 'Neues im Kurs:' lists 'Arbeitsmaterial hinzugefügt:', 'Formular Zuordnungsvorschlag WPF', and 'Arbeitsmaterial aktualisiert: Notenrechner MIM Excel-Sheet (PO bis SS11)'. At the bottom of the sidebar, it says 'Neue Forenbeiträge:'.

Abbildung 44 - Startseite eines Kurses auf der Moodle-E-Learning-Plattform der THM

Die Entwicklung von Moodle wurde 1999 von Martin Dougiamas gestartet, die erste Version erschien im August 2002. Die Plattform wurde fortan stetig weiterentwickelt, im November 2010 folgte dann die Version 2.0 mit zahlreichen Überarbeitungen und Erweiterungen. (17) Derzeit wird Moodle in über 50.000 registrierten Installationen in mehr als 200 Ländern genutzt. (16)

Die Software ist in der Skriptsprache PHP geschrieben und wird auf einem zentralen Webserver betrieben. Dort wird ebenfalls ein Datenbank-Management-System benötigt, in dem die anfallenden Daten abgelegt werden. Die Nutzer greifen dabei über einen Webbrowser auf die Plattform zu.

7.2 Erfüllung der Integrationsanforderungen

Mit Hilfe der auf der Moodle-Projektseite bereitgestellten Dokumentation für Entwickler und einer Testinstallation der Plattform soll zunächst geklärt werden, ob alle in Abschnitt 5 aufgestellten Anforderungen erfüllt werden können, sodass eine Integration der Echtzeit-Web-Analyse-Plattform in Moodle möglich ist.

Alle Angaben beziehen sich dabei auf die zum Zeitpunkt dieser Arbeit aktuelle Version 2.0.3.

- **Erweiterbarkeit:**

Bei einem Blick in die Datei- und Ordner-Struktur einer Standard-Moodle-Installation fällt auf, dass sich zahlreiche Funktionen in der Bezeichnung von Ordnern erkennen lassen. So befindet sich beispielsweise unterhalb des Verzeichnisses */mod* jeweils ein Ordner für die unterschiedlichen Typen von Arbeitsmaterialien und Aktivitäten in einem Kurs. Dies lässt auf eine hohe Modularisierung schließen.

Dieser Eindruck wird bei einem Blick in die Dokumentation bestärkt. In einem sehr umfangreichen Wiki werden zahlreiche verschiedene Plug-In-Typen genannt, die erstellt werden können, um die Funktion von Moodle zu erweitern. (18) Zudem wird mit vielen Beispielen erläutert, wie auf Elemente des Software-Kerns von Moodle zugegriffen werden kann, um bestimmte Daten auszulesen oder Funktionen auszuführen.

+ Anforderung erfüllt!

- **Manuell ausgelöste Nutzerinteraktionen:**

Da innerhalb der Standard-Oberfläche von Moodle fast jede Aktion eines Nutzers (Aufrufen eines Arbeitsmaterials, Ausfüllen von Tests, usw...) mit einem Neuladen der gesamten Seite verbunden ist, kann hier der Seitenabruf als Indikator für eine manuelle Benutzeraktion genutzt werden. Ungeachtet hierbei bleiben nun nur noch Aktionen, die kein Neuladen der Seite erzeugen. Diese sind zum Beispiel Drag&Drop-Aktionen oder das Nachladen von Inhalten über AJAX. Moodle lässt sich also bei diesem Punkt wie eine "normale" Website behandeln.

+ Anforderung erfüllt!

- **Netzwerkverbindung zwischen beiden Systemen/Plattformen:**

Da beide Systeme im Regelfall öffentlich zugänglich sind, muss dieser Punkt nicht weiter beachtet werden. Da sowohl die Web-Analyse-Plattform, als auch Moodle in PHP geschrieben sind, wäre es sogar technisch möglich, beide Plattformen auf dem gleichen Server zu betreiben.

+ Anforderung erfüllt!

- **Einbindung des Tracking-Codes:**

Da viele Funktionen innerhalb der Oberfläche von Moodle auf JavaScript basieren, kann auch der JavaScript-Tracking-Code der Web-Analyse-Plattform genutzt werden. Bei der Recherche von Möglichkeiten, wie dieser integriert werden kann, ist eine Orientierung an der Integration des Google-Analytics-Codes sinnvoll. Hierzu finden sich in zahlreichen Internetforen bereits verschiedene Lösungen. So ist es möglich, den Code direkt in die Template-Dateien des Themes einzubetten oder eine eigene Box zu programmieren, die dann auf die Seiten eingehängt werden kann.

+ Anforderung erfüllt!

- **Zuordnung der Interaktionen in Bereiche:**

Da Moodle auf der Aufteilung der Inhalte in Kurse basiert, erscheint es sinnvoll, auch diese Einteilung für die Zuordnung von Interaktionen zu nutzen. So kann ein Trainer direkt die Aktivität in seinem Kurs einsehen oder ein Administrator über alle Kurse hinweg erkennen, in welchem Kurs aktuell die meisten Interaktionen stattfinden.

Für die Zuordnung wird also die `setSection()`-Funktion des Tracking-Codes genutzt. Als technischer Parameter würde sich in Moodle die ID des Kurses, aber auch der Kurzname des Kurses anbieten.

✚ Anforderung erfüllt!

- **Erweiterbarkeit des GUI:**

Bei einer Standardinstallation von Moodle sind bereits zahlreiche *Berichte* vorinstalliert. Ein solcher *Bericht* wäre auch geeignet, um die Auswertung der Web-Analyse-Plattform zu integrieren. Da es sich bei einem Bericht um einen der möglichen Plug-In-Typen handelt und die HTML-Ausgabe komplett von dessen Programmierung abhängt, ist also eine Erweiterung kein Problem. Über bestimmte Funktionen der Moodle-API können zudem die notwendigen Menüeinträge erzeugt werden. (18)

✚ Anforderung erfüllt!

- **Schutz vor unberechtigtem Zugriff:**

Moodle verfügt über ein fest integriertes Rollen- und Berechtigungssystem. Somit ist es problemlos möglich, bestimmte Seiten und Inhalte nur einem ausgewählten Benutzerkreis, einer Gruppe, zugänglich zu machen. Im konkreten Anwendungsfall sollten normale Nutzer überhaupt keinen Zugriff auf die Statistiken haben, Trainer nur auf die Daten ihrer Kurse und Administratoren Zugriff auf alle Daten.

In Moodle kann für jede Rolle genau festgelegt werden, welche Aktionen erlaubt sind und welche nicht. Das zu erstellende Plug-In muss also lediglich eine neue *Fähigkeit* hinzufügen, die dann für die jeweiligen Rollen aktiviert wird.

✚ Anforderung erfüllt!

7.3 Integration des Trackingcodes

Um den Tracking-Code nun in die Oberfläche der Moodle-Installation zu integrieren, soll der Ansatz des Erstellens einer eigenen Box weiterverfolgt werden. Dieser bietet den Vorteil, dass später über die in Moodle vorhandenen Mechanismen genau bestimmt werden kann, auf welchen Seiten und Bereichen die Box und somit der Tracking-Code eingebunden werden soll. Ebenso kann innerhalb der Programmierung der Box auf das Moodle-API zugegriffen werden.

Das Anlegen einer Box geschieht durch das Hinzufügen eines Verzeichnisses unterhalb von */blocks*. Der Name des Verzeichnisses ist dabei frei wählbar. In ihm muss sich dann neben einer Datei *version.php*, die Informationen zum Versionsstand des Plug-Ins beinhaltet, die PHP-Datei mit der Benennung in der Form *block-<verzeichnisname>.php* befinden. In ihr erfolgt die Programmierung des Blocks. Dazu wird die von Moodle bereitgestellte Klasse *block_base* erweitert.

```
class block_livestats extends block_base {

    function init() {
        $this->title = 'Livestatistiken';
    }

    function get_content() {
        if ($this->content !== NULL) {
        }
        $this->content = new stdClass;
        $this->content->text = '<script type="text/javascript">
            // Üblicher JavaScript-Code
            </script>
            <noscript>
            </noscript>
        ';
        return $this->content;
    }

    function instance_allow_multiple() {
        return false;
    }

    function hide_header() {
        return true;
    }
}
```

Durch das Überschreiben der in der Klasse *block_base* vorgegebenen Methoden kann das Verhalten des Blocks individuell angepasst werden. So wird sichergestellt, dass nur eine Instanz der Box auf einer Seite erlaubt ist und kein Header der Box angezeigt wird.

Nach dem Anlegen dieser Skripte erscheint in der Plug-In-Prüfung der folgende Eintrag:

blocks/glossary_random	Zufälliger Glossareintrag	standardmäßig
blocks/html	Textblock	standardmäßig
blocks/livestats	livestats	nicht standardmäßig (soll installiert werden)
blocks/login	Login	standardmäßig
blocks/mentees	Mentoren	standardmäßig

Abbildung 45 - Neu erkanntes Plug-In in der Administrationsoberfläche von Moodle

Da für das Plug-In keine Anpassungen oder Erweiterungen an der Datenbank notwendig sind, ist die Installation in einem Schritt abgeschlossen. Der Block erscheint nun in der Liste aller verfügbaren Blöcke:

<ul style="list-style-type: none"> ▶ Kurse ▶ Bewertungen ▶ Lokales ▶ Sprache ▼ Plugins <ul style="list-style-type: none"> ▶ Aktivitäten ▼ Blöcke <ul style="list-style-type: none"> Übersicht Kursliste 	Zufälliger Glossareintrag	0	2007101509		Löschen
	Textblock	0	2010071900		Löschen
	Livestats	0	2011052002		Löschen
	Login	0	2007101509		Löschen
	Mentoren	0	2007101509		Löschen
	Mitteilungen	0	2007101509		Löschen
	Netzwerkserver	0	2010112900		Löschen

Abbildung 46 - Administration der Blöcke in Moodle

Er kann nun hinzugefügt werden:

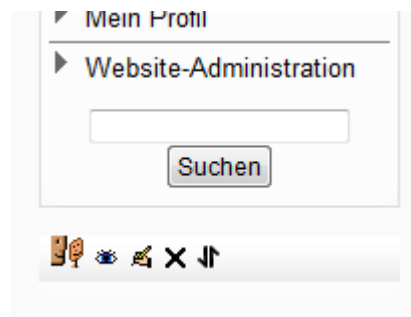


Abbildung 47 - Hinzugefügter Block mit Tracking-Code

Obwohl der Header deaktiviert ist, wird er im Bearbeitungsmodus der Seite noch angezeigt, damit ein Zugriff auf die Einstellungen des Blocks möglich ist. Befindet sich die Seite im Normalzustand, ist nur ein kleiner grauer Rahmen zu erkennen. Damit auch dieser nicht erscheint, wird unterhalb des Tracking-Codes noch der folgende CSS-Code eingefügt:

```
<style type="text/css">
  .block_livestats {
    border: none;
  }
</style>
```

Der CSS-Code wird so immer zusammen mit dem Tracking-Code ausgegeben.

