30 minutes in real life,
3 seconds in IT

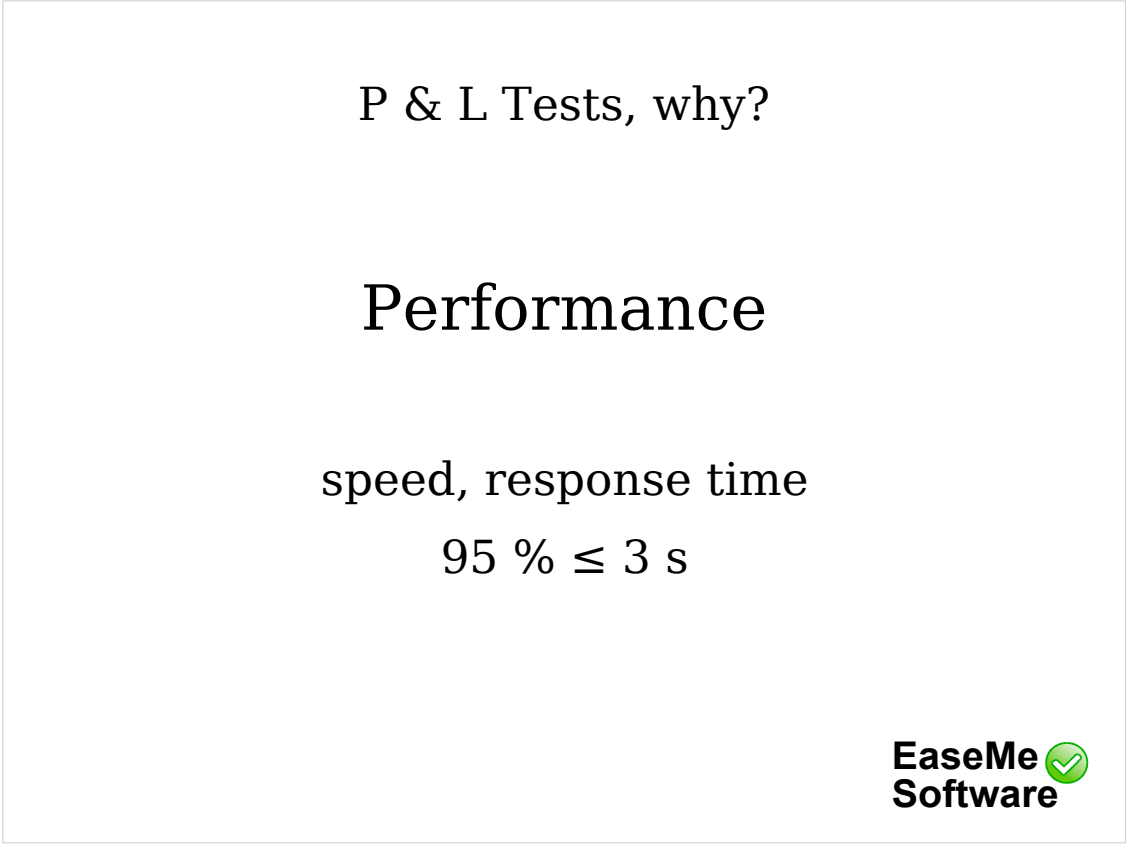
EaseMe
Software

In real life, waiting 30 minutes is, obviously, very frustrating :)

An IT user, however, begins to feel frustration after only 3 seconds, in most situations.

Performance and load tests are very important to make users happy. An otherwise perfect application isn't worth much if response times are slow, or if it can't handle as many users as necessary.

P & L tests have to eliminate a number of risks to make sure an application has got enough performance and capacity, is stable and robust, an is possible to monitor and troubleshoot.



Performance in an application means, in my opinion, speed or response time. Speed in batches and response times in GUI's.

