

IICM  
Technische Universität Graz

IT-Seminararbeit

# **Open Source Webshops**

Performance Evaluierung

Betreuer: Denis Helic

Studenten: Johannes J. Klinger  
Gottfried Schipfer

**Keywords:**

Open Source Webshop, osCommerce, PhPepperShop, XAMP, Performancetest, Webtest, OpenSTA, PHP Tunning, mmCache

**Kurzbeschreibung**

Thema dieser Semiararbeit ist die Evaluierung der externen Leistung von Open Source Webshopsystemen, dh. es wird der gesamte Anwenderprozess untersucht und nicht auf die einzelnen Belastungen (interne Leistung) der Komponenten wie Webserver, Datenbank und Middleware eingegangen. Die Ziele der Arbeit sind die Messung der Antwortzeiten und des Durchsatzes in Form von Online-Transaktionsprozessen (OLTP) ähnlich des TPC-C Standards, nur mit dem Unterschied, dass wir den gesamten Einkaufsprozess von Neukunden abbilden und als atomare Transaktion betrachten. Ein weiteres Ziel war es Optimierungspotenzial aufzuzeigen und einfache Maßnahmen durchzuführen.

# Inhaltsverzeichnis

1.	Einleitung.....	3
2.	Überblick .....	4
2.1	XAMP-System .....	4
2.1.1	(X) - Betriebssystem .....	4
2.1.2	Apache Webserver .....	4
2.1.3	MySQL Datenbank .....	5
2.1.4	PHP.....	5
2.2	Webshops .....	5
2.2.1	osCommerce.....	6
2.2.2	PhPepperShop .....	7
2.2.3	Übersicht der Features der Webshops.....	8
2.3	Performancetest.....	9
2.3.1	Definition der verschiedenen Testarten.....	9
2.3.2	Mögliche Ziele der Performance-Analyse .....	9
2.3.3	Voraussetzungen für Performancetests .....	9
2.3.4	Tipps für die Durchführung.....	10
2.3.5	Testtools .....	10
3.	Setup .....	14
3.1	Testkonfiguration .....	14
3.2	Installation am Server.....	15
3.3	Installation am Test-Host .....	16
3.4	Erzeugen einer Referenzdatenbank.....	16
3.5	Optimierungsmöglichkeiten .....	17
3.5.1	Optimierung an der Hardware.....	17
3.5.2	Optimierung an der Software .....	18
4.	Test .....	19
4.1	Grundsatz-Überlegung .....	19
4.1.1	Use-Cases .....	20
4.2	Testerstellung .....	22
4.2.1	Erstellen von Skripten zur Simulation des Userverhaltens .....	22
4.2.2	Erstellen der Tests .....	22
4.3	Durchführung der Tests.....	22
5.	Interpretation der Ergebnisse .....	23
5.1	Vergleich von OpenSTA und ab (apache bench).....	23
5.2	Ermittlung der max. Einkaufstransaktionen.....	24
5.2.1	Vorbereitungen.....	24
5.2.2	Ermittlung der max. Einkaufstransaktionen bei osCommerce.....	24
5.2.3	Ermittlung der max. Einkaufstransaktionen bei PhPepperShop .....	26
5.3	Lasttests.....	28
5.3.1	Lasttest bei osCommerce .....	28
5.3.2	Lasttest bei PhPepperShop .....	30
5.3.3	Speedup von OnlineTransaktionen .....	32
	Literaturverzeichnis.....	34

# 1. Einleitung

Immer mehr Unternehmen wollen im Internet nicht nur mit einer Homepage vertreten sein sondern auch ihre Produkte in einem Webshop zum Verkauf feil bieten. Beim Blick auf die Preise professioneller Webshop-Systeme bieten sich bei schmalen Budget Webshops auf Basis von Open Source durch ihre Kostenlosigkeit gerade zu an. Der Einsatz dieser auf XAMP – basierenden Webshopssystemen wird von führenden IT- Beratungsunternehmen meist jedoch nur für kleine und mittlere Unternehmen empfohlen. Über die tatsächliche Leistungsfähigkeit solcher Systeme wird aber keine Aussage getroffen.

Ziel dieser Arbeit ist es einen Überblick zu geben, einerseits über die dem Webshop zugrundeliege Plattform XAMP als auch über die Leistungsfähigkeit der Webshops selbst. Es werden die einzelnen Bestandteile eines XAMP-Systems vorgestellt wie auch möglichen Varianten aufgezeigt. Von den derzeit über 20 gängigen Open Source Webshops wurden zwei aufgrund ihrer Verbreitung und unterschiedlichen Zielgruppen als Repräsentanten ausgewählt. Sie werden näher beschrieben und ihrer Merkmale gegenüber gestellt.

Des weiteren werden die verschiedenen Testarten definiert und die Ziele von Performance Analysen spezifiziert. Um diese Tests zu realisieren werden aktuelle Test-Tools vorgestellt. Aufgrund der Verfügbarkeit werden hier ausschließlich Freeware und Open Source Tools näher beschrieben. Da es zur Durchführung der Performance Evaluierung der Shopsysteme erforderlich ist, daß die Testtools bestimmte Features wie DOM – Adressierung beherrschen, wird die Entscheidung für den Einsatz eines Open Source Performance- Testtools getroffen. Um die Ergebnisse der Tests mit anderen Systemen vergleichen zu können werden der verwendete Server und Test-Host seitens Hard- und Software und deren Installation beschrieben. Kurz werden hier auch mögliche Optimierungen aufgezeigt.

Der zweite Teil der Arbeit beschreibt die praktische Durchführung der Tests und die Interpretation der Testergebnisse. Hierfür werden zuerst Userszenarien ermittelt und analysiert. Das Testtool simuliert das definierte Userverhalten beliebig vieler virtueller Benutzer. Damit ist es möglich beliebige Lastsituationen nachzubilden. Die Ergebnisse des Testtools geben Aufschluß über die Leistungsfähigkeit der Webshopssysteme.

## 2. Überblick

### 2.1 XAMP-System

XAMP steht als Abkürzung für ein System bestehend aus dem folgenden Komponenten:

- X** Das **X** steht als Variable für die verschiedenen Betriebssysteme  
Linux (LAMP), Windows (WAMP), MacOS X, OS/2, andere UNIX-Derivate
- A** Apache Webserver
- M** MySQL Datenbank
- P** PHP / Perl / Python Skriptsprache

In Abwandlung gibt es noch eine andere Auslegung der Abkürzung, in der MySQL durch PostgreSQL ersetzt wird. XAMP: **X** (Variables Betriebssystem), **A**pache, **M**iddleware (PHP/Perl/Python) und **P**ostgreSQL.

Wenn beim Apache zusätzlich noch SSL über OpenSSL eingesetzt wird, spricht man auch von XAMPS-Systemen.

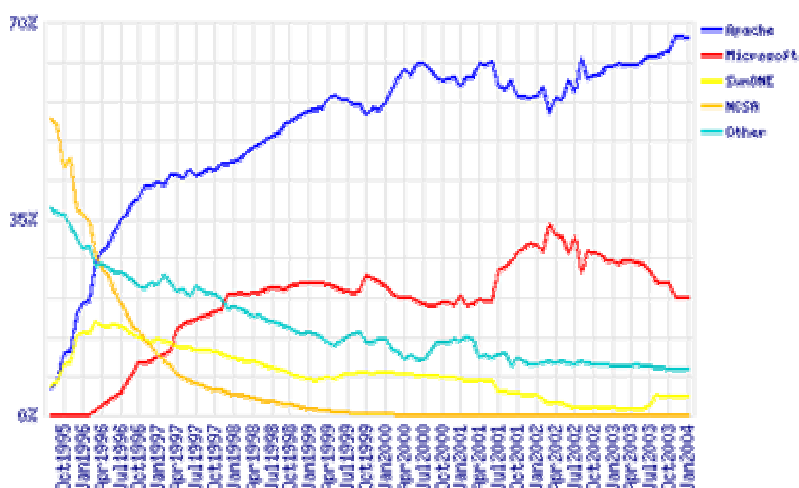
#### 2.1.1 (X) - Betriebssystem

Zwar läuft ein XAMP-System auf verschiedenen Plattformen, trotzdem sind hier UNIX Betriebssysteme, vor allem Linux zu favorisieren, da die anderen Teile von XAMP ursprünglich eben dafür entwickelt wurde und erst später auf die anderen Plattformen portiert wurde. Außerdem ist es auch der Grundgedanke von XAMP, das es komplett aus Open Source Projekten besteht und für jedermann frei zugänglich sein soll.

#### 2.1.2 Apache Webserver

Apache ist der derzeit mit Abstand am meisten benutzte Webserver (siehe Grafik). Das liegt vor allem an seiner freien Verfügbarkeit, seiner Stabilität und vor allem an seiner vielfältigen Konfigurierbarkeit und dem modularen Aufbau. Dieser ist auch für die gute Anbindung von Skriptsprachen an den Webserver verantwortlich. So kann man Perl und PHP nicht nur über das CGI (Common Gateway Interface) sondern direkt über ein Modul anbinden.

Apache selbst entstand 1995 aus einer Gruppe von Benutzern die damals den Webserver des National Center for Super Computer Applications (NCSA) verwendeten. Als das Projekt des NCSA ins Stocken geriet, begannen eben diese User aus dem Code des NCSA-Webserver einen neuen Server zu entwickeln. Um die Fehlerkorrekturen und die Weiterentwicklung zu koordinieren, wurde die „Apache Group“ oder „a patchy group“ als zentrales Forum gegründet.



### 2.1.3 MySQL Datenbank

MySQL ist ein relationales Datenbank Managementsystem (rdbms), welches von der Firma T.c.X. DataKonsult in Schweden entwickelt wurde. Es bietet eine Multi-User, Multi-threaded SQL (Structured Query Language) Datenbankengine, welche schnell, robust und einfach im Gebrauch ist. Da es auf Open Source basiert, ist sie sehr weit verbreitet und eingesetzt.

### 2.1.4 PHP

Die Abkürzung PHP steht für PHP: Hypertext Preprocessor. Der Syntax dieser Interpretersprache ist dem von C/C++, Java bzw. JavaScript ähnlich. Der PHP-Programmcode ist direkt in der HTML-Seite eingebettet. Die durch PHP dynamisch erstellte Seite wird am Server erzeugt und dann als reines HTML an den Client übermittelt, d. h. der Client sieht nur den Output und nie den Code.

PHP entstand 1995 aus den Personal Home Page Tools / Forms Interpreter (PHP/FI), einem Projekt von Rasmus Lerdorf. Ursprünglich eine Sammlung von Perl Skripts, wurde es später immer mehr erweitert und in C geschrieben. 1997 erschien die überarbeitete zweite Version, die bereits von 1% der Domains im Internet verwendet wurde. 1998 kam die dritte, von zwei Studenten völlig überarbeitete Version. Sie war schneller, stabiler und vor allem die Datenbankschnittstellen wurden ausgebaut. Hier bekam sie auch den jetzigen Namen: PHP. Mit der derzeit aktuellen Version 4 kam abermals ein Performanceschub, aber auch erweiterte Modularität. Man ging vollständig vom ursprünglich funktionalen zum Objektorientierten Syntax über. Derzeit ist PHP in etwa 10% der Domains im Internet installiert.

## 2.2 Webshops

Viele Unternehmen setzten mehr und mehr auf den Online-Handel, genügend wickeln ihre Geschäfte ausschließlich nur über das Internet ab. Dementsprechend groß ist auch der Markt für E-Commerce-Lösungen. Die Palette reicht hier von einfachen Open-Source-Shops über Out-of-the-Box-Standartlösungen im mittleren Preisbereich bis hin zu den kompletten E-Commerce-Suits mit horrenden Lizenzkosten. Vor allem für kleine und mittlere Unternehmen, deren Budget beschränkt ist, bieten sich hier diese kostenfreien Open-Source-Lösungen an. Bevorzugte Programmiersprache ist hier Perl und vor allem PHP. Die derzeit am verbreitetsten auf PHP programmierten Shops sind osCommerce, pgMarket, phPay, PhPepperShop und der der phpshop. Sie unterscheiden sich teilweise stark in der Benutzerfreundlichkeit seitens des Shop-Besuchers und des Administrators, ebenso im Erscheinungsbild und der Möglichkeit, dieses ohne hohen Programmieraufwand zu verändern.

Unsere Auswahl fiel hier auf zwei Shops:

#### **osCommerce,**

da er am meisten verwendet wird und sicherlich zu den grafisch am aufwendigsten aufgebauten Webshops gehört.

