Web Performance Testing and Measurement: a complete approach

G. Cassone, G. Elia, D. Gotta, F. Mola, A. Pinnola

www.telecomitalialab.com

Abstract

The fast growing number of Web sites for mission critical applications (e-commerce, e-business, contents management and advertisement) makes web site performance and availability a business need, and a distinctive feature for success in the market. Poor performance or low availability may have strong negative consequences both on the ability of the company in attracting and retaining its own customers and on its ability to obtain high revenues from the investment in new technology. Controlling performances of web site and back-end systems (where e-business transactions run) is a key factor for every company.

During recent years, TILAB has progressively gained considerable methodological & practical experience and know-how in Performance Testing and Measurement of Computer systems including the newest Web Performance Testing and Measurement Techniques.

Web Performance Testing is executed through testing campaigns for stressing the web site and back-end systems with the amount of load simulating the real conditions of the field or to evaluate if the site/application will support the expected load following special situations (e.g. an advertisement campaign). That allows you to guarantee system performances under that load and to identify and help in fixing possible issues.

Web Performance Measurement aims at analysis and fast characterization of system and user behaviour in order to give fast feedback on any issues. In order to achieve that, TILAB has realized two tools: WEBSAT (WEB Server Application Time) and BMPOP. WEBSAT is based on web server log files and completes the tool suite of the commercial off the shelf products used for Web Performance Evaluation/Measurement activities. BMPOP (BenchMark Point Of Presence) is used for Web Performance Measurement from the end-to-end perspective.

This paper shows the present status of TILab approach to Web Performance Testing and Measurement. The paper also presents two Case Studies applied by TILab to Web Performance Evaluation of Telecom Italia SDH Network Management System and to the Web Site of an important Italian telecommunication service operator. Finally some Conclusions are given in order to clarify the methodological and experimental trends foreseen by TILab in the fast growing field of Web Performance Testing and Measurement.

1. Introduction

Thanks to the fast evolution of internet-based technologies during recent years, many Companies are migrating their mission-critical applications towards the WEB, and they choose the WEB as an important source of earnings. But to be successful, they need effective tools, resources and methodologies to provide the user with a high service level.

Passing from LAN Client-Server (C/S) to Internet Web-based applications, your audience increases but your risks increase as well. So, now, application performances need greater attention. In the USA a survey has found that a user waits just 8 seconds for a page to download completely before leaving the site. In Italy, this limit may be higher, but anyway, providing high performance is a key factor in the success of the site.

A web application is different from an “old” C/S application: lack of knowledge of clients, lack of knowledge of users, more risks, a more complex architecture (firewalls, proxies, DNS, etc).

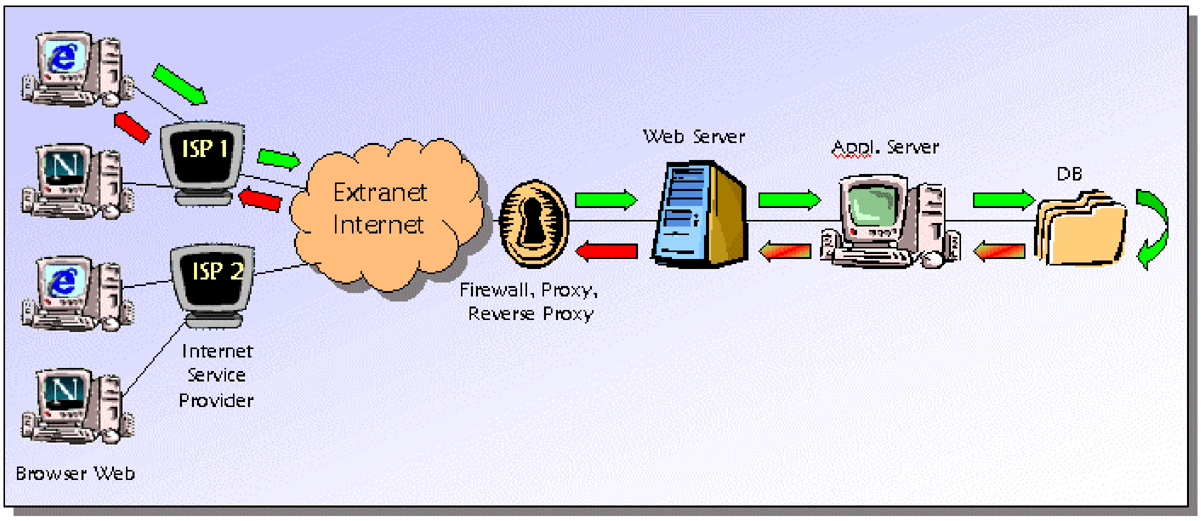


Figure 1: Architecture of web-based applications

To measure web application performance effectively and define what improvement it needs, it’s quite important to take into account its architecture (see Figure 1). This includes:

- **Web Browser**: the software client on which web applications run. It’s independent of the application.
- **Internet Service Providers (ISP)**: provide different speed and type of Internet access.
- The “**Big Internet**” is the world wide communication infrastructure between browser and server web.
- A **Firewall** is the interface between company intranet and Internet. They can filter the ingoing and outgoing traffic according to some rules defined by the web administrator.
- Other network elements can be found on the boundary between intranet and Internet: **proxies and reverse proxies**. Proxies help an intranet client finding an external site and besides they have the important function of caching, keeping in memory the most requested pages from the users of LAN. **Reverse-proxies** mask intranet internal servers providing to Internet some sort of virtual address that is forwarded to the right server.
- **Web Servers** is the applications able to meet requests from clients (browsers). It forwards the requests to the web-application that can be on the same machine or on another server devoted for that purpose (application server).
- **Application server** is the machine where the code of the applications runs. This machine can coincide with the web server, but if a company has many applications is better to allocate them on different servers, letting the web server the function of interface.
- The **Data-base** holds the data of the applications. Access data can heavy. When they manage many data, access time could be too high. For that reason it’s better to allocate a machine just for this function (DB server).

Even with this complex architecture, it's important to assure users with a good service level. This level, usually, is the response time that they experience in their browser loading a page or completing a transaction.

A survey over 117 companies with an income of, at least, $200 million highlighted that 94% of the companies who scaled correctly had done web performance testing, while just 32% of the companies that didn't scale correctly had done that activity. So web performance testing activity is important, but is also important how earlier it has been done during the development of the application.

This paper shows the present status of Telecom Italia Lab (TILab) approach to Web Performance Testing and Measurement. The paper presents also two Case Studies applied by TILab to the Web Performance Evaluation of Telecom Italia SDH Network Management System and to the Web Site of an important Italian telecommunication service operator.

Finally some conclusion are given in order to clarify the methodological and experimental trends foreseen by TILab in the fast growing field of Web Performance Testing and Measurement.

2. A complete approach to WEB Performance Testing & Measurement

Web Performance Testing and Measurement aim at providing a measure of the actual performances of a web application, and an evaluation of performances that the application could provide, following a change of load; identifying, moreover, possible bottlenecks and providing useful advice about how to fix problems (tuning of system parameters, modification of software or hardware upgrade). Next paragraphs list objectives and components needed to achieve them and all the key elements are highlighted in order to provide an effective client satisfaction.