Since all applications, more or less, suffer from unpredictable peaks with slower response times, a strict requirement on response time isn't very realistic. More realistic is to require that response times should be 3 seconds or faster for 95 % of the transactions/calls.

P & L Tests, why?

Capacity

n users initiating n transactions per second

expected load and load limit

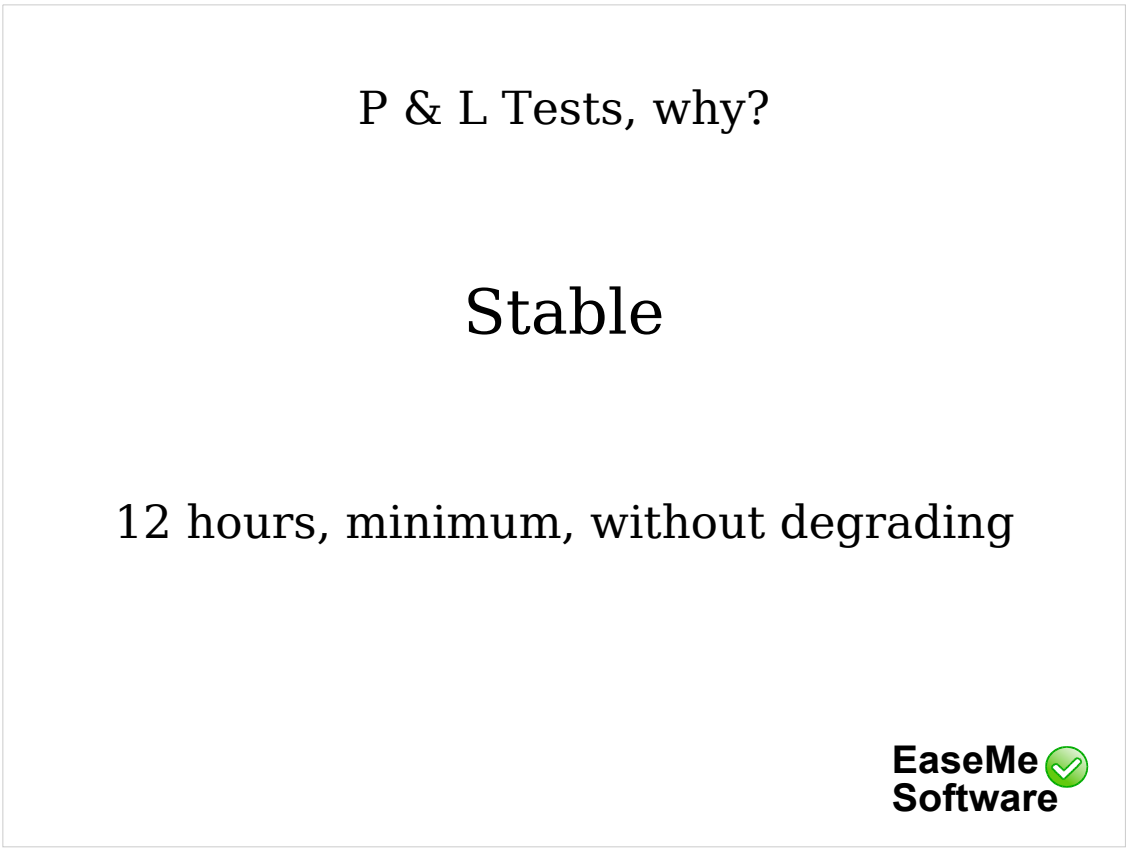
scaling

EaseMe Software

It is necessary to test if an application has got the capacity to serve the expected number of users (or threads in a calling system) and/or transactions without slower response times.

Besides that, it is good to know just have many users/transactions that the application can serve, again without slower response times.

As a side effect of capacity tests,it will be revealed if scaling has a negative effect on response times.



To make sure that the application doesn't degrade in responsiveness over time, it is necessary to execute stability tests. CPU utilization shouldn't increase, memory shouldn't be continuously consumed, and response times shouldn't slow down over time.

