Does dynamic meet static?
*Researching the performance of static versus dynamic websites*

Erik van Beijnen

Master’s Thesis
Information Science

Radboud University Nijmegen

*June, 2009*
Abstract

Content Management Systems (CMSs) are widely used on the web to maintain websites. A regular visitor though, shouldn’t be bothered by if a website is displayed by a CMS. Visitors are more concerned about how long they will wait for the requested website. Does the presence of a CMS negatively affect this waiting time?

This thesis investigates waiting time differences between websites with or without a CMS. Both types of websites are exposed to a load test and the different turnaround times are measured.

To analyze the turnaround times, the Mean Opinion Score is introduced to measure the quality of a system.

Although the results indicate that the time difference (between the different types of websites) is significant, for a visitor this difference appears to be negligible.
Preface

During a climb, a mountaineer fell down and broke both his legs. No one saw it, so nobody would call for help, and no rescue team would be sent within the first few days. He realized he needed its own strength to bring himself to safety. Because the journey would take a long time he kept setting small goals, such as: “I will reach that rock within one hour,” and “I will reach that plant before sunset”. Every achievement was a small step to safety. Using these achievable goals and will-power, the mountaineer within a few days returned to safety.

Frits Vaandrager told me this story, and it has stayed with me ever since. While writing my thesis, I continually set goals. It has taken me several years, but every achieved goal led me further to “safety”; the completion of this thesis.

I would like to thank my supervisor, Frits Vaandrager for his tremendous support and useful feedback. 
Thanks to Jan Tretmans and the company GX for starting up this research and provide me the opportunity to gain experience within a real company. 
Of course, many thanks to Colours, the internet company I work for now, for providing me all resources I needed in order to perform the eventual tests.

At last, but definitely not least, special thanks to my family, friends, and colleagues, for their support and encouragement.
# Table of contents

ABSTRACT ................................................................................................. I  
PREFACE .................................................................................................. III  
TABLE OF CONTENTS ........................................................................ IV  
LIST OF FIGURES ................................................................................ VI  
LIST OF TABLES .................................................................................. VIII  
1. INTRODUCTION ............................................................................. 1  
   1.1 PERFORMANCE EVALUATION .................................................. 4  
      1.1.1 Quality factors................................................................. 4  
      1.1.2 Time behavior ............................................................... 5  
   1.2 CONTENT MANAGEMENT SYSTEMS ....................................... 6  
   1.3 OBJECTIVES AND RESTRICTIONS .......................................... 7  
   1.4 THESIS STRUCTURE .................................................................. 8  
2. PERFORMANCE EVALUATION RESEARCH ................................... 9  
   2.1 PROBLEM IDENTIFICATION AND REQUIREMENTS ANALYSIS .... 10  
   2.2 CHARACTERIZATION ............................................................... 11  
      2.2.1 Workload characterization techniques ................................ 12  
      2.2.1.1 Customer Behavior Model Graph ..................................... 12  
      2.2.1.2 Customer Visit Model .................................................. 13  
   2.3 DESIGN METHODOLOGY ............................................................ 14  
      2.3.1 Experiments .................................................................. 14  
      2.3.2 Modeling ...................................................................... 15  
   2.4 ANALYSIS AND VALIDATION .................................................... 15  
3. PERFORMANCE EVALUATION ACTION PLAN .............................. 16  
   3.2 WORKLOAD CHARACTERIZATION ......................................... 17  
   3.3 EXPERIMENTS ON THE SYSTEM ............................................... 18  
   3.4 COMPARISON OF LOAD TEST RESULTS ................................ 19  
      3.4.1 Mathematical and Statistical values ..................................... 19  
      3.4.2 Mean Opinion Score ......................................................... 20  
      3.4.2.1 Define the known MOS values ........................................ 20  
      3.4.2.2 Define the corresponding logarithmic function and curve .... 22  
4. PERFORMANCE EVALUATION TEST CASE .................................... 24  
   4.1 THE TESTED WEBSITE - EURAIL.COM .................................. 24  
      4.1.1 About Eurail.com ............................................................. 24  
   4.2 PROBLEM IDENTIFICATION - EURAIL.COM ......................... 25  
      4.3.1 Defining the most visited pages ......................................... 27  
      4.3.2 Define the workload per tested page ................................. 28  
5. PERFORMANCE EVALUATION RESULTS ..................................... 30  
   5.1 TEST RESULTS: EURAIL_SELECT_PASS .................................. 31  
      5.1.1 Mathematical and statistical results ..................................... 31  
      5.1.2 Mean Opinion Score results ............................................. 32  
   5.2 TEST RESULTS: HOME ............................................................ 35
List of figures

Figure 1-1: The static vs. dynamic process of webpage retrieval .......................... 2
Figure 2-1: Overview of Performance Evaluation Methodology [Herz01] ............ 9
Figure 2-2: The system with its Environment, Requirements and Constraints [Herz01] ............................................................................................................ 11
Figure 2-3: An example of a Customer Behavior Model Graph [Alme02] ........... 12
Figure 3-1: Performance Evaluation Action Plan ............................................. 16
Figure 3-2: Length of time online shoppers will wait for a web page to render .... 21
Figure 3-3: Example mean opinion score chart .............................................. 23
Figure 4-1: The required standard deviation according to a normal distribution ........................................................................................................ 26
Figure 4-2: Page view trend Eurail.com (November 2007 - February 2008) ....... 28
Figure 5-1: Turnaround times page: eurail_select_pass .................................. 31
Figure 5-2: Average, median and standard deviation results page:  eurail_select_pass ......................................................... 32
Figure 5-3: MOS curve: eurail_select_pass ................................................... 33
Figure 5-4: Static MOS values: Eurail_select_pass ....................................... 33
Figure 5-5: Dynamic MOS values: Eurail_select_pass ................................... 34
Figure 5-6: Turnaround times, page: Home ................................................... 35
Figure 5-7: Turnaround times page: Home (< 100 ms) ..................................... 36
Figure 5-8: Average, median and standard deviation results page: Home ......... 37
Figure 5-9: MOS curve: Home .................................................................... 38
Figure 5-10: Static MOS values: Home ......................................................... 38
Figure 5-11: Dynamic MOS values: Home .................................................... 39
Figure 6-1: MOS values of all tested pages ................................................... 41
Figure 6-2: Average turnaround times of all tested pages .............................. 41
Figure A4-0-1: Turnaround times page: 1_prepare_your_trip ......................... 50
Figure A4-0-2: Average, median and standard deviation results page: 1_prepare_your_trip ................................................................. 50
Figure A4-0-3: Turnaround times page: eurail_select_pass ............................ 51
Figure A4-0-4: Average, median and standard deviation results page: eurail_select_pass ................................................................. 51
Figure A4-0-5: Turnaround times page: 1_global_pass .................................. 52
Figure A4-0-6: Turnaround times page: 1_global_pass (< 70 ms) ................. 52
Figure A4-0-7: Average, median and standard deviation results page: 1_global_pass ................................................................. 53
Figure A4-0-8: Turnaround times page: 1_our_products .................................. 54
Figure A4-0-9: Average, median and standard deviation results page: 1_our_products ................................................................. 54
Figure A4-0-10: Turnaround times page: Eurail_regional_pass ..................... 55
Figure A4-0-11: Average, median and standard deviation results page: Eurail_regional_pass ................................................................. 55
List of tables

Table 1-1: Percentage of households in The Netherlands with connection to the internet [CBS07] ..........................................................1
Table 1-2: McCall’s software quality factor model [Gali04] ......................4
Table 1-3: ISO 9126-1 defined Quality Characteristics and (their) sub-characteristics [Zeis+96] ...............................................................5
Table 1-4: ISO 9126-1 Sub-characteristic Time behavior Indicators [Zeis+96] ...6
Table 2-1: Application of the Customer Visit Model to Characterize a Session [Alme02] .................................................................................13
Table 3-1: Mean opinion score (MOS) .....................................................20
1. Introduction

It began in the early 1990s. The Internet became accessible to everyday households. Prior to this, only a fraction of households had access to the internet. In the Netherlands, the number of households with an internet connection increases rapidly every year (see Table 1-1).