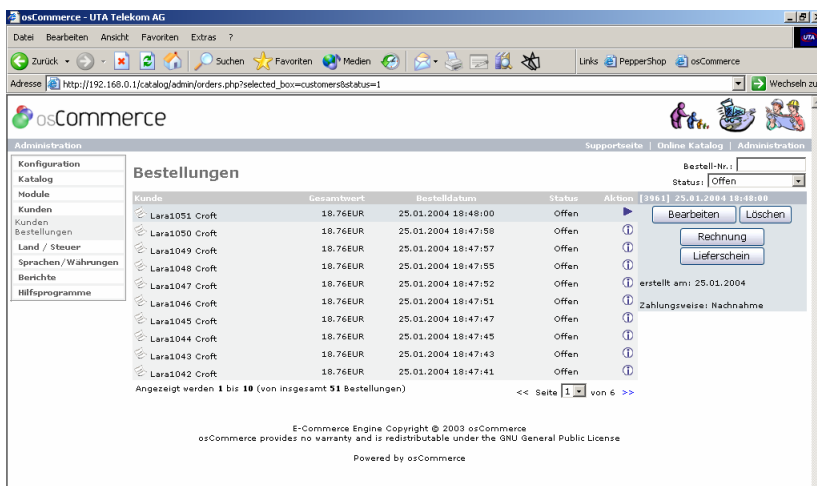
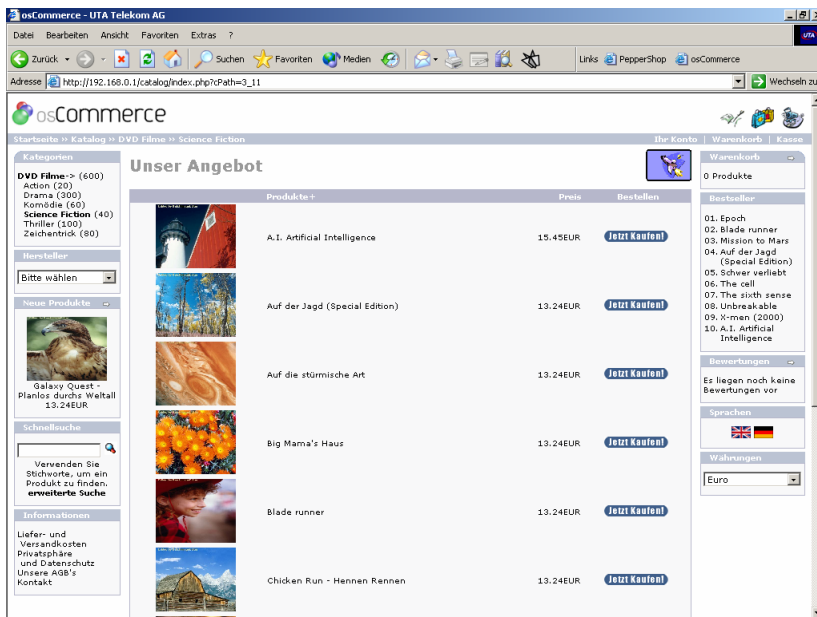
#### **PhPepperShop,**

durch seine Übersichtlichkeit und ausgezeichneten Dokumentation und nicht zuletzt deshalb, da er von zwei Studenten (José Fontanil und Reto Glanzmann von der Züricher Hochschule Winterthur) im Rahmen ihrer Diplomarbeit entwickelt wurde.

### 2.2.1 osCommerce

Schon das Standard-Erscheinungsbild ist gut gelungen und macht einen professionellen Eindruck. Gimmicks wie „Neu im Sortiment“ oder Hitlisten putzen die Optik weiter auf. Der Besucher des Shops findet sich leicht zurecht, da er bei der Navigation ähnlichen kommerziellen Produkten gleicht.

Administrativ ist der Shop über ein übersichtliches Webinterface zu verwalten, es gibt zahlreiche Einstellungsmöglichkeiten wie Internationalisierung, verschiedene Steuern und Währungen. Bei Bedarf lässt er sich noch mit weiteren Modulen für Bezahl- und Versandmodelle erweitern. Bei der Verwaltung der Benutzer- und Artikeldaten fiel uns vor allem der fehlende Im- und Export von CSV-Dateien oder ähnlichen negativ auf.

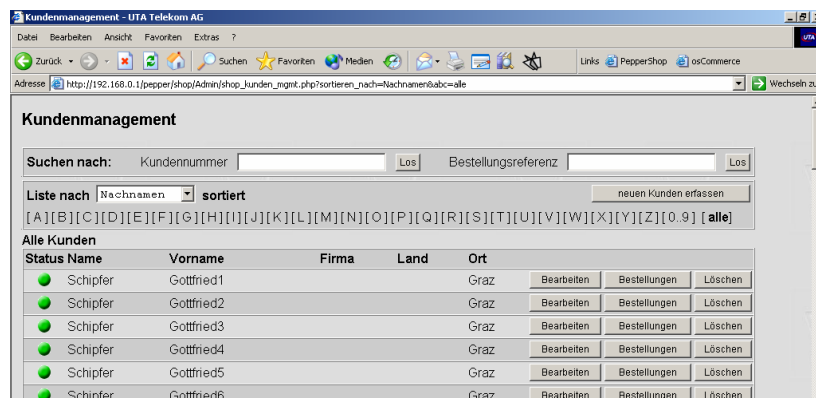
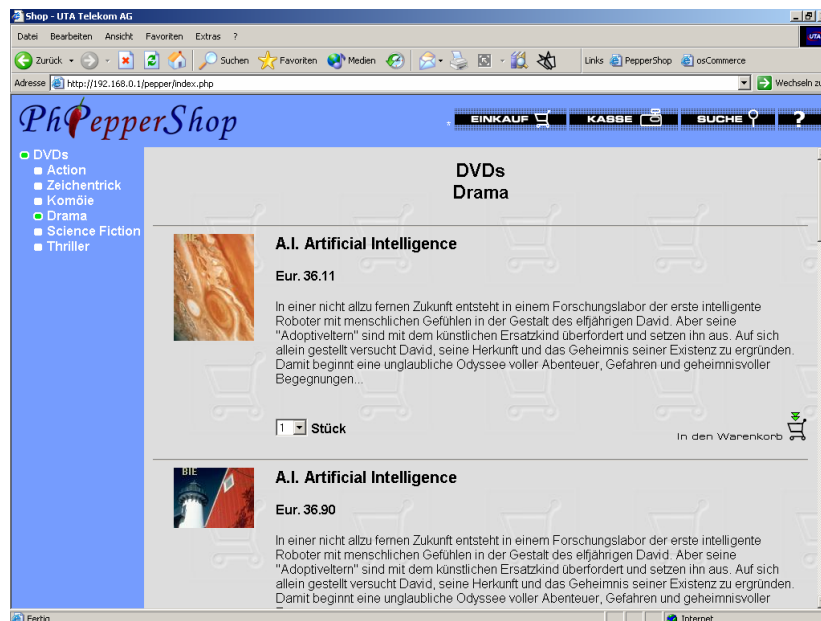


## 2.2.2 PhPepperShop

Die Startseite von PhPepperShop erscheint nach der Installation eher etwas karg. Nach einigen Anpassungen allerdings besticht der Webshop durch seine Übersichtlichkeit, intuitive Bedienung und vor allem seinem flotten Aufbau. Der Aufbau der Artikelübersicht in einer Kategorie weicht von den anderen Webshops ab. Hier werden die Artikeldetails gleich in der Artikelübersicht angezeigt, was dazu führt, das man die maximal Anzahl der angezeigten Artikel pro Seite eher niedriger einstellen muss. Bei Artikeln, die wenig Beschreibung benötigen, erhält man so aber sogar eine bessere Übersicht, da man sich hier das Öffnen der Detailfenster spart. Hervorzuheben ist auch die Möglichkeit, einen Artikel mit beliebig viele Varianten und Optionen zu versehen. Die Administrationsoberfläche ist eher einfach gehalten, beinhaltet aber alle benötigten Funktionen, sowie auch eine automatische Datenbanksicherung und ein Im- und Exporttool für CSV-Dateien.

PhPepperShop ist nur bis zur Version 1.2 frei verfügbar, die hier getestete Version 1.4 SR1 bekommt man nur, wenn man die Entwickler sponsert. Man kann den Shop auch individuell anpassen und auch gleich von den Entwicklern hosten lassen. Hervorzuheben ist ausgezeichnete Dokumentation des Shops und die gut frequentierten Foren.

Wir danken an dieser Stelle José Fontanil, einer der Entwickler des Shops dafür, dass er uns spontan die neuste Version zur Verfügung stellte und uns auch bei Fragen per email unterstützte.



### 2.2.3 Übersicht der Features der Webshops

<b>Webshop</b>	<b>osCommerce 2.2</b>	<b>PhPepperShop 1.4 SR1</b>
Homepage	www.oscommerce.com	www.phpeppershop.com
<b>Katalog</b>		
Artikel / Unterkategorien	beliebig/beliebig	beliebig/beliebig
Attribute / Varianten	feste Anzahl/beliebig	beliebig/beliebig
Artikel-ID für Varianten	nein	nein
Mehrfachzuordnung	nein	ja
Produktverfügbarkeit	ja	nein
<b>Auftragsverfolgung</b>		
Auftragsbestätigung/Lieferstatus	ja/ja	ja/nein
Produktverfügbarkeit	ja	nein
<b>Personalisierung</b>		
Kundenprofile	ja	ja
Mehrere Lieferanschriften	ja	nein
<b>Suche</b>		
Volltext / Detail	ja/ja	ja/nein
Boolsche Verknüpfungen	ja	ja (UND)
<b>Produktdarstellung</b>		
Baum/Liste/Detail/Sortierung	ja/nein/ja/ja	nein/nein/ja/nein
<b>Warenkorb</b>		
Datenerhaltung	Session übergreifend	Session übergreifend
<b>Bezahlung</b>		
Vorkasse	ja	ja
Nachnahme	ja	ja
Einzug-Lastschriftverfahren	nein	ja
Rechnung	ja	ja
Bankeinzug	ja	ja
Kreditkarten	ja	ja
<b>Internationalisierung</b>		
Sprachen/Währungen/Steuersätze	ja/ja/ja	nein/nein/ja
<b>Marketing</b>		
Hitliste	ja	nein
„Neu im Sortiment“	ja	nein
Banner	ja	nein
<b>Verwaltung</b>		
Datenbackup	ja	ja
Im- und Exporttool	nein	ja
Statistik	ja	nein

## 2.3 Performancetest

Der Begriff Performancetest wird häufig in Zusammenhang mit Lasttest und Stresstest gebraucht und ist leider nicht einheitlich definiert. Um hier eine Verwirrung zu vermeiden empfiehlt sich eine terminologische Klärung: "Leistung bzw. Performance definiert sich als eine Aussage über die Fähigkeit eines Anwendungssystems, eine Aufgabe oder eine Aufgabenmenge innerhalb einer bestimmten Zeitspanne bewältigen zu können", das heißt sie wird durch nicht funktionale Qualitätskriterien beschrieben.

### 2.3.1 Definition der verschiedenen Testarten

- **Loadtest,**  
Test mit festgelegter Auslastung
- **Stresstest**  
Test mit Überlast
- **Performancetest**  
Test mit unterschiedlichen Laststufen

Das zeigt, dass sie mit Load-Testtools durchaus auch Performance- und Stresstests durchführen können.

Bei der quantitativen Bewertung der Performance sind die *Antwortzeit* und der *Durchsatz*, der die Anzahl der Transaktionen pro Zeiteinheit angibt, von Hauptinteresse.

Es ist zwischen externer Leistung (gesamte Anwenderprozess wird betrachtet) und interner Leistung (einzelne Komponenten) zu unterscheiden.

### 2.3.2 Mögliche Ziele der Performance-Analyse

- Messung des Antwortzeitverhaltens und des Durchsatzes
- Analyse der Skalierbarkeit
- Ermittlung des Optimierungspotentials
- Untersuchung des Verhaltens unter Stress

Ein anderer Aspekt sind Benchmarks, die maßgeblich durch die Institutionen TPC und SPEC definiert werden. Standardisierte Benchmarks können aber meist nur zur Untersuchung der Basistechnologien herangezogen werden, z.B. zum Performance-Vergleich zweier Webserver. Im Regelfall sind aber eigene Kennzahlen erforderlich.