In my opinion, a 12 hour test is sufficient.

P & L Tests, why?

Robust

Overload behaviour, fail-over

EaseMe Software

What happens if load suddenly increases, consider Black Friday as an example. Can the system scale fast enough? How much slower will response times get? The application should definitely not crash, even if momentarily slower response times might be acceptable.

Another important question to ask, and test, is if response times gets back to a normal level when the load does.

If the application is equipped with redundancy systems, it is good to know what a fail-over does to the users.

P & L Tests, why?

Monitoring

alarms, not user calls

troubleshooting

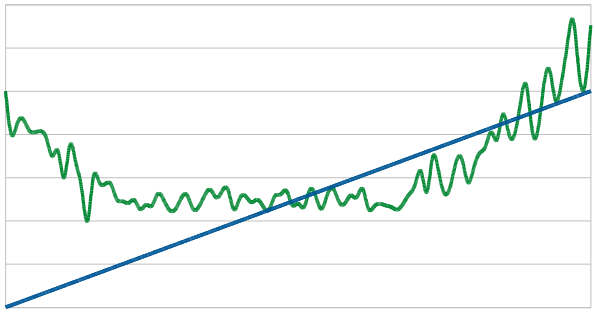
EaseMe Software

In production, an application should throw an alarm if there are non-functional issues. That is much better than if users/customers calls in and complains.

This aspect is tested by provoking the application with overload or by “un-tuning” (creating internal queues).

When testing, sufficient monitoring is necessary for effective troubleshooting and identification of bottlenecks.

P & L Tests, how?



load and response time

EaseMe Software

When load gradually increases there are, generally speaking, three different phases of response time behavior: warmup, normal and overload.

In the warmup phase, response times are slower as caches gets filled, connections are established and initial scaling takes places. In this phase, measurements varies a lot and is not suitable for testing.

Response times are normally much more stable in the normal phase. This phase is perfect for performance and stability tests.

When load exceeds capacity, response times often increases in an exponential way. It's the transition between the normal and the overload phases that capacity tests aims to identify.

In the end of the overload phase, response times usually stabilizes as timeouts kicks in and the application responds with errors.

Q: Type of change? A: Type of test

Small code change/fix or platform change?

Load from statistics

Compare results

EaseMe Software

If the test object is a small change or bug fix in the application, or a change in the platform a comparison approach is possible. Examples of platform change could a DBMS upgrade or switch of hardware.

First, baseline tests of different aspects (performance, capacity...) are executed. Then, after the upgrade of the test environment, another series of tests are executed, and finally results are compared to the baseline.

With this type of change, load can be simulated based on production statistics.

Q: Type of change? A: Type of test

New application or new function?