2.1. Objectives

There are different objectives achievable with a web performance testing and measurement. End-users take advantage of a better application, system managers use that information to improve performances of their systems and, last but not least, management can obtain useful information about the business of their company.

"End-User objectives":

❖ To find average response time of pages and transactions, slowest and fastest pages;

❖ To make sure main pages (e.g. home page) can be downloaded within acceptable time (e.g. 10 seconds);

❖ To find out maximum number of concurrent users, sessions and transactions that the application is able to support still providing a high level of service;

❖ To find out maximum number of concurrent users, sessions and transactions that the application is able to support without system crash;

❖ To characterize more frequent user paths, the most used starting and exiting page;

❖ To identify main reasons of site abandonment.

"System" objectives:

❖ To correlate system resource utilization with load;

❖ To find out possible actual hardware bottlenecks and prevent new ones (capacity planning);

❖ To tune all the web application components to support as much load as possible using actual hardware;

❖ To find out how the application works when overloaded;

"Management"objectives:

❖ To provide an objective measure of the usage of the site (e.g. for an e-commerce site, it could be the number of electronic carts and the number of objects sold);

❖ To provide a "business view" of the previous data (e.g. how performance issues have affected the business);

2.2. Components

To achieve the objectives just described, a complete approach is needed. This approach should be based on sound methodology, a set of tools and a high level of know-how. The Telecom Italia Lab approach is based on the following components:

- **Web Performance Objectives** – to identify, according to the previous classification (end-user, system, management), what kind of results you need. Actually it's possible to address all three categories or just one or two;
- **Web Performance Testing** – to verify, in test plant, that the application is able to support the entire load expected and even more (e.g. the load could increase in an unpredictable way after an advertising campaign). For a more detailed analysis, this activity could be done both inside and outside proxy and firewall;
- **Web Performance Measurement** – to measure the actual performance of the application in the field. This could be done by identifying user behaviour, back-end system response time, maximum number of concurrent users (**Web Log Analysis**), monitoring, at the same time, hardware and software resources utilization (**Resource Monitoring**) and the end user experience (**End-to-end Monitoring**);
- **Problem Resolution and Activity Results** – to support client by providing the source required to fix any bottlenecks;
- **Capacity Planning** – to ensure that adequate hardware and software resources will be available when needed in the future. Capacity planning activity is carried out using all information from other components

All the components just described can be combined according to the needs of the client. Figure 2 shows a possible sequence utilizing the components.

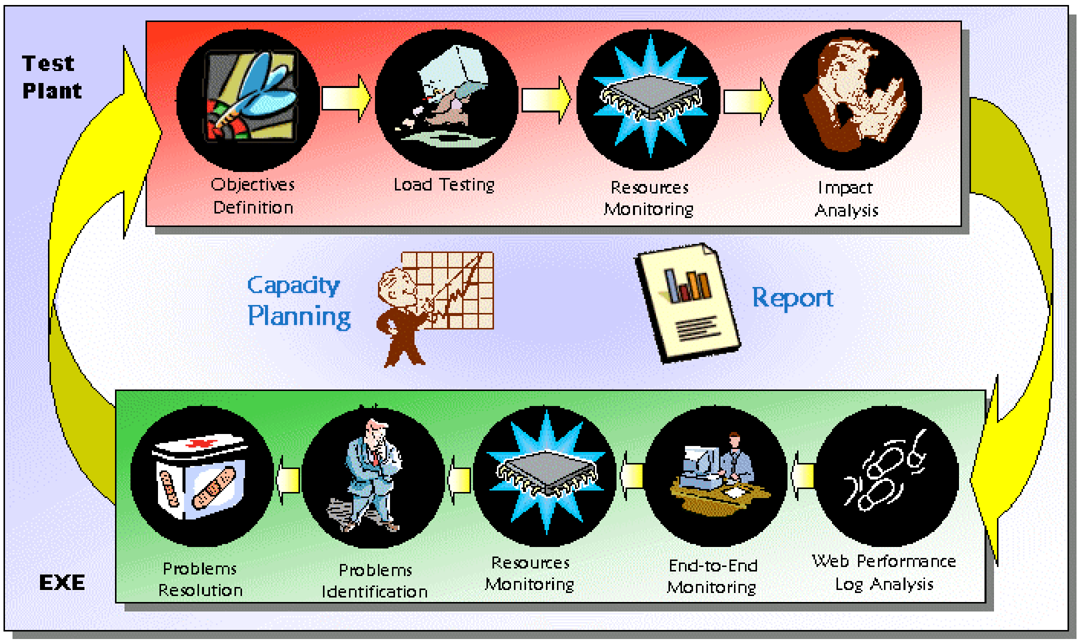


Figure 2: Example of a complete approach

2.3. Test environments and Measurement issues

Experience has taught us that is very useful to have a test plant, a "twin environment", where you can

- Test critical activities like release changing;
- Test greater load than that experienced in field (it's not always feasible to do that job in the field);
- Tune all your systems in order to minimize the risks of a direct introduction of a new release in the field.

A test plant is expensive, but you can gain lot of advantages. Besides, all information extracted from the field can be used afterwards to calibrate activities in test plant (only by monitoring the field is possible to extract correct information about user profile).

To make sure that web performance testing and measurement activity provides useful results for an effective evaluation of performances and bottlenecks, in order to predict behaviour of the application changing the load, it's important to have the following data:

- **Response Time**: time for downloading pages and performing main transactions on the user side (e.g. to download a page) and of the back-end systems side (e.g. to access a DB);
- **Application Data**: a measure of the service provided (e.g. for an e-commerce site it could be the number of the electronic carts and the number of objects sold);
- **User profile**: all information useful to characterize the end-user behaviour (most and least requested pages, paths followed during navigation, bandwidth, session average time,...);
- **Resources Data**: all the monitoring data of resources of the servers on which the application runs (e.g. Cpu utilization, Disk utilization, run queue, etc);
- **System Management Activity**: a summary of all the system management activity carried out over all the components of the architecture of the application during the testing period (system restart, tuning of buffers, of cache, of the hardware and of other system parameters);
- **Expected load**: expected growth of the load according to the valuation of the marketing department;

3. Discussion of the components

This chapter presents all the components of the complete approach, described briefly in the previous chapter, and also provide a brief description of all the activities involved and the tools used.

3.1. Web Performance Objectives

A Web Performance Testing and Measurement Activity should begin with a discussion and definition of the activity's objectives and a review of the established performance objectives of the application or system. In Chapter 2 general objectives have been discussed; specific objectives will be defined due to the multiple perspectives that are involved in the process. The output of this activity should be a document that recalls the scope and the objectives of the activity and the expected behaviour of the application or of the system, if relevant.

3.2. Web Performance Testing

The aim of Web Performance Testing [4] is to evaluate performances of the web application and all the back-end systems (DB and application server) changing the load. Performance testing activity guarantees system performances according to a defined load, identifying where response time is too high.

Automated tools are used to generate and emulate the load of the end users. These kinds of tests are very useful to verify, in test plant, new releases of the application or to stress it with a greater load (useful, for example, before the launch of a new advertising campaign). An artificial traffic is generated to reproduce the activity of a certain number of users of different types and, if long response time are measured, to find out all the system bottlenecks. After a first tuning activity, all the tests should be executed again.