<table>
<thead>
<tr>
<th>Year</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1998</td>
<td>14</td>
</tr>
<tr>
<td>1999</td>
<td>23</td>
</tr>
<tr>
<td>2000</td>
<td>38</td>
</tr>
<tr>
<td>2001</td>
<td>48</td>
</tr>
<tr>
<td>2002</td>
<td>55</td>
</tr>
<tr>
<td>2003</td>
<td>59</td>
</tr>
<tr>
<td>2004</td>
<td>64</td>
</tr>
<tr>
<td>2005</td>
<td>78</td>
</tr>
<tr>
<td>2006</td>
<td>80</td>
</tr>
<tr>
<td>2007</td>
<td>83</td>
</tr>
<tr>
<td>2008</td>
<td>86</td>
</tr>
</tbody>
</table>

Table 1-1: Percentage of households in The Netherlands with connection to the internet [CBS07]

Given the large and varied backgrounds of daily internet users, a website is an effective way to inform visitors on products and services. Maintaining up-to-date webpage content assists in keeping these visitors and it ensures their return. Content Management Systems (CMSs) have been developed to fulfill this need, allowing webmasters to efficiently and effectively update websites.

Since its initial introduction, CMS functionalities have been expanded. In the early stages of CMS development, only a small portion of website content could be added and updated. Modern CMSs offer a variety of functionalities such as: version management, validation, multi-language, multi-site management, sharing content and additional possibilities to support website maintenance. Regardless of a website’s content, whether it contain plain text, images, multimedia, or other binary content, it can be saved and maintained in most modern CMSs.
Initially websites only contained static HTML files. These static files were used to display the website when it was accessed. When a dynamic page is requested from a website (i.e. where content has been saved into a CMS), a more complicated process is followed to display the website than when a static page is requested. This process differs with each CMS, but can include for example: site-selection, language selection, and deriving content from different locations in order to be able to display the requested page (see Figure 1-1).

A page request from a static website

A page request from a website using a CMS (dynamic)

Figure 1-1: The static vs. dynamic process of webpage retrieval

Getting a single file in a static website may appear less time-consuming than the process of presenting a dynamic page using a CMS. Although the speed of this process has improved, compared with earlier versions of CMSs, does this process now meet the speed of its static variant?
This thesis addresses the following problem: *CMSs have proven their usefulness in the web-site maintenance process. Using a CMS for website maintenance is becoming more a rule than an exception; even when a website does not need all the functionalities a CMS offers. Everyday visitors of a website however, still suspect the same performance of the website. They are oblivious to whether the website is being run by a ‘state-of-the-art’ CMS. They only want requested web-content in an acceptable time.*

*Based on this knowledge, my research question is:*

| Is the (current) quality of a website that retrieves its content from a CMS, compared with its static variant, still acceptable in terms of performance? |

A performance evaluation is implemented to answer this question.

Two important terms in this research are *performance evaluation* and *content management system*. In this chapter, these two terms are explained, the objectives and restrictions of this research are defined, and the structure of this thesis is introduced.
1.1 Performance evaluation

1.1.1 Quality factors

When evaluating a computer system, the quality of that system is verified. To be of good quality, software has to meet all requirements. These requirements should cover the entire system, including all software attributes and software use. This includes, but is not limited to, usability, reusability, and maintainability abilities. Issues related to software attributes, use, and maintenance (as defined in software requirements documents), can be classified into content groups called quality factors.

A classic model of software quality factors was suggested by McCall [Galo04]. McCall’s factor model classifies all software requirements into 11 quality factors. These factors are grouped into three categories: product operation, product revision, and product transition. The division of these quality factors per category is displayed in Table 1-2.

<table>
<thead>
<tr>
<th>McCall’s software quality factor model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Product transition:</strong></td>
</tr>
<tr>
<td>Portability</td>
</tr>
<tr>
<td>Reusability</td>
</tr>
<tr>
<td>Interoperability</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Table 1-2: McCall’s software quality factor model [Gali04]

McCall’s model has been simplified by the ISO 9126-1 into the ISO 9126-1 quality model. This model is now commonly accepted in the state-of-the-art of product quality specification. It proposes a set of six independent high-level quality characteristics, which are defined as a set of attributes of a software product by which its quality is described and evaluated. These quality characteristics are refined into sub-characteristics. These quality (sub) characteristics are shown in Table 1-3.
ISO 9126-1 defined Quality Characteristics and (their) sub-characteristics

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Reliability</th>
<th>Usability</th>
<th>Efficiency</th>
<th>Maintainability</th>
<th>Portability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suitability</td>
<td>Maturity</td>
<td>Understandability</td>
<td>Time behavior</td>
<td>Analyzability</td>
<td>Adaptability</td>
</tr>
<tr>
<td>Accuracy</td>
<td>Fault tolerance</td>
<td>Learnability</td>
<td></td>
<td>Changeability</td>
<td>Installability</td>
</tr>
<tr>
<td>Interoperability</td>
<td>Recoverability</td>
<td>Operability</td>
<td></td>
<td>Stability</td>
<td>Conformance</td>
</tr>
<tr>
<td>Compliance</td>
<td></td>
<td>Explicitness</td>
<td></td>
<td>Testability</td>
<td></td>
</tr>
<tr>
<td>Security</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1-3: ISO 9126-1 defined Quality Characteristics and (their) sub-characteristics

[Zeis+96]

In the ISO 9126-1 quality model, each sub-characteristic contains various (related) indicators. These indicators enable measurement of the sub-characteristic.

1.1.2 Time behavior

All ISO 9126-1 sub-characteristics are used when evaluating a software system to verify the quality of that system (see Table 1-3 for a list of all characteristics). However, not all sub-characteristics are relevant when only the system’s performance is evaluated. To determine which sub-characteristics are relevant, Ulrich Herzog’s [Herz01] definition of performance evaluation is used. Herzog defines performance evaluation as:

Investigating and optimizing the dynamic time-varying behavior within and between individual components of transportation and processing systems.

This definition highlights that the behavior of a system in terms of time, is the key aspect of a system’s performance. Based on the ISO 9126-1 sub-characteristics, time behavior is the crucial characteristic in performance evaluation.

Time behavior is defined in the ISO 9126-1 as:

Attributes of software that bear on response and processing times and on throughput rates in performing its function.

Time behavior consists of 16 indicators shown in Table 1-4

---

1 The ISO 9126-1 formulation of the Time behavior Indicators can be found in appendix 1
Based on the ISO 9126-1 standard, I define performance evaluation as:

Investigating and optimizing the behavior of a system, in terms of time behavior, using the indicators given in ISO 9126-1.

1.2 Content Management Systems

A definition of content management is given in [Bos05] as: “The process of managing electronic content through its lifecycle - from creation, review, storage, and dissemination to destruction.” A Content Management System (CMS) is an application that supports this process (most times a web-application\(^2\)). A CMS is used for managing websites (construction and maintenance of different web pages) and web content (text, images, sounds, videos, and animations).

Because CMSs simplify website maintenance, even if this website has large content or a complex structure, they are used throughout the Web.

\(^2\) A web-application is an application that’s accessed with a Web browser over a network such as the Internet or an intranet.
1.3 Objectives and restrictions

This thesis focuses on evaluating the performance of a content management system. The aim is to answer the question; is the performance of a website driven by a CMS acceptable, as compared to an identical website without a CMS (i.e. a website containing only static HTML files)?

This thesis has the following restrictions:

- The focus is on performance methods and techniques that can be applied to the Web;
- Performance evaluation and implementation is applicable to the, so called, presentation layer of a content management system.