Um die einzelnen Seitenaufrufe einem Kurs zuordnen zu können, muss der Code des Blocks noch ein wenig ergänzt werden:

```
if(strlen($COURSE->shortname)>0) {
    $this->content->text .= 'impTrack.setSection("'.$COURSE->shortname.'");';
}
```

Befindet man sich nicht innerhalb eines Kurses, sondern zum Beispiel auf der Kursübersichtsseite, ist der Kurzname des Kurses automatisch der Kurzname der Moodle-Installation. Eine Identifizierung einzelner Seiten ist im Gegensatz zu einem klassischen CMS hier nicht möglich, da jede Seite für jeden Benutzer individuell erzeugt wird.

Um die Parameter des Tracking-Codes nicht fest in der Programmierung des Blocks hinterlegen zu müssen, soll nun noch für Administratoren die Möglichkeit geschaffen werden, diese Parameter in der Oberfläche von Moodle leicht zu bearbeiten.

Hierzu muss in der Klasse des Blocks zunächst die Konfiguration aktiviert werden (19):

```
function has_config() {
    return true;
}
```

Im Anschluss kann in der Datei *settings.php* die Gestaltung des Formulars vorgenommen werden. Das Erstellen des HTML-Markups, sowie die Logik zum Speichern und Auslesen der Werte übernimmt dabei der Moodle-Core. Zum Einfügen zweier Textfelder sind lediglich die beiden folgenden Befehle nötig (20):

```
defined('MOODLE_INTERNAL') || die;

if ($ADMIN->fulltree) {
    $settings->add(new admin_setting_configtext('block_livestats_serverurl',
        get_string('serverurl', 'block_livestats'),
        get_string('serverurldescription', 'block_livestats'),
        'mydomain.org/server'));

    $settings->add(new admin_setting_configtext('block_livestats_projectid',
        get_string('projectid', 'block_livestats'),
        get_string('projectiddescription', 'block_livestats'), '1'));
}
```

Für einen Administrator ist somit das folgende Formular in den Einstellungen sichtbar.

Abbildung 48 - Einstellungen des Blocks in der Administratorenansicht

In der Programmierung des Blocks kann über die globale Variable `$CFG` auf die Werte zugegriffen werden.

```
$this->content->text .= '<noscript>

</noscript>';
```

7.4 Integration der Auswertung

Um einen neuen *Bericht* mit individueller Programmierung in eine Moodle-Installation zu integrieren, sind zunächst folgende Schritte notwendig (21):

Unterhalb des Verzeichnisses `/admin/reports` muss ein neuer Ordner mit der Bezeichnung des zu erstellenden Reports angelegt werden. In diesem Ordner muss im Anschluss die Datei `settings.php` erstellt werden. Hiermit wird der Bericht im Administrations-Menü registriert:

```
<?php
defined('MOODLE_INTERNAL') || die;
$ADMIN->add('reports', new admin_externalpage('reportlivestats',
'Livestats', "$CFG->wwwroot/$CFG->admin/report/livestats/index.php"));
?>
```

In dem darin referenzierten Skript `index.php` erfolgt dann die Programmierung des Berichts:

```
require_once(dirname(__FILE__).'../../../../../config.php');
require_once($CFG->libdir.'/adminlib.php');

// Check permissions.
require_login();
$systemcontext = get_context_instance(CONTEXT_SYSTEM);

// Print the header.
admin_externalpage_setup('reportlivestats');
echo $OUTPUT->header();

// Print the settings form.
echo $OUTPUT->box_start('generalbox boxwidthwide boxaligncenter');
echo 'Simple Text';
echo $OUTPUT->box_end();

echo '<p>noch mehr HTML-Ausgabe</p>';

// Footer.
echo $OUTPUT->footer();
```

Im Gegensatz zum im vorherigen Abschnitt angelegten Block ist hier keine weitere Installation notwendig. Der Bericht erscheint direkt nach dem Hochladen der Dateien im Administrations-Menü:



Abbildung 49 - Anzeige des Berichts im Administrations-Menü

Ein Aufruf zeigt die zuvor erstellte simple HTML-Ausgabe. Sie ist dabei in die üblichen Moodle-Seitenelemente eingebettet:



Abbildung 50 - HTML-Ausgabe des Berichts

Die Integration kann nun wieder über einen Iframe erfolgen. Da, wie in Abschnitt 7.3 beschrieben, in Moodle jeder Seitenabruf dem dazugehörigen Kurs zugeordnet wird, muss in der Auswertung auch eine Filterung nach Kursen ermöglicht werden. Die Liste aller im System verfügbaren Kurse kann mittels der folgenden Befehle abgerufen und als Auswahlmengü dargestellt werden:

```
$courses = $DB->get_records('course');
$options = array();
foreach($courses as $course) {
    $options[$course->shortname] = $course->fullname;
}
echo html_writer::select($options, 'course', $courseFilter,
    array('' => ''));
```

Der Filter kann somit wieder als Parameter an die im Iframe einzubettende URL angefügt werden. Ein Aufruf des Berichts ohne Filter zeigt so alle im System angefallenen Seitenaufrufe.

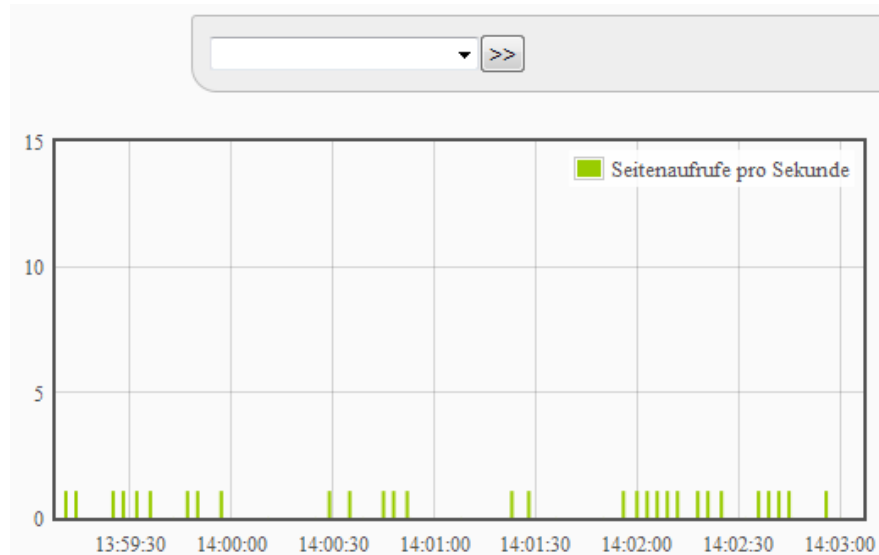


Abbildung 51 - Anzeige aller Seitenabrufe

Im Gegensatz dazu werden mit aktiviertem Filter nur die Seitenaufrufe angezeigt, die im gerade ausgewählten Kurs stattfanden.

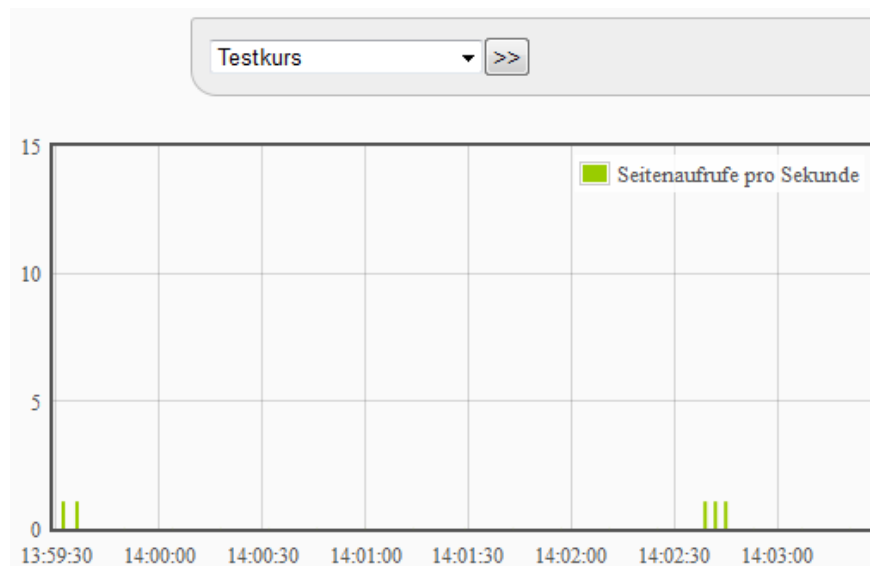


Abbildung 52 - Anzeige der gefilterten Seitenaufrufe

Im Gegensatz zu Blöcken bietet Moodle für Reports offenbar keine eigene Konfigurationsoberfläche an. Es bleibt daher momentan nur die Möglichkeit, die Konfigurationsparameter fest in der PHP-Programmierung des Reports zu hinterlegen.

8 Integrationsbeispiel Piwik

8.1 Vorstellung von Piwik

Das Projekt Piwik stellt sich selbst als Open-Source-Alternative zu *Google Analytics* dar. (22) Es handelt sich somit um eine klassische Web-Analyse-Plattform. Im Gegensatz zu vielen anderen Angeboten kann diese jedoch auf einem eigenen Server betrieben werden. Es wird lediglich ein PHP-Webserver sowie ein MySQL-Datenbankserver benötigt.

Das System wird seit 2008 unter anderem von Matthieu Aubry entwickelt und liegt derzeit in Version 1.4 vor. Die erste öffentliche Version erschien dabei im März 2009.(23)

Piwik bietet zur Auswertung eine moderne webbasierte Oberfläche an, die ausführliche Recherchen nach den verschiedensten Merkmalen erlaubt.



Abbildung 53 - Ansicht der Auswertungsoberfläche von Piwik (22)

Das System setzt ebenfalls auf ein clientseitiges Tracking mittels eines JavaScript-Codes, der in die zu analysierenden Websites eingebaut werden muss. Es verfügt darüber hinaus über weitere Schnittstellen, mit deren Hilfe Seitenaufrufe erfasst werden können.