### 2.3.3 Voraussetzungen für Performancetests

- Testdatenbestand
- Userprofile
- Reproduzierbare Baseline
- Definition des System unter Test
- Lastserver für die Steuerung der Clients
- Realistische Testumgebung
- Testtreiber mit Testskripten

## 2.3.4 Tipps für die Durchführung

Performance-Anforderungen und Systemumgebung sind exakt und korrekt zu spezifizieren  
Die Requierements – Analyse ist durchzuführen  
Mit den Performance-Untersuchungen frühestmöglich beginnen  
Möglichkeiten zur Systemoptimierung sind permanent zu suchen und zu dokumentieren, aber es sollten nicht permanent Optimierungen durchgeführt werden.  
Performancetests sind auch während Betrieb und Wartung in betracht ziehen, damit frühzeitig Probleme und Engpässe erkannt werden können.  
Beim Einsatz von Werkzeugen für Last und Stresstests ist die Toolauswahl und die Einsatzplanung entscheidend. Entsprechend aufwendig sind Produktbewertungen  
Qualifiziertes Personal für Begleitung bzw. Durchführung von Performancetests muss verfügbar sein

## 2.3.5 Testtools

Dass die Auswahl des geeigneten Performancetesttools eine grundlegende und schwierige Aufgabe ist wurde bereits erwähnt. Da wir über kein Budget verfügten, mussten wir unsere Suche auf Freeware und OpenSource Load- und Performance-Testtools einschränken. Da die Mehrzahl der Webshop-Besucher Windowssysteme verwenden, entschieden wir uns für den Testhost die gleiche Plattform einzusetzen. Dies schränkte die Auswahl der Testumgebung weiter ein. Für die Performance-Evaluierung der Webshopsysteme wurden folgende Tools in Betracht gezogen:

### 2.3.5.1 Einfache Loadtest-Tools:

openload  
ab (apachebench)

Hier handelt es sich um kleine Kommandozeilentools mit denen ein Dokument (z.B. index.php) mehrfach angefordert werden kann. Die Anzahl der parallelen Abfragen ist ebenso einstellbar. Bsp. Syntax für den Test der Startseite von osCommerce:

```
ab -n 1000 -c 20 http://192.168.0.1/catalog/index.php

Document Path:      /catalog/index.php
Document Length:    29137 bytes
Concurrency Level:  20

Time taken for tests: 165.277657 seconds
Complete requests:  1000
Failed requests:    994
   (Connect: 0, Length: 994, Exceptions: 0)
Write errors:       0
Total transferred:  29721155 bytes
HTML transferred:  29245155 bytes
Requests per second: 6.05 [#/sec] (mean)
Time per request:   3305.553 [ms] (mean)
Time per request:   165.278 [ms] (mean, across all concurrent requests)
Transfer rate:      175.61 [Kbytes/sec] received
```

Wir haben ab verwendet um die Ergebnisse der Testumgebung OpenSTA zu überprüfen. Hier ist darauf zu achten, dass apachebench wirklich nur eine einzige Datei anfordert und nicht alle Inhalte mit anfordert wie es ein aufgezeichnetes Testskript bzw. der Internet-Browser macht.

### 2.3.5.2 Trialversionen und Freeware der Marktführer

#### **Mercury Astra Loadtest**

Die Trialversion vom Marktführer Mercury Interactive dem Entwickler des quasi Industriestandard Load-Test Tools LoadRunner. Es hat sehr viele Features, aber auch so starke Einschränkungen, dass damit kaum sinnvolle Tests durchführbar sind.

#### **Webpartner Performance Center**

Ein ebenfalls sehr populäres und interessantes Tool mit kostenloser voll-funktionsfähiger 30 Tage Testversion. Laut Webseite hat unsere Hardware die minimalen Systemanforderungen erfüllt, der erste Versuch einen Test zu erstellen wurde jedoch mit einer Fehlermeldung und dem Wunsch nach mindestens 800Mbyte RAM quittiert.

#### **IBM Web Performance Tools**

Freeware, die mittlerweile Teil eines kommerziellen Produkts ist, und leider nicht mehr Verfügbar ist

#### **Microsoft WAS**

Ein relativ gut gelungenes weit verbreitetes Freewaretool, leider beherrscht es keine DOM – Adressierung und konnte deswegen für unsere Tests nicht verwendet werden.

### 2.3.5.3 Open Source Test-Tools

#### **Dieseltest**

Ein in Delphi programmiertes Loadtest-Tool mit Scriptrecorder, ein kurzer Test hat aber gezeigt, dass die Scripts zwar editierbar aber nicht programmierbar sind und daher war auch dieses Tool nicht für unsere Tests geeignet ist.

#### **OpenSTA**

Die Open Systems Testing Architektur, basierend auf Corba wurde für HTTP und HTTPS Performance Tests entwickelt.

Die Hautanforderung an unsere Testumgebung war es beliebig viele virtuelle User (VU's) mit realistische Verhalten simulieren und modellieren zu können, das Tool verfügt über eine eigene Scriptsprache SCL die dies ermöglicht, und ist unsere Wahl für die Performance Evaluierung der Webshops. Außerdem ist OpenSTA das einzige Tool welches DOM – Adressierung beherrscht, diese Feature benötigen wir um aus dem versteckten Formularfeld beim PhPepperShop die Kunden\_ID auslesen zu können.

### 2.3.5.4 Erfahrungen mit OpenSTA

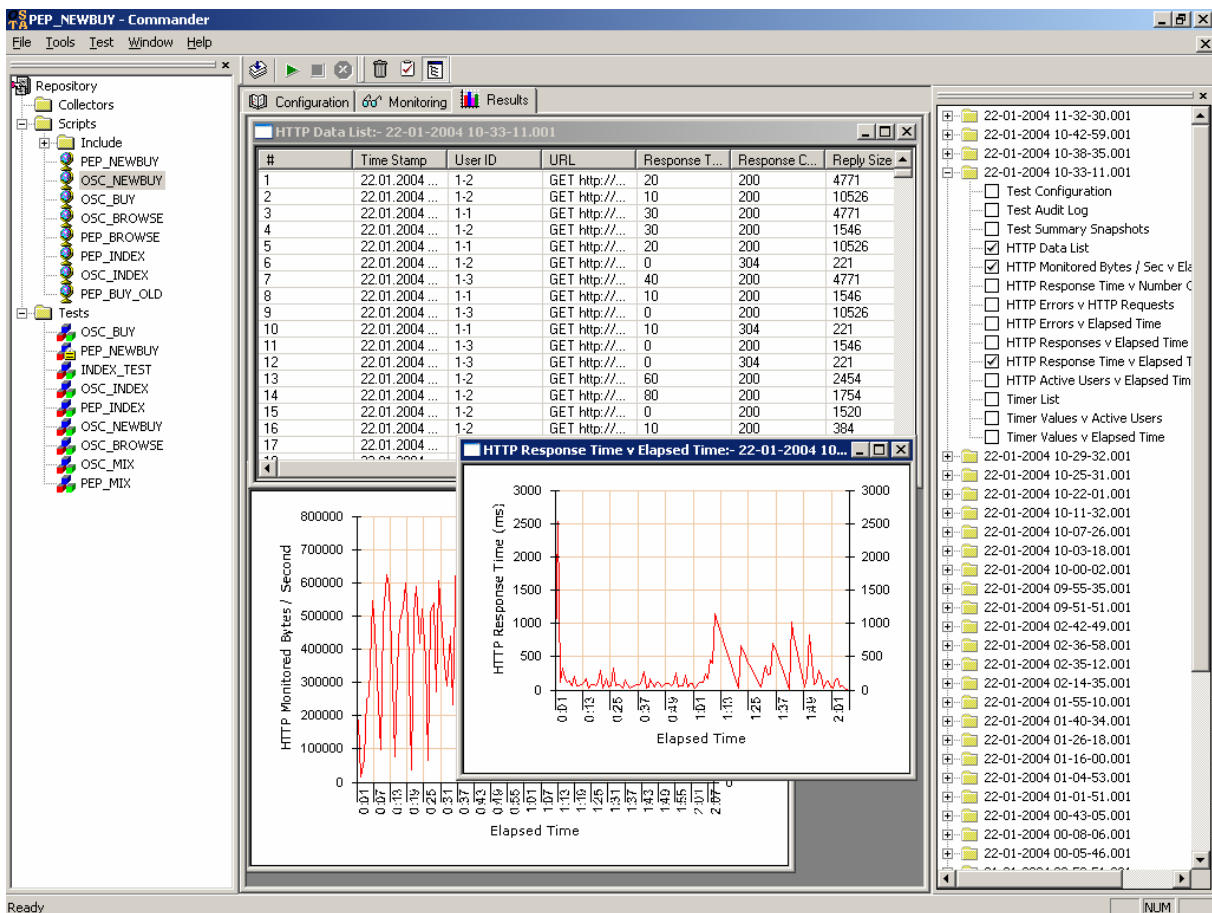
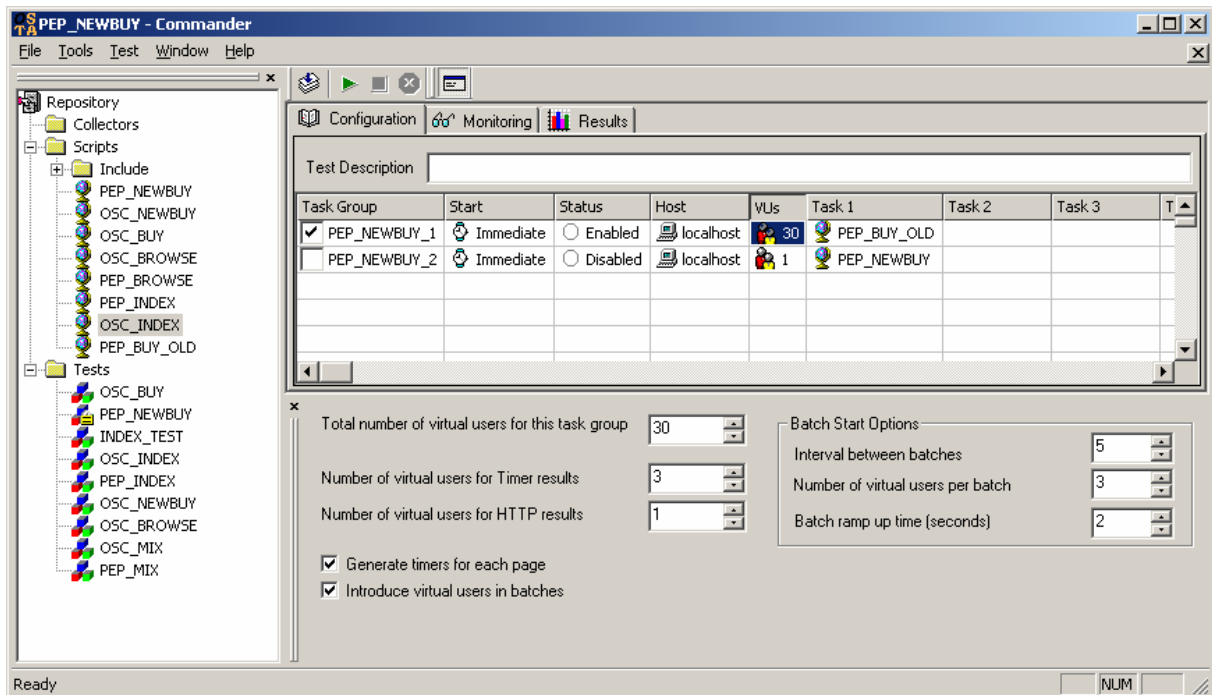
Installation ist wie bei fast allen Windowsanwendungen sehr einfach.

Um seinen ersten Loadtest durchzuführen sollte man auf jeden Fall einen Blick auf den „Getting Started Guide“ werfen. Das Tool ist im Allgemeinen mit kommerziellen Load-Testtools vergleichbar und auch ähnlich zu bedienen. Skripte aufnehmen und modellieren scheint sehr einfach zu sein, wer aber komplexere Webanwendungen wie Webshops testet muss sich intensiv mit der Modellierung der Skripte beschäftigen. Darauf wird noch ausführlich im Abschnitt Usecases eingegangen.

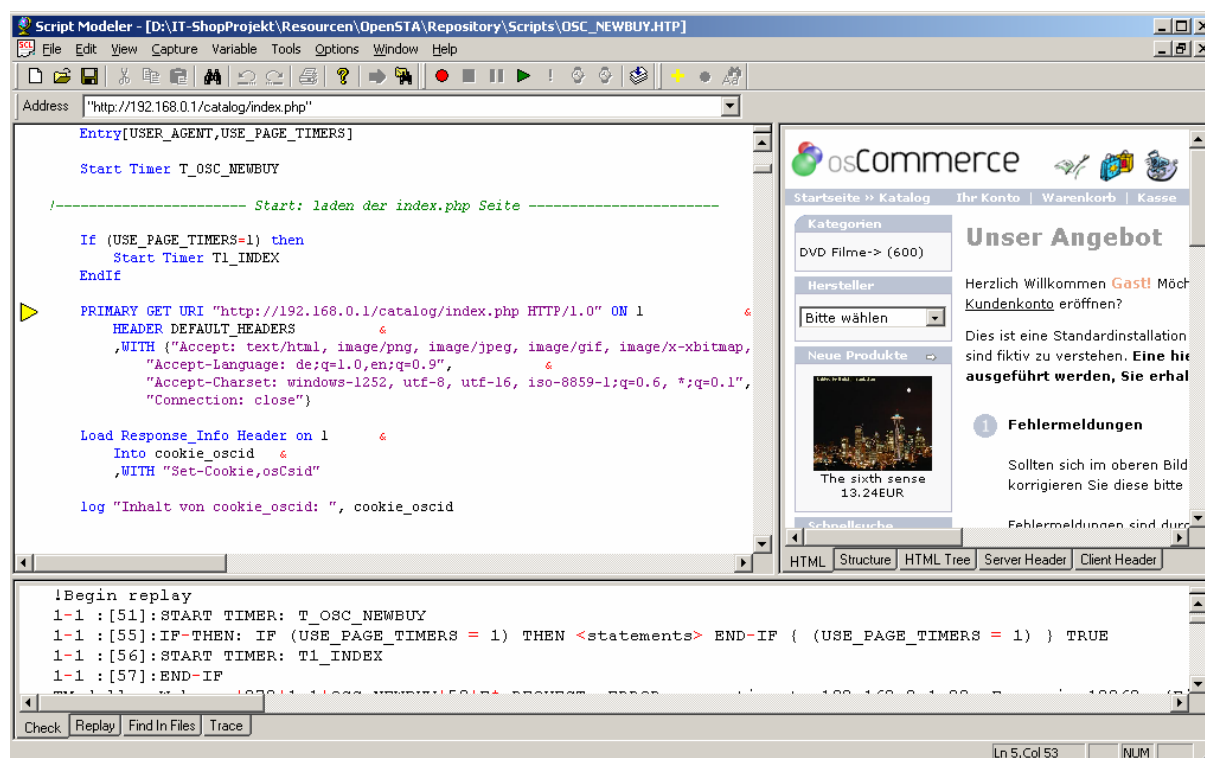
Um Tests durchführen hat man sehr viele Parameter zur Verfügung, ein einfacher Loadtest ist einfach zu erstellen, um einen möglichst realistischen Test durchzuführen muss man sich intensiv mit den geboten Möglichkeiten auseinandersetzen.