All the test activities can be executed on-site or in a remote way. The focus of the first kind of activity is the application (all the network involved between server web and browser is not considered) and it's possible to extract lot of useful and detailed information (especially if, during the test, all the resources of the systems involved are under monitoring). With a remote test, as all the elements involved are considered (user browser, ISP, network and application) it's possible to obtain an end-to-end measure of performance, but it's impossible to get lot of details on every element.

According to the type of load and timing, it's possible to identify different types of tests:

- **Smoke Test** – A brief test, just to check if the application is really ready to be tested (e.g. if it takes 5 minutes to download the home page it isn't worth going on to test other pages)
- **Load Test** – For these kinds of tests, the application is subject to a variable increasing load (until the peak load is reached). It's useful to understand how the application (software + hardware) will react in the field.
- **Stress Test** - For these kinds of tests, the application is subject to a load bigger than the one actually expected. It's useful to evaluate the consequences of an unexpected huge load (e.g. after an advertisement campaign)
- **Spike Testing** – For these kinds of tests, the application is subject to burst loads. It's useful to evaluate applications used by lot of users at the same time (high concurrent user rate)
- **Stability Testing** – For these kinds of tests, the application is subject to an average load for a long period of time, It's useful to find out problems like memory leaks.

Web Performance Testing Tools

Automatic tools are used to execute performance testing when you need to simulate real user activities. They work as follow:

- Identify transactions and pages to test;
- Record user activity (tool feature). A script will be generated;
- Identify data and parameters of the application (workload characterization);
- Modify the script according to the data just identified, in order to reflect the activity of several users (e.g. in an e-commerce site, every user buys different things);
- Playback the script increasing the numbers of simulated users (virtual users);
- Extract response time of main user transactions or pages.

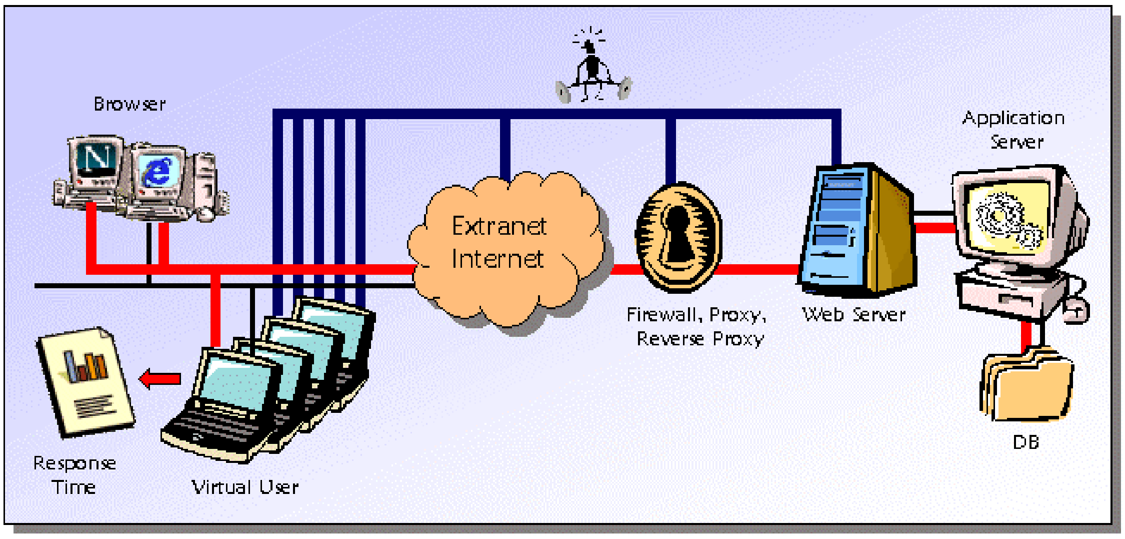


Figure 3: Performance Testing Tools

It's important to create workloads with high accuracy and as real as possible. That means knowing correctly the real user profile. The newest Web Log Analysis techniques and tools (see next paragraph) can help to solve this problem.

3.3. Web Log Analysis

Introduced in the previous paragraph, Web Log Analysis (WLA) is a relatively new activity that can help to create a real and accurate workload. Every web server stores a lot of information in a web server log file.

This file contains detailed information about the number of accesses to the site, geographical classification of users, distribution of accesses over the day, average navigation time for every user, page visited, user paths, ingoing and outgoing pages, browsers used, keywords used searching for the site, Operating systems, etc.

Using all that information it is possible to create workloads that can really simulate user activities.

Web Log Analysis Tools

There are many commercial web log analysis tools. Some of them are complex. They let you extract detailed information, useful if you need to attract more users (marketing department). There are some other simpler tools that don't require many configuration activities that are easy to use and, anyway, let you extract all the information you might need to create an accurate workload.

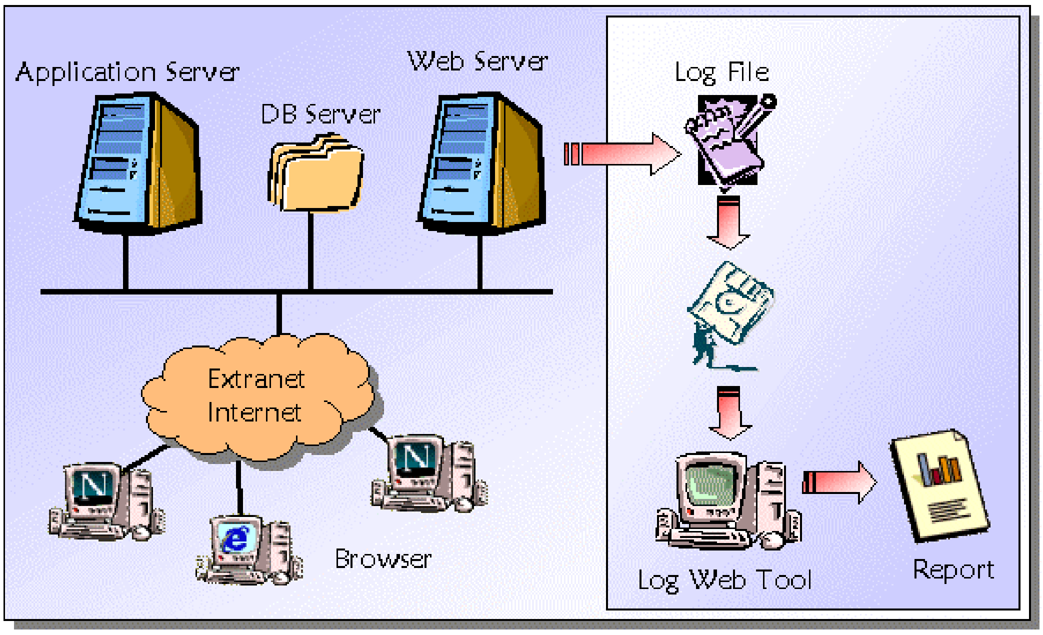


Figure 4: Web log analysis tools

TILab noticed that most commercial WLA tools don't use all the information of a web log file. Among them there's a value that represents back-end system response time. That is the time a server web takes to execute a request coming from the browser. This is not the response time experienced by the user, but it's a useful metric of performances of the application (e.g. if a cgi script takes 2 minutes to be executed by the back-end systems it isn't surely working well).