Auch für den Zugriff auf die erfassten und aufbereiteten Daten existieren Schnittstellen. So wird unter anderem eine iPhone- und Android-App angeboten, die sich diesen Schnittstellen bedient und die Daten auf mobilen Geräten zur Anzeige bringt.

Um auch bei größeren Datenmengen eine schnelle Recherche zu ermöglichen, werden die angefallenen Daten regelmäßig durch ein Skript komprimiert. Dieses Vorgehen führt jedoch dazu, dass die Daten je nach Intervall dieser Komprimierung zeitverzögert zur Verfügung stehen.

8.2 Integrationsmöglichkeiten

Da es sich bei Piwik selbst schon um ein Web-Analyse-Tool handelt, ist eine Integration der Echtzeit-Web-Analyse-Plattform natürlich überflüssig. Viel interessanter ist hingegen die Verknüpfung beider Systeme.

Ein Schwachpunkt der im Entwicklungsprojekt erstellten Plattform ist es, dass kein Zugriff mehr auf die Daten möglich ist, sobald die eingestellten Speicherzeiträume überschritten wurden. Somit ist zum Beispiel ein Vergleich der aktuellen Daten mit etwas länger zurückliegenden Daten nicht möglich.

Um dieses Problem zu lösen, wurde bereits im Entwicklungsprojekt ein Transfer-Modul für die Übertragung der Daten in ein Langzeit-Archiv-System angedacht. Das Modul befindet sich dabei an folgender Stelle:

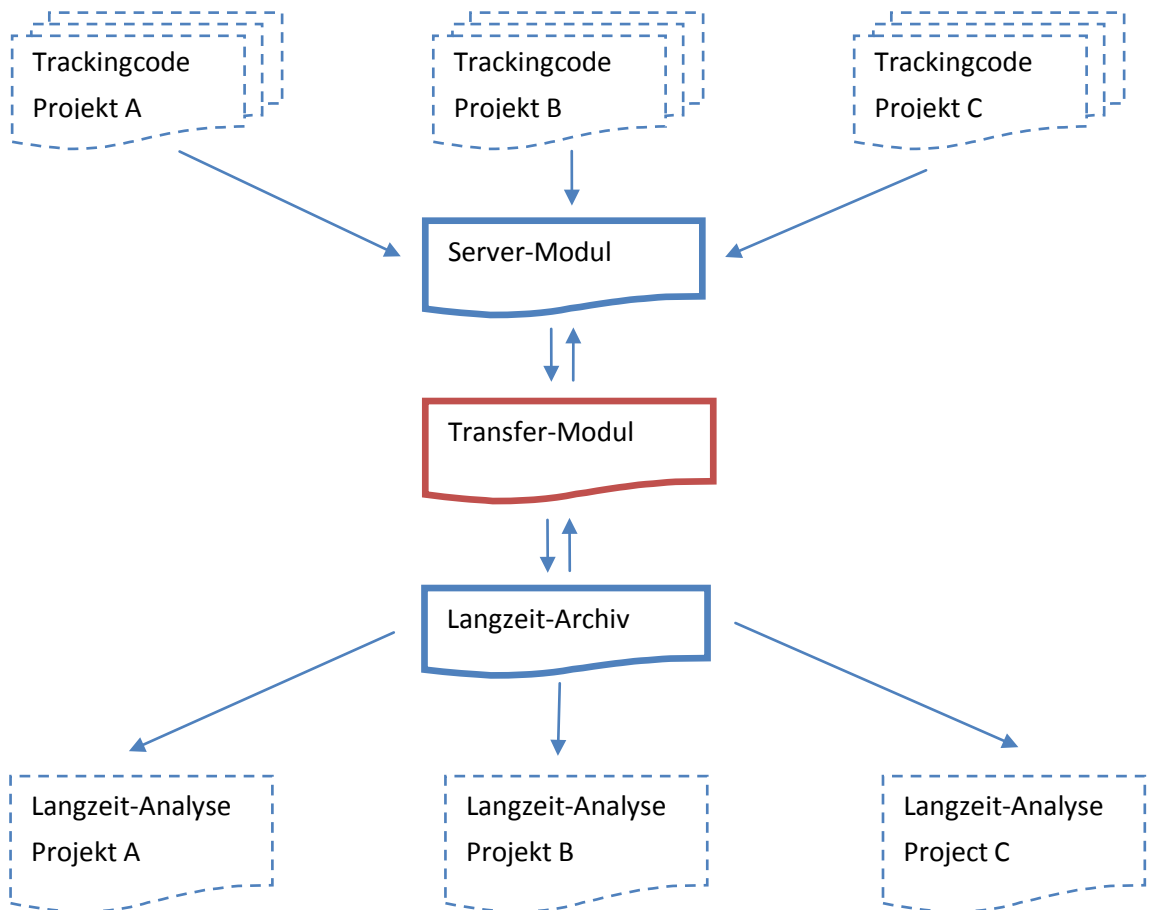


Abbildung 54 - Anordnung des Transfer-Moduls (22)

Es sorgt dafür, dass die erfassten Daten aus dem Echtzeit-Analyse-System abgefragt werden, bei Bedarf konvertiert und dann in das Langzeit-System, hier Piwik, eingespielt

werden. Um dies im konkreten Fall zu erreichen, müssen folgende Einzelschritte umgesetzt werden:

- **Abruf der Rohdaten aus dem Echtzeit-Web-Analyse-System:**

In der bestehenden Form ist es nicht möglich, die mit dem System erfassten Daten in ihrer Rohform wieder aus dem System auszulesen. Alle verfügbaren Abfragemethoden gruppieren die Daten anhand der jeweiligen Merkmale.

Es muss also dem Server-Modul eine neue Export-Methode hinzugefügt werden, die alle Seitenabrufe mit allen dazugehörigen Daten für einen bestimmten Zeitraum zurückgibt. Dies ist notwendig, damit für einen späteren Import der Daten in Piwik wieder möglichst viele Eigenschaften erfasst werden können, so dass auch diese nach einem längeren Zeitraum recherchiert werden können.

✚ Anforderung nach Erweiterung erfüllt!

- **Import der Daten in Piwik:**

Hierzu gibt es zwei prinzipielle Realisierungsmöglichkeiten. Da es sich um ein quelloffenes System handelt, wäre es möglich, zu erfahren, wie die Datenbankstruktur aufgebaut ist, um so die Daten direkt in die MySQL-Datenbank zu schreiben. Dieses Vorgehen ist jedoch nicht zu empfehlen, da die Programmierung des Transfer-Moduls bei jeder Aktualisierung von Piwik überprüft und ggfs. angepasst werden muss.

Neben der Erfassung von Seitenabrufen über einen JavaScript-Code bietet Piwik eine bereits vorgefertigte PHP-Klasse, mit der ebenfalls Seitenaufrufe erfasst werden können. Diese Klasse verfügt über Methoden, mit deren Hilfe jeder Parameter des zu erfassenden Seitenaufrufes manipulierbar ist. Besonders wichtig dabei, seit kurzem kann auch der Zeitpunkt des Seitenaufrufes manuell gesetzt werden. (24) Dies ermöglicht so auch das Erfassen von zurückliegenden Seitenaufrufen.

Dieses Vorgehen ist vermutlich etwas langsamer als der direkte Import in die Datenbank, es nutzt jedoch offiziell angebotene und dokumentierte Software-Schnittstellen und ist daher der ersten Variante vorzuziehen.

✚ Anforderung erfüllt!

- **Zuordnung der Eigenschaften und Koordination des Transfers:**

Ebenfalls muss eine Zuordnung der Eigenschaften zwischen beiden System erfolgen. Das Transfer-Modul muss hierzu die Eigenschaften eines Seitenabrufes aus dem Echtzeit-Web-Analyse-System den Eigenschaften eines Seitenaufrufes in Piwik zuordnen. Dabei sind vermutlich Konvertierung der Datenformate, etc. zu erwarten. Ebenso muss die Zuordnung der Seitenabrufe zu einem Besucher korrekt übertragen werden, damit es in beiden Systemen nicht zu unterschiedlichen Werten kommt. Da das Piwik-Tracking-API auch hierzu eine manuelle Zuordnung erlaubt, ist dies realisierbar.

Damit keine Daten mehrfach übertragen werden, muss das Transfermodul in der Lage sein, den Transfer entsprechend zu koordinieren. Es kann hierzu zum Beispiel abspeichern, welcher Datensatz als letztes bei einem Transfervorgang bearbeitet wurde.

+ Anforderung erfüllt!

8.3 Erweiterung des Server-Moduls

Zum Abruf der erfassten Seitenabrufe aus dem Server-Modul wird dieses um die Export-Methode *getRawPageImpressionforLastXSeconds()* erweitert. Als Grundlage kann dafür die schon vorhandene Methode *getRawPageImpressionCountforLastXSeconds()* genutzt werden. Die Implementierung im Memcached-Plugin sieht dabei wie folgt aus:

```
/**
 * (non-PHPdoc)
 * @see core/interfaces/StorageBackend::getRawPageImpressionforLastXSeconds()
 */
public function getRawPageImpressionforLastXSeconds($projectId,$seconds=60){
    $pis = array();
    $time = time();

    for($i=0; $i<$seconds; $i++) {

        $second = $time-$i;
        $key = 'raw_pi_'. $projectId. '_'. $second;
        $timeSlot = $this->memcacheConnection->get($key);

        if($timeSlot == false) {

            continue;

        } else {

            foreach ($timeSlot as $pi) {
                $pi['time'] = $second;
                $pis[] = $pi;
            }

        }

    }

    return array('pis' => $pis);
}
```

Jeder abgerufenen Page-Impression wird dabei ein Zeitstempel hinzugefügt. Dieser wird aus der der Bezeichnung des jeweiligen Time-Slots generiert, aus der die Page-Impression stammt. Die daraus resultierende Datenstruktur hat im XML-Exportformat für zwei beispielhafte Page-Impressions die folgende Form:

```

- <export>
- <pis>
  - <element>
    <isImageTracker>true</isImageTracker>
    <pageURL>http://server1/moodle/</pageURL>
    <pageIdentifier>123456</pageIdentifier>
    <pageTitle>Moodle-Testinstallation</pageTitle>
    <section>Test</section>
  - <userAgent>
    Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
  </userAgent>
  <screenResolution>1440x900</screenResolution>
  <remoteIp>192.168.0.132</remoteIp>
  <time>1307261306</time>
</element>
- <element>
  <isImageTracker>true</isImageTracker>
  <pageURL>http://server1/moodle/</pageURL>
  <pageIdentifier>123456</pageIdentifier>
  <pageTitle>Moodle-Testinstallation</pageTitle>
  <section>Test</section>
- <userAgent>
  Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
</userAgent>
<screenResolution>1440x900</screenResolution>
<remoteIp>192.168.0.132</remoteIp>
<time>1307261274</time>
</element>
</pis>
</export>