## OpenSTA besteht aus folgenden GUI Komponenten:

Der **Commander**, ist die Kernkomponente und dient zum Erstellen der Tests, dem Einstellen der Testparameter, der Durchführung der Tests und dem Analysieren der Ergebnisse.



Der **Script Modeller** steuert die Aufzeichnung der Skripte mittels Browser und dient zum Programmieren und Debuggen.



## Probleme bei der Arbeit mit OpenSTA

Beim Aufzeichnen der Skripte stürzten einige Male sowohl der vorerst verwendete Internetbrowser IE 6.0 als auch der Scriptrecorder (gateway.exe) ab. Das Problem ist aus Foren bekannt und ließ sich ausschließlich mit einer Neuinstallation von OpenSTA beheben. Beim Durchführen von Stresstests (> 70 VU) passiert es häufig, dass der Test nicht von selbst beendet, da der Testexecuter noch virtuelle User anzeigt, obwohl keine Anfragen mehr an Server gesendet werden.

Man sollte nicht versuchen, die Ergebnisdiagramme mittels [STRG]+[C] zu kopieren, da dies zum Absturz von OpenSTA führt.

Stürzt OpenSTA ab, reicht es nicht aus OpenSTA mittels Taskmanager zu beenden, da noch weitere zu OpenSTA gehörende Prozessen laufen und die gesamten Systemressourcen beanspruchen.

Bei der Deinstallation des Programms werden nicht alle Registryeinträge gelöscht, was dazu führt, dass bei einer Neuinstallation die alten Parameter eingestellt bleiben.

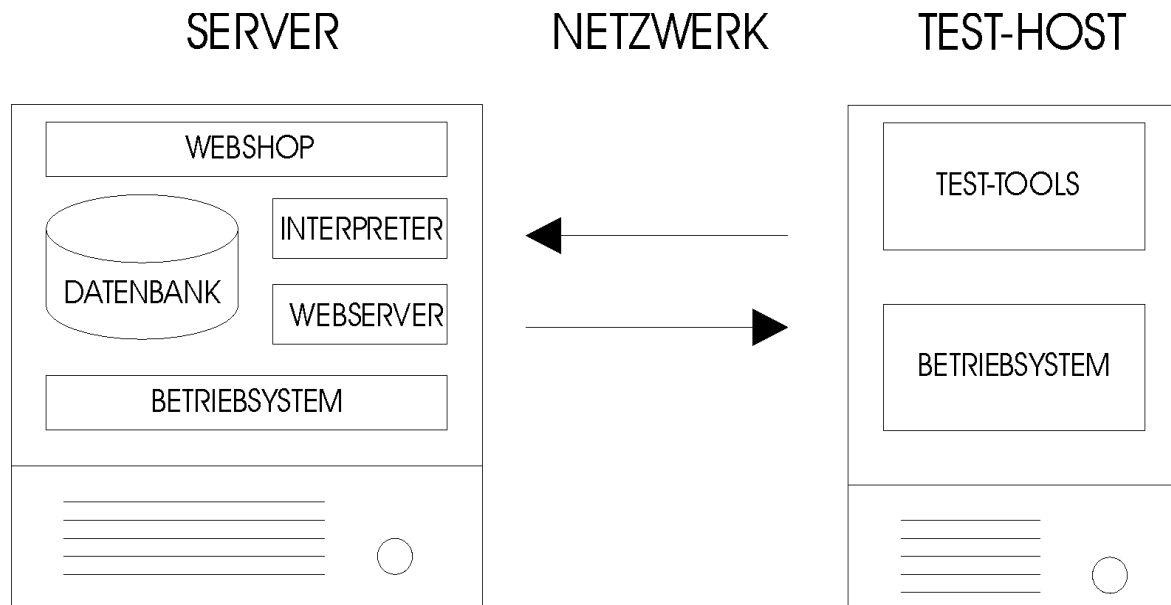
Bei Verwendung von Anti-Spy Programmen wird ein Registry-Eintrag von OpenSTA als möglicher Browser-Hijacker identifiziert. Nach irrtümlicher Entfernung des Eintrags müssen beim Aufzeichnen von Skripten die Proxy-Einstellungen des Browsers manuell durchgeführt werden.

Die Cacheeinstellungen des verwendeten Browsers können die Testergebnisse deutlich verfälschen. Beim IE 6.0 kann der Cache nicht vollständig deaktiviert werden, darum empfiehlt sich der Einsatz von Opera.

Trotz dieser Probleme ist OpenSTA eine gute Wahl für Performacetests, da kaum ein anderes Open Source Test-Tool ähnliche Funktionalität bei akzeptabler Stabilität bietet.

### 3. Setup

#### 3.1 Testkonfiguration



Hardware am Server	
CPU	1 GHz Celeron
Speicher	348 MB SDRAM
Festplatte	40 GB IDE
NIC	10/100 Mbit

Hardware am Test-Host	
CPU	1 GHz Duron
Speicher	256 MB SDRAM
Festplatte	40 GB IDE
NIC	10/100 Mbit

Software am Server	
Betriebssystem	SuSE Linux 8.2 pro
Webserver	Apache 2.0.48
Datenbank	MySQL 4.0.16
Interpreter	PHP 4.3.3
Webshop	ocCommerce 1.2
	PhPepperShop 1.4

Software am Test-Host	
Betriebssystem	Windows XP pro
Test-Tools	OpenSTA 1.4.2

Hardware im Netzwerk	
Switch	16 x 10/100 Mbit

## 3.2 Installation am Server

Basis für die XAMP-Installation ist Linux in der Form der Distribution von SuSE in der Version 8.2 professional. Es wurden nur die rudimentären Bestandteile des Betriebssystems installiert, auf unnötigen Ballast wie XServer usw. wurde bewusst verzichtet.

Apache, MySQL und PHP wurden über das Paket *XAMPP for Linux* von der Webseite <http://www.apachefriends.com> auf dem Server installiert. In diesem Paket liegen die Programme bereits als Binary vor und sind schon vorkonfiguriert. Außerdem im Paket enthalten ist auch Perl, mmCache (Cache für PHP) und ein SSL-Modul für Apache.

Abgerundet wird das ganze durch ein nettes PHP-Frontend, mit dem man eine Webstatistik und eine Übersicht über die installierten Programme und deren Haupteinstellungen hat.

### Kompletter Inhalt des XAMP-Pakets:

Programm	Version	Programm	Version	Programm	Version
Apache	2.0.48	gdbm	1.8.0	FreeTDS	0.60
MySQL	4.0.16	zlib	1.1.4	gettext	0.11.5
PHP	4.3.3	expat	1.2	IMAP C-Client	2002b
Perl	5.8.0	ming	0.2a	OpenLDAP (client lib)	2.1.22
ProFTPD	1.2.9	Sablotron	1.0	Turck MMCache	2.4.4
phpMyAdmin	2.5.4	libxml2	2.4.26	mcrypt	2.5.7
OpenSSL	0.9.7c	Webalizer	2.01	mhash	0.8.18
Freetype2	2.1.0	pdf class	009e	SQLite	2.8.6
libjpeg	6b	ncurses	5.8	cURL	7.10.7
libpng	1.2.2	mod_perl	1.99_08		

### Installation des XAMP-Pakets

Anmeldung als Superuser  
 Entpacken des Pakets  
 Ausführen des Installscripts  
 Starten der Server mittels  
`/opt/lampp/lamp start`

### Installation von osCommerce

Hier reicht das Entpacken der Dateien in das gewünschte Verzeichnis und dem Aufruf des Install-Skripts über den Browser mittels `[Host-IP]/catalog/install`  
 Über das Webinterface können alle Einstellungen bezüglich Shoppparameter und Datenbankbindung getätigt werden. Nach erfolgreicher Installation ist das Installationsverzeichnis zu löschen, bzw. umzubenennen und die Konfigurationsdateien mit den entsprechenden Rechten zu schützen.  
 Eine ausführliche Installationsanleitung ist im Paket enthalten.

### Installation von PhPepperShop

Die Installation gestaltet sich einfach: es wird ein Perl-Skript gestartet, das Anfragen über den Installationsordner, Webshopnamen, Usernamen und dergleichen macht und PhPepperShop dementsprechend installiert.

Achtung: Bei Verwendung des XAMP-Pakets findet das Installationsskript den MySQL-Client nicht automatisch. Hier muss der Pfad /opt/lampp/bin manuell angegeben werden. Auch hier ist eine ausführliche Installationsanleitung im Paket enthalten.

### Anpassen der Shops

Bei beiden Webshops wurde für die Artikelübersicht die Anzahl der angezeigten Artikel pro Seite auf zehn eingestellt. Bei osCommerce wurde die Banner-Werbung und das „Neue Produkte“-Features deaktiviert. Die Zweitwährung „Dollar“ und die Sprache „Spanisch“ wurden gelöscht, „Englisch“ konnte nicht entfernt werden, da dies sonst zu Fehlfunktionen im Shop geführt hätte. Dadurch wurde erreicht, dass die Shopsystem einigermaßen vergleichbar wurden.

## 3.3 Installation am Test-Host

Als Testtool wurde **OpenSTA 1.4.2** gewählt, welches nur für WindowsNT/2000/XP lauffähig ist. Um die Arbeiten zwischen Server und Test-Host zu erleichtern, wurden noch zusätzliche Tools installiert.

Nachstehend die Liste mit den verwendeten Programmen:

Programm	Version	Beschreibung
OpenSTA	1.4.2	Testtool
PuTTY	0.53b	Telnet / SSH – Client
MySQL-Front	3.0	Grafisches MySQL Frontend mit Im- und Exportfunktionen
WS FTP LE	5.08	FTP-Client mit grafischer Oberfläche
Batch Image Editor	2.0 Trial	Tool zum Konvertieren von Produktbildern

## 3.4 Erzeugen einer Referenzdatenbank

Um die Test erst überhaupt zu ermöglichen, mussten die Datenbanken von osCommerce und PhPepperShop mit identischen Artikeldaten gefüllt werden. Es wurden 20 Artikel mit verschiedenen Namen, Beschreibungen, Preisen und Artikelbildern erzeugt. Die Artikelbilder sind über einen Pfad-Eintrag in der Artikeltabelle mit dem jeweiligen Artikel verknüpft. Aus diesen 20 Artikeln wurden dann 600 Mutationen geschaffen. Diese wurden dann ungleichmäßig in sechs Kategorien aufgeteilt.

Kategorien
<b>DVD Filme-&gt;</b> (600)
Action (20)
Drama (300)
Komödie (60)
Science Fiction (40)
Thriller (100)
Zeichentrick (80)

### Import in osCommerce

osCommerce enthält kein Importtool, deshalb wurden die Artikeldaten manuell über MySQLFront (ein grafisches Freeware Frontend für MySQL) per CSV-Datei in die osCommerce-Datenbank eingespielt. Die Artikelbilder wurden über einen WS\_FTP (ein grafischer FTP-Client) in das entsprechende Verzeichnis hochgeladen.

Um ein Produkt erfolgreich anzulegen, müssen Daten in mindestens drei Tabellen eingetragen werden: products, products\_descriptions, products\_to\_categories. Wird dies nicht gemacht, wird durch einen Fehler der Produktpreis aller Produkte auf 0 Euro gesetzt. Dieser Fehler kann nur durch wiederherstellen der kompletten Datenbank wieder behoben werden.

### **Import in PhPepperShop**

Die Artikeldaten wurden über das Importtool von PhPepperShop per CSV-Datei problemlos eingespielt, die Artikelbilder über FTP in das entsprechende Zeichnis hochgeladen. Bei den Artikelbildern ist zu beachten, dass PhPepperShop von jedem Bild immer zwei Versionen davon verwendet. Einmal das originale Artikelbild, welches in der Detailansicht angezeigt wird, und einmal ein auf eine festgelegte Größe verkleinertes Artikelbild (Thumbnail), welches in der Listenansicht verwendet wird. Das verkleinerte Artikelbild wird normalerweise von PhPepperShop selbst erzeugt, wenn ein neuer Artikel angelegt wird. In unserem Fall erzeugten wir diese Bilder mit Hilfe des Programms Batch Imge Editor. Unterschieden werden die beiden Versionen eines Bildes durch den Appendix `_gr` (für groß) und `_kl` (für klein).