A content management system’s presentation layer refers to the way an everyday visitor experiences and interacts with the system (public interface a.k.a. front-end). It can be compared to a television screen. A program is watched, but the deeper processes such as the broadcasting and placing of images on a screen remains a mystery to most viewers. The presentation layer can be regarded as the ‘shell’ of the CMS.
1.4 Thesis structure

Chapter 2 presents research relating to this topic. Consideration is given to Ulrich Herzog’s performance evaluation methodology. All steps within the methodology are explained and the various related methods, techniques, and tools used, are also introduced.

Chapter 3 presents the action plan based on Ulrich Herzog’s performance evaluation methodology. This action plan contains defined steps, in order to be able to compare the quality of a system of both with and without a CMS. This chapter also introduces the Mean Opinion Score, which indicates the quality of a system.

Chapter 4 introduces the initial performance evaluation of the high traffic website, Eurail.com, fulfilling the first steps of the action plan (analysis and workload characterization).

Chapter 5 contains the experimental section of this research. Information and results from the load test of the website Eurail.com, both with and without CMS, are presented. The Mean Opinion Score of all tested pages within the website are also provided.

In Chapter 6, the main conclusion of this thesis is presented. An answer to the research question is given and the introduced Mean Opinion Score is evaluated.
2. Performance Evaluation research

This chapter provides a summary of the research into different methods, techniques, and tools, to evaluate performance on the web. These methods, techniques and tools can assist in the development of a way to better understand performance results.

This chapter uses the performance evaluation methodology introduced by Ulrich Herzog as a basis for investigation (see Figure 2-1). Other methods, techniques, and tools applicable to this methodology are also presented in this chapter.

![Figure 2-1: Overview of Performance Evaluation Methodology [Herz01]](image-url)
2.1 Problem identification and requirements analysis

Before evaluating a system’s performance, the problem must be identified and a thorough requirements analysis completed.

In [Kula+03], a requirement is defined as something that a computer application must do for its users. It is a specific function, feature, quality, or principle that the system must provide for it to merit its existence. Requirements constitute part of the scope of a software development project. The requirements also dictate how a system should respond to user interaction. It should do specific things when poked or prodded by the user.

Requirements let you know what the system must do for its users. In any evaluation, it is important to have requirements. Otherwise one can never be sure that the system behaves as required. Requirements should also be complete. When a problem occurs, this problem should always be able to be traced back to a requirement.

Requirements have a direct influence on the quality of software. The IEEE3 definition of software quality is as follows:

<table>
<thead>
<tr>
<th>Software quality is:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The degree to which a system, component, or process meets specified requirements.</td>
</tr>
<tr>
<td>2. The degree to which a system, component, or process meets customer or user needs or expectations</td>
</tr>
</tbody>
</table>

Software quality - IEEE definition [Gali04]

According to [Gali04], the number 1 cause of software errors is a faulty requirement definition. A thorough requirements analysis therefore improves software quality.

When evaluating software performance, only requirements relating to performance should be included. As discussed in paragraph 1.1, when evaluating performance, a system is evaluated using the ISO 9126-1 time behavior sub-characteristics. This means that the requirements that relate to performance are those that are measured using the time behavior indicators (see Table 1-4).

---

3 IEEE (Institute of Electrical and Electronics Engineers, inc.) is the world's leading professional association for the advancement of technology.
Given that performance evaluation requires performance requirements, the weaker the requirements, the less reliable and less useful the outcome of the performance evaluation will be.

2.2 Characterization

Once a problem has been identified and the appropriate performance requirements are identified, the first evaluation steps can begin. These initial steps are referred to as workload characterizations and system parameter specification. The workload of a computer system is defined by [Ferr78] as the set of inputs the system receives from its environment. According to [Alme02], workload characterization is the process of precisely describing the system’s global workload in terms of its main components. The other evaluation step, system parameters, refers to the specifications (configurations and settings) of the system being evaluated.

Figure 2-2 displays a general performance evaluation scenario to illustrate the need for characterization evaluation. This figure highlights that the environment generates system requests (the workload), while the system (which is bound by the system parameters) consists of one or more components trying to satisfy these requests. An optimal system structure and operating mode is reached if the system fulfils all its performance requirements concerning the, so called, Quality of Service (QoS), as well as fulfilling all technical and economical constraints.

![Figure 2-2: The system with its Environment, Requirements and Constraints [Herz01]](Image)

To summarize, appropriate workload characterization is required to generate relevant requests, while system parameters are required to determine what kind of system the (actual) QoS refers to.

Various techniques and tools are available to characterize a system’s workload. Some of these are discussed in the following sub-paragraphs.
2.2.1 Workload characterization techniques

To generate a relevant workload in a performance evaluation, there are various techniques available to characterize the workload. A few of these are discussed in this paragraph.

2.2.1.1 Customer Behavior Model Graph

In Web-based environments, users interact with the site through a series of consecutive and related requests, called sessions. It has been observed that different customers exhibit different navigational patterns. A Customer Behavior Model Graph (CBMG) [Alme02] can be used to capture the navigational pattern of a visitor through a website.

An example of a CBMG is provided in Figure 2-3. This CBMG determines how a visitor moves on an e-commerce site. How does the user move from one state to another? This is represented by a matrix of transition probabilities. Probabilities are associated with transitions as in a Markov Chain.

Consider the CBMG from Figure 2-3. The probability that a customer goes from ‘entry’ to ‘browse’ is 0.50. The probability that he/she goes from ‘browse’ to ‘search’ is 0.60. The probability that he/she goes from ‘entry’ to ‘search’ via ‘browse’ is 0.50 x 0.60 = 0.30, and so on.

In [Alme02], CBMGs are explained in further detail.

---

4 A Markov Chain provides information about different states in a system and the probabilities the system will switch from one state to another.
2.2.1.2 Customer Visit Model

An alternate, and less detailed, representation of a session entails representing a session as a vector of visit ratios to each state of the CBMG. The visit ratio is the number of visits to a state during a session [Alme02].

Table 2-1 presents an example of three sessions described by the number of visits to each state of the CBMG. Note that the states ‘entry’ and ‘exit’ are not represented in the Customer Visit Model (CVM) since the number of visits to these states is always one. Session 1 represents a session of a customer who browsed through the site, did a few searches, but did not login or buy anything. In session 2, the customer logged in but did not register because he or she was already a registered customer. This customer abandoned the site before paying, even though two items had been added to the shopping cart. Finally, session 3 represents a new customer who registers with the site, adds one item to the shopping cart, and pays for it.

<table>
<thead>
<tr>
<th></th>
<th>Session 1</th>
<th>Session 2</th>
<th>Session 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Browse</td>
<td>4</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>Search</td>
<td>5</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Login</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Pay</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Register</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Add to Cart</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Select</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 2-1: Application of the Customer Visit Model to Characterize a Session [Alme02]

The information from a CVM can be used to describe the global workload of different states of the site.
2.3 Design methodology

The design methodology employed in this study is an iterative method to design an optimized structure of the software system in relation to performance. Within this design methodology, performance can be validated. When performance is deemed unacceptable (as defined in step one, performance requirements, see section 2.1), there should be a return to the characterization step (see section 2.2) to redefine either the workload, the system parameters, or both, until performance is acceptable.

The design methodology distinguishes between two separate, yet complementary, approaches: *experiments* on the real system (measurements) and *modeling*.

2.3.1 Experiments

In experimentation, a so called *load generator* is used to generate workload on the real system. The behavior of the system can be measured and analyzed according to whether it is deemed acceptable or not (i.e. does it meet the performance requirements?).

A load generator mimics browser behavior: It continuously submits requests to the website, waits for a period of time after the site sends a reply to the request (the think time), and then submits a new request. A load generator can emulate thousands of concurrent users. Each emulated browser is called a virtual user, which is a key load-testing concept. A load test is valid only if virtual users’ behavior has characteristics similar to those of actual users. [Mena02]

An example of a load generator is Apache’s JMeter [Jmet]. JMeter can be used to simulate a heavy workload on a system to test its strength or to analyze overall performance under different workloads. It can also be used to create a graphical analysis of the performance.