To fill that gap, TILab has created **WEBSAT** (Web Server Application Time). WEBSAT is a software that uses web server log files to produce a table containing the user, and the IP addresses, of the users and, mainly, the execution time of every object requested by the browser.

3.4. End-to-End Monitoring

Using a complete approach means evaluating performances provided by every element of the chain from the user to the back-end systems. This implies that it's also important to monitor performances of the network, and the Internet Service Provider (ISP), that users use to connect to your site. If one ISP requires a longer access time than others, it could be published to induce users to change their ISPs.

End-to-end monitoring means to measure the availability and response time of different ISPs. There are two different approaches: Java Instrumentation and Ghost Instrumentation.

Java Instrumentation – is based on java-applets installed on the web server downloaded by the browser of the user. These applets record all response times experienced by the user and send them to a manager installed on the same LAN of the server web to produce reports. This solution is not completely "non-intrusive" but let you measure the real response time of every user of your application.

Ghost Transaction – is based on real clients installed on the Internet according to the real geographical distribution of users (this information can be extracted by a WLA tool). These clients are trained to execute the same transaction and activity of users, so they act as a probe on the Internet. This approach is completely “non-intrusive”, but needs a distribution of clients that should reflect the user distribution.

End-to-End Monitoring tools

There are two different kinds of end-to-end monitoring tools (intrusive and non-intrusive according to the classification described in the previous paragraph). They provide reports about the end-to-end response time (i.e. the response time experienced by the user).

TILab has created **BMPOP** a software that is able to simulate an user who uses a connection to an ISP to download a web page.

BMPOP is based on the Ghost transaction. It needs a network of clients to simulate the geographical distribution of users. At a fixed period of time, BMPOP tries to connect to Internet through one or more ISPs and tries to download a page of the site under test. Reports show information about ISP (connection availability) and information about the access time to download the page (end-to-end measurement).

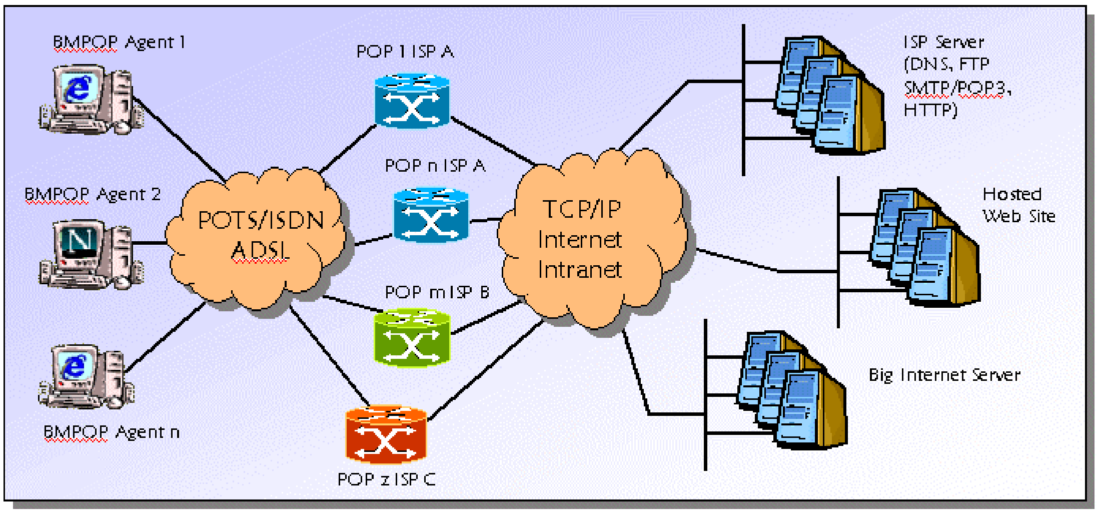


Figure 5: Transactions measured by BMPOP

3.5. Resources Monitoring

A complete approach requires that every single parameter of the system must be monitored. One of the main causes of poor performances is an excessive utilization of hardware and software resources. All systems, performing their activities use system resources like cpu, disks, memory etc. It’s important to correlate resources utilization with all the activity that the application does. This helps to find out if the system, where the application runs, needs software tuning or hardware upgrade.

A performance monitoring should be executed both during a load test and, periodically, just to supervise the normal activity in the field.

Resource Monitoring Tools

Resource monitoring tools need two kinds of software (see Figure 6): a client (for every server we want to monitor) and a manager. The client stores all the resource utilization information about his server. The manager collects all these information (metrics) and allows the user to generate lot of graphics. There are about three hundred metrics available in a tool like this, but only a few are essential (cpu utilization, disk utilization, memory page out activity, run-queue utilization, pagination activity).

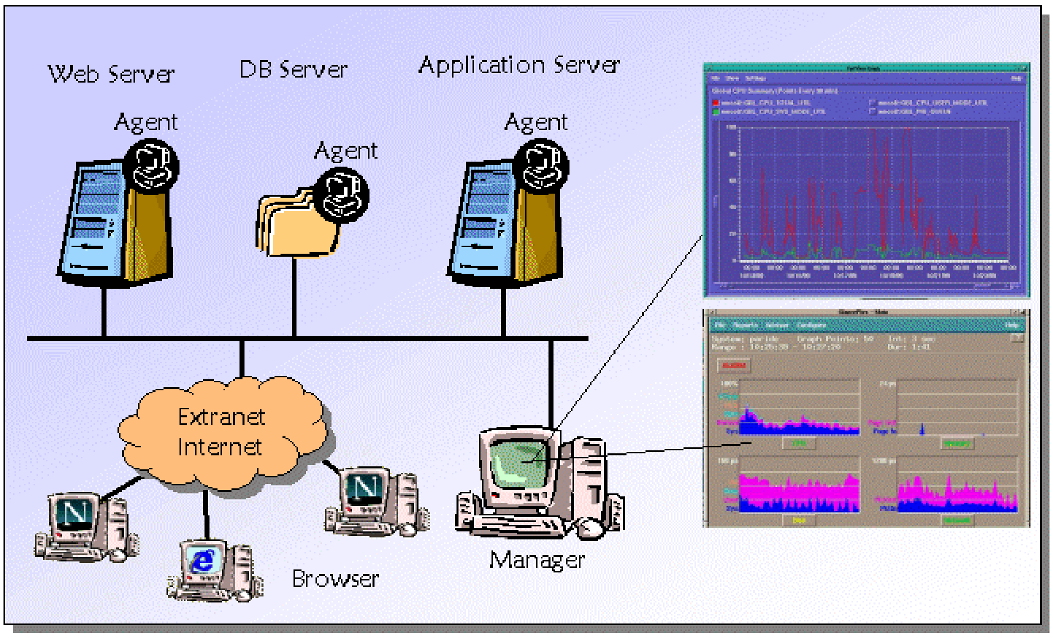


Figure 6: Resource monitoring tools

3.6. Capacity Planning

Capacity Planning is the process of ensuring that adequate hardware and software resources will be available when needed in the future. If a business on the Internet is successful, Web sites outgrow their existing infrastructure and capacity. So, one important management problem is capacity planning of the web sites, and also of the back-end systems when needed, and Capacity Planning is an important component in approaching problems in Web Performance.

