Webtuesday Zürich
Load Testing of Web Applications

namics

Zurich, October 9, 2007

Jürg Stuker, CEO & Partner

Nutrition Facts

Serving Size about 60 Minutes (1 WebTuesday)

	% Daily Value*
Performance Tuning	1%
Load Test Basics	25%
My Experience	15%
Questions & Answers	20%
Tool Demo	39%

No significant source of Business Consulting.
*** Percent Daily Values are based on a tech diet.**

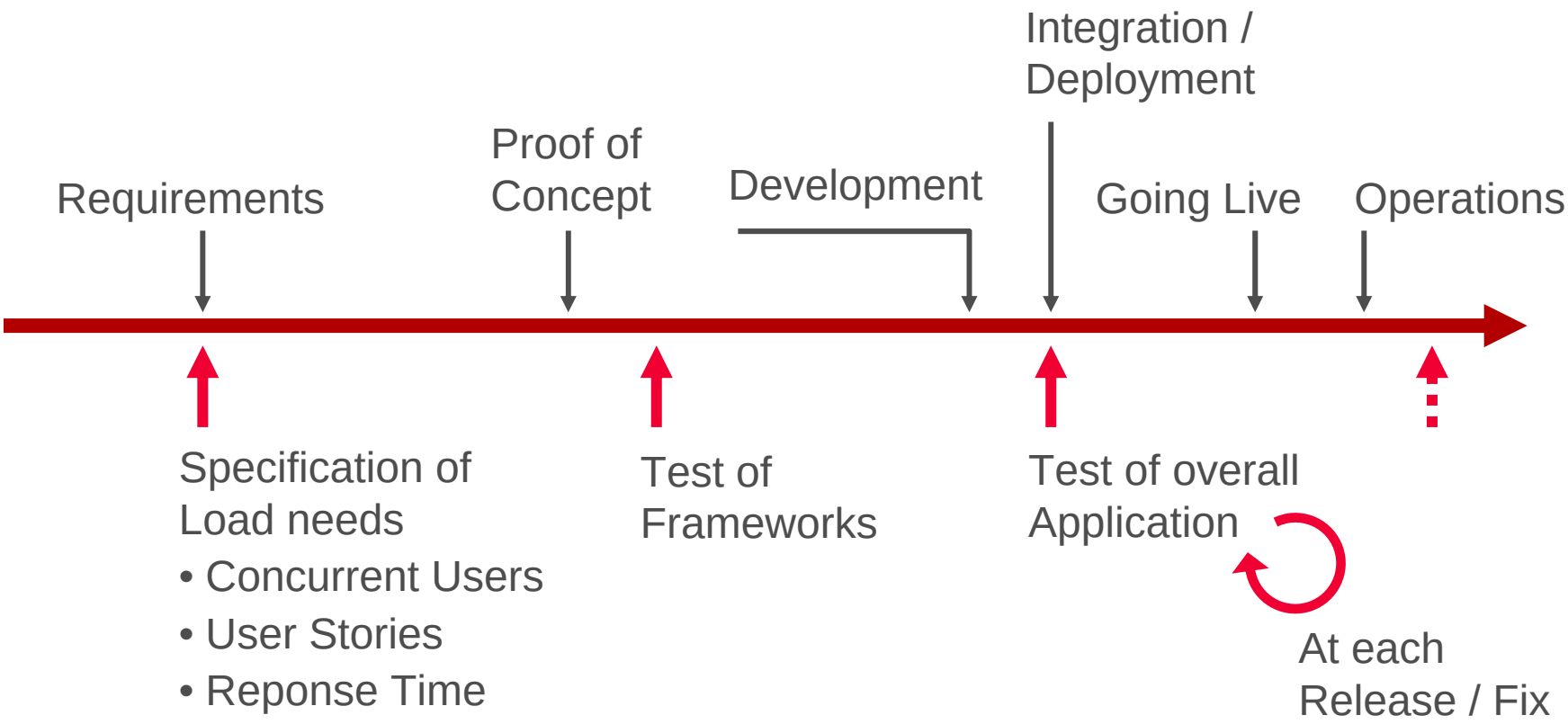
Why, When and Where?

Why should I test?

- QUALITY!