In recent years, there have been several load generators introduced such as WebAppLoader [Wolt03] or Tangram II [Leao+00]. Load generators are also built into software development tools such as Microsoft Visual Studio.
### 2.3.2 Modeling

“Model-based performance evaluation is about finding those 10% of the system that explains 90% of its behavior”

Alan Scherr - IBM’s time-sharing pioneer

Performance models are used to predict performance after an aspect of the workload or the site’s architecture has changed. Two types of models are available: *simulation models* and *analytical models*. [Alme02]

- Analytical models specify the interactions between various components of a web system via formulas.
- Simulation models mimic the behavior of the system by running a simulation program.

System behavior can be simulated using Colored Petri Nets (CP-nets). CP-nets are a graphical modeling language that models both the states of a system and the events that change the system from one state to another. CP-nets combine the strengths of Petri nets and programming languages. The utilization of Petri nets is well-suited for describing concurrent and synchronizing actions in distributed systems. Programming languages can be used to define data types and manipulation of data.

During a CP-net simulation, the CP-net can contain and generate a large amount of quantitative information regarding the performance of a system, such as: queue length, response time, throughput, etc [Well02].

More information regarding Petri nets can be found in [Balb01, Well02].

### 2.4 Analysis and validation

The test results are analyzed and validated. If the results do not meet the stated requirements, either the system, the test parameters, or both, can be modified and the test can be repeated.

Finally, the system should be optimized and meet all requirements.
3. Performance Evaluation Action Plan

To answer the research question, the following two performance evaluations of a single system are completed and compared:

1. **A CMS driven website**
   A performance evaluation is carried out on a website that is fully driven by a CMS. This CMS driven website is hereafter referred to as the *dynamic website*.

2. **The same website without a CMS**
   A static copy of the dynamic website is made and a performance evaluation of this static version is undertaken. This static copy of the dynamic website is hereafter referred to as the *static website*.

Based on Hertz’s Performance Evaluation Methodology [Herz01], the research performance evaluation within this study can be described using the following (non-iterative) steps (see Figure 3-1):

1. **Problem identification**
2. **Workload characterization**
3. **Experiments on the system**
   a. Load test on the system driven by a CMS (the dynamic website)
   b. Load test on the system without CMS (the static website)
4. **Comparison of load test results**

![Figure 3-1: Performance Evaluation Action Plan](image)

The steps within the performance evaluation action plan are discussed in the following sections.
3.1 Problem identification

The main issue within this performance research is the time-behavior of a website. Because this research is addressed to the front-end (public interface) of a dynamic website, the other main issue is the website user itself. The website user is the most important person who interacts with the front-end of the system.

Using the time behavior Indicators, introduced in paragraph 1.1.2, the single indicator focusing the best on how a website user experiences a website is turnaround time. If a website has a long loading time, a visitor is more likely to be dissatisfied (or vice versa).

According to ISO 9126-1, the definition of turnaround time indicates:
The speed of processing by measuring the elapsed time between the beginning of process requirement and gaining the result of the process.

Theoretically, as both the static and the dynamic websites are exposed to the same experiments, each websites’ turnaround times can be compared. In practice however, static websites out-perform dynamic websites. A method which supports the measurement and comparison of these differences is presented and described in paragraph 3.4.

3.2 Workload characterization

Although there are numerous characterization techniques available, the Customer Behavior Model Graph (CBMG) is used in this study. This model was selected because it is sufficient, visually representative, and easily explainable. This graph also presents the most visited (occupied) pages of a website and the navigation paths used.
Together with recent website visitor statistics, a realistic and reliable CBMG can be made. Using these statistics, a realistic workload for each of the website pages can be calculated.

Only the workloads of the heavily visited pages are characterized. The reasons why the other pages are not characterized are as follows:

1. From a visitor’s perspective, heavily visited pages within a website are the most important.
2. Pages less frequently visited are less susceptible to performance evaluation; characterization of these pages would not add value to test results.
3. Since at high-traffic websites the number of other pages is enormous, testing these pages would require an enormous undertaking. Since this effort does not gain lots of value to the test results, it’s not efficient to test these pages.

The workload of the characterized pages is expressed in requests per second.

### 3.3 Experiments on the system

A real system was used for this performance evaluation as results based on actual systems are more realistic and reliable than those completed on imitation systems.

Within this performance evaluation, both the static and dynamic websites are exposed to identical workloads within a load test. These load tests are executed under identical circumstances using the same server. Thus, the only uncontrolled variable present is the website (System Under Test).

For each page-request, the measured turnaround time per page is recorded in milliseconds. These times are used within the next step of the performance evaluation.
3.4 Comparison of load test results

As all tested pages of the dynamic and static websites were exposed to an identical number of requests, a single graph per page for each of the websites can be compiled to visualize the load test results.

Two additional methods and techniques are used to compare each system’s results: Mathematical and Statistical values and Mean Opinion Score.

3.4.1 Mathematical and Statistical values

Alongside visual representation of the turnaround times, the following mathematical and statistical values are calculated based on each page’s turnaround time per website condition (dynamic or static).

**Average turnaround time**

Average turnaround time can be calculated for each page within each website condition (dynamic or static). The average turnaround time is calculated by summing all turnaround times and divided this by the number of results. This average value can be used to identify which page, on average, performs the best as measured by turnaround times.

**Median**

Median turnaround time refers to a single page’s central turnaround time per website condition (dynamic or static), when all observations are arranged from lowest to highest. It is the turnaround time separating one half of the population from the other.

This value can be used to investigate if there are large peaks in the test results. If the median is lower than the average, there must be one or more requests influencing the average turnaround time by increasing this average value.

**Standard deviation**

The standard deviation measures the distribution of the turnaround times. A low standard deviation indicates stable turnaround times, while a high standard deviation indicates instable, or fluctuating, turnaround times.

This value can be used to measure page’s stability. The lower the standard deviation of the turnaround time is, the more stable a page and its corresponding system are.
3.4.2 Mean Opinion Score

The mean opinion score (MOS) was originally developed to indicate the perceived quality of multimedia data services (audio, voice telephony, or video), such as voice over IP. The MOS is expressed as a single number between 1 and 5, where 1 corresponds to the lowest perceived quality and 5 the highest (see Table 3-1 for descriptors [Vos06]).

<table>
<thead>
<tr>
<th>MOS</th>
<th>Quality</th>
<th>Impairment</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Excellent</td>
<td>Imperceptible</td>
</tr>
<tr>
<td>4</td>
<td>Good</td>
<td>Perceptible but not annoying</td>
</tr>
<tr>
<td>3</td>
<td>Fair</td>
<td>Slightly annoying</td>
</tr>
<tr>
<td>2</td>
<td>Poor</td>
<td>Annoying</td>
</tr>
<tr>
<td>1</td>
<td>Bad</td>
<td>Very annoying</td>
</tr>
</tbody>
</table>

Table 3-1: Mean opinion score (MOS)

In my opinion this is a well formed numerical indication to indicate the quality (in terms of performance) of the System under Test. I will try to make it applicable within the performance evaluation by defining the MOS values (see section 3.4.2.1).

3.4.2.1 Define the known MOS values

User experience follows a logarithmic curve [Vos06]. Within this study, the turnaround times from 0 onwards are plotted on the X-axis, while the MOS values from 1 to 5 are plotted on the Y-axis.

To individually compare each tested page requires a separate logarithmic function and curve for each page. At least two known values are necessary to calculate the logarithmic function. The two most practical values to determine are:

- “Good” turnaround time
- “Poor” turnaround time
Good turnaround time
Good turnaround time can be regarded as one that is neither annoying to the visitor nor barely noticeable (i.e. imperceptible, see Table 3-1). Good turnaround time is based on the average turnaround time of the static load tests. This turnaround time is not by definition the fastest; it represents the system that should not be deemed annoying for the visitor.