After the Web Performance testing and measurement activities, enough data will be available to attempt to predict two important things: how much spare capacity do you have on your system and how long before you have to upgrade the existing infrastructure. Also you should be able to predict when the future load levels are expected to saturate the site and what are the most effective hardware and software upgrades that will allow your system to expand and sustain the load over a specified period of time.

By means of simple models and knowledge of the system workload and architecture, direction can be given to the Web management team to improve or upgrade one or more components of the system. Furthermore, actions can be defined to tune the system resources in order to maximize performance in the near future, by means of adjusting system parameters. So Capacity Planning is a natural follow-up action of the Performance Testing and Measurement activities.

3.7. Activity Reporting and Problem Resolution

Activity reporting and Problem Resolution are the final stage of a Web Performance Activity, and what the whole process is aimed at. Several Reports can be target and several problems can be fixed to different professional: system managers (identified hardware bottlenecks), software and site developers (pages too big and too slow, software routine or access to DB not optimised), managers (capacity planning data, System scalability data, ISP accessibility data), marketing (users characterization, users quantification).

4. Case Study 1 - Performance testing of a mission critical Web Based Network Management System

As a case study, we describe the application of the methodology to the Network Manager of the Telecom Italia's multivendor SDH network (named SGSDH-NM). The case study stresses the specific needs and constraints related to driving a performance testing activity on a strategic management application, based on WEB technologies.

SDH (Synchronous Digital Hierarchy) is the world-wide standard for transmitting payloads on digital networks. The management of the telecommunication network and services is a strategic issue for the network operator, since providing the highest quality level of the service under any traffic condition is mandatory to maintain customer loyalty and gain new share of the very competitive and dynamic telecommunication market.

SGSDH-NM was developed, according to TMN and to the main international SDH management standards, in order to reach an integrated management solution for a multivendor SDH network [6], while retaining all the advantages offered by SDH at management level, within a centralized system at network level.

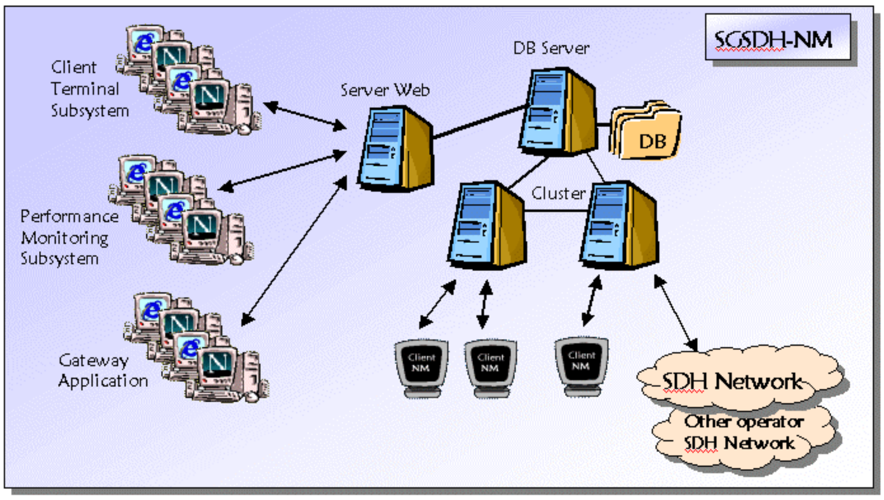
SGSDH-NM provides functions for the management of the SDH network as a whole (e.g. end-to-end path set-up, network configuration, alarms on network resources and end-to-end performance analysis) interacting with the Element Management Layer, which provides functions for the management of SDH equipments (e.g. equipment configuration, alarm surveillance, equipment maintenance and performance data collection).

Some figures, in order to understand the complexity and the extension of Telecom Italia SDH network and, as a consequence, the large amount of managed data and the complexity level involved with the network management system follow hereafter: at the beginning of this year, SGSDH-NM interacts with **63** Element Managers**,** of three different technologies, and manages a network made up of **7062** Network Elements, and **200.000** paths. The network extension is rapidly increasing showing, nowadays, a circuit’s delivery throughput of **8.000** paths routed per month.

4.1.SGSDH-NM architecture

SGSDH-NM is a complex system, made up of many components:

- A client - server application that enables communication among 70 clients and a server configuration made up of three different machines: two Application Servers and one DB Server.
- Many WEB applications devoted to the management of specific functional components, interacting with application servers and DB server:
    - The *Clients Terminal Subsystem*, a WEB application providing access to the network by external users in order to check path configuration or fault conditions on proprietary resources.
    - The *Performance Monitoring Subsystem*, a WEB application granting access to either internal or external users to performance monitoring information on the network circuits.
    - A *Gateway application* implementing the communication with an Operation Support System providing data layer functions for the whole transport network



One of the more relevant issues related to this configuration is the accessibility to the management data, through the WEB applications, from users belonging to other companies who connect to the system in order to get management information on private circuits.

The need to grant good performance and availability on client systems is, therefore, a crucial factor in success not only inside Telecom Italia itself, where the main target pursued is to support in a proper way operational processes as well as to provide management services in an efficient and cost effective way, but particularly outside the company, where poor performances on the WEB applications could impact on the relationship between customer and service provider, affecting credibility and image.

4.2. SGSDH-NM Performance testing

A Performance testing activity for such a complex and strategic system is made up of many components strictly related to one another, which is possible to group into two main areas:

- ❑ Performance evaluation of new software releases before their introduction in the field (comparative characterization between different software versions) in Test Plant environment.
- ❑ Performance measurement and analysis in the field.

This approach reflects the constraints of a specific domain characterized by a continuous introduction of new functionality or components in an assessed environment showing typically, in the field conditions, adequate level of performances.

The strategic relevance of the system, as well as the need to add new features or upgrades evaluating interoperability issues and fixing bugs before the deployment phase, has led to the reproduction in a testing lab environment the field architecture, of a strategy not always carried out for cost reasons, but in this specific case necessary in order to support adequately the operational processes implemented by the company and to assure the business goals related to the circuits delivery throughput.

A key factor of the approach adopted is therefore the capacity to anticipate performance criticisms in a testing environment, preventing the negative impacts on the business of the company before the new features roll out phase, as well as to monitor the behaviour of the system in the field taking into account either workload conditions or physical resources occupations and performance trends.

The main problem encountered, given a system configuration similar to the real one, has been how to generate, in testing lab environment, realistic workload conditions and to stress the system in order to identify main problems of performance.

4.3.Performance Evaluation in the Test Plant Environment: Load testing

The performance evaluation activity carried out in Test Plant environment in the specific context is aimed at assuring an adequate level of performance from new software releases in comparison with the performance offered by the old software versions present in the field and previously evaluated in the testing environment. The evaluation is based on workload component identification, workload generation trough emulators, and performance measurement either on each workload component, or on concurrent workloads.