```

Abbildung 55 - Anordnung des Transfer-Moduls (22)

8.4 Realisierung des Transfer-Moduls

Es wird dabei auf die schon aus dem Server- und Präsentationsmodul bekannte Datei und Verzeichnisstruktur zurückgegriffen, um möglichst viele Teile der schon erfolgten Programmierung übernehmen zu können. Dabei werden folgende Verzeichnisse angelegt:

- **configuration:** Enthält die *config.xml*-Datei, in der die globalen Einstellungen des Transfer-Moduls, sowie die Projektzuordnungen zwischen den beiden Plattformen hinterlegt werden.
- **cron:** Beinhaltet ein PHP-Skript, das später zyklisch durch einen Cron-Job aufgerufen wird und den Transfer vornimmt.
- **inc:** Beinhaltet komplette Software-Module, die genutzt werden sollen. Im konkreten Fall das Piwik-PHP-Tracking-Modul, das fertig zum Download angeboten wird.
- **lib:** Kernprogrammierung des Moduls. Hier wird lediglich die *ConfigurationManager*-Klasse aus den vorhandenen Modulen benötigt, die die Konfigurationsdatei einlesen und auswerten kann.
- **tmp:** Hier kann das Modul temporäre Daten in Form von Dateien ablegen, in denen für die einzelnen Projekte jeweils gespeichert wird, bis zu welchem Zeitpunkt die Daten transferiert wurden.

Die XML-Konfigurationsdatei erhält auf Grund der Anforderungen die folgende Struktur.

```
<?xml version="1.0" encoding="UTF-8"?>
<config>
  <settings>
    <dataserver>http://127.0.0.1/server/</dataserver>
    <piwikUrl>http://127.0.0.1/piwik/</piwikUrl>
    <piwikTokenAuth>d8e24685db2669ac6f0cb0655756eb25</piwikTokenAuth>
  </settings>
  <projects>
    <project>
      <livestatId>1</livestatId>
      <piwikId>1</piwikId>
    </project>
  </projects>
</config>
```

Innerhalb des Knotens *projects* können beliebig viele weitere *project*-Unterknoten eingefügt werden, für die die Daten transferiert werden sollen.

Das zentrale Skript des Moduls ist unter `/cron/index.php` zu finden. Nachdem es die notwendigen Komponenten inkludiert und die Klassen instanziiert hat, werden die Daten aus dem Echtzeit-Analyse-Tool angerufen und an das Piwik-Tracking-API weitergegeben.

```

$settings = $configManager->getSettings();
$projects = $configManager->getProjects();

$lastProcessedTimestamp = 0;
if(file_exists(dirname(__FILE__).'../tmp/lastProcessed_'.
    $project['piwikId'].'_'.$project['piwikId'].'.txt')) {
    $lastProcessedTimestamp = trim(file_get_contents(dirname(__FILE__)
        .'../tmp/lastProcessed_'. $project['livestatId']
        .'_' . $project['piwikId'].'.txt'));
}

require_once dirname(__FILE__).'../inc/PiwikTracker.php';

foreach($projects as $project) {
    $url = $settings['dataserver'].'export/index.php?format=xml&pid='
        . $project['livestatId'].'&method=getRawPageImpressionforLastXSeconds
        &params[seconds]=100';
    $xml = @simplexml_load_file($url);

    foreach ($xml->pis->element as $pi) {
        $tracker = new PiwikTracker($project['piwikId'], $settings['piwikUrl']);
        $resolution = explode('x', (string) $pi->screenResolution);
        $tracker->setResolution((int) $resolution[0], (int) $resolution[1]);
        $tracker->setUrl((string) $pi->pageURL);
        $tracker->setUserAgent((string) $pi->userAgent);

        $tracker->setTokenAuth($settings['piwikTokenAuth']);
        $tracker->setIp((string) $pi->remoteIp);
        $tracker->setForceVisitDateTime(date("Y-m-d H:i:s",
            (string) $pi->time));
        $tracker->setVisitorId(substr(md5((string) $pi->remoteIp), 0, 16));

        $tracker->doTrackPageView((string) $pi->pageTitle);
    }
}

```

Besonders interessant dabei ist der Befehl `setTokenAuth()`. Nicht alle Parameter des Piwik-Tracking-API sind frei setzbar. So ist es bei einem normalen Aufruf nicht möglich, die IP und den Zeitpunkt des Besuches zu setzen. Dies ist nur für einen Nutzer mit Admin-Rechten möglich. Damit nicht der Benutzername und das Passwort fest hinterlegt werden müssen, stellt Piwik für jeden Administrator einen Token zur Verfügung, mit dessen Hilfe er über die API in der Lage ist, diese Werte zu manipulieren. Der Token ist dabei in den API-Einstellungen der Piwik-Oberfläche zu finden:

Benutzerauthentifizierung

Wenn Sie Daten mit einem Script, einem Cronjob, etc. abrufen wollen, müssen Sie den Parameter an die URLs anhängen, deren API-Aufrufe eine Authentifizierung benötigen.

```
&token_auth=d8e24685db2669ac6f0cb0655756eb25
```

Der token_auth ist so geheim wie Ihr Login und Passwort, teilen Sie es niemandem mit!

Abbildung 56 - Ansicht des token_auth in der Piwik-Oberfläche

In der Variablen *lastProcessedTimestamp* wird dabei immer der Timestamp des zuletzt bearbeiteten Seitenaufrufes festgehalten. Zu Beginn wird dieser, falls vorhanden, aus einer Datei ausgelesen und zum Ende mit dem aktualisierten Wert in diese Datei zurückgeschrieben.

Besonderes Augenmerk verdient auch der Befehl *setVisitorId()*. Er dient dazu, die einzelnen Seitenabrufe einem Besucher zuzuordnen. Der Parameter besteht dabei aus einer 16-stelligen Hexadezimalzeichenkette. Im einfachsten Fall ist dies also ein gekürzter Hash der IP des Besuchers.

Da jedoch oftmals Benutzer, zum Beispiel in einem Firmennetzwerk, über ein und die gleiche IP surfen, ist diese Methode eher ungeeignet. Oftmals wird hierzu beim ersten Seitenaufruf ein Cookie mit einem zufälligen Wert auf dem Rechner des Besuchers abgelegt, mit dessen Hilfe er wiedererkannt werden kann. Da die Echtzeit-Webanalyse-Plattform diese Methode jedoch derzeit nicht zur Verfügung stellt, müssen möglichst viele Parameter herangezogen werden, die die Nutzer beschreiben. Eine mögliche Optimierung wäre daher:

```
$visitorId = '';
if(isset($pi->remoteIp)) $visitorId .= (string) $pi->remoteIp;
if(isset($pi->screenResolution)) $visitorId .= (string) $pi->screenResolution;
if(isset($pi->userAgent)) $visitorId .= (string) $pi->userAgent;
$tracker->setVisitorId(substr(md5($visitorId), 0, 16));
```

Für einen ersten Test werden zunächst einige Page-Impressions erzeugt und darauf direkt einmal das Cron-Script des Transfermoduls aufgerufen. Die Seitenaufrufe sind danach unmittelbar im Piwik-Live-Widget zu erkennen:

Besucher in Echtzeit		
Datum	Besuche	Seitenansichten
Letzte 24 Stunden	1	4
Letzte 30 Minuten	1	4
Mo 13 Jun - 19:51:12 (0s) - IP: 192.168.0.132		
Direkte Zugriffe		
Seiten:		
		detailliertes Besucher-Log anzeigen

Abbildung 57 - Seitenaufrufe im Piwik-Live-Widget

Zu erkennen ist, dass alle Seitenaufrufe einem Besucher zugeordnet wurden und auch Eigenschaften wie Browser, Betriebssystem und IP stimmen. Ebenso wurden auch die besuchten Seiten getrennt erkannt (farbige Ordner Symbole). Weitere Details hierzu sind auch im ausführlichen Besucher-Log zu finden:

Datum	Besucher	Herkunftsseite	Aktionen
Mo 13 Jun - 19:51:21 IP: 192.168.0.132 Provider: Speedport		Direkte Zugriffe	4 Aktionen - 0s 1. moodiestest: Anmerkungen http://server1/moodle/notes/index.php?filtertype=course&filterselect=0 2. moodiestest: TeilnehmerInnen http://server1/moodle/user/index.php?id=1 3. Kurs sichern: moodiestest: Voreinstellungen http://server1/moodle/backup/backup.php?id=1 4. Moodle-Testinstallation http://server1/moodle/

Abbildung 58 - Protokoll Seitenaufrufe im Piwik-Backend

Erfolgen nun weitere Zugriffe, werden diese ebenfalls dem gleichen Besucher zugeordnet, da alle Parameter, die die *VisitorID* beeinflussen, gleich geblieben sind.