Poor turnaround time
In a study on consumer reactions to various online shopping experiences, Jupiter Research [Jupi06] concluded that page rendering should be kept to no more than four seconds (see Figure 3-2).

![Figure 3-2: Length of time online shoppers will wait for a web page to render](image)

This four-second rule is regarded as the statistical boundary between “annoying” and “very annoying” (resulting in a visitor leaving the website). Poor turnaround time is therefore set to 4000 milliseconds (4 seconds).
3.4.2.2 Define the corresponding logarithmic function and curve

Once these two values are identified (the good and poor turnaround time), a single logarithmic function can be applied.

For example, if a page’s average turnaround time is 2000 milliseconds, then the following are the known parameters: [4, 2000] and [2, 4000] \(^5\).

The single corresponding logarithmic function is the following \(^6\):

\[
Y = 25.9316 - (2.88539 \times e^{\log X})
\]

Where:
\[
e = \text{the unique real mathematical constant (2.71828...)}
\]

This can be rewritten as:

\[
MOS = 25.9316 - (2.88539 \times e^{\log TT})
\]

Where:
\[
MOS = \text{the mean opinion score (1 <= MOS <= 5)}
\]
\[
TT = \text{the measured turnaround time in milliseconds (>= 0)}
\]
\[
e = \text{the unique real mathematical constant (2.71828...)}
\]

An example of the corresponding MOS chart can be seen in Figure 3-3.

\(^5\) These know values are in the form of [MOS value, turnaround time]
\(^6\) This logarithmic function is plotted with the help of a mathematical expert of the University of Technology in Eindhoven
Figure 3-3: Example mean opinion score chart
4. Performance Evaluation Test Case

This chapter is the beginning of exposing a real website to the performance evaluation action plan introduced in chapter three. First, the selected website is introduced, then the results of the first two steps of the performance evaluation action plan are listed.

4.1 The tested website - Eurail.com

High traffic websites have multiple simultaneous and multiple returning visitors. These websites are often susceptible to performance effects because if a visitor has a negative performance experience (i.e. a long waiting time before gaining requested information), this visitor is likely to not only abandon the site, but not return to the site. The visitor may even go to a competitor’s website. For an e-commerce website, negative performance will influence selling rate and possibly have fatal consequences for the website.

For these reasons, when selecting a website for performance evaluation a high traffic (e-commerce) website using a CMS was selected. Eurail.com responded positively to the request to perform a performance evaluation study of its website.

4.1.1 About Eurail.com

The website Eurail.com sells rail passes for European train travel. Eurail.Com is a subsidiary of the Eurail Group, a cooperation of 30 European railway companies and several shipping lines. The website specializes in selling Eurail Passes online.

The company is based in Utrecht (the Netherlands) with other branches operating in the United States and Australia. Eurail advertises itself as offering Rail Passes to non-European residents who wish to explore Europe by rail.

Eurail.com is powered by the CMS Smartsite.
4.2 Problem identification - Eurail.com

Eurail.com is a high traffic e-commerce website. A lack of website performance leads to a loss of visitors and, hence, also a loss of customers.

Within this performance evaluation, the dynamic website should keep up with the static version, with only minimal differences between the two. A stricter requirement can be formalized as:

<table>
<thead>
<tr>
<th>Requirement #1</th>
</tr>
</thead>
<tbody>
<tr>
<td>The turnaround time for a page request may be perceptible but should not be annoying from the visitor’s perspective. Therefore, the average mean opinion score for each tested pages should be at least 4.</td>
</tr>
</tbody>
</table>

Since conclusions will be based on an enormous amount of measured values there should also be assured that these values are reliable. If the measured turnaround times are too much spread out (at the time of measuring), the system tested would be too unstable. The measured turnaround times then will be unreliable. There should be assured that for each tested (dynamic and static) page almost all measured turnaround times (99 percent) should differ with a maximum of 1500 milliseconds from the average. Assuming a normal distribution, coverage of 99.6% requires a standard deviation of 500ms (see Figure 4-1).
This leads to the following requirement:

**Requirement #2**  
*Since the measured values may not be statistically significant, the standard deviation for each tested page may not exceed 500 milliseconds.*
4.3 Workload characterization - Eurail.com

To characterize the website’s workload, statistics from the live website (from the period November 2007 - March 2008) were collected and analyzed. A Customer Behavior Model Graph (CBMG) was also created (from the period 21 January - 20 February, 2008). This CBMG can be found in appendix 2.

4.3.1 Defining the most visited pages

Based on the CBMG, there are several paths that a regular website visitor follows. The seven most frequently used pathways are:

Path 1:
Home → 1_prepare_your_trip

Path 2:
Home → eurail_select_pass

Path 3:
Home → 1_global_pass

Path 4:
Home → 1_our_products

Path 5:
Home → eurail_regional_pass

Path 6:
Home → 1_where_to_go

Path 7:
Home → visit other page or exit site

Based on this information, the following seven pages are selected for performance evaluation:

1. Home
2. 1_prepare_your_trip
3. eurail_select_pass
4. 1_global_pass
5. 1_our_products
6. eurail_regional_pass
7. 1_where_to_go

As discussed in paragraph 3.2, all remaining pages are not included in the test.
4.3.2 Define the workload per tested page

According to the statistics, the busiest day was 3 March with a total of 108,542 page views. The busiest hour in the investigated period was at 9 pm with 6,483 page views. Using this data, the highest perceived workload is at least 1.81 requests per second (6,483 page views / 3600 seconds).

Viewing the website’s page view trend (see Figure 4-1), the number of visitors per day increases by approximately 10 percent per month from December onwards.

To match this increasing trend, the workload is adjusted with a factor of 1.5. The workload that is used within the performance test requires adjustment to **2.72 requests per second**. Based on this rate of growth, the test will remain applicable for five to six months.
Using the defined workload and the CBMG, the following workload is defined per tested page:

**Home**

2.72 requests per second

**1_prepare_your_trip**

2.72 * 0.09 = 0.25 requests per second

**eurail_select_pass**

2.72 * 0.15 = 0.41 requests per second

**1_global_pass**

2.72 * 0.1 = 0.27 requests per second

**1_our_products**

2.72 * 0.05 = 0.14 requests per second

**eurail_regional_pass**

2.72 * 0.08 = 0.22 requests per second

**1_where_to_go**

2.72 * 0.04 = 0.11 requests per second
5. Performance Evaluation Results

Within two test rounds of 15 minutes, both static and dynamic versions of the Eurail.com website were exposed to a load test with the predefined characterized workloads (see paragraph 4.3.2). Appendix 3 presents the experimental setup; it explains the environment in which the tests have been accomplished and the tools used within these tests.

This chapter presents the test results of the two pages with the highest load: eurail_select_pass and Homepage. A complete overview of all test results can be found in appendix 4.

The following abbreviations and terms are used:

- **rps**: requests per second
- **ms**: milliseconds
- **time static**: the results of the test with the static web-site (without CMS)
- **time dynamic**: the results of the test with the dynamic web-site (with CMS)
5.1 Test results: eurail_select_pass

5.1.1 Mathematical and statistical results

Both the static and dynamic variants of the eurail_select_pass page were subjected to a load of 0.41 requests per second. Within a 15-minute time frame, this results in 369 requests per website variant. All measured turnaround times are displayed in Figure 5-1.

As seen in Figure 5-1, there appears to be a slight difference between the static and dynamic page. The results of both pages generally fluctuate between 0 and 15 ms, however the dynamic page contains several peaks at 30 ms and even at 60 ms.

Unlike the static page, the dynamic page appears to be unstable. This is supported by the average, median, and standard deviation results (see Figure 5-2).
Due to the few large peaks in the turnaround time data (see Figure 5-1), the average turnaround time of the dynamic page is higher than the static page (see Figure 5-2). These peaks have also had an affect on the median and standard deviation of the turnaround times. Since the standard deviation is less than 500 ms, it meets the 2\textsuperscript{nd} requirement.