In our experience, load testing is one of the more practical and reliable ways to evaluate system performance in a testing environment as well as to support the system tuning process, and offers, moreover, the not negligible advantage that it can be reused in the field, in order to analyse system behaviour under stress conditions in the real environment, anticipating the occurrence of applications crisis.

Load generation can’t be split, anyway, from workload characterization derived by the observation of the system behaviour in the field and must necessarily derive from the knowledge of the actual functionalities supported by the system.

4.4.Two different work load sources, two different load generators

In the specific contest evaluated, two main workload sources are present: the load coming from the network, such as incoming notifications from the Element Manager Layer, and the load due to the operations requested on the client’s side, that means workload related to operations on both Work Stations and WEB clients causing several transactions between the system components.

One of the main differences between a generic WEB site and a WEB based application implementing management services is, in fact, that the load due to user navigation is just one component to deal with, moreover, requests done by users typically trigger complex transactions, because the system is not only a data repository, but actually carries out several management functions.

The need to cope with two different workload sources has been managed using two different load generators, the first one with the purpose of emulating the load created by the network, the second one with the objective to emulate the user's behaviour.

MOMIX: A proprietary Element Managers emulator acting as network workload generator

TILab has developed a toolkit for the implementation of TMN Thorough Agent Emulators, called MOMIX [7]. A thorough agent emulator is a program which interacts with the network management system behaving as one or more real agents: all the object interactions specified in the information model (the description of the communication interface between NM and EM layers, according to TMN standards) are supported, without the need of further programming/adaptation.

The MOMIX tool allows the execution of either characterization or stress test cases, on specific or concurrent network loads.

The usage of a commercial tool for user's load emulation: the impact of a realistic load testing

User workload, within the specific context, is carried out by a commercial toolkit providing users emulation functions based on capturing real user transactions, recording the communication on editable scripts and replaying the captured behaviour for several users.

The main goal of this testing component is to detect response time and failure rate under various loading conditions, analysing HW resource consume, and determining system breakpoints.

In order to implement realistic load conditions, a crucial factor is to take into account a large variety of user requests, that means spending time and energy defining significant test scripts and properly parameterizing them. This technique is, in fact, the more effective the longer the range of data used during the playback phase the closest to the habits of real users.

The development of scripts in the tool environment takes, actually, familiarity both with the application under test and the tool itself, particularly in order to deal with capture accuracy as well as parameterisation issue. Two important issues to take in to account are: to avoid caching mechanism during load playback and accurately design the user behaviour, introducing, if required, real conditions, such as "think time".

A correct understanding of user behaviour in the field is a significant issue in order either to define adequately test sessions or to properly interpret test sessions results, while it gives important inputs in the test designing phase as well as increasing awareness of how stressful the test session has been in comparison with actual usage of the system in the target environment.

A significant advantage presented by this approach is that the scripts implemented can be re-run multiple times, insuring a repeatable process. When software upgrades don't affect transactions between client and server (regarding WEB applications, that means upgrades that don't impact on HTTP transactions between the browser client and the web server), the same scripts can be used several times and the data collected by the tool, regarding response time and failure percentage, could significantly help in order to perform comparisons between different software versions or characterizing possible variations of system's performance in the field, as long as the amount of managed data is increasing.

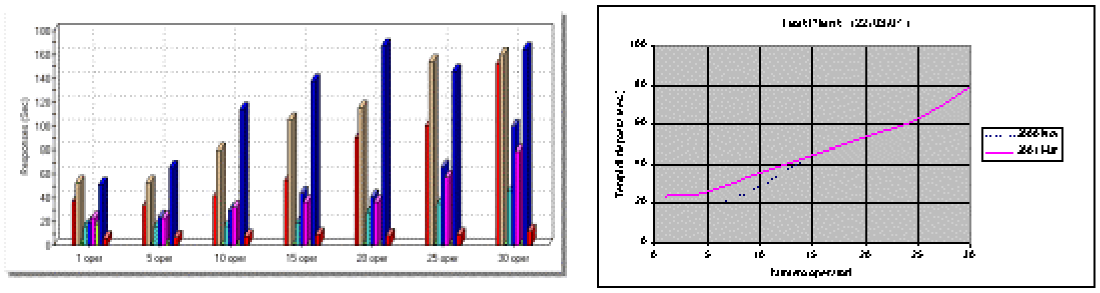


Figure 7: Example of output produced by a load test session

In the specific application contest, the modularity of the System Under Test (SUT) allows the identification of a repository of scripts building suites oriented to the evaluation of each application element (such as the main client server application, the “Clients Terminal Subsystem”, the “Performance Monitoring Subsystem” and so on). The tool chosen supports concurrent replication of different test suites granting a realistic emulation of users workload on an application presenting more user access points.

4.5. Evaluating SUT performance in a lab environment

Load generation is just one step of the process involved with the SUT performance evaluation. A second important component is the usage of measurement and performance monitoring tools.

As regards this item, two different strategies has been chosen:

- The usage of a proprietary tool devoted to measurement in order to support characterization tests on a specific functional area.
- The usage of a commercial tool in order to evaluate HW resource consume trends, during larger periods of observation.

This approach shows how the combination of more techniques, depending on the SUT capability focused, can be a good solution in order to achieve a more complete vision about the whole system behaviour.

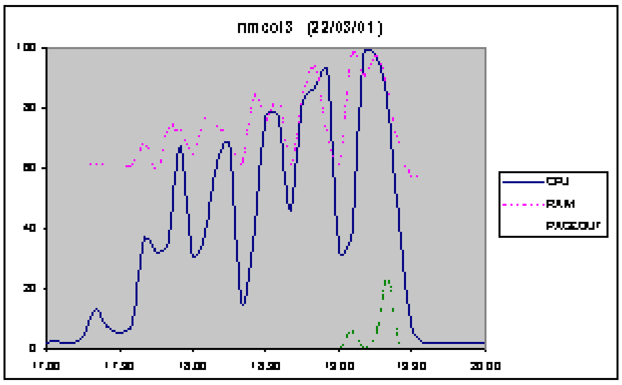


Figure 8: an example of HW resource monitoring during a load session

4.6. Performance measurement and analysis in the field

To accurately monitor the behaviour and performances offered by the applications in the field is the most important element of the process that starts with performance evaluation in a lab environment and ends with improved performance.

Observation and measurement in the field is actually the most relevant source of information, without which is impossible to perform effective tests, appropriate evaluations of test results in a lab environment or anticipate breakdowns and potential criticisms.

The approach driven in the specific contest includes:

- System behaviour and physical resource consume monitoring
- Users workload characterization, specifically Web Log Analysis techniques for WEB application components
- Performance measurement trough load testing

The approach used is based on a high level of reuse of the same tools and methodology adopted in the lab environment. This solution gives the opportunity either to reduce instrumentation costs or to compare results in an easy way, saving time as well as upgrading resources expertise.

System behaviour and physical resources usage in the field monitoring

The HW resources monitoring phase is carried out within the same commercial tool used in the lab environment for concurrent workloads analysis. A key factor of success in the complex process of interpreting collected data is, anyway, the correlation between resources consumes and workload conditions. In the specific context, this is performed either evaluating the operations and the amount of data managed by the system during the observation period, extracting metrics from the DB (SQL scripts), or characterizing, for the WEB based elements, users behaviour trough WLA.