Besucher in Echtzeit		
Datum	Besuche	Seitenansichten
Letzte 24 Stunden	1	6
Letzte 30 Minuten	1	6
Mo 13 Jun - 19:53:09 (1 Minuten 48s) - IP: 192.168.0.132		
Direkte Zugriffe		
Seiten:		
		detailliertes Besucher-Log anzeigen


Abbildung 59 - Seitenaufrufe im Piwik-Live-Widget



Erst nach der Veränderung eines Parameters, zum Beispiel durch den Wechsel des Browsers, wird ein zweiter Besucher angelegt:


Besucher in Echtzeit		
Datum	Besuche	Seitenansichten
Letzte 24 Stunden	2	10
Letzte 30 Minuten	2	10
Mo 13 Jun - 19:54:00 (0s) - IP: 192.168.0.132		
Direkte Zugriffe		
Seiten:		
Mo 13 Jun - 19:53:09 (1 Minuten 48s) - IP: 192.168.0.132		
Direkte Zugriffe		
Seiten:		
		detailliertes Besucher-Log anzeigen

Abbildung 60 - Seitenaufrufe im Piwik-Live-Widget

Neben den Piwik-Live-Daten sind die übertragenen Daten auch in allen anderen Modulen erkennbar und es können beliebige Auswertungen durchgeführt werden, sofern die entsprechenden Parameter übertragen wurden:

Zeitspanne: 2011-06-13 

Seitenname	Seitenansichten	Einmalige Seitenansichten 	Absprungrate	Durchschn. Zeit pro Seite	Ausstiegsrate
moodletest: Anmerkungen	3	2	0%	12 Minuten 58s	0%
Moodle-Testinstallation	3	2	0%	1 Minuten 9s	50%
Forum wird bearbeitet	3	1	0%	0s	0%
Schlagworte	2	1	0%	1s	100%
 moodletest: Teilnehmer	3	1	0%	0s	0%
moodletest: Meine Startseite	1	1	0%	6 Minuten 46s	0%
Kurs: Testkurs	1	1	0%	0s	0%
Kurs sichern: moodletest: Voreinstellungen	1	1	0%	0s	0%
Nachrichtenforum	3	1	0%	2 Minuten 16s	0%

1-9 von 9 



  [Niedrige Entwicklung ausschließen](#)

Abbildung 61 - Piwik Seitentitel-Anzeige

8.5 Skalierbarkeit

Abschließend muss noch einmal darauf hingewiesen werden, dass das Transfer-Modul zwar mit wenig Programmieraufwand eine weitreichende Funktionalität, jedoch auch einige Probleme mit sich bringt.

Die Echtzeit-Web-Analyse-Plattform wurde explizit für stark frequentierte Seiten konzipiert. Bei einer Analyse zeigte sich damals, dass Piwik hier nicht als Alternative geeignet ist, da es auf Grund der aufwendigen Software-Architektur nicht für eine sehr hohe Anzahl von Seitenaufrufen/Besuchern geeignet ist. Mit dem Einsatz des Transfer-Moduls in der beschriebenen Form ist dieses Problem nach wie vor vorhanden. Die Seitenaufrufe werden zwar nicht im Live-Betrieb im System erfasst, die gesamte Menge muss jedoch anschließend in einem noch kürzeren Zeitraum importiert werden. Dies funktioniert nur solange, wie der Import innerhalb der Intervalle, in denen der Transfer gestartet wird, komplett fertiggestellt werden kann.

Im Anschluss müssen die Daten außerdem von Piwik erneut komprimiert werden. Hier wird zum Beispiel für eine Seite mit "mehreren 10.000 Besuchern" eine Zeit von über einer halben Stunde genannt. (25)

Linux/Unix: How to setup a crontab to automatically archive the reports?

A crontab is a time-based scheduling service in Unix-like server. The crontab requires php-cli or php-cgi installed. You will also need SSH access to your server in order to set it up:

```
# crontab -e
```

and then add the lines:

```
MAILTO="youremail@example.com"  
5 * * * * www-data /path/to/piwik/misc/cron/archive.sh > /dev/null
```

The Piwik archive script will run every hour. Generally, it completes in less than one minute. On larger websites (10,000 visits and more), Piwik archive can take up to 30 minutes.

Abbildung 62 - Piwik-Dokumentation zur Einrichtung der Daten-Archivierung (22)

Dies führt dazu, dass das Transfer-Modul nur für Websites geeignet ist, die auch im regulären Betrieb mit Piwik analysiert werden können. Die Kombination mit der Echtzeit-Web-Analyse-Plattform bietet jedoch den Vorteil, dass nur ein Tracking-Code in die zu analysierende Website eingebaut werden muss.

Soll die Kombination auch für stärker frequentierte Websites genutzt werden, sollte eine Lösung gefunden werden, die die Daten nicht wie beschrieben über das API in Piwik einspielt, sondern direkt in die Datenbank schreibt und entsprechende Komprimierungen direkt beim Import vornimmt. Dies bringt allerdings gleichzeitig die geschilderten Nachteile mit sich.

9 Integrationsbeispiel Mobile-App

9.1 Motivation

Ruft man das bestehende Präsentations-Modul der Web-Analyse-Plattform einmal mit Hilfe eines Browsers auf einem derzeit handelsüblichen Smartphone auf, stellt man fest, dass die Anzeige der Diagramme in der gleichen Qualität erfolgt wie auf einem Desktop-Rechner.

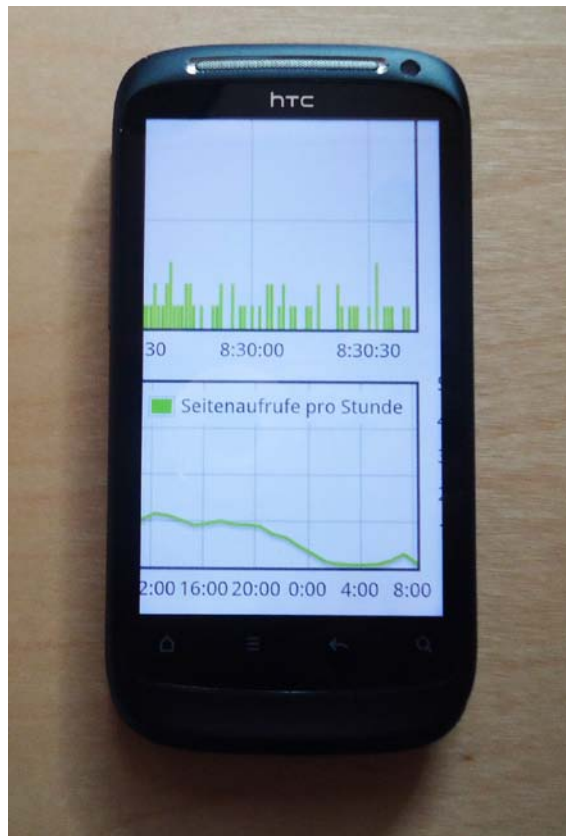


Abbildung 63 - Darstellung des Präsentations-Modul auf einem Android-Smartphone

Dies liegt darin begründet, dass zahlreiche Browser auf mobilen Endgeräten (vor allem iOS- und Android-Geräte) auf der *WebKit*-HTML-Rendering-Bibliothek aufbauen (26). Sie verfügt über ein ausgereiftes HTML- und CSS-Rendering, sowie über eine JavaScript-Umgebung. Dies sind zugleich die einzigen Anforderungen des im Präsentations-Modul genutzten JQuery-Plug-Ins zur Darstellung der Diagramme.

Einziges Hindernis bei der Nutzung sind die Dimensionen und Anordnungen der Diagramme. Da das Präsentations-Modul meist für Bildschirme mit hohen Auflösungen konfiguriert ist, ist es über ein Smartphone eher umständlich zu bedienen, weil wegen der dort eher vorherrschenden kleinen Bildschirm-Auflösungen häufig gescrollt werden muss.

Alternativ kann natürlich der Zoom verkleinert werden, dies hat jedoch dann zur Folge, dass die Diagramme so gut wie nicht mehr zu erkennen sind.

Abhilfe könnte hier eine App schaffen, die die Analyse-Daten mit Hilfe der bekannten Techniken auf einem Smartphone in einer passenden Größe visualisiert, zusätzlich jedoch auch die Navigation zwischen den einzelnen Diagrammen erleichtert. Das heißt, es muss lediglich ein Rahmen um das bestehende Präsentationsmodul gebaut werden, damit es auch problemlos auf mobilen Geräten genutzt werden kann.

Für die Navigation soll dabei auf die bekannten GUI-Elemente (Menüs, Toolbars, Buttons, etc...) von den jeweiligen Plattformen zurückgegriffen werden.

9.2 Vorteile einer Web-App

Eine mögliche Herangehensweise besteht nun darin, für die meist genutzten mobilen Betriebssysteme mit Hilfe des jeweiligen SDKs eine native App zu entwickeln. Dies ist jedoch ein sehr hoher Aufwand für nur eine App, die letztlich auch nur eine Website anzeigen soll.

In den letzten Monaten entstanden jedoch immer mehr HTML-, CSS- und JavaScript-Bibliotheken, die versuchen, mit diesen Technologien das Look&Feel einer nativen App auf einer Website zu imitieren. Mit Hilfe von HTML5 sind inzwischen auch Funktionen realisierbar, die in einer klassischen Website nicht umsetzbar sind. Hierzu zählt zum Beispiel das Speichern von Daten in einem Offline-Datastorage. So bleibt die App auch ohne Netzwerkverbindung lauffähig (27).

Ein weiterer Vorteil besteht darin, dass die Verbreitung der App ohne die Verfügbarkeit im Software-Markt der jeweiligen Anbieter gewährleistet ist. Da keine Installation auf dem einzelnen Endgerät notwendig ist, können zudem Updates deutlich einfacher und schneller umgesetzt werden.

9.3 Erweiterung des Präsentations-Moduls

Für den konkreten Anwendungsfall ist die Erstellung einer Web-App nahezu ideal. Anstatt ein neues komplett eigenständiges Modul zu entwickeln, muss das Präsentations-Modul nur so erweitert werden, dass es eine alternative Ansicht bietet. Es können somit große Teile der bestehenden Programmierung genutzt werden.

Da im Präsentationsmodul bereits das JavaScript-Framework *jQuery* zum Einsatz kommt, bietet es sich an, auch die dazugehörige Bibliothek *jQuery Mobile* zu nutzen. Sie liegt derzeit in der Version Beta1 vor, unterstützt jedoch schon zahlreiche Endgeräte (28).



Abbildung 64 - JQuery Mobile Demo auf verschiedenen Endgeräten (28)

Zu Beginn soll dabei ein Auswahl-Menü stehen, in dem der Nutzer sich für ein bestimmtes Diagramm entscheidet, dass dann im Anschluss dargestellt wird. Über einen Zurück-Button kommt der Nutzer dabei wieder zur Auswahlliste zurück.

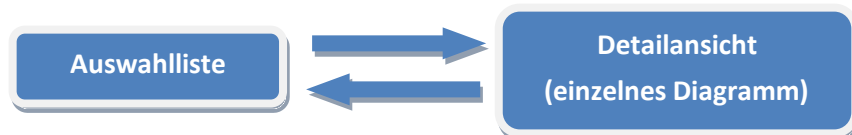


Abbildung 65 - Bedienungsfluss Mobile App

Die zu erstellende App besteht also aus einer Übersichtsseite, sowie je einer Seite für jedes Diagramm. Im Gegensatz zu einer klassischen Website ist es im Falle von *jQuery Mobile* so, dass alle für den Benutzer sichtbaren Seiten physisch in einer Seite, bzw. einer Datei angelegt werden. Über das *id*-Attribut der einzelnen Bereiche kann dann eine Navigation erfolgen (29). Optional sind dabei so genannte Transitions, also Übergangseffekte, zwischen den Seiten möglich, wie man sie von nativen Apps kennt.