### 5.1.2 Mean Opinion Score results

For the page “Eurail\_select\_pass” a good turnaround time is the static page’s average turnaround time (9.31 ms), while the poor turnaround time is set at 4000 ms (see paragraph 3.4.2.1 for further information).

From these values, the following logarithmic function is derived:

\[
MOS = 4.73597 - (0.329872 \times e^{\log_{TT}})
\]

Where:

- \(MOS\) = the mean opinion score (1 <= MOS <= 5)
- \(TT\) = the measured turnaround time in milliseconds (>= 0)
- \(e\) = the unique real mathematical constant (2.71828...)

The corresponding curve can be seen in Figure 5-3.
Using this logarithm, all measured turnaround times can be converted to an MOS value. All calculated MOS values for both the static and dynamic page are presented in Figure 5-4 and 5-5.
Figure 5-5 indicates that the static MOS value only fluctuates between 3.8 and 5. Although like in Figure 5-1 the dynamic page again contains peaks, compared to the peaks in the turnaround time graph, these peaks are not as large. The average MOS for the static page is 4.30 and for the dynamic page it is 4.01.
5.2 Test results: Home

In contrast with the eurail_select_pass page, the Homepage was exposed to a much higher load (2.72 requests per second as opposed to 0.41 requests per second). This increase in load is reflected in the difference in test results.

5.2.1 Mathematical and statistical results

Within a 15-minute time frame, both the static and dynamic variant of the homepage was requested 2448 times. The turnaround times of these requests are displayed in Figure 5-6.

![Figure 5-6: Turnaround times, page: Home](image)

With its relatively small workload, the eurail_select_pass page contained occasional turnaround time peaks of 30 or 60 ms. The turnaround times for the homepage however, display more frequent peaks in turnaround time and increased peaks (up to 2000 ms). This difference is not surprising given the workload for the homepage was increased by at least six times than that of the eurail_select_pass page. Compared to the static page, the dynamic page also contains more peaks in turnaround time.

Although the peaks are easily observed in Figure 5-6, any other request information is hidden due to the scale of the graph. Figure 5-7 re-presents the turnaround times on a graph using a different scale, where the y-axis is limited to 100 ms. Differences between the static and dynamic pages now become visible.
Figure 5-7 displays a “gap” between the static and dynamic turnaround times. This gap may be explained by the high workload the tested page (Homepage) is subjected to. The turnaround time of the static page fluctuates between 0 and 15 ms, while the dynamic page fluctuates between 30 and 47 ms. It appears that at an increased workload, the dynamic variant of the homepage has a structurally higher turnaround time than the static variant.

Analyzing these different turnaround times there can be gathered more information about the tested page. The high peaks within Figure 5-6 reveal some about the stability of the dynamic tested page. The calculated average, median and standard deviation in Figure 5-8 clarify this.
The high average and the small median turnaround times reveal the effects of the peaks in the turnaround time data. The difference in the static and dynamic standard deviation turnaround times indicates that, under a high workload, the dynamic page is not as stable as the static page. Nevertheless, the standard deviation meets the 2nd requirement.

5.2.2 Mean Opinion Score results

To calculate the logarithmic MOS function, the good turnaround time is set to 12.98 ms (the average static turnaround time) and the poor turnaround time is set to 4000 ms.

The corresponding logarithmic MOS function is:

\[
MOS = 4.89112 - (0.34857 \times e^{\text{Log}_{TT}})
\]

Where:
- \(MOS\) = the mean opinion score (1 <= MOS <= 5)
- \(TT\) = the measured turnaround time in milliseconds (>= 0)
- \(e\) = the unique real mathematical constant (2.71828...)

The corresponding curve can be seen in Figure 5-9.
This MOS curve shows that the satisfaction of the user, in terms of MOS value, descends rapidly when the turnaround time increases. This has a direct influence on the calculated MOS values of both the static and dynamic page (see Figure 5-10 and Figure 5-11).
The data indicate that the dynamic homepage contained some peaks in turnaround time of almost 2000 ms. This seems to be not as close to the defined “poor” turnaround time of 4000 ms. The corresponding calculated MOS values though, have peaks up to 2.25. These MOS values are, in contrast to the corresponding turnaround times, close to the “poor” MOS value of 2.

The average MOS values of the dynamic and static homepages are not as close to each other as the *eurail_select_pass* page. The average MOS value of the static homepage is 4.28, while the average MOS value of the dynamic homepage is 3.62. This means that the dynamic homepage did not meet the defined 1st requirement of a MOS of 4.
6. Conclusions

Within this thesis, the performance evaluation method of Ulrich Herzog has been used to evaluate the performance of both a website using a Content Management System and an identical static website. Also method originally designed to indicate the quality of multimedia data services, the mean opinion score, has been adapted in order to use it within this performance evaluation.

Within the introduction I have explained that current use of a Content Management System to run a website is more a rule then an exception. Is the quality of this websites still acceptable compared to its static variant? To put concisely, does dynamic performance meet static performance?

To investigate this question, the high traffic website Eurail.com, which runs on the Content Management System called Smartsite, was analyzed and subjected to several load tests. These load tests were completed on the seven most visited pages within the website. The exact same tests were also completed on the static variant of the website (i.e. on the same hardware under the same circumstances).

Turnaround times of both the static and dynamic pages were compared and the numerical quality value, the MOS, calculated for each page. Figure 6-1 presents the final MOS of each tested static and dynamic page.

The main requirement of each dynamic page was to keep up with the static page: the MOS should be at least 4 (“good”). Figure 6-1 displays that only two of the seven tested dynamic pages did not meet this requirement. The page “1_where_to_go” had an MOS of 3.93 and the most frequently visited “Home” page had an MOS of 3.62.

Although these two pages failed to meet predetermined requirements, differences in average turnaround times for these pages compared to the static variants, showed that turnaround times only varied between approximately 12 and 43 milliseconds. While these differences may appear to be significant, to a (human) website visitor these differences are negligible.
Figure 6-1: MOS values of all tested pages

Figure 6-2: Average turnaround times of all tested pages
According to the predetermined criteria, the website has failed the performance evaluation. However, it could be argued that an MOS value of 4 may have been too high and that an MOS higher than 3.5 is also adequate. If the website is judged according to this lower MOS value, it has met the performance requirement.

The results indicate that there is a significant difference between the static and dynamic pages. The website with connection to a CMS has slower turnaround times than its static variant and according to the measures of central tendency the turnaround times are less stable. Despite these discrepancies, the actual differences are negligible for a website visitor.

In answer to the research question, in my opinion, the tested website (Eurail.com) with CMS provided a performance comparable to its static variant. As this study only investigated a single website and a solitary CMS, no definitive conclusions can be made based on this study. This being said, my experiences with other CMSs (such as Sitecore and GX) and the results of this study, lead me to have high performance expectations of other modern CMSs.

I believe that nowadays, dynamic does meet static.
Appendix 1: The ISO 9126-1 Time behavior Indicators

batch turnaround time
The time that passes between start and finish of background processing.

batch capacity
Number of information items which can be processed sequentially during background processing.

processing time
The average and maximum time a user needs for a certain processing task, with a certain usage load.

processing capacity
Number of processing tasks of a certain type that the user can perform during a certain period with a certain usage load.

average internal transaction time
Average time a certain internal processing task occupies with a certain usage load.

maximum internal transaction time
Maximum time a certain internal processing task occupies with a certain usage load.

turnaround time
Indicates the speed of processing by measuring the elapsed time between the beginning of process requirement and gaining the result of the process.

response time
Indicates the speed of processing by measuring the elapsed time between the end of inquiry or request to the computer system and the start of response.

CPU elapsed time
Indicates the speed of processing by measuring the elapsed time between the start of program execution and the end of program execution.