## **3.5 Optimierungsmöglichkeiten**

Die Liste der hier aufgezeigten Möglichkeiten ist bei weitem nicht vollständig. Aufgrund der Komplexität der einzelnen Komponenten eines XAMP-Systems wäre dies auch ein sehr aufwendiges Unterfangen. Oft sind die Optimierungsmöglichkeiten auch nicht durchführbar, da kein dezidiertes Server verwendet wird, sondern ein Angebot eines Webhosters. Hier ist die Hardware vorgegeben und auch viele Einstellungen am Betriebssystem, Webserver, PHP und Datenbank sind fest vorgegeben und nicht konfigurierbar. Einzig am Webshop sind dann Tuningmaßnahmen möglich.

### **3.5.1 Optimierung an der Hardware**

#### **CPU**

Das Kompilieren der PHP-Skripten beansprucht sehr viel CPU-Zeit. Zwar kann man durch das Cachen der kompilierten Skripten viel an Rechenleistung einsparen, trotzdem bleibt der Prozessor bei stark PHP-lastigen Seiten der Flaschenhals. MySQL benötigt je nach Seitenaufbau in etwa gleich viel Rechenzeit wie PHP. Läuft die Datenbank auf dem gleichen Server, erhöht das die CPU-Belastung abermals.

#### **Speicher**

Pro Apache/PHP-Prozess werden etwa 3MB RAM benötigt, ein MySQL-Prozess benötigt etwa 2 MB RAM. Der Speicher daher vor allem bei vielen parallelen Zugriffen zum Problem. Fazit: Speicher kann man nie genug haben.

#### **Festplatte**

Ein Webserver verbringt einen Großteil seiner Zeit mit dem Zugriff auf die Festplatten, und da diese auch heute noch relativ langsam sind, stellen sie oftmals die Hauptursache für einen Performanceverlust da. Hier sind vor allem SCSI-Systemen den Vorzug zu geben. Eine weitere Verbesserung auch in Richtung Datensicherheit stellen dann RAID-Systeme mit entsprechenden Cache-Controllern da. Bei Highperformance-Anforderungen sollten SAN-Lösungen in Erwägung gezogen werden.

#### **Netzwerk**

Der Flaschenhals bei rein statischen Seiten. Möglichkeiten zur Reduzierung der Datenmenge sind: Caching und Output-Kompression der Daten. Letzteres reduziert den Traffic um bis zu 90%. Die zusätzliche Komprimierung wirkt sich allerdings wieder negativ auf die CPU-Belastung aus.

### **Einsatz mehrerer Server**

Eine mögliche Variante wäre die Aufteilung von Webserver und Datenbank auf verschiedene Server, die natürlich über eine entsprechende Netzwerkverbindung verfügen müssen. Ebenso könnten statische Webseiten von einem anderen Server bedient werden. Auch ein Proxy-Caching Server wie z.B. *Squid* würden den Durchsatz erhöhen.

## **3.5.2 Optimierung an der Software**

### **Betriebssystem (Linux)**

Basis sollte eine Installation ohne jeglichen Overhead an unnötigen Prozessen sein. Vor allem eben ohne grafischen System wie ein Windowmanager oder Xserver. Man sollte auch überlegen, ob ein FTP-Server benötigt wird oder auch die ganzen kleinen Tools, die normalerweise ganz praktisch sind, aber bei reinem Server-Betrieb unnötig sind. Mögliche wäre auch eine Beschleunigung des Festplattenzugriffs mittels `hdparm`.

### **Webserver (Apache)**

Alle Einstellungen von Apache werden in der Datei `httpd.conf` gespeichert. Optimierungsmöglichkeiten wären etwa das deaktivieren von unnötigen Modulen, eine Erhöhung der `SendBufferSize` auf die Größe der größten verarbeiteten Web Seite. Eine andere Möglichkeit wäre die Umleitung von statischen Seiten auf einen anderen Webserver wie z.B. *Tux* oder *httpd*, welche hier effizienter arbeiten. Eine Reduktion der Datenmenge ist durch Aktivierung der Output-Compression unter des Apache-Modul `mod_gzip` möglich.

### **MySQL**

Die gravierendsten Geschwindigkeitsoptimierungen sind hier nicht an der Datenbank selbst, sondern am Datenbankclient zu erzielen. Eine Optimierung der SQL-Abfragen an MySQL bringt bei komplexeren Datenbankstrukturen oft gravierende Geschwindigkeitsunterschiede (siehe Performanceanalyse der Artikeldarstellung im PhPepperShop).

### **PHP**

PHP ist vor allem deshalb langsam, da es eine Interpretersprache ist. Das ständige kompilieren der Skripten benötigt sehr viele CPU-Ressourcen. Hier gibt es die Möglichkeit die bereits kompilierten Skripten zu cachen. Programme wie z.B. *Turck mmCache* bringen je nach Komplexität der Web-Seiten Geschwindigkeitszuwächse zwischen 25 bis 400 Prozent. Eine andere Optimierung aus der Sicht des Webshop-Besuchers wäre die Kompression der HTML-Daten mittels `ob_gzhandler`. Der Flaschenhals bei den meisten Benutzern ist noch immer das Netzwerk (Modem-Benutzer). Die Komprimierung kann den Download der Webshop-Seiten stark beschleunigen. Nachteilig ist natürlich wieder die zusätzliche CPU-Belastung auf Seite des Servers

### **Webshop**

Hier sind die größten Performance-Gewinne sowie auch Verluste möglich. Die beste Hardware hilft nichts, wenn die Algorithmen und SQL-Abfragen des Webshops ineffizient sind.

Auf Seiten des Webshop-Administrators ist der grafische Aufbau des Webshops wichtig. Mit großen Bildern überladene Startseiten z.B. trüben durch lange Ladezeiten den ersten Eindruck des Besuchers. Wichtig ist auch eine effiziente Ansicht der Artikellisten durch Beschränkung der angezeigten Artikel pro Seite.

## 4. Test

### 4.1 Grundsatz-Überlegung

Unser Hauptziel ist die Messung des Durchsatzes in Form von Einkaufstransaktionen pro Minute mittels OpenSTA. Es wurden bei beiden Shops identische Produkte angelegt. Das Userprofil wurde unter Use-Cases definiert. Das „System under Test“ ist der Serverrechner auf dem die Webshopsysteme laufen und die 100Mbit Netzwerkverbindung zu unserem Test-Host, wobei die Netzwerkverbindung nur bei statischen Inhalten berücksichtigt werden muss. Die Antwortzeit die unser Testtool misst setzt sich also aus Datenübertragungszeit des Netzwerks, Serverantwortzeit und erneuter Datenübertragungszeit über das Netzwerk zusammen. Das ist genau jene Zeit die ein Benutzer auf die angeforderte Seite wartet. Leider ist es mit OpenSTA nicht möglich eine reduzierte Bandbreite wie sie z.B. ein Modembenutzer zur Verfügung hat zu simulieren. Für eine Performance Analyse ist weiters eine realistische Testumgebung (Livesystem) erforderlich, dies ist bei uns nicht der Fall darum bezeichnen wir unsere Arbeit auch als Performance Evaluierung und nicht als Performance Analyse. Durch Tests wurde das Optimierungspotential mittels Turck mmCache ermittelt, basierend auf den Ergebnissen wurde diese Optimierung durchgeführt.

### 4.1.1 Use-Cases

Als Use-cases wurden folgende Aktivitäten der Benutzer ermittelt:

Skript Name	Beschreibung	Schritte	# der Seiten	# der GET's	Skriptzeit
OSC_INDEX	Shop-Startseite ansehen	1. Startseite öffnen	1	21	1,95
PEP_INDEX			1	13	1,31
OSC_BROWSE	Stöbern	1. Startseite öffnen 2. Hauptkategorie <i>DVD</i> öffnen 3. Unterkategorie Komödie öffnen 4. Produkt <i>A.I.</i> auswählen 5. großes Produktbild ansehen und schließen 6. neue Unterkategorie <i>Thriller</i> auswählen 7. Seite 4 auswählen 8. Produkt <i>Heartbreakers</i> auswählen 9. großes Produktbild von <i>Heartbreakers</i> ansehen	9	62	45,09
PEP_BROWSE		1. Startseite öffnen 2. Hauptkategorie <i>DVD</i> öffnen 3. Unterkategorie Komödie öffnen 4. großes Produktbild von <i>A.I.</i> ansehen und schließen 5. neue Unterkategorie <i>Thriller</i> auswählen 6. Seite 4 auswählen 7. großes Produktbild von <i>Heartbreakers</i> ansehen	7	45	27,49
OSC_BUY	Einkaufen (registrierter Kunde)	1. Startseite öffnen 2. Hauptkategorie <i>DVD</i> öffnen 3. Unterkategorie <i>Science Fiction</i> öffnen 4. Produkt <i>Epoch</i> in den Warenkorb legen 5. Kassa 6. Anmelden 7. Versandoptionen mit Weiter bestätigen 8. Zahlungsweise Nachnahme auswählen und bestätigen 9. Bestellung bestätigen	10	81	36,55

Skript Name	Beschreibung	Schritte	# der Seiten	# der GET's	Skriptzeit
OSC_NEWBUY	Einkaufen (Neukunde)	<ol style="list-style-type: none"> <li>1. Startseite öffnen</li> <li>2. Hauptkategorie <i>DVD</i> öffnen</li> <li>3. Unterkategorie <i>Science Fiction</i> öffnen</li> <li>4. Produkt <i>Epoch</i> in den Warenkorb legen</li> <li>5. Kassa</li> <li>6. Neukundenanmelden</li> <li>7. Feedbackseite „Anmeldung erfolgreich“</li> <li>8. Versandoptionen mit Weiter bestätigen</li> <li>9. Zahlungsweise <i>Nachnahme</i> auswählen und bestätigen</li> <li>10. Bestellung bestätigen</li> </ol>	11	81	126,21
PEP_NEWBUY		<ol style="list-style-type: none"> <li>1. Startseite öffnen</li> <li>2. Hauptkategorie <i>DVD</i> öffnen</li> <li>3. Unterkategorie <i>Science Fiction</i> öffnen</li> <li>4. Produkt <i>Epoch</i> in den Warenkorb legen</li> <li>5. Kassa</li> <li>6. Anmelden</li> <li>7. Kunden- und Versanddaten eintragen, Zahlungsweise <i>Nachnahme</i> auswählen und bestätigen</li> <li>8. Bestelldaten anzeigen und bestätigen</li> <li>9. AGB akzeptieren und bestätigen</li> </ol>	9	74	74,02

Nach Analyse der Use-Cases stellten wir fest, dass alle Skripte eine ähnliche Struktur haben:

- Startseite
- Navigieren
- Warenkorb
- (Neukunden-) Anmeldung
- Einkaufsabwicklung

Da die Skripte OSC\_NEWBUY und PEP\_NEWBUY fast alle Aktivitäten der anderen Skripte enthalten verwenden wir ausschließlich diese für unsere Performance Evaluierung.

## 4.2 Testerstellung

### 4.2.1 Erstellen von Skripten zur Simulation des Userverhaltens

Zuerst wurde das Benutzerverhalten mittels Proxy (von OpenSTA) und Internet Browser Opera aufgezeichnet. Das daraus entstandene SCL – Script wurde folgender Maßen modelliert.

- auskommentieren alle WAIT – Befehle
- ersetzen des Cookiewerts durch die Cookievariable
- Benutzerdaten und Produkt parametrisiert
- bei PhPepperShop wird die Kunden\_ID mittels DOM-Adressierung ausgelesen und für die Bestellabwicklung gespeichert
- hinzufügen von Log-Messages für Analysezwecke
- Timer sinnvolle Bezeichnungen geben und entsprechend Positionieren

### 4.2.2 Erstellen der Tests

Aus den modellierten Skripten werden Tests erstellt. Die Virtuellen User werden innerhalb von 2s gestartet, dies entspricht am ehesten einem realen Userverhalten, wenn z.B. ein Angebot gemacht wird interessieren sich innerhalb kurzer Zeit sehr viele Kunden für dieses und kaufen dieses. Das alle virtuellen Kunden wirklich gleichzeitig (innerhalb von 10 ms) den Einkauf beginnen und immer exakt zur gleichen Zeit Ihre Anfragen stellen ist unrealistisch und wird deswegen von uns nicht getestet.

## 4.3 Durchführung der Tests

Für jeden einzelnen Test wurden folgende Schritte durchgeführt:

- Einstellen der entsprechenden Testparameter
- Leeren der Datenbank
- Beobachtung der Server-Last mittels `top` bzw. `vmstat` über `ssh`
- Monitoring der Active Users, HTTP Errors und Notes während des Tests mit OpenSTA
- Überprüfung der Datenbank-Einträge
- Auswertung der Ergebnisse
  - Response Time
  - Timer List
  - HTTP Errors
  - Datendurchsatz
  - Durchschnittliche Response time
  - Datendurchsatz und durchschnittliche Response time werden mittels Tabellenkalkulation aus der HTTP Data List errechnet

## 5. Interpretation der Ergebnisse

### 5.1 Vergleich von OpenSTA und ab (apache bench)

Da wir überprüfen wollten wie exakt die Messungen des wesentlich komplexeren OpenSTA sind, verglichen wir die Ergebnisse mit dem Referenz-Benchmarktool ab (apache bench).