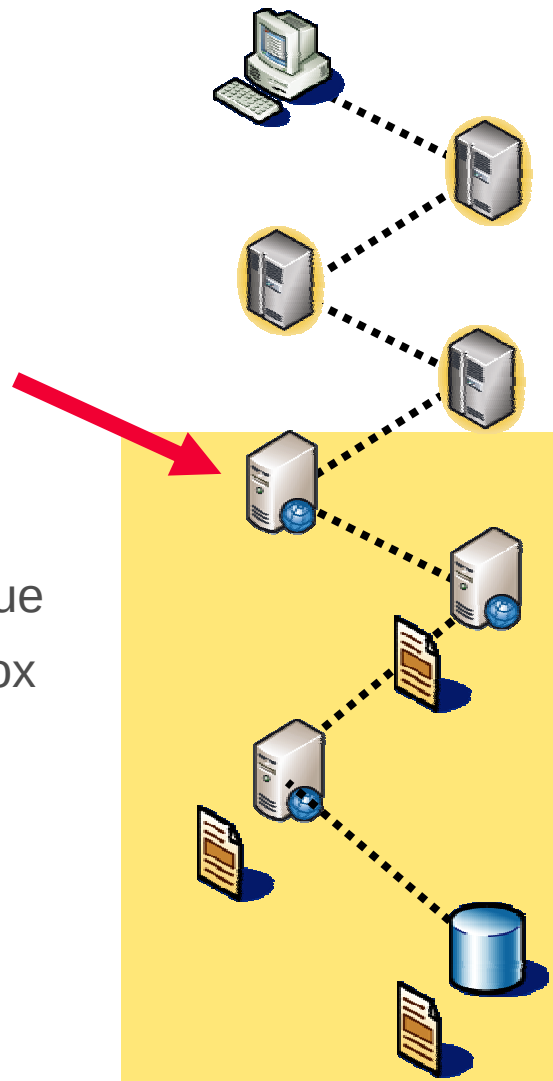
- Load capacity and performance
  - How many users can the application serve (in parallel)?
  - How long does it take to deliver pages?
  - How much bandwith does the system do?
- Stability
  - Over time → Memory leaks & overflows
  - Under load → Concurrency & deadlocks
- Fittness of Infrastructure
- Reproduction of problems in order to fix them

When to test?


Requirements
Proof of Concept
Development
Integration / Deployment
Going Live
Operations
Specification of Load needs
• Concurrent Users
• User Stories
• Reponse Time
Test of Frameworks
Test of overall Application
At each Release / Fix

Where to test?

1. Do not test the environment first
2. Treat application as a black box
3. When there is an issue
   - Look into black box
   - Change point of measurement



- Client
- Firewall
- HTTP LoadBalancer
- Reverse Proxy
- Webserver
- Frontend (Server)
  - Application Code
- Backend Server
  - Application Code
- Data Container
  - Application Code

A Word of Caution

- Load testing generates load ;-)