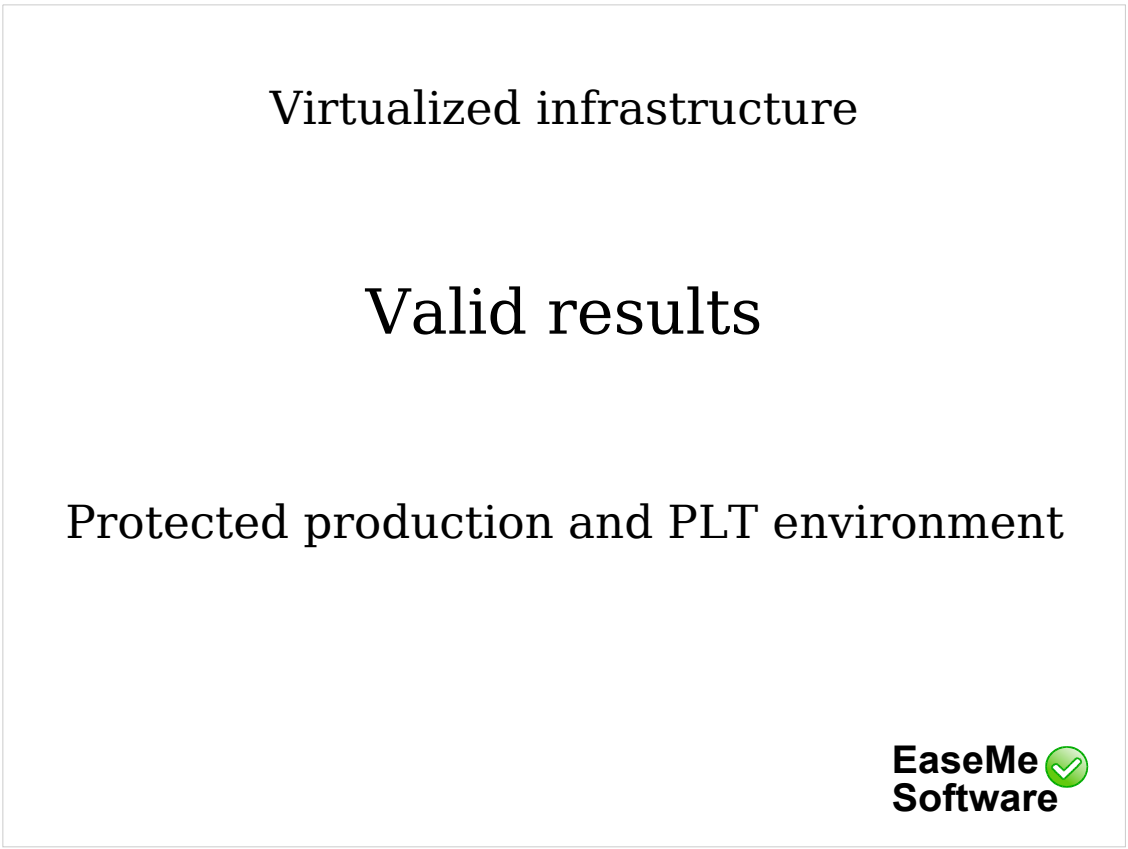
Estimate load

Evaluate results

EaseMe Software

A new application or a new function is a bit more tricky to test. No production statistics are available and results can't be compared to a baseline.

Instead, load patterns has to be estimated and test results has to be evaluated.



In a virtualized world, the idea is that applications should not suffer no matter how it is deployed in the background by the hypervisor. So it should be possible, in theory, to have one virtualized server platform that hosts both the production and the test environments.

This could be, however, very risky for production and very problematic for P & L test environments. Production can suffer from heavy tests and test results would be unreliable.

It is very important that production and P & L test environments are separated from each other and from other test environments. Either completely separated on a hardware level or by quotas of some sort.

Virtualized infrastructure

Conclusion

Proof that system will work in test or production?

EaseMe Software

What's the conclusion of a successful test? Does a successful test result guarantee successful production? No, not really, unfortunately.

If the test environment is an EXACT copy of production the best possible conclusion is: “The application works with the load patterns tested”. If it's not possible to play-back integrity logs from production, it is virtually impossible to simulate production load patterns.

If there's no EXACT copy of production, or if the platform is virtualized, the best possible conclusion is: “The application works with the load patterns tested in the test environment”. On a virtualized platform, the different components of an application will most likely be handled in different ways by the hypervisor, depending on the load patterns.

One way to do it, is to execute appropriate tests and try to find impediments that would risk production.

Agile P & L Testing

Nightly tests

Automated scheduled tests

Automated analysis of results

EaseMe Software

Moving towards a more and more agile world, with an increase in the number of delivered changes of software, P & L testing must adapt and be more efficient than it traditionally has been (manual test and manual compilation and analysis of results).

This is also true if a system contains many different products that requires life-cycle updates.

I think the answer to being more effective is nightly P & L tests. Automated scheduled tests with automated compilation of measurements. With nightly tests, there are always a couple of baselines at hand and therefore new code changes and platform changes can be tested every other day or so.