#### Testkonfiguration:

Iterationen: 200  
parallele User: 1

Dokumentenpfad: <http://192.168.0.1/catalog/index.php>  
Beschreibung: komplexe php-Seite mit Datenbankzugriffen  
Dokumentgröße: 24,14 kbytes

	Ohne Cache		Mit mmCache 2.4.4	
	OpenSTA	ab	OpenSTA	ab
Mittlere Response time [ms]	240,52	240,45	152,05	140,20
Requests / s	4,16	4,16	6,58	7,13

Dokumentenpfad: <http://192.168.0.1/pepper/index.php>  
Beschreibung: enthält ausschließlich echo-Befehle  
Dokumentgröße: 4,25 kbytes

	Ohne Cache		Mit mmCache 2.4.4	
	OpenSTA	ab	OpenSTA	ab
Mittlere Response time [ms]	110,48	110,36	15,15	13,97
Requests / s	9,05	9,06	66,01	71,58

#### Beobachtung:

Die OpenSTA ist minimal langsamer, dies liegt daran weil im Skript bereits das Cookie extrahiert wird.

Der Einsatz von mmCache oder ähnliche Produkten ist sehr empfehlenswert.

Für alle weiteren Tests ist mmCache aktiviert!

## 5.2 Ermittlung der max. Einkaufstransaktionen

### 5.2.1 Vorbereitungen

Vor jedem Testlauf sind bestehende Kunden- und Bestelungsdaten zu löschen.  
(SQL-Script: Ressourcen/clean\_shops.sql)

osCommerce:

```
DELETE FROM `osc`.`address_book`;
DELETE FROM `osc`.`customers`;
DELETE FROM `osc`.`customers_basket`;
DELETE FROM `osc`.`customers_info`;
DELETE FROM `osc`.`orders`;
DELETE FROM `osc`.`orders_products`;
DELETE FROM `osc`.`orders_status_history`;
DELETE FROM `osc`.`orders_total`;
```

PhPepperShop:

```
DELETE FROM `pepper`.`artikel_bestellung`;
DELETE FROM `pepper`.`bestellung`;
DELETE FROM `pepper`.`bestellung_kunde`;
DELETE FROM `pepper`.`kunde`;
```

Aus den modellierten Skripten wurden die WAIT- Befehle entfernt.

Eine Transaktion besteht aus der Auswahl eines Produkts, dem Anlegen des Kundenkontos und dem Kauf des Produkts.

Die mittlere Response time und die empfangenen Daten werden aus der HTTP-Datalist von OpenSTA errechnet.

Werden mehrere VU's exakt zur gleichen Zeit gestartet, entsteht ein unrealistisches Lastverhalten. Bei gleichzeitigen Start der VU's kommt es zu einer sehr hohen max.

Response time [34s@4 VU's bei osCommerce], die durchschnittliche Response time bleibt jedoch unverändert.

Darum werden die VU's innerhalb von zwei Sekunden gestartet (Rampe).

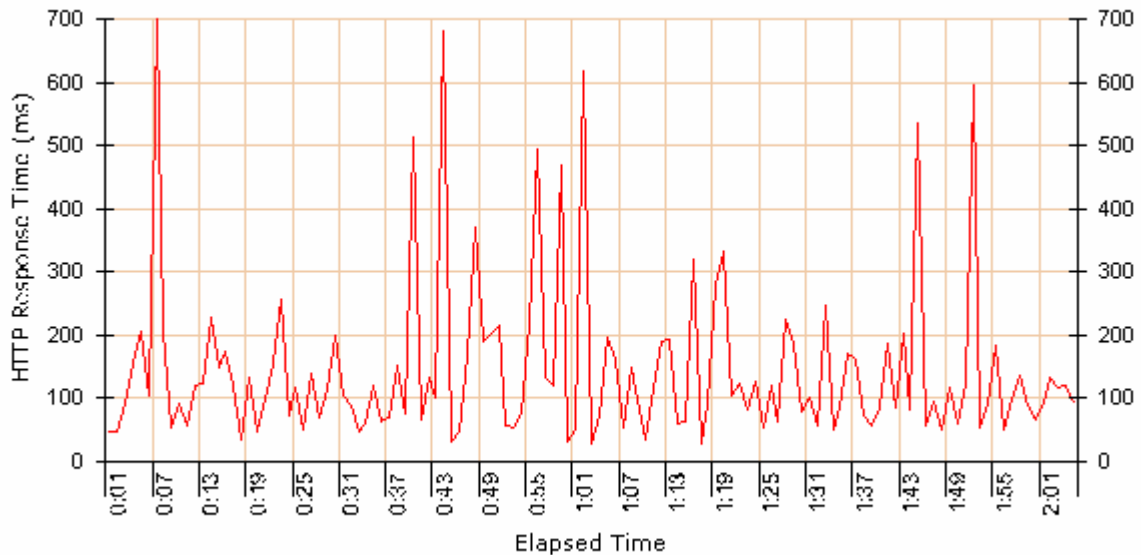
### 5.2.2 Ermittlung der max. Einkaufstransaktionen bei osCommerce

Test: OSC\_NEWBUY  
 Testzeit: 120s  
 Single Task Time: 2,58s  
 Single Task Data: 612 Kbytes  
 Requests: 81

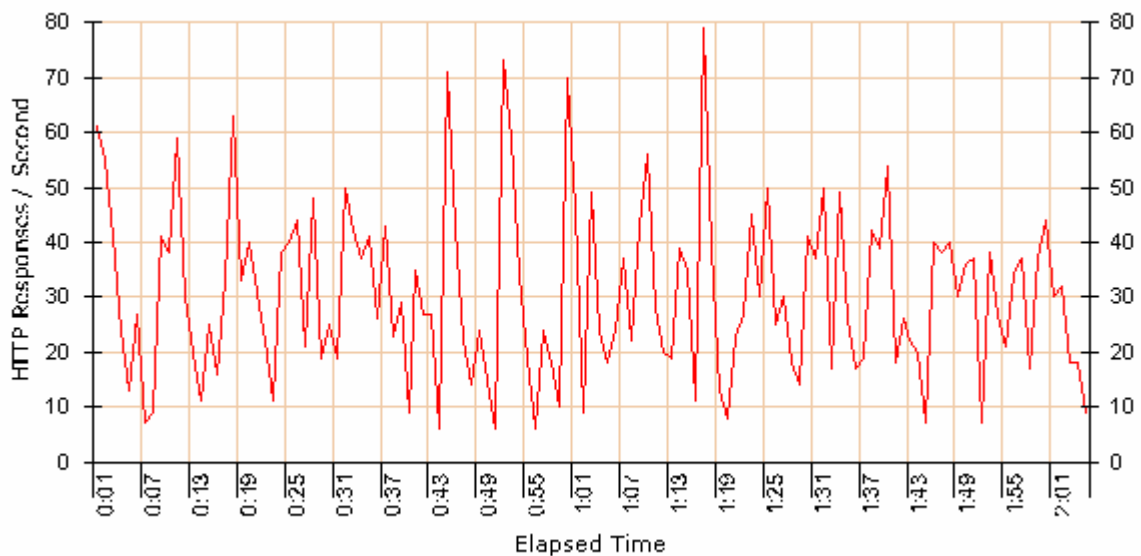
VU's	1	2	3	4	5	10	20	50
Transaktionen / min	23,5	25	27,5	27	27	26,5	25,5	26
avg. Response time [ms]	32,7	58,4	84,7	114,1	147,2	319,4	673,2	2074,7
max. Response time [ms]	400	951	3735	5848	6324	13319	19718	62039
Errors	0	0	0	0	0	0	0	0
empf. Daten [kbytes/s]	240,0	270,6	280,8	275,7	275,7	275,7	260,1	263,9

Die CPU ist bereits ab drei parallelen VU's voll ausgelastet. Hier erreicht der Server auch die höchste Transaktionsrate mit 27,5 Trans./min. Das bedeutet, das theoretisch 55 Neukunden in zwei Minuten einen Einkauf über osCommerce tätigen können. Die Transaktionsrate ist über einen weiten Lastbereich relativ konstant, die Response time hingegen steigt linear mit der Last.

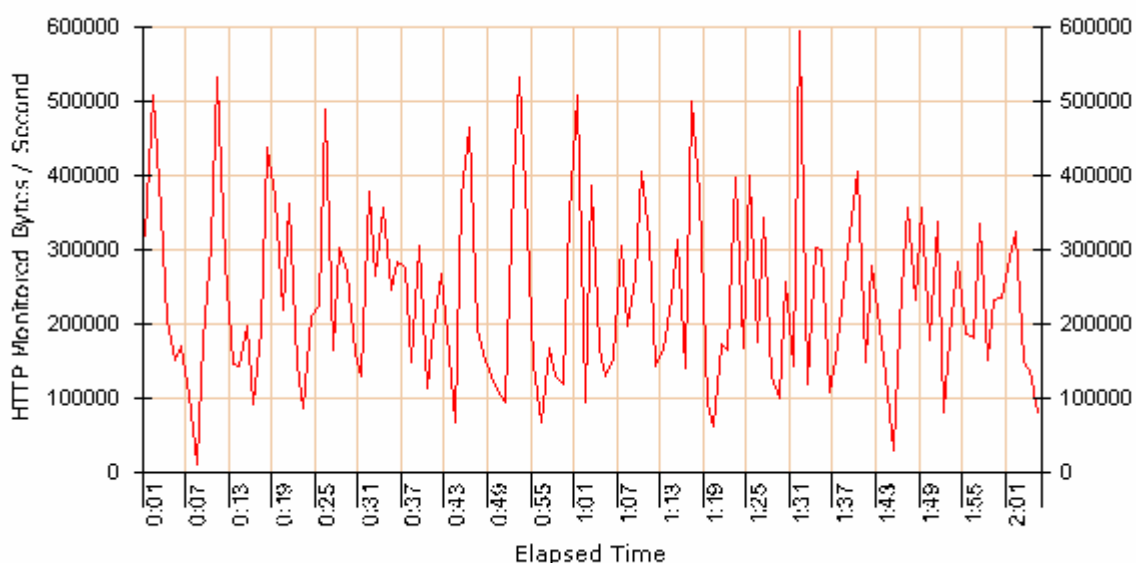
### Response Time bei max. Performance



### Anzahl der Responses pro Sekunde bei max. Performance



## Datendurchsatz bei maximaler Performance (3 VU)



## 5.2.3 Ermittlung der max. Einkaufstransaktionen bei PhPepperShop

Test: PEP\_NEWBUY  
 Testzeit: 120s  
 Single Task Time: 0,82s  
 Single Task Data: 268 Kbytes  
 Requests: 74

VU's	1	5	9	10	15	20	30	70
Transaktionen / min	72	81	89	84	77,5	78	64	28,5
avg. Response time [ms]	16,5	64,2	104,6	120,8	205,0	287,9	791,1	-1
max. Response time [ms]	961	5507	7200	7580	11977	19307	21118	-
Errors	0	0	0	0	0	0	0	234
empf. Daten [kbytes]	321,8	362,0	397,7	375,4	346,3	348,6	274,0	-

Anmerkung zu Test mit 70 VU's:

Dieser Test soll eine Überlastung des Webservers simulieren.

Dadurch kam es zu folgenden Fehlern:

In der Benutzerdatenbank wurden vier Benutzer mit leeren Datenfeldern angelegt, welche auch keine Bestellungen durchführen konnten.

Im Error-Log der Testumgebung traten 234 Fehler auf, davon:

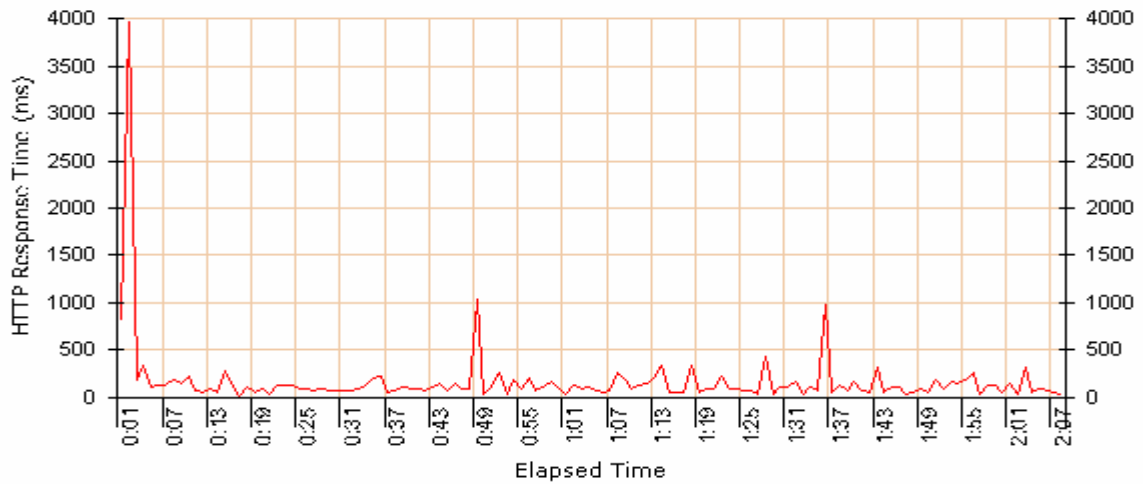
136 x ERROR 10038: (Ein Vorgang bezog sich auf ein Objekt das kein Socket ist.)