Das Grundgerüst des HTML-Markups hat dabei die folgende Form:

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- JS- und CSS-Verweise -->
  </head>
  <body>

    <div data-role="page" id="menu">
      <ul data-role="listview" data-theme="c" data-dividertheme="f">
        <li><a href="#page_projectldiagram1">Diagrammtitel 1</a></li>
        <li><a href="#page_projectldiagram2">Diagrammtitel 2</a></li>
      </ul>
    </div>

    <div data-role="page" id="page_projectldiagram1">
      <div data-role="header">
        <a href="#menu" data-role="button" data-icon="arrow-l">Back</a>
        <h1>Diagramm 1</h1>
      </div>
      <div data-role="content">
        <div id="projectldiagram1" style="width:360px;height:250px;"></div>
      </div>
      <div data-role="footer">
        <h4>Page Footer</h4>
      </div>
    </div>
    <script language="javascript" type="text/javascript">
      function updateDiagram_projectldiagram1() {
        /* JS Code zum Update des Diagrams */
      }
      updateDiagram_projectldiagram1();
      window.setInterval("updateDiagram_projectldiagram1()", 1500);
    </script>

    <div data-role="page" id="page_projectldiagram2">
      <!-- Diagramm2 -->
    </div>
    <script language="javascript" type="text/javascript">
      /* JS Code Diagram 2 */
    </script>

  </body>
</html>
```

Die Grundstruktur des Präsentations-Moduls muss also nicht sehr stark verändert werden. Die komplette Konfiguration der Diagramme kann wie bisher aus den bestehenden Konfigurationsdateien ausgelesen und verarbeitet werden. Lediglich die Größe und Positionierung der Diagramme weicht von einer Anzeige auf einem Bildschirm ab.

Hinweis: Das JQuery-Plugin *Flot* verlangt zum Zeichnen der Diagramme eine fixe Breite und Höhe in Pixeln. Diese Werte variieren jedoch bei mobilen Endgeräten je nach Typ sehr stark. Eine feste Konfiguration wie bei einer Bildschirmanzeige erscheint daher nicht sinnvoll. Eine mögliche Lösung besteht darin, die jeweiligen Werte per JavaScript auszulesen und das `<div>`-Element entsprechend dynamisch anzupassen. Hinzu kommt eine

Änderung der Auflösung zur Laufzeit, die durch das Kippen des Gerätes hervorgerufen wird. Für einen ersten Test soll daher zunächst auf eine statische Größe der Diagramme von 360x250px zurückgegriffen werden.

Um das beschriebene HTML-Markup zu erzeugen, wird die im Präsentation-Modul vorhandene Klasse *ScreenGUITemplate* wie folgt erweitert.

Tabelle 1 - Gegenüberstellung der Funktionen der Klasse *ScreenGUITemplate*

Bestehende Methode	Hinzugefügte Methode	Bemerkungen
getMainPage()	getMainMobilePage()	Liefert den kompletten HTML-Inhalt für die mobile Anzeige.
-	getMobileMenu()	Da bei der konventionellen Ausgabe keine Navigation möglich ist, wurde diese Funktion neu hinzugefügt. Sie erstellt ein Menü mit je einem Eintrag pro Diagramm. Die Bezeichnung wird dabei aus einem neu hinzugefügten Attribut "title" aus der Konfigurationsdatei gezogen.
getHeader()	getMobileHeader()	Liefert den HTML-Code für den Kopfbereich der Seite. In der mobilen Variante wird ein HTML5-Header gesetzt, sowie zusätzlich die <i>jQuery Mobile</i> Bibliothek eingebunden.
getFooter()	getMobileFooter()	Nur geringfügige Anpassungen notwendig.
getDiagramHTML()	getMobileDiagramHTML()	Liefert den HTML-Code für eine <i>jQuery Mobile</i> Seite. Die Seite setzt sich dabei aus Header (mit Zurück-Button), Inhaltsbereich (Diagramm) und Footer zusammen.

getDiagramJS()	-	Zunächst keine Anpassungen notwendig, daher keine weitere Funktion notwendig.
buildPage()	buildMobilePage()	Nur geringfügige Anpassungen notwendig.

Um nicht ständig auf einem mobilen Endgerät testen zu müssen, kann ein HTML5-fähiger Desktop-Webbrowser verwendet werden. Um die Auflösung des Displays entsprechend zu simulieren, reicht ein Verkleinern des Fensters zunächst vollkommen aus. Die erstellte Web-App zeigt sich bei einer Konfiguration mit drei Projekten mit je drei Diagrammen im Mozilla Firefox 4 wie folgt:

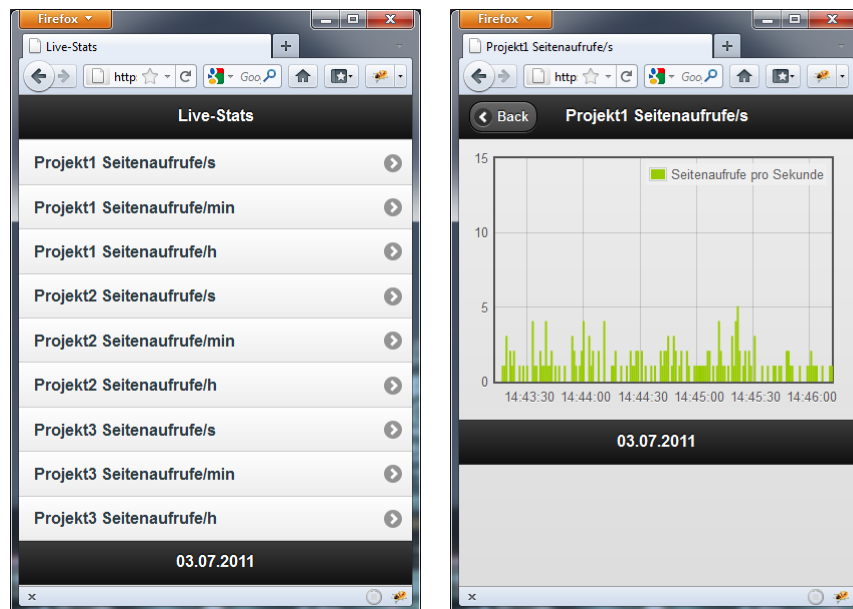


Abbildung 66 - Präsentations-Modul als Web-App

Mit Hilfe dieser Oberfläche ist nun auch die Nutzung des Präsentations-Moduls auf einem modernen mobilen Endgerät problemlos möglich.

Die erstellte Web-App wurde primär mit Betriebssystem *Android* in dem darin enthaltenen Webbrowser in Version 2.1 und 2.3 getestet. Da jedoch auch weitere Browser, wie *Apples Safari*, der ebenfalls häufig auf mobilen Endgeräten zu finden ist, auf der *WebKit*-Engine basieren, ist davon auszugehen, dass auch Plattformen wie *iPhone*, *iPod* und *iPad* problemlos unterstützt werden. (30) Für eine noch breitere Unterstützung sorgt das Framework *jQuery Mobile* von Haus aus. So werden in der Gerätegruppe mit "A-Grade" zahlreiche weitere Plattformen wie *Windows Phone 7* oder *Palm WebOs 3.0* unterstützt (31).

9.4 Weitere Optimierungen der Web-App

Ruft man die mobile Ansicht über einen Browser auf einem handelsüblichen Smartphone auf, stellt man fest, dass die Oberfläche deutlich träger reagiert als auf dem Desktop oder bei vergleichbaren Web-Apps. Dies ist in erster Linie auf die begrenzte Leistung der Geräte zurückzuführen.

Hinzu kommt, dass wie bei der ursprünglichen Anzeige auf einem Monitor nach wie vor alle Diagramme regelmäßig aktualisiert werden, auch wenn diese gerade gar nicht angezeigt werden. Diese Aktualisierung erfolgt in Abhängigkeit von dem in der Konfigurationsdatei hinterlegten Zeitintervall. Sinnvoller ist es daher, die Diagramme erst beim Aufrufen der jeweiligen Seite zu erstellen. Es wird dabei auf das Event-API von *jQuery Mobile* zurückgegriffen. Hierüber ist es möglich, frei definierbaren JavaScript-Code beim Auftreten von bestimmten Events, zum Beispiel dem Aufruf einer Seite, auszuführen. Der Kopfbereich der HTML-Seite wird daher um folgenden Code ergänzt:

```
<script type="text/javascript" src="static/js/jquery.min.js"></script>
<script type="text/javascript">
  $(document).bind("mobileinit", function(){
    $('div').live('pagehide', function(event, ui){
      if(ui.nextPage.attr('id') != 'menu') {
        eval(ui.nextPage.attr('id').
          replace('page_', 'updateDiagram_')+ "()");
      }
    });
  });
</script>
```

Wird nun eine Seite aufgerufen, deren Bezeichner nicht *menu* ist, wird aus deren Bezeichner der Name der entsprechenden JavaScript-Funktion zum Zeichnen gebaut und diese dann einmal aufgerufen. Hierdurch arbeitet die Web-App deutlich ressourcenschonender, vor allem bei einer großen Anzahl an konfigurierten Diagrammen.

Weitere denkbare Verbesserungen wären:

- Durch den Nutzer aktivierbares, regelmäßiges Aktualisieren des gerade angezeigten Diagramms.
- Als Ergänzung zu Diagrammen Unterstützung weiterer Elementtypen
- Optische Gruppierung der Diagramme im Menu nach Projekten

10 Fazit und Ausblick

10.1 Aktueller Stand

Durch die zu Beginn dieser Arbeit durchgeführte Analyse und der Optimierung des Systems, konnten zahlreiche Probleme in der während des Entwicklungsprojektes umgesetzten Programmierung erkannt und behoben werden. Somit arbeiten nun die damals geplanten und umgesetzten Funktionen zur Messung des Besucherverhaltens einer Website stabil und zuverlässig. Durch gezielte Erweiterung konnte das Präsentations-Modul in einer Weise ausgebaut werden, dass die darin dargestellten Daten nun intuitiver wahrgenommen und verstanden werden können.

Die Integration des Tracking-Codes und des Präsentations-Moduls in das Content-Management-System Typo3 und die E-Learning-Plattform Moodle haben gezeigt, dass das System vielseitig verwendbar ist und problemlos in bestehende Web-Applikationen integriert werden kann, sofern diese die notwendigen Anforderungen erfüllen.

Die bei der Arbeit entstandenen Extensions / Plug-Ins zeigen exemplarisch das notwendige Vorgehen, um das Echtzeit-Web-Analyse-System zu integrieren. Sie sind dabei jedoch noch nicht in dem Stadium, dass sie als komplett fertige Erweiterungen für die beiden Plattformen angesehen werden können. Es fehlen weiterführende Funktionen, wie eine Lokalisierung, die zwar nicht unbedingt für einen Test der Integration notwendig sind, jedoch eine "gute" Extension oder Plug-In auszeichnen.

Um weitere Erkenntnisse über den Praxisbetrieb zu erhalten, ist es geplant, das Moodle-Plug-In auf der E-Learning-Plattform der THM zu installieren.

Durch die Erstellung des Transfer-Moduls zu Piwik zeigt sich, dass die Daten auch nach der Erfassung und Verarbeitung durch die Echtzeit-Webanalyse-Plattform noch in vielfältiger Form auch außerhalb des Systems genutzt und weiterverarbeitet werden können. Die Funktionalität des Moduls wurde vollständig implementiert.

Die geschaffene Web-App erweitert das ursprünglich angedachte Nutzungsumfeld des Präsentations-Moduls, ohne dabei das grundlegende Konzept des Moduls zu verletzen. Sie basiert derzeit noch auf einer Beta-Version des JQuery Mobile Frameworks, welches aktualisiert werden sollte, sobald eine finale Version erschienen ist.

Zusammenfassend lässt sich sagen, dass die Plattform ideal geeignet ist, um das aktuelle Besucherverhalten auf einer stark frequentierten Website zu visualisieren. Durch die zahlreichen und einfach anzusteuern Schnittstellen ist sie in nahezu jedes andere System integrierbar.

10.2 Ausblick

Um die Entwicklung des Systems weiter aufrecht zu erhalten, bieten sich zwei Vorgehensweisen an: Dies ist zu einem die Gründung eines Open-Source-Projektes. Plattformen wie *Google Code* oder *SourceForge* stellen hierzu eine gute Grundlage dar, weil bereits zahlreiche andere Open-Source-Anwendungen auf ihnen verwaltet werden. Somit besteht die Möglichkeit, dass andere Website-Betreiber auf das Projekt aufmerksam werden und es auf ihren Seiten nutzen und sich im Idealfall auch an der Weiterentwicklung beteiligen. Da das System selbst bereits auf zahlreichen Open-Source-Projekten aufbaut und keine anderen lizenzrechtlich geschützten Module beinhaltet, steht dem nichts entgegen. Außerdem kann über Foren- und Blog-Einträge versucht werden, den Bekanntheitsgrad des Systems zu erhöhen.

Eine weitere Möglichkeit besteht im professionellen Vertrieb der Plattform. Zwar sind seit dem Beginn der Entwicklung der Plattform zahlreiche weitere Angebote auf den Web-Analyse-Markt gekommen und vorhandene Angebote wurden ausgeweitet, ein direkt vergleichbares System ist jedoch nach wie vor nicht verfügbar.

Dass ein Bedarf an einer solchen Lösung besteht, zeigt unter anderem ein Bericht über die IT-Infrastruktur hinter der Website *wimbledon.com*. *IBM* nutzt hier ebenfalls eine große Anzeige, um die Anzahl der Seitenaufrufe auf der Website zu visualisieren. (32)

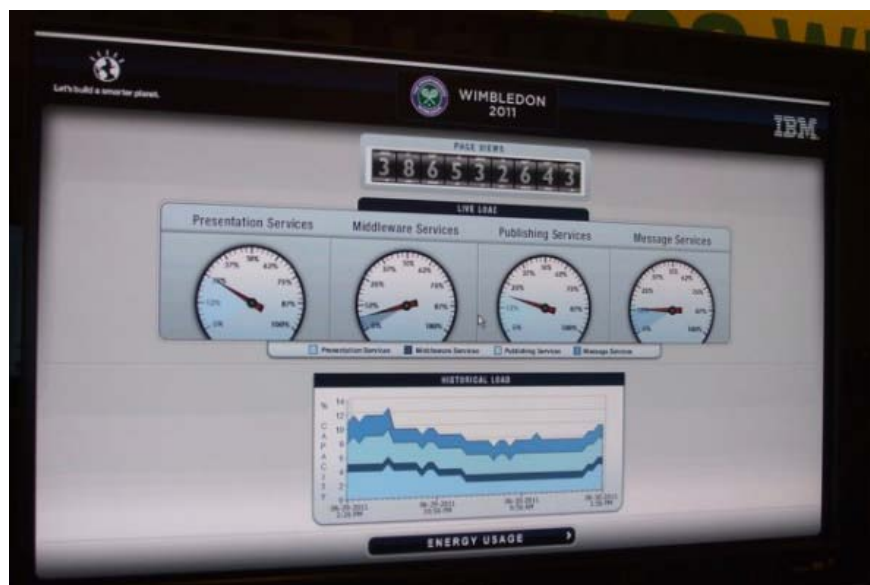


Abbildung 67 - Die Wimbledon-Website wird mit einem IBM-Tool überwacht (32)

Ebenso positiv ist in diesem Zusammenhang festzuhalten, dass die Bereitschaft von Unternehmen, die professionell Websites betreiben, für Web-Analytics-Produkte zu zahlen, stetig zunimmt (33).

11 Literatur- und Quellenverzeichnis

1. Updating graphs with AJAX. *Flot Examples*. [Online] 09. 07 2011. [Zitat vom: 09. 07 2011.] <http://people.iola.dk/olau/flot/examples/ajax.html>.
2. **Patalong, Frank**. Browser-Marktanteile - Internet Explorer weiter im Aufwind. *Spiegel Online*. [Online] 03. 08 2011. [Zitat vom: 2011. 08 20.] <http://www.spiegel.de/netzwelt/web/0,1518,709769,00.html>.
3. **Trevor Bounford, Alastair Campbell**. Statische Diagramme / Balkendiagramme. *Digitale Diagramme*. München : StiebenerVerlag, 2001.
4. Säulendiagramm. *wikipedia.org*. [Online] 11. 08 2011. [Zitat vom: 21. 08 2011.] <http://de.wikipedia.org/wiki/S%C3%A4ulendiagramm>.
5. Cross-Site-Scripting. *wikipedia.org*. [Online] 23. 05 2011. [Zitat vom: 05. 06 2011.] http://de.wikipedia.org/wiki/Cross-Site_Scripting.
6. Application programming interface. *wikipedia.org*. [Online] 07. 09 2011. [Zitat vom: 07. 09 2011.] http://en.wikipedia.org/wiki/Application_programming_interface.
7. **Stern, Hal, Damstra, David und Williams, Brad Williams**. DON'THACKCORE! *Professional WordPress®: Design and Development*. Indianapolis : Wiley Publishing, Inc., 2010.
8. Typo3. *wikipedia.org*. [Online] 16. 06 2011. [Zitat vom: 23. 06 2011.] <http://de.wikipedia.org/wiki/TYPO3>.
9. The Typo3 references blog. *T3Blog.com*. [Online] 23. 06 2011. [Zitat vom: 23. 06 2011.] <http://www.t3blog.com/>.
10. Extension Repository. *TYPO3 - The Enterprise Content Management System*. [Online] 23. 06 2011. [Zitat vom: 2011. 06 23.] <http://typo3.org/extensions/repository/>.
11. **Kamper, Steffen**. Conditions. *sk-typo3.de*. [Online] 2006. [Zitat vom: 07. 09 2011.] <http://www.sk-typo3.de/Conditions.94.0.html>.
12. COBJ_ARRAY (COA, COA_INT). *TYPO3.net - Das deutsche TYPO3-Portal*. [Online] [Zitat vom: 07. 09 2011.] http://www.typo3.net/tsref/cobject/cobj_array/.
13. **RRZN, Regionales Rechenzentrum für Niedersachsen**. Eine Extension mit dem Kickstarter erstellen. *Typo3-Dokumentation*. [Online] [Zitat vom: 08. 09 2011.] <http://www.rrzn.uni-hannover.de/fileadmin/multimedia/typo3/greifswald/material/anleitungen/ErsteExtension.pdf>.

14. **Müller, Steffen.** Combining Fluid ViewHelpers and TypoScript in TYPO3 - 5 basic examples. *T3node.com A TYPO3 BLOG*. [Online] 29. 07 2010. [Zitat vom: 24. 06 2011.] <http://www.t3node.com/blog/combining-fluid-viewhelpers-and-typoscript-in-typo3-5-basic-examples/>.
15. **Vlcek, Maik.** Backend / BE Extension – Konfiguration von Konstanten direkt nach Installation möglich machen. *Maik Vlcek - Software Developer*. [Online] 07. 05 2007. [Zitat vom: 24. 08 2011.] <http://mediavrog.net/blog/2007/05/28/typo3/extensions/backend-be-extension-konfiguration-von-konstanten-direkt-nach-installation-moglich-machen/>.
16. Moodle. *wikipedia.org*. [Online] 16. 05 2011. [Zitat vom: 02. 06 2011.] <http://de.wikipedia.org/wiki/Moodle>.
17. Releases. *Moodle Docs*. [Online] 30. 05 2011. [Zitat vom: 02. 06 2011.] http://docs.moodle.org/en/Moodle_version_history#Moodle_1.0.
18. Developer Documentation. *Moodle Developer Documentation*. [Online] 02. 06 2011. [Zitat vom: 02. 06 2011.] http://docs.moodle.org/.../Development:Developer_documentation.
19. Blocks/Appendix A. *Moodle Developer Documentation*. [Online] 19. 07 2011. [Zitat vom: 27. 07 2011.] http://docs.moodle.org/dev/Blocks/Appendix_A#has_config.28.29.