User behaviour characterization: WEB LOG Analysis

An important advantage linked to a WEB application environment is given by the features provided in order to trace user behaviour, an often complex issue with traditional client-server architecture. The functionality provided by WEB servers to log user requests tracing a large amount of parameters is a very useful opportunity in order to collect information regarding end users activity on the system. Commercial tools devoted to the WEB LOGS processing can be integrated with homemade software aimed at extracting and presenting data of interest, such as “response time” perceived on the client side for specific transactions.

Performance measurement trough load testing

The test suites implemented for load testing in the lab, can also run in the field in order to compare performance features of the different software versions rolled out as well as to monitor possible performance degrade depending on the increasing workload due to the managed network growth. This solution is not always practicable, but if possible, can significantly help in collecting incontrovertible data (such as “response time”, “failure rate”) in correlation with given workload conditions.

While load testing in the field is run as a source of comparative data, it must be executed without any other workloads. Not all applications or sites could be tested in such conditions because users’ access is in most cases unpredictable, but the application context allows this approach being the SUT a management system accessed typically by end users at rather predictable times.

When the lab HW configuration is slightly different from the target one, running load testing suites either in the lab or in the field, can significantly help in order to properly scale lab results with prediction purposes. This experience has been made for the management system evaluated as long as the target configuration has been progressively upgraded introducing more powerful servers.

4.7. Case study 1 Conclusions

The virtuous loop between testing lab experience and “in the field” monitoring

The process shown puts as evidence for dealing with performance of an application providing support to organization’s critical business area, demands an integrated approach made up of many synergic components. A key factor in order to improve the effectiveness of the process is the feed-back mechanism between the testing lab and the field environments.

Load testing execution and results analysis in the lab can’t be split by workloads characterization, understanding of users behaviour and system monitoring in the field.

Conversely, the evaluation of system performance in the field can’t be separated from the experience made in the lab concerning test suites assessment, workloads analysis, correlation between HW resources consume and operational results.

The experience shared demonstrates that the method for approaching the task of performance evaluation and improvement has necessarily to take into account specificity and constraints present in the application context; the knowledge of the application context is in turn a basic starting point of a global process made up of operational choices and a combination of different techniques, that, as a whole, can contribute to achieving the goal of improved performance.

5. Case study 2 – Performance measurement of a web application

This case study is an example of how to carry out a performance measurement activity of a web application already in the field.

An important Italian telecommunication service operator had received some complaining by users not completely satisfied by performances (i.e. response time) experienced using a web-based chat service. To cope with this problem, TILab has executed a measurement campaign with BMPOP (TILab End-to-end Monitoring tool – see par. 3.4), to measure the real downloading response time of the home page of the service, from different ISPs.

The test executed with BMPOP was:

1. A RAS dial-up connection from two different Point of Presences (POPs) of two different ISPs, located in Rome and Turin, using an ISDN modem
2. Access to the home page (this is a dynamic page because it shows the name of people connected at that moment)

Step 1 lets you know about the availability of POPs. Step 2 gives a measure of response times experienced by users of different cities and ISPs, loading the home page of the application under test. Next paragraphs describe and analyse data obtained with tool BMPOP.

5.1. Network Access Tests

Figure 9 and Figure 10 report about connection tests from the two ISPs from 5 April 10:48 to 6 April 15:16. A green bar shows a successful test, while a red bar shows a failed test.

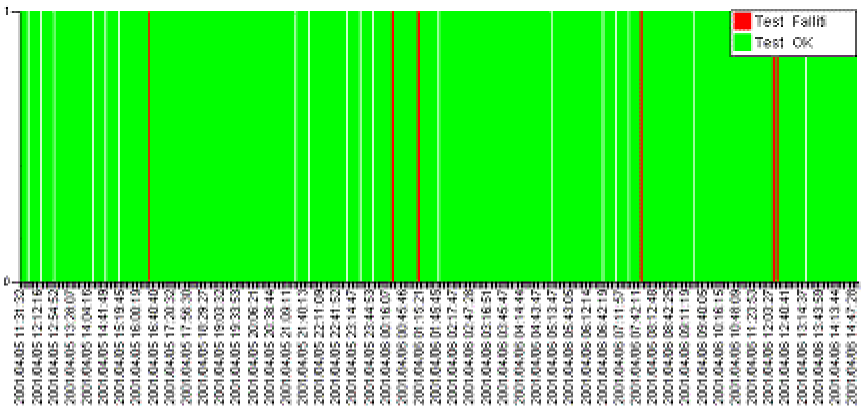


Figure 9: network access from provider A

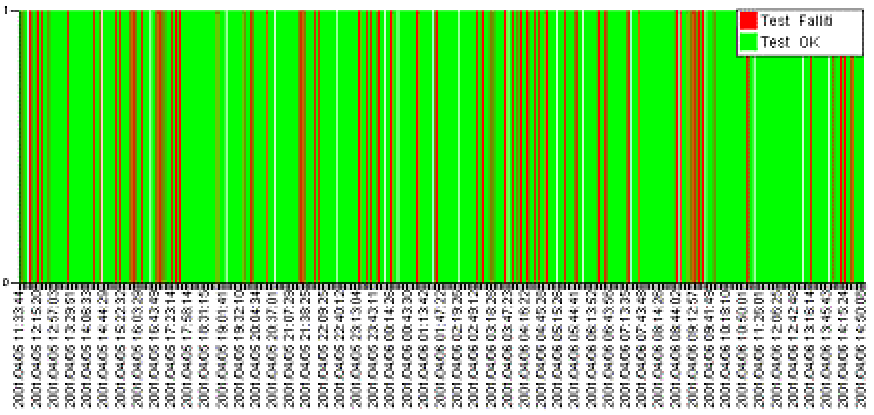


Figure 10: network access from provider B

5.2. Home page downloading Tests

Table 1 shows the summary of home page downloading test results, from the different ISPs and from different cities. If a home page downloading test exceeds 2 minutes it is considered failed.

Access	City	Successes(%)	Number of Tests	Downloading time(secs)
Provider A	ROMA	100.0 %	161	34.3 s
	TORINO	100.0 %	226	35.5 s
Total		**100.0 %**	**387**	**35.0 s**
Provider B	TORINO	94.7 %	227	43.1 s
	ROMA	96.8 %	221	43.3 s
Total		**95.8 %**	**448**	**43.2 s**

Table 1: home page downloading test results

A user, using Provider A to connect to Internet, doesn't have problems downloading the home page. Different results are obtained using Provider B. In that case 4.3% of total tests fail.

Figure 11 shows home page downloading time using provider A as a connection to Internet. After 6 April 02:00 response times remain stable with an average of 29 seconds (during the previous period it was 40 seconds). It seems that response times have no relation with the time of the day.