CPU execution time
Indicates the speed of processing by measuring the CPU time between the start of program execution and the end of program execution.
I/O processing time
The time required for I/O operation between main memory and external storage.

waiting time
The time between program stop by the interruption of O/S or resources, and program restart.

network processing time
Indicates the speed of network processing by measuring the time from the beginning of data transmission to the end of the transmission between host computer and terminal.

terminal processing time
The time between the beginning of terminal processing and the end of terminal processing.

throughput
Indicates the ability of the system by measuring the amount of jobs processed in a unit of time.

number of processed transactions
Indicates the ability of the system by measuring the number of transactions done in a unit of time.
Appendix 2: CBMG - Eurail.com

The following CBMG has been set up using visitor statistics from the Eurail.com live website.
Appendix 3: Experimental setup

This appendix presents the experimental setup from the performance evaluation of Eurail.com. It explains the environment in which the tests have been accomplished and the tools used within these tests.

A3.1 Environment used

The staging environment of the tested website (Eurail.com) has been selected to perform the load test on.

The staging environment exists of the same website and CMS implementation as on the live environment. It can only being accessed by authorized users (e.g. website owners and server administrators) and can be used as a (temporary) stage to test new or revised functionalities before they are made live.

The use of this staging server for this experiment has two advantages:

1. When something goes wrong during the test it won’t damage the live environment. A normal internet user will not notice anything of this experiment.
2. Since the staging server will be used to test (new) functionalities of the website before going live, the hard- and software are similar to those on the live environment.

This staging environment of the tested website exists of one dedicated web server and one shared database server.

These servers have the following technical details:

Dedicated web server:
- Web server: IIS6
- Intel Xeon CPU 2.66 GHz
- 768 MB. RAM

Shared database:
- Database server: MS SQL Server 2005
- Intel Xeon CPU 2.33 GHz
- 4 GB. RAM
The web root of the web server exists of the program code of the tested website with CMS Smartsite to run the website (dynamic website) as well as this exact same website without CMS (static website). The tested website has been made static using the (free) tool HTTrack Website Copier [Httr] (see A3.2.1 for more details).

A laptop (which is connected to the internet via wired LAN) has been used to generate the different page visits (load) and gain the required response data. The (free) tool Apache JMeter [Jmet] has been installed and configured to perform these steps (see A3.2.2 for more details).

This laptop has the following technical details:
- OS: Windows XP Professional; Service Pack 3
- Tools installed: HTTrack 3.41-2, JMeter 2.3.1
- Intel CPU; 1.83 GHz.
- 2 GB. RAM
A3.2 Tools used

The following tools have been used in order to accomplish the different load tests in this research.

A3.2.1 HTTrack Website Copier

*Used version within this research: 3.41-2*

HTTrack allows you to download a World Wide Web site from the Internet to a local directory, building recursively all directories, getting HTML, images, and other files from the server to your computer. HTTrack arranges the original site’s relative link-structure [Httr].

HTTrack has been used to make a static version of the tested website. First, the dynamic version was downloaded (using HTTrack) to a local directory. Second, this local directory was uploaded to the original web server of the tested website. Doing this, the web server contains a dynamic version of the website which gains its content out of the CMS, as well as a static version which has no (dynamic) connection with the CMS at all.

A3.2.2 Apache JMeter

*Used version within this research: 2.3.1*

JMeter can be used to simulate a heavy load on a server, network or object to test its strength or to analyze overall performance under different load types. You can use it to make a graphical analysis of performance or to test your server/script/object behavior under heavy concurrent load [Jmet].

JMeter was used to perform a load test on the static and dynamic website. Both the static and the dynamic website were tested (on the same hardware within the same environment) and were exposed to a specific (previously characterized) workload within two test rounds of 15 minutes. The results of each request were saved in an XML file. Later, on this XML data was used to calculate, visualize and compare the different test results.
Appendix 4: Performance Evaluation Results

This appendix consists of all test results from the performance evaluation of Eurail.com.

The following abbreviations and terms are used:

- **rps**: requests per second
- **ms**: milliseconds
- **time static**: the results of the test with the static web-site (without CMS)
- **time dynamic**: the results of the test with the dynamic web-site (with CMS)

**A4.1 Test results including mathematical and statistical values**

This section presents the results of all tested pages. Each tested page contains:
- a single graph with all measured turnaround times for both static and dynamic website;
- a single graph with the average, median and standard deviation of the turnaround time for the different websites.

*Note: Within some charts there are large peaks and many request information is hidden due to the scale of the graph. Within these cases, I have made a second chart with a smaller y-axis. These can be recognized with “< x ms”, where x is the maximum milliseconds on the y-axis.*
A4.1.1 Test results: 1_prepare_your_trip

Figure A4-0-1: Turnaround times page: 1_prepare_your_trip

Figure A4-0-2: Average, median and standard deviation results page: 1_prepare_your_trip
A4.1.2 Test results: eurail_select_pass

Figure A4-0-3: Turnaround times page: eurail_select_pass

Figure A4-0-4: Average, median and standard deviation results page: eurail_select_pass
A4.1.3 Test results: 1_global_pass

Figure A4-0-5: Turnaround times page: 1_global_pass

Figure A4-0-6: Turnaround times page: 1_global_pass (< 70 ms)
Figure A4-0-7: Average, median and standard deviation results page: 1_global_pass
A4.1.4 Test results: 1_our_products

![Figure A4-0-8: Turnaround times page: 1_our_products](image1)

![Figure A4-0-9: Average, median and standard deviation results page: 1_our_products](image2)
A4.1.5 Test results: eurail_regional_pass

Figure A4-0-10: Turnaround times page: Eurail_regional_pass

Figure A4-0-11: Average, median and standard deviation results page: Eurail_regional_pass
A4.1.6 Test results: 1_where_to_go

**Figure A4-0-12: Turnaround times page: 1_where_to_go**

**Figure A4-0-13: Turnaround times page: 1_where_to_go (< 50 ms)**
Figure A4-0-14: Average, median and standard deviation results page: 1_where_to_go
A4.1.7 Test results: Home

Figure A4-0-15: Turnaround times page: Home

Figure A4-0-16: Turnaround times page: Home (< 100 ms)
Figure A4-0-17: Average, median and standard deviation results page: Home
**A4.2 Mean opinion score (MOS)**

The following paragraphs presents the various MOSs. For each tested page the MOS chart is calculated, and the different MOS values for each request are shown.

**A4.2.1 MOS: 1_prepare_your_trip**

**A4.2.1.1 Defined MOS function**

Good turnaround time (MOS = 4): 9.48 ms.
Poor turnaround time (MOS = 2): 4000 ms.

Corresponding logarithmic MOS function:

\[
MOS = 4.74416 - (0.330859 \times e^{\log_{TT}})
\]

Where:
- \(MOS\) = the mean opinion score (1 <= MOS <= 5)
- \(TT\) = the measured turnaround time in milliseconds (>= 0)
- \(e\) = the unique real mathematical constant (2.71828...)

![MOS curve: 1_prepare_your_trip](image-url)
A4.2.1.2 MOS values per request

Figure A4-0-19: Static MOS values: 1_prepare_your_trip

Figure A4-0-20: Dynamic MOS values: 1_prepare_your_trip

Average static MOS: 4.30
Average dynamic MOS: 4.27
A4.2.2 MOS: eurail_select_pass

A4.2.2.1 Defined MOS function

Good turnaround time (MOS = 4): 9.31 ms.
Poor turnaround time (MOS = 2): 4000 ms.

Corresponding logarithmic MOS function:

\[ \text{MOS} = 4.73597 - (0.329872 \times e^{\log TT}) \]

Where:
MOS = the mean opinion score (1 <= MOS <= 5)
TT = the measured turnaround time in milliseconds (>= 0)
e = the unique real mathematical constant (2.71828...)