Tools

Record and replay

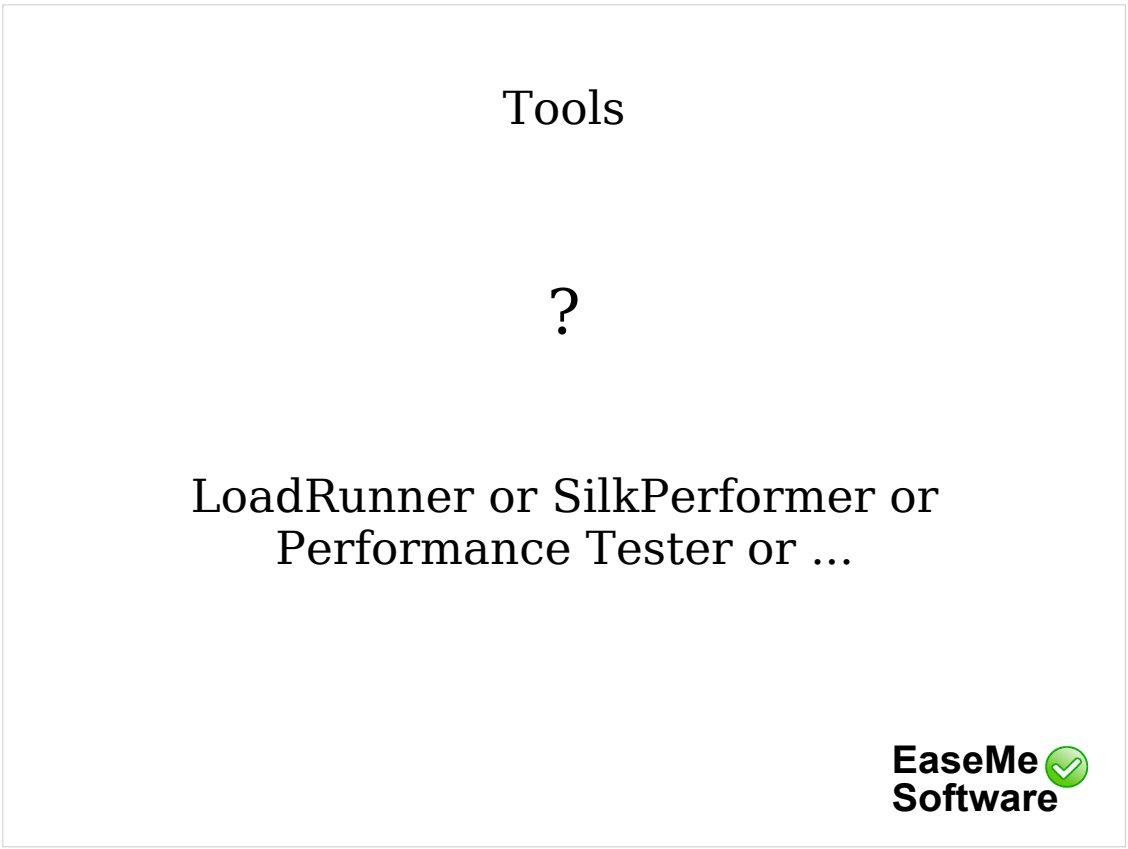
Record and simulate client traffic

Record and replay user actions

When the tester executes a functional test case, a recorder captures the communication between the client and the server. From that recording a script gets generated. After that, the tester parameterizes input fields and creates data pools (data files). Additional steps are session handling and design of the load patterns.

So called load agents executes the script in a pre-defined number of threads (virtual users).