98 x ERROR 10060: (Ein Verbindungsversuch ist fehlgeschlagen)

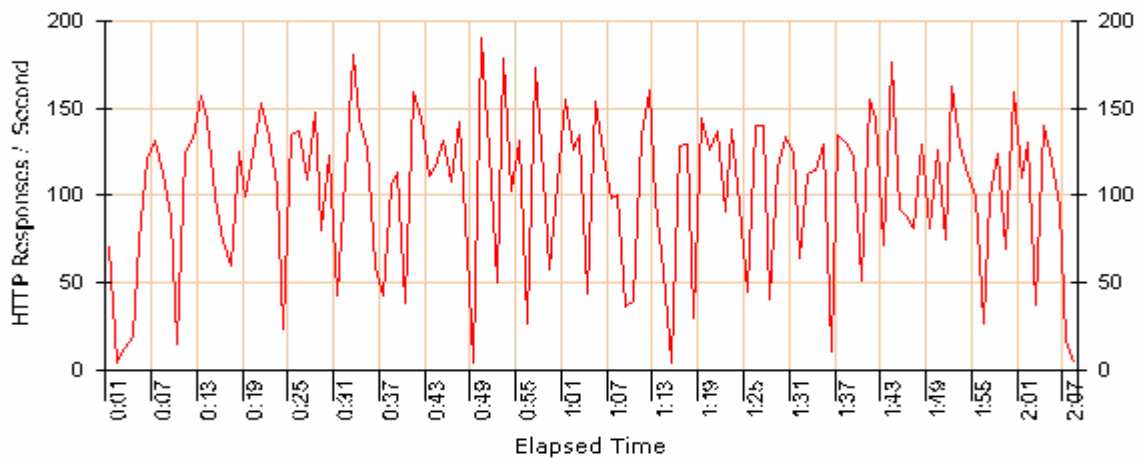
Diese Fehler sind auf die Überlast des Servers durch die hohen Benutzeranzahl und den daraus resultierenden Server timeouts zurückzuführen.

<sup>1</sup> Test abgebrochen

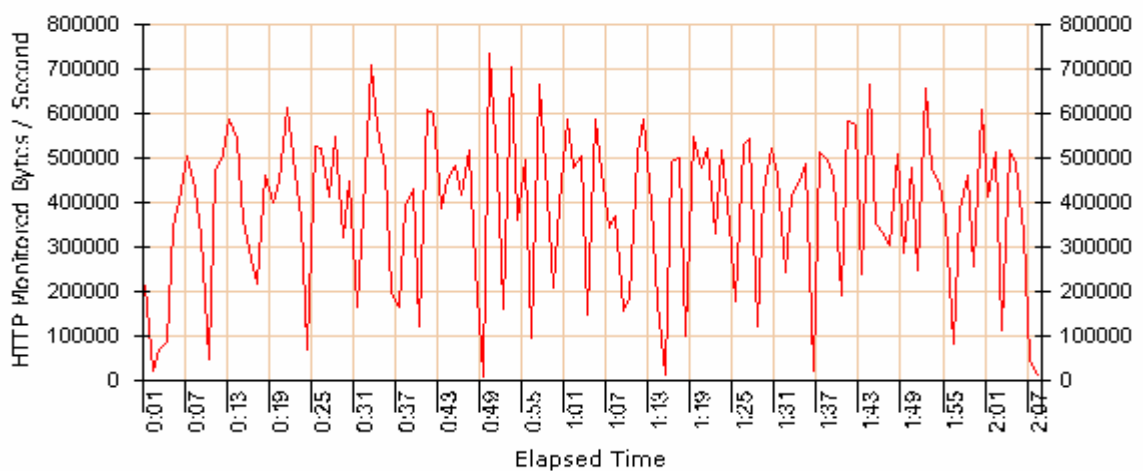
## Response Time bei max. Performance



## Anzahl der Responses pro Sekunde bei max. Performance



## Datendurchsatz bei maximaler Performance

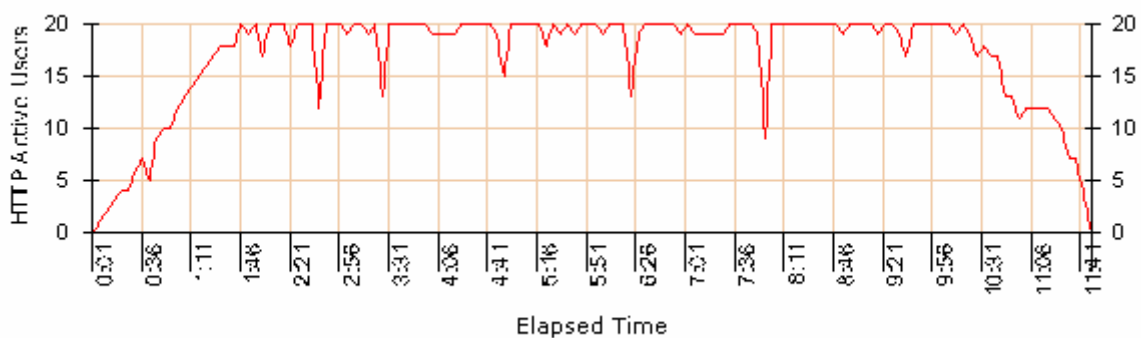


## 5.3 Lasttests

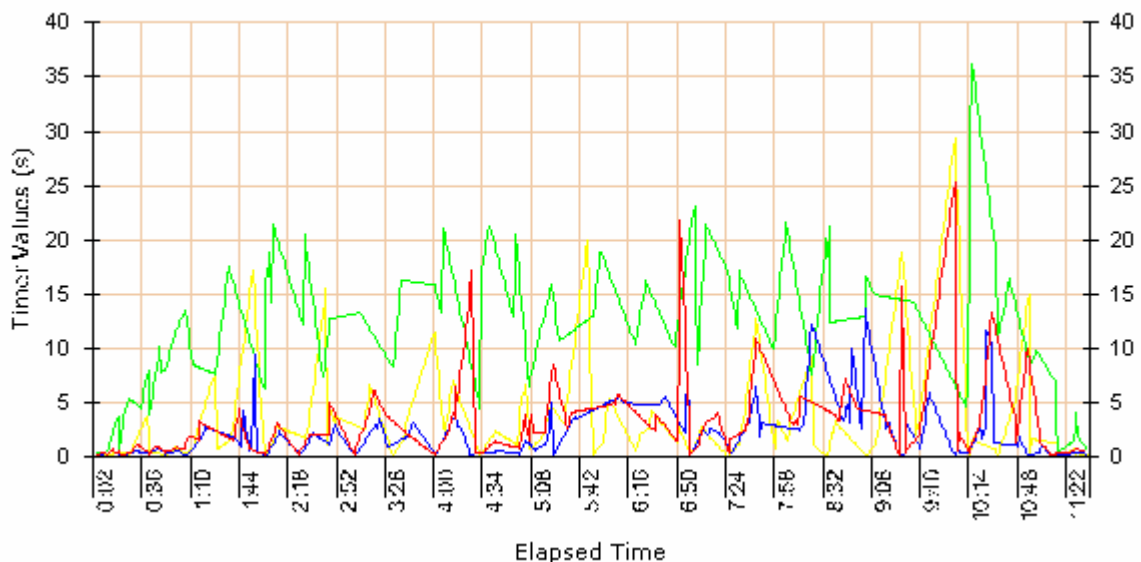
### 5.3.1 Lasttest bei osCommerce

Test: OSC\_NEWBUY  
 Testzeit: 600s  
 Single Task Time: 2,58s  
 Single Task Data: 612 Kbytes  
 VU's: 20  
 Iterationsverzögerung: variable (1 bis 4s)

Anzahl der aktiven Benutzer



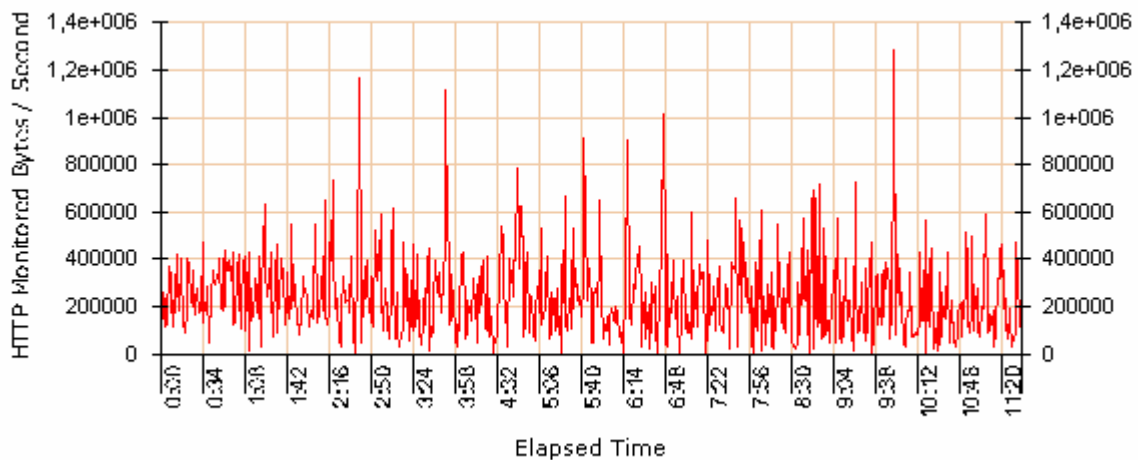
Timer für den Zahlungsprozess:



rot ..... Auswahl der Versandart  
 blau .... Auswahl der Zahlungsweise  
 grün.... Bestätigung und abschicken der Bestellung (zeitkritisch)  
 gelb .... Bestätigung der erfolgreichen Bestellung (vom Shop)

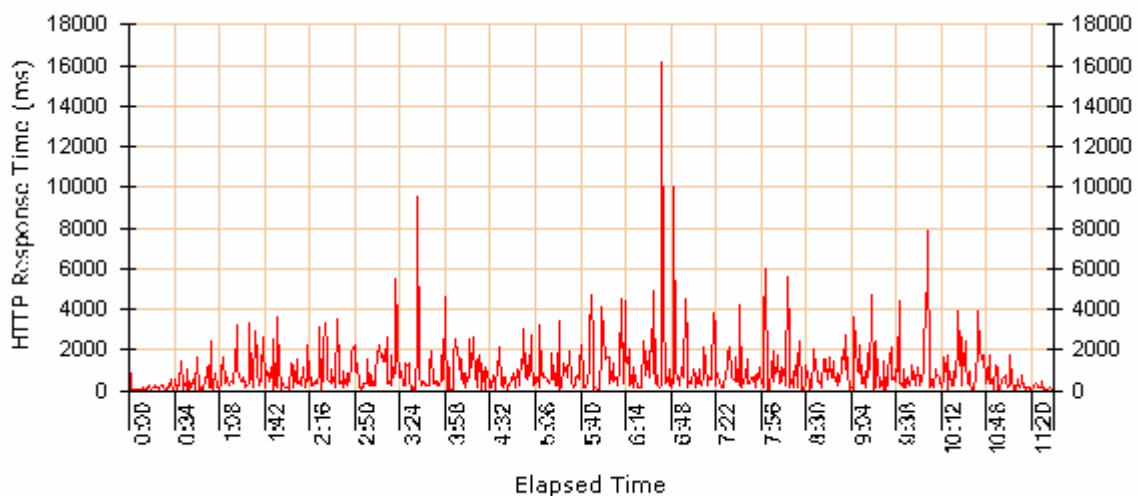
Hier ist deutlich zu erkennen, dass die einzelnen Seitenladezeiten bereits sehr häufig die 10s Grenze für akzeptable Ladezeiten überschreiten. Man könnte Vermuten, dass ein realer Onlinekunde bei solchen Antwortzeiten den Einkauf mit großer Wahrscheinlichkeit abbrechen würde. Betrachtet man jedoch die durchschnittliche Antwortzeit über den gesamten Einkauf (931,7ms) so erkennt man, dass diese Antwortzeiten durchaus vertretbar sind. Durch „Test surfen“ mit einem Browser (Opera und IE 6) wurden die Ladezeiten überprüft, der Webshop zeigt keine Fehler (wie ungeladene Bilder oder mySQL Fehler) nach eigenem Empfinden (als langjähriger Analog-Modem Benutzer) ist das Antwortzeitverhalten durchaus noch akzeptabel. Man muss sich aber Bewusst sein, dass diese Antwortzeiten in der Realität noch mit der Datenübertragungszeit addiert werden müssen.