Figure A4-0-21: MOS curve: eurail_select_pass
A4.2.1.2 MOS values per request

Average static MOS: 4.30
Average dynamic MOS: 4.01
A4.2.3 MOS: 1_global_pass

A4.2.3.1 Defined MOS function

Good turnaround time (MOS = 4): 11.57 ms.
Poor turnaround time (MOS = 2): 4000 ms.

Corresponding logarithmic MOS function:

\[ MOS = 4.83769 - (0.342136 \times e^{\log_{TT}}) \]

Where:
- \( MOS \) = the mean opinion score (1 <= MOS <= 5)
- \( TT \) = the measured turnaround time in milliseconds (>= 0)
- \( e \) = the unique real mathematical constant (2.71828...)

Figure A4-0-24: MOS curve: 1_global_pass
A4.2.3.2 MOS values per request

Average static MOS: 4.20
Average dynamic MOS: 4.00
A4.2.4 MOS: 1_our_products

A4.2.4.1 Defined MOS function

Good turnaround time (MOS = 4): 10.12 ms.
Poor turnaround time (MOS = 2): 4000 ms.

Corresponding logarithmic MOS function:

\[ MOS = 4.77414 - (0.334474 \times e^{\log_{TT}}) \]

Where:
- \( MOS = \) the mean opinion score (1 <= MOS <= 5)
- \( TT = \) the measured turnaround time in milliseconds (>= 0)
- \( e = \) the unique real mathematical constant (2.71828...)

Figure A4-0-27: MOS curve: 1_our_products
A4.2.4.2 MOS values per request

Figure A4-0-28: Static MOS values: 1_our_products

Figure A4-0-29: Dynamic MOS values: 1_our_products

Average static MOS: 4.26
Average dynamic MOS: 4.12
A4.2.5 MOS: eurail_regional_pass

A4.2.5.1 Defined MOS function

Good turnaround time (MOS = 4): 10.52 ms.
Poor turnaround time (MOS = 2): 4000 ms.

Corresponding logarithmic MOS function:

\[ MOS = 4.79225 - (0.336657 \times e^{\log_{TT}}) \]

Where:
- MOS = the mean opinion score (1 <= MOS <= 5)
- TT = the measured turnaround time in milliseconds (>= 0)
- e = the unique real mathematical constant (2.71828...)

Figure A4-0-30: MOS curve: Eurail_regional_pass
A4.2.5.2 MOS values per request

**eurail_regional_pass - 0.22 rps - Static MOS**

![Static MOS values: Eurail_regional_pass](image1)

**eurail_regional_pass - 0.22 rps - Dynamic MOS**

![Dynamic MOS values: Eurail_regional_pass](image2)

**Average static MOS:** 4.27

**Average dynamic MOS:** 4.00
A4.2.6 MOS: 1_where_to_go

A4.2.6.1 Defined MOS function

Good turnaround time (MOS = 4): 9.29 ms.
Poor turnaround time (MOS = 2): 4000 ms.

Corresponding logarithmic MOS function:

\[ MOS = 4.735 - (0.329755 \times e^{\log_{TT}}) \]

Where:
- \( MOS \) = the mean opinion score (1 <= MOS <= 5)
- \( TT \) = the measured turnaround time in milliseconds (>= 0)
- \( e \) = the unique real mathematical constant (2.71828...)

Figure A4-0-33: MOS curve: 1_where_to_go
A4.2.6.2 MOS values per request

Average static MOS: 4.30
Average dynamic MOS: 3.93
A4.2.7 MOS: Home

A4.2.7.1 Defined MOS function

Good turnaround time (MOS = 4): 12.89 ms.
Poor turnaround time (MOS = 2): 4000 ms.

Corresponding logarithmic MOS function:

\[ MOS = 4.89112 - (0.34857 \cdot e^{\log_{TT}}) \]

Where:
- \( MOS \) = the mean opinion score \((1 \leq MOS \leq 5)\)
- \( TT \) = the measured turnaround time in milliseconds \((\geq 0)\)
- \( e \) = the unique real mathematical constant \((2.71828\ldots)\)

Figure A4-0-36: MOS curve: Home
A4.2.7.2 MOS values per request

Figure A4-0-37: Static MOS values: Home

Figure A4-0-38: Dynamic MOS values: Home

Average static MOS: 4.28
Average dynamic MOS: 3.62
References

[Alme02] Virgilio A.F. Almeida
Capacity Planning for Web Services, Techniques and Methodology
Performance Evaluation of Complex Systems: Techniques and Tools, Lectures in Computer Science Vol. 2459,

[Balb01] Gianfranco Balbo
Introduction to Stochastic Petri Nets

[Barb07] Scott Barber
How Fast Does a Website Need To Be?
Ongoing research, PerfTestPlus, Inc., 2007
www.perftestplus.com/resources/how_fast.pdf

[Bos05] R. Bos
User Interface Design for Content Management Systems
Master’s Thesis, Utrecht University, INF/SCR-04-84, July 2005

[Britt] Encyclopedia Britannica Online
www.britannica.com

[CBS07] Centraal Bureau voor de Statistiek (CBS) the Netherlands
www.cbs.nl

[Crem+02] Paolo Cremonesi, Giuseppe Serazzi
End-to-End Performance of Web Services
Performance Evaluation of Complex Systems: Techniques and Tools, Lectures in Computer Science Vol. 2459,

[Dale] Van Dale Dutch dictionary
www.vandale.nl

[Ferr78] Domenico Ferrari
Computer Systems Performance Evaluation
[Gali04] Daniel Galin
Software Quality Assurance

[Herz01] Ulrich Herzog
Performance Issues of Multimedia Applications

[Httr] HTTrack Website Copier
www.httrack.com

[Itco08] ITcommercie Web Content Management Special
ITcommercie, 2008

[Jmet] Apache JMeter
http://jakarta.apache.org/jmeter/

[Jupi06] Jupiter Research
Retail Web Site Performance, Consumer Reaction to a Poor Online Shopping Experience
Jupiter Research, June, 2006

[Kula+03] Daryl Kulak, Eamonn Guiney
Use Cases: Requirements in Context, Second Edition

[Leao+00] Rosa M.M. Leão, Edmundo de Souza e Silva, Sidney C. de Lucena
A Set of Tools for Traffic Modeling, Analysis and Experimentation

[Losa+03] Francisca Losavio, Ledis Chirinos, Nicole Lévy and Amar Ramdane-Cherif
Quality Characteristics for Software Architecture

[Mena02] Daniel A. Menascé
Load Testing of Web Sites
IEEE Internet Computing, July-August 2002, page 70-74
[Niel94] Jakob Nielsen
Usability Engineering

[Paix+00] G.T. Paixão, W. Meira Jr., V.A.F. Almeida, D.A. Menascé, A.M. Pereira
Design and Implementation of a Tool for Measuring the
Performance of Complex E-commerce Sites
Computer Performance Evaluation: Modeling Techniques and
Tools, Lecture Notes in Computer Science Vol. 1786,

[Vos06] Controlling the KPI <-> Cost balance using network counters
Presentation by Esther Vos, senior consultant Planning
Performance and Quality, TNO, the Netherlands, 2006

[Well02] Lisa Wells
Performance Analysis using Coloured Petri Nets
PhD Dissertation, Department of Computer Science, University of
Aarhus Denmark, July 31, 2002

[Wolt03] Katinka Wolter, Kristian Kasprowicz
WebAppLoader: A Simulation Tool Set for Evaluating
WebApplication Performance
Computer Performance Evaluation: Modeling Techniques and
Tools, Lectures in Computer Science Vol. 2794,

[Zeis+96] B. van Zeist, P. Hendriks, R. Paulussen, J. Trienekens
Kwaliteit van softwareprodukten (druk 1)

A Comprehensive Toolset for Workload Characterization,
Performance Modeling, and Online Control
Computer Performance Evaluation: Modeling Techniques and
Tools, Lectures in Computer Science Vol. 2794,