20. Class List. *Moodle API Reference*. [Online] 27. 07 2011. [Zitat vom: 27. 07 2011.] http://xref.moodle.org/nav.html?_classes/index.html.
21. Admin Reports. *Moodle Developer Documentation*. [Online] 20. 08 2009. [Zitat vom: 01. 08 2011.] http://docs.moodle.org/dev/Admin_reports.
22. *Piwik - Open source web analytics*. [Online] [Zitat vom: 2. 6 2011.] <http://piwik.org/>.
23. Piwik. *wikipedia.org*. [Online] 23. 05 2011. [Zitat vom: 02. 06 2011.] <http://de.wikipedia.org/wiki/Piwik>.
24. Piwik Tracking API. *Piwik*. [Online] [Zitat vom: 3. 6 2011.] <http://piwik.org/docs/tracking-api/>.
25. How to setup auto archiving of your reports? *Pwik Docs*. [Online] 05. 06 2011. [Zitat vom: 05. 06 2011.] <http://piwik.org/docs/setup-auto-archiving/>.
26. WebKit - Verwendung. *wikipedia.org*. [Online] 22. 06 2011. [Zitat vom: 26. 06 2011.] <http://de.wikipedia.org/wiki/WebKit#Verwendung>.

27. **Ebner, Alexander.** Entscheidungshilfe für iPhone-Entwickler: iPhone Apps – Nativ oder als Web-App? *t3n Magazin* . 2010, 18.
28. Touch-Optimized Web Framework for Smartphones & Tablets. *JQuery Mobile*. [Online] 26. 06 2011. [Zitat vom: 26. 06 2011.] <http://jquerymobile.com/>.
29. **Schannak, Sven.** jQuery Mobile: Grundlegende Konzepte und Funktionsweisen. *t3n.de*. [Online] 17. 08 2011. [Zitat vom: 2011. 08 20.] <http://t3n.de/news/jquery-mobile-grundlegende-konzepte-funktionsweisen-326279/>.
30. WebKit. *wikipedia.org*. [Online] 01. 08 2011. [Zitat vom: 13. 08 2011.] <http://de.wikipedia.org/wiki/WebKit>.
31. Supported platforms. *jQuery Mobile Docs*. [Online] 20. 08 2011. [Zitat vom: 20. 08 2011.] <http://jquerymobile.com/demos/1.0b2/#/demos/1.0b2/docs/about/platforms.html>.
32. **Hedemann, Falk.** *t3n.de. Wimbledon Behind the Scenes: Über gigantische Datenmengen zwischen Tennis und Erdbeeren.* [Online] 01. 07 2011. [Zitat vom: 04. 08 2011.] <http://t3n.de/news/wimbledon-scenes-gigantische-datenmengen-zwischen-tennis-318146/2/>.
33. **AddSugar GmbH Consulting.** Trends im Webanalyse-Markt: Google dominiert im Massenmarkt, Top-Portale wechseln zu Profilösungen. *AddSugar CONSULTING*. [Online] 14. 06 2011. [Zitat vom: 20. 08 2011.] <http://www.addsugar.de/component/content/article/97-trends-im-webanalyse-markt-google-dominiert-im-massenmarkt-top-portale-wechseln-zu-profiloesungen>.

12 Diagramm-, Tabellen- und Abbildungsverzeichnis

Abbildung 1 - Präsentations-Modul der entwickelten Plattform während eines Lasttests am Ende des Entwicklungsprojektes	1
Abbildung 2 - Grundstruktur der Module	4
Abbildung 3 - Konfiguration einer Website in der Administrationsoberfläche	5
Abbildung 4 - Demo-Liniendiagramm von Flot (1).....	8
Abbildung 5 - Liniendiagramm zur Visualisierung der Seitenaufrufe pro Sekunde	9
Abbildung 6 - Liniendiagramm zur Visualisierung der Seitenaufrufe pro Minute und Stunde	9
Abbildung 7 - Liniendiagramm im zeitlichen Verlauf	10
Abbildung 8 - Seitenaufrufe der letzten 24 Stunden.	11
Abbildung 9 - Liniendiagramm im zeitlichen Verlauf	11
Abbildung 10 - Ansicht der Sicherheitsmeldung im IE-Script-Debugger.....	12
Abbildung 11 - Aufruf des Tracking-Scripts	12
Abbildung 12 - Säulendiagramm im Präsentations-Modul	14
Abbildung 13 - Säulendiagramm mit logarithmischer y-Achse	16
Abbildung 14 - Gleichzeitig linear und logarithmisch skalierte Achse	16
Abbildung 15 - Liniendiagramm mit Darstellung eines größeren Zeitraumes.....	17
Abbildung 16 - Liniendiagramm mit zusätzlicher Zähler-Anzeige.....	18
Abbildung 17 - Erfolgreich Aufruf des Tracking-Pixels im Internet-Explorer.....	20
Abbildung 18 - Google Analytics Pixel-Abruf	21
Abbildung 19 - Ansicht einer Seite im Seiten-Modul im Typo3-Backend.....	27
Abbildung 20 - Screenshot der im Introduction-Package enthaltenen Website	31
Abbildung 21 - Neu angelegtes Typoscript-Template <i>lib.livestats</i>	31
Abbildung 22 - Konstanten des Typoscript-Template <i>lib.livestats</i>	32
Abbildung 23 - Setup des Typoscript-Template <i>lib.livestats</i>	32
Abbildung 24 - Einbinden von <i>lib.livestats</i> in <i>root_blocks</i>	33
Abbildung 25 - Template-Struktur des Introduction-Packages im Template-Analyser	33
Abbildung 26 - Template-Struktur des Introduction-Packages im Template-Analyser	34
Abbildung 27 - Finale HTML-Ausgabe des Tracking-Codes	34
Abbildung 28 - Typoscript-Code zum Auslesen der Seiten-ID	35
Abbildung 29 - Ausgabe der Seiten-ID	35
Abbildung 30 - Conditions zur Abfrage der Position im Seitenbaum.....	36
Abbildung 31 - Typoscript-Code zum Einbinden des Seitenbaumbereiches.....	36

Abbildung 32 - Ausgabe des Seitenbereichs.....	36
Abbildung 33 - Anlegen einer Extension im Kickstarter	37
Abbildung 34 - Anlegen einer Extension im Kickstarter	38
Abbildung 35 - Anlegen einer Extension im Kickstarter	38
Abbildung 36 - Erweiterung des Typoscript-Setups von plugin.tx_livestats I	39
Abbildung 37 - Erweiterung des Typoscript-Setups von plugin.tx_livestats II	40
Abbildung 38 - Anlegen der Extension mit Backend-Moduls im Kickstarter.....	41
Abbildung 39 - Geöffnetes Backend-Modul im Typo3-Backend.....	41
Abbildung 40 - Mittels Iframe eingebettetes Präsentations-Modul.....	42
Abbildung 41 - Statistik der Seite mit der ID 64.....	45
Abbildung 42 - Statistik der Seite mit der ID 65.....	45
Abbildung 43 - Konfiguration der Extension livestats_backend	46
Abbildung 44 - Startseite eines Kurses auf der Moodle-E-Learning-Plattform der THM.....	47
Abbildung 45 - Neu erkanntes Plug-In in der Administrationsoberfläche von Moodle.....	52
Abbildung 46 - Administration der Blöcke in Moodle.....	52
Abbildung 47 - Hinzugefügter Block mit Tracking-Code.....	52
Abbildung 48 - Einstellungen des Blocks in der Administratorenansicht.....	53
Abbildung 49 - Anzeige des Berichts im Administrations-Menü	55
Abbildung 50 - HTML-Ausgabe des Berichts.....	55
Abbildung 51 - Anzeige aller Seitenabrufe	56
Abbildung 52 - Anzeige der gefilterten Seitenaufrufe	56
Abbildung 53 - Ansicht der Auswertungsoberfläche von Piwik (22).....	57
Abbildung 54 - Anordnung des Transfer-Moduls (22)	58
Abbildung 55 - Anordnung des Transfer-Moduls (22)	62
Abbildung 56 - Ansicht des token_auth in der Piwik-Oberfläche	64
Abbildung 57 - Seitenaufrufe im Piwik-Live-Widget.....	65
Abbildung 58 - Protokoll Seitenaufrufe im Piwik-Backend.....	66
Abbildung 59 - Seitenaufrufe im Piwik-Live-Widget.....	66
Abbildung 60 - Seitenaufrufe im Piwik-Live-Widget.....	66
Abbildung 61 - Piwik Seitentitel-Anzeige	67
Abbildung 62 - Piwik-Dokumentation zur Einrichtung der Daten-Archivierung (22).....	68
Abbildung 63 - Darstellung des Präsentations-Modul auf einem Android-Smartphone	69
Abbildung 64 - JQuery Mobile Demo auf verschiedenen Endgeräten (28).....	71
Abbildung 65 - Bedienungsfluss Mobile App.....	71

Abbildung 66 - Präsentations-Modul als Web-App.....	74
Abbildung 67 - Die Wimbledon-Website wird mit einem IBM-Tool überwacht (32)	77

13 Abkürzungsverzeichnis

ACL	Access Control List, Zugriffssteuerungsliste
AJAX	A synchronous J avaScript and X ML
API	Application Programming Interface, Programmierschnittstelle
CSS	Cascading Style Sheets, stufenförmige geschachtelte Gestaltungsvorlagen
CMS	Content-Management-System, Inhaltsverwaltungssystem
COA	Content Object Array, Inhaltselement in Typoscript
GIF	Graphics Interchange Format, Grafikaustausch-Format
GUI	G raphical U ser I nterface, grafische Benutzeroberfläche
HTML	Hypertext Markup Language, Hypertext-Auszeichnungssprache
HTTP	Hypertext Transfer Protocol, Hypertext-Übertragungsprotokoll
HTTPS	Hypertext Transfer Protocol Secure, sicheres Hypertext-Übertragungsprotokoll
Iframe	Inlineframe, HTML-Element
IP	Internet Protocol, Netzwerkprotokoll
LMS	Learning Management System, Lernplattform
PHP	Hypertext Preprocessor, Skriptsprache
SDK	Software Development Kit,
SQL	Structured Query Language, Datenbankabfragesprache
TS	Typoscript
URL	Uniform Resource Locator, einheitlicher Quellenanzeiger
XML	Extensible Markup Language, erweiterbare Auszeichnungssprache
XSS	Cross-Site Scripting, Seitenübergreifendes Scripting

14 Anlagenverzeichnis

14.1 CD-ROM

- Masterarbeit in Form einer PDF-Datei
- Quellcode des überarbeiteten Server-Moduls
- Quellcode des überarbeiteten Präsentations-Moduls (inkl. Mobile-Ansicht)
- Typo3-Extension zur Integration des Tracking-Code
- Typo3-Extension zur Integration der Auswertung im Backend
- Moodle-Plug-In zur Integration des Tracking-Codes
- Moodle-Admin-Report zur Nutzung der Auswertung im Administrations-Menü
- Quellcode des Piwik-Transfer-Moduls