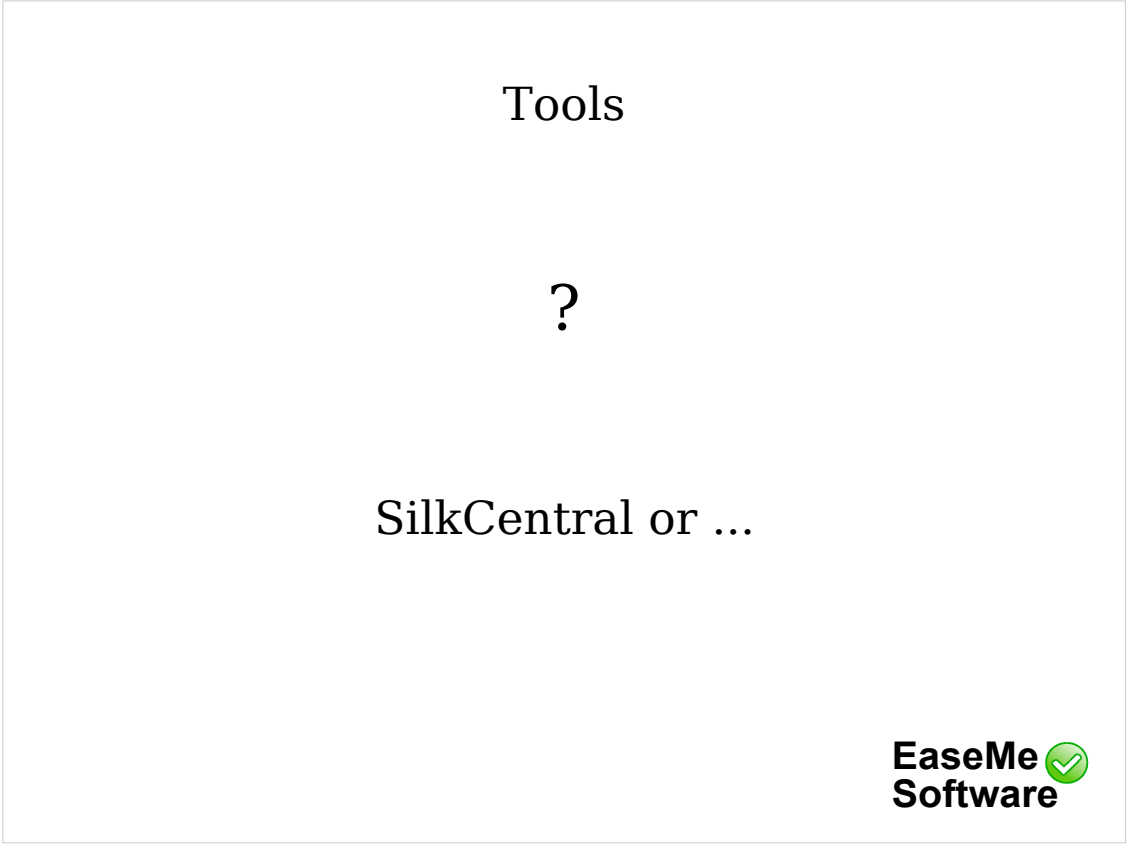
Now, this approach only tests the back end. It is appropriate to also have a few virtual users run the complete client application in order to measure the actual response times the users experience.



I've been using SilkPerformer for many years. Recently some colleagues of mine evaluated LoadRunner and filled me in on the pros and cons. A couple of years ago, I attended a demo of Performance Tester, and had the opportunity to ask in-depth questions.

With that knowledge base, I would only consider SilkPerformer or LoadRunner for full-scale P & L tests.

Naturally, I lean towards SilkPerformer as a good tool because it's the tool I'm used to. But then again, LoadRunner hasn't revealed no functionality that SilkPerformer hasn't got. Besides, LoadRunner is (much) more expensive. But that's my take on it. I'm sure there are people who thinks differently :)



When it comes to my urge to implement nightly tests I will, of course, evaluate SilkCentral. It integrates well with SilkPerformer and it's possible to schedule nightly tests. It'll be very interesting to see what it can do when it comes to compilation and analysis of test results.