### Empfangene Daten pro Sekunde



Mittlere Datenübertragungsrate: 236,2 kbyte/s

### Antwortzeitverhalten

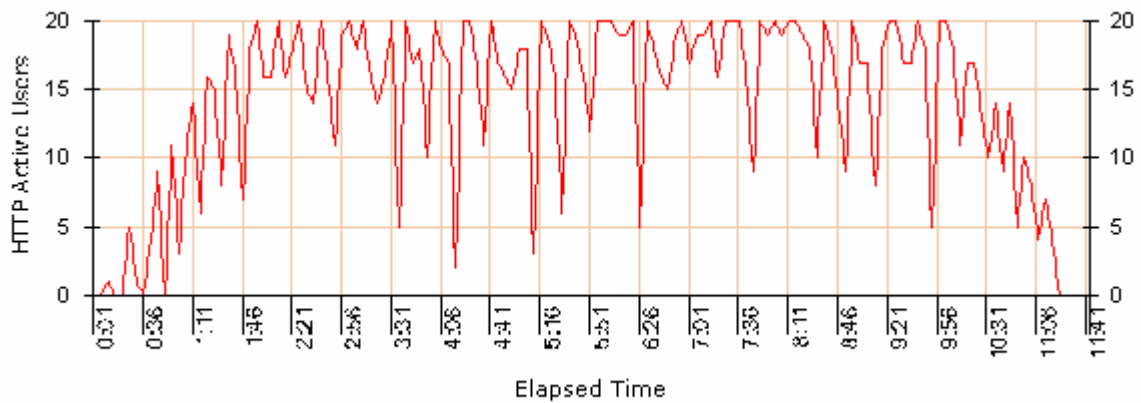


Mittler Antwortzeit: 931,7 ms

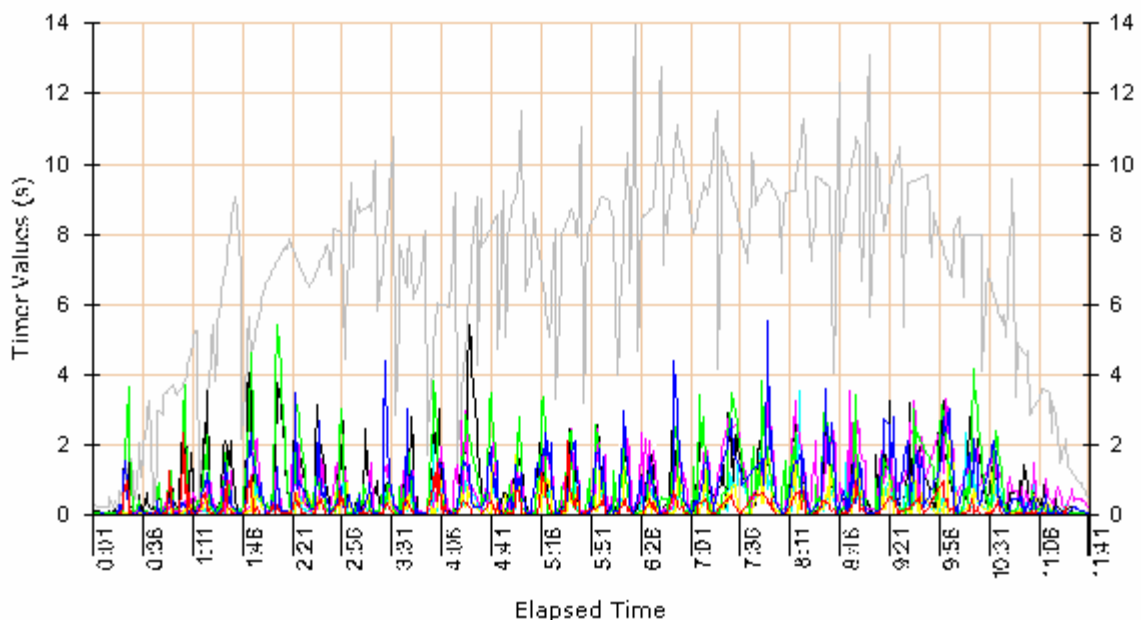
### 5.3.2 Lasttest bei PhPepperShop

Test: PEP\_NEWBUY  
Testzeit: 600s  
Single Task Time: 0,82s  
Single Task Data: 268 Kbytes  
VU's: 20  
Iterationsverzögerung: variable (1 bis 4s)

Anzahl der aktiven Benutzer



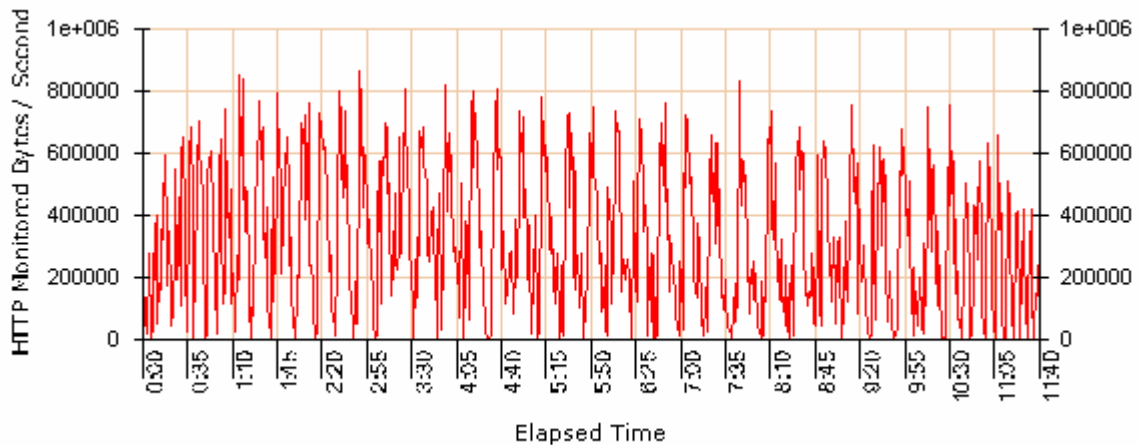
Timer für alle Webseiten des Einkaufsprozesses



grau ... Bestätigung der Erfolgreichen Bestellung  
alle anderen PHP- Seiten sind unkritisch

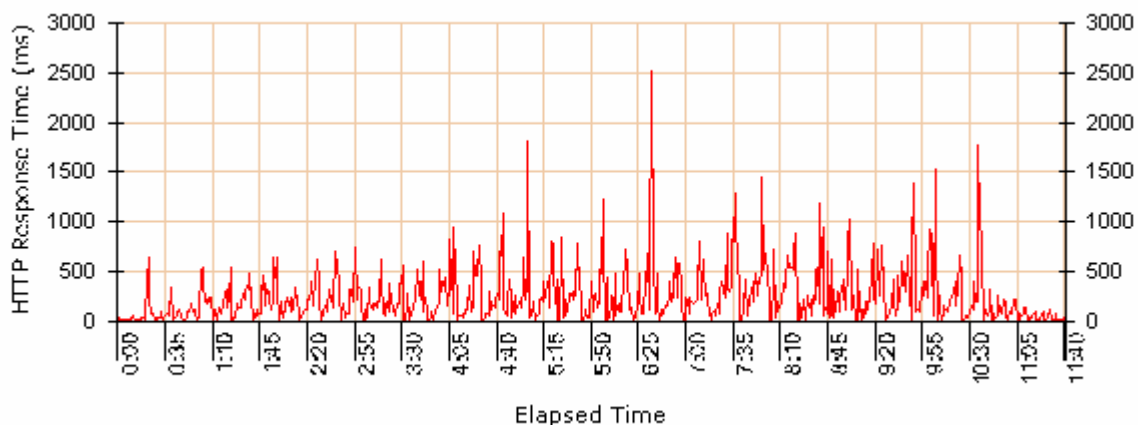
Der PhPepperShop ist unter dieser Belastung durchaus noch in der Realität einsetzbar. Die einzige Seite die eine kritisch lange Antwortzeit hat ist die Bestellungsbestätigungsseite. Hier ist der Einkauf bereits getätigt, somit hat diese lange Antwortzeit keinen Einfluss auf die Kaufentscheidung mehr.

#### Datendurchsatz in Bytes/s



Durchschnittlicher Datendurchsatz: 322,6 kbyte/s

#### Antwortzeitverhalten



Durchschnittliche Antwortzeit: 247,2 ms

### 5.3.3 Speedup von OnlineTransaktionen

Die maximale Anzahl paralleler VU's ist 20.

Alle 5 s wird ein neuer VU gestartet, bis 20 VU's online einkaufen. Ist ein Einkauf abgeschlossen, so wartet der VU zwischen einer und vier Sekunden bis er den nächsten Onlineeinkauf durchführt.

Die VU's kaufen bis zum Ende der Testzeit von 600 s.

Webshop	Einkaufstransaktionen/min		
	ohne mmCache	mit mmCache	Speedup
<b>osCommerce</b>	17,1	25,1	+47%
<b>PhPepperShop</b>	29,6	79,4	+268%

Es ist ein deutlicher Performance Gewinn mit mmCache zu erzielen. Bei PhPepperShop erreicht man um 268% mehr OnlineTransaktionen, bei osCommerce kann man zumindest 47% Performancegewinn feststellen. Der geringere Geschwindigkeitszuwachs beim osCommerce ist darauf zurückzuführen, das er komplexer aufgebaut ist und wahrscheinlich auch schon seine interne Struktur bereits optimiert ist.

## 6. Zusammenfassung

Um einen Webshops auf Basis von PHP einer größeren Anzahl von Usern anbieten zu können, benötigt man einen leistungsstarken Server. Hier spielt in erster Linie die Performance der CPU und der Speicherausbau eine Rolle. Bei nur einem dedizierten Server wäre eine starke Anbindung an das Internet Verschwendung. So würden erst ca. 30 Stück des bei diesen Tests verwendeten Server-Systems eine 100 Mbit-Leitung voll beanspruchen. Eine große Steigerung der Performance eines Shops wäre dadurch zu erreichen, wenn man versucht, statt der dynamisch generierten Seiten Statische zu verwenden. Bei der Startseite oder sich selten ändernden Katalogteilen wie Übersichten oder ähnlichen könnten diese vom Admin erzeugt werden. Professionelle Shopssysteme z. B. generieren solche statischen Abbilder von selbst.

Die beiden Webshops sind stabile Plattformen, die ihre Kinderkrankheiten überwunden haben und auch sicher genug sind, um wirklich eingesetzt werden zu können. Zwar vermisst man vor allem in der Administration das eine oder andere Feature, doch durch die Öffentlichkeit des Codes werden die Systeme immer weiterentwickelt, oder man nutzt die Möglichkeit selbst Hand anzulegen und die fehlenden Teile zu programmieren.

Der PhPepperShop überzeugt durch seine klare und innovative Architektur und ist auch besser an den Gebrauch im Deutschsprachigen Raum angepasst. Es verwendet bei der Generierung der Seiten Frames und spart sich somit eine komplizierte Tabellenstruktur wie osCommerce. Besser wäre hier der Einsatz von CSS (Stylesheets). Aufgrund seines einfacheren Aufbau und Verzicht auf aufwendige Features besitzt er auch die bessere Performance.

OsCommerce besitzt ein ansprechendes Design ohne das man viele Anpassungen vornehmen muss. Der von ihm gebotene Komfort wird allerdings mit der vollständig dynamisch Generierung der Seiten und der damit schlechteren Performance erkaufte. Eigenartig erscheint, dass bei der Standardinstallation beim Einkauf keine Bestätigung der AGB getätigt werden muss. Um dieses rechtliche Problem zu beseitigen muss man den entsprechenden Text selbst einbinden.

## Literaturverzeichnis

### Für die Seminararbeit verwendete Bücher

Kabir Mohamed: Die Apache Server 2 Bibel, 2. Auflage 2003, mitp-Verlag

McCarty Bill: PHP4 IT- Tutorial, 2002, mitp- Verlag

### Magazine

Hübner Malte: Sonderangebote, sechs Open-Source-Shopsysteme im Vergleich, iX 10/2003, S. 62: E-Commerce

Achim Wagenknecht: Kostenlos geöffnet, Open-Source-Shops im Vergleich, Internet Professionell Ausgabe 07/2003, online unter

<http://www.vnunet.de/testticker/internet/article.asp?ArticleID=7170> (20.01.2004)

### Internetressourcen

Sourceforge, die Quelle für Open Source Software, <http://sourceforge.net> (21.01.2004)

XAMPP 1.4 für Linux – Open Source Softwarepaket, <http://www.apachefriends.org> (21.01.2004)

OpenSTA 1.4.2, Performance - Testtool, <http://www.opensta.org> (21.01.2004)

Putty, SSH Client, <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html> (21.01.2004)

MySQL-Front, <http://www.mysqlfront.de> (21.01.2004)

### Open Source Shops

osCommerce, <http://www.oscommerce.com> (20.01.2004)

PhPepperShop, <http://www.phpeppershop.com> (20.01.2004)

Der PhPepperShop 1.4 SR1 wurde uns freundlicherweise vom Entwickler kostenlos zur Verfügung gestellt, frei ist momentan erst die Version 1.2 erhältlich.

### Papers und Präsentationen

Kneschke Jan: Webserver Performance Tuning,

[http://www.phpconference.de/2003/slides/business\\_track/kneschke\\_webserver-performance-tuning.pdf](http://www.phpconference.de/2003/slides/business_track/kneschke_webserver-performance-tuning.pdf) (21.01.2004)

Petrasch Roland: Empfehlungen zum Thema Performance Test,

[http://www.softwarequality.de/Petrasch\\_TAV16\\_PerformanceTest.PDF](http://www.softwarequality.de/Petrasch_TAV16_PerformanceTest.PDF) (20.01.2004)

Kopjev Dimitri: Seminararbeit „Lasttest und Webtest“,

<http://www4.in.tum.de/~pretschn/teaching/testsem02-ausarb/kopjev-lastweb.pdf> (20.01.2004)