<u>Note</u>: a red dot in the figure means a failed test

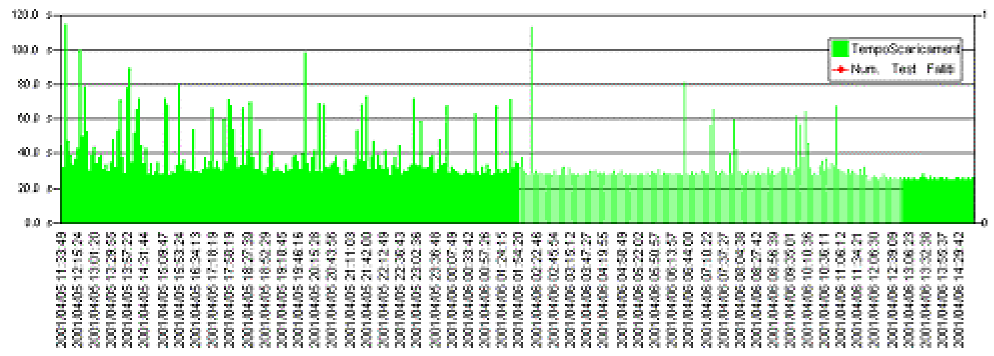


Figure 11: home page downloading time using Provider A

Figure 12 shows home page downloading time using provider B as a connection to Internet. In that case response times have a strong correlation with working hours (09:30-13:00 and 14:00-18:00). In that interval, the average response time is 80 seconds, outside is just 31.

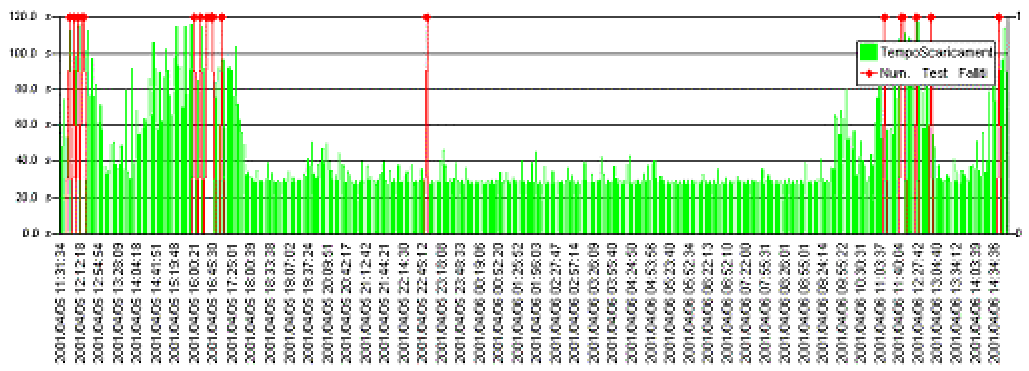


Figure 12: home page downloading time using Provider B

5.3. *Other tests*

This different behaviour from different ISPs warned TILab. It seemed that some users could have experienced poor performance because of their ISP. So, we decided to go on repeating all tests with other ISPs (most of the free Italian ISPs) using PSTN access.

Figure 13 shows network access error distribution per provider. It's easy to note that provider D is the worst.

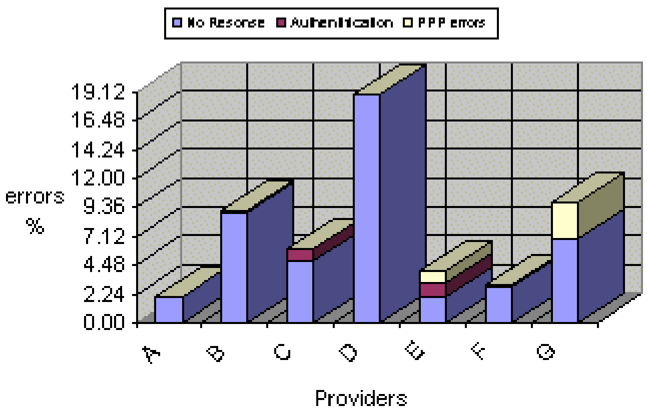


Figure 13: Network access error distribution (from 6 Apr to 9 Apr)

5.4. *Case study 2 conclusions*

Home page downloading time is never less than 20 seconds, while the maximum response time exceeds 2 minutes. The average response time waves between 30 and 40 seconds based on the provider used to connect to Internet. Monitoring hardware recourses of systems and good response time experienced by a user of the service from the same LAN of the web server seems that most of problems can be attributed to ISPs.

Analysing Web Server Log file seems that response times have no correlation with the number of users connected.

During all tests we took into account:

- Performances of a web server are affected by caching activity. At the beginning of every test, all the data in cache were cancelled.
- In this context it's possible considering 30 seconds an acceptable downloading time. Anyway it is far from being good (less than 10 seconds).
- The home page has a dimension of 83388 Byte. Using a good connection to Internet, the possible minimum response time depends on the modem speed, but it's no faster than the time showed on the following table

Modem Speed	Download Time
14,4k	48,33 seconds
28,8k	25,16 seconds
56k	13,79 seconds

To improve general performances of the chat service, two possible solutions are

1. Reducing the dimension of the page (now about 80 KB) and the number of images (81);
2. Review of caching activity parameters of the server-proxy.

To provide more detailed analysis, a load and stress test activity will be executed to correlate response time with amount of load (number of concurrent users)

6. Final Conclusions

Poor performance or low availability may have strong negative consequences both on the ability of the company in attracting and retaining its own customers and on its ability to obtain high revenues from the investment in new technology. Controlling performances of web site and back-end systems (where e-business transactions run) is a key factor for every company.

We think that the capacity of an e-business application is difficult to estimate or simulate because usually systems are too complex to model a priori. To deliver proper performances you need a load testing activity (on-site o remotely) simulating a real scenario (Web Log Analysis), combined with hardware resource utilization monitoring and end-to-end monitoring of web site (using different ISPs).

TILab is deeply involved in the development of methods and tools for Web Performance Testing and Measurement. Our reports and analysis, about the performance of a web-based application, allows us to compare network performances, application performances and service levels as perceived by end users. We are evolving this approach including information about load test results (including simple models on hardware resource utilization) to anticipate performance problems and to highlight differences from actual and estimated behaviour.

7. References

[1] R. Jain - “The Art of Computer System Performance Analysis”, John Wiley & Sons, 1991

[2] P. J. Ross Taguchi – “Techniques for Quality Engineering”, McGraw-Hill, 1996

[3] M. Richeldi, G. Ferro, D. Gotta - “Predicting the performance under overload conditions of operations systems of TLC networks”, CMG 1997

[4] L. Biffi, D.Gotta - “Performance Evaluation of web applications”, CMG 1998

[5] Zona Assessment Paper Issue 33 Internet Performance Services

[6] F. Broccolini - "The Telecom Italia SDH Management System", London, 1996

[7] N. Bersia, G. Lo Russo, R. Manione, M. Porta - "TMN OS validation and testing via thorough Agent Emulation", NOMS-96, Kyoto, 1996

[8] A. C. Siller, Jr. and M. Shafi, Eds. SONET/SDH C. – “A Sourcebook of Synchronous Networking”, IEEE Press, 1996

Acknowledgment

The authors wish to thank M.P. Maccario, N. Reale e M. Coppola for their valuable contribution about BMPOP application (TILAB End-to-End Monitoring Tool).