1. Introduction

An application server is a framework that provides developers with facilities to create web applications and a server on which we can run them. Application Server Frameworks contain a detailed service layer model where the application server acts as a group of components accessible to the code developer through a regular API outlined for the platform itself. For web applications, these parts are sometimes performed within the same running environment as their web server(s), and their main job is to support the development of dynamic pages. However, several application servers target way more than simply web page generation: they implement services like clustering, fail-over, and load-balancing, thus developers can concentrate on implementing the business logic. Application server also refer to the computer hardware on which the services run.

Application servers are platforms where web applications or desktop applications run. Application servers comprise of web server connectors, PC programming languages, runtime libraries, database connectors, and the organization code expected to be deployed, design, oversee, and connect these parts on a web host. An application server keeps running behind a web server (e.g. Apache or Microsoft Internet Information Services (IIS)) and quite often before a SQL database (e.g. PostgreSQL, MySQL, or Oracle). Web applications are codes which keep running on application servers and are composed in the language(s) the application server supports and call the runtime libraries and parts the application server offers.

Numerous application servers exist. The decision of choosing a server architecture impacts the cost, execution, reliability, adaptability, and viability of a web application. Exclusive application servers give system benefits in an all-around characterized yet proprietary manner. The application engineers create programs as per the specification of the application server. This project aims to compare the performance of a selected set of web server technologies and analyse the results based on various circumstances and conclude with which web server architecture should be used under what situations. The benchmarking of these various web technologies will be done using two popular benchmarking tools called "LoadRunner" and "JMeter". The project also aims to compare the two benchmarking tools, analysing their features and results.

1.1. Purpose of the project

In the rapid development of Web today, many sites are faced with new problems, such as the problem of multiuser requests and high concurrency. The dynamic scripting language JavaScript has become enormously popular for client and is widely used in Web development. Node stands for one new technology in JavaScript. Node is a platform built on Chrome's JavaScript runtime

for easily building fast, scalable network applications [**?**]. Node.js uses an event-driven, non- blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices [**?**]. Node.js popularity surveys performed by official website indicate that the average downloads are over 35,000 since the version 0.10 released in March 2013. Corporations are quickly realizing the importance of Node.js and five major PAAS providers have supported Node.js [**?**]. Nowadays, JavaScript has been the first popular language in GitHub with 177,352 repositories and growing [**?**]. And talking about evaluation of Web technologies' performance, many researchers have done the related work. But the work described in this thesis differs from others in two aspects. Firstly, we consider from both objective systematic tests (benchmark) and realistic user behaviour tests (scenario).

Not long ago even a 2-second page response time was considered as an acceptable one. However, web users have become increasingly impatient when it comes to speed these days. Earlier, speed was considered a feature and now it is deemed a necessity. Additionally, technological innovation in mobile space has raised the bar for speed. Hence, speed makes a lot of economic sense now. A recent research found that 250-450 milliseconds are the numbers that decide the winner in the race for web speed. Research also indicates that the slower the site, the lesser would be the number of clicks and transactions performed on the site which would eventually result in the loss of users.

In order to perform these tests on the various Web technologies we will make use of the testing tools such as JMeter and LoadRunner. In addition to testing the various Web technologies, this project will also focus on the quality of the testing tools, the differences between them and if they produce different results under certain circumstances. Complex systems make increasing demands on web servers, high volumes can overwhelm systems if they are not scaled correctly. Multiple objects can interfere when one process is handling a request for a specific user. Fixes need to be identified early in the project so that server crashes and vulnerabilities can be caught in advance Clients have scalability concerns and we must warranty some level of scalability with industry accepted metrics. In order to achieve this, our web servers.

This paper focuses on the impact on Web performance from three different Web technologies: Node, PHP and .NET. The security and scalability issues are beyond the scope of the thesis. We mainly use the benchmark tests and scenario tests. In addition, one universal method of Web development techniqueâĂŹs evaluation based on the performance comparison is proposed in the paper, which can be used to evaluate any new Web technology. The main contributions of this paper are listed as follows.

1. We consider new web technologies like Node, PHP and .NET in our experiment and analyse the results of it. Then we compare them making a conclusion of which situation they ought to be used in.
2. By means of benchmark tests and scenario tests, we can evaluate performance from both objective systematic tests (benchmark) and realistic user behaviour tests (scenario). There is often a dual impact on Web server performance, from

the calculation, and from the number of users. The research herein has taken each of these effects in account.

The rest of this research is organized as follows - Chapter 2 discusses related work. Chapter 3 describes the test bed and configurations in the research. Chapter 4 details the methodology and experimental design of tests. Chapter 5 presents and analyses the results of all tests. Chapter 6 makes a conclusion of the paper with a summary of study and a future direction.