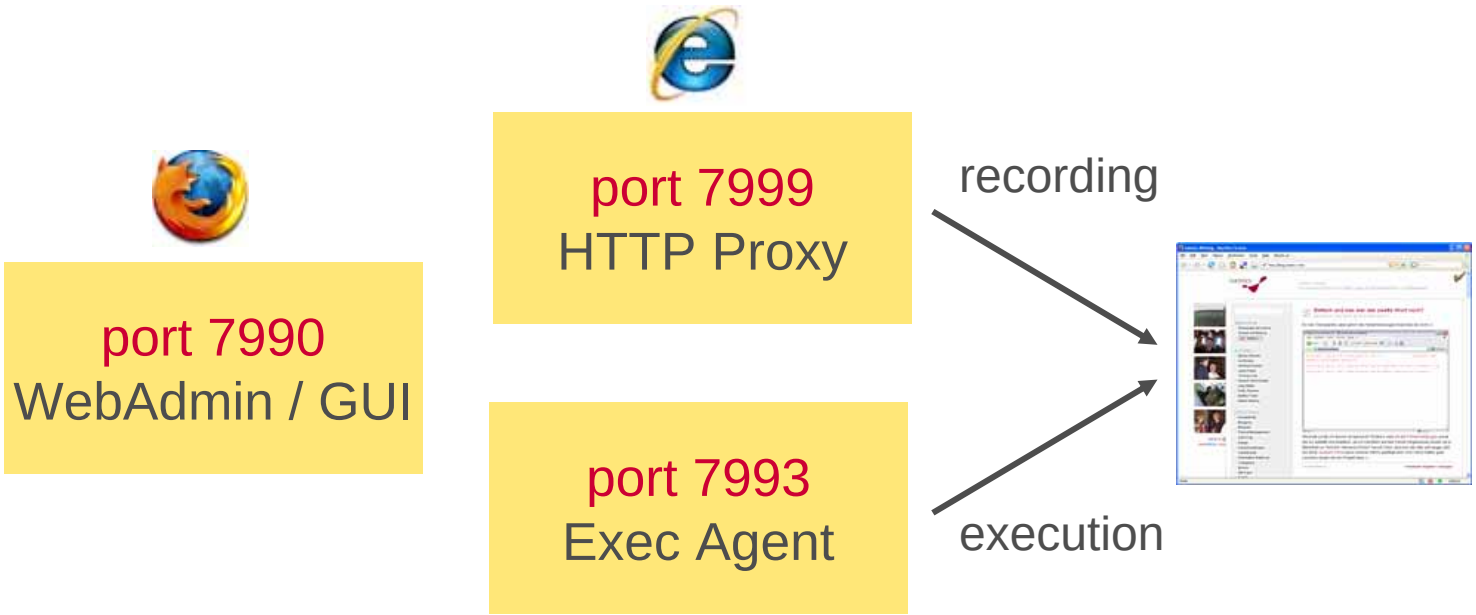
Tools

Many... I use

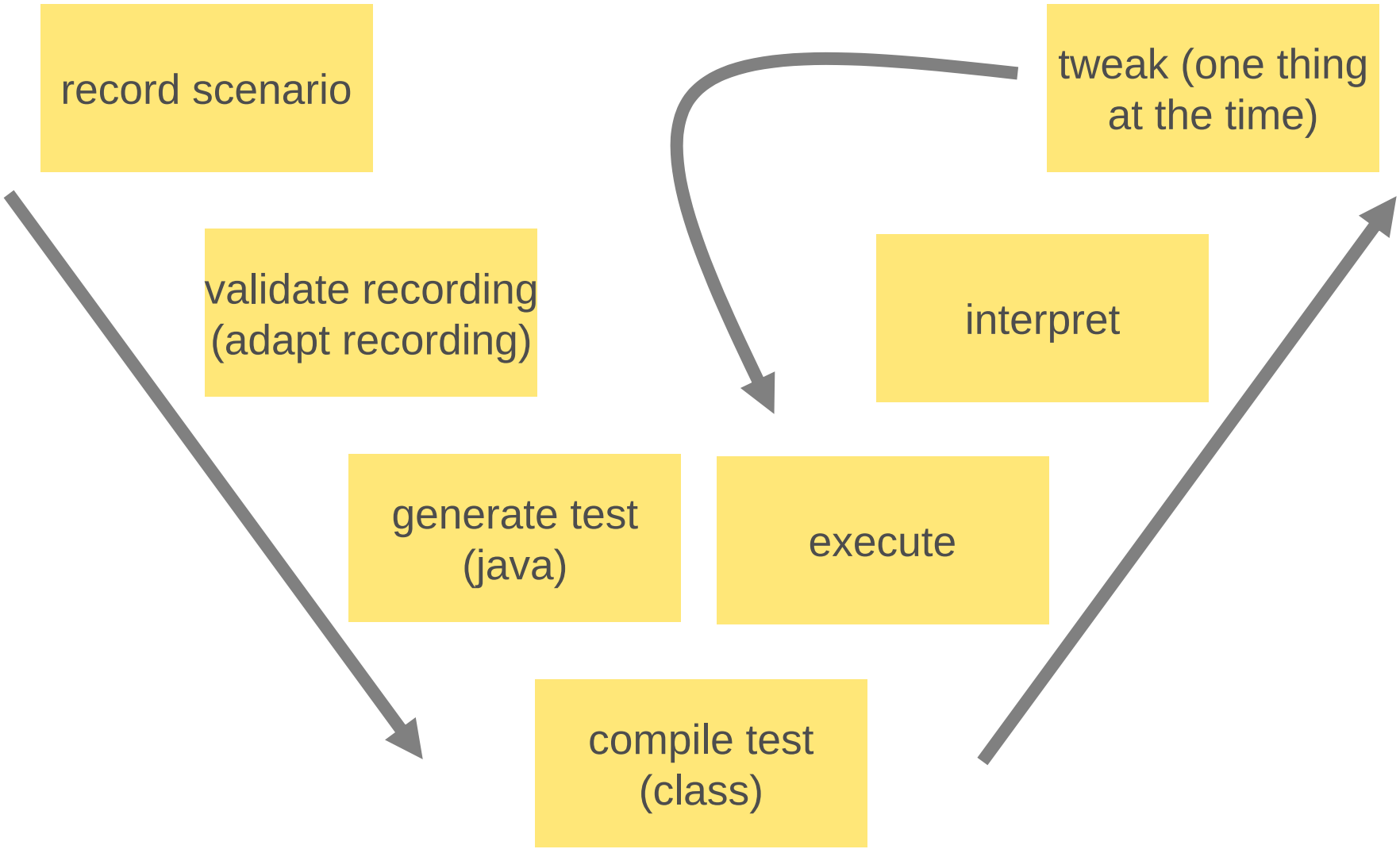
- ApacheBench (hammer on application)
  - http://httpd.apache.org/docs/2.2/programs/ab.html
- Firexfox Firebug extension (single items and headers)
  - https://addons.mozilla.org/en-US/firefox/addon/1843
- Proxy Sniffer
  - http://www.proxy-sniffer.com/
  - Built and maintained in Switzerland by David Fischer
  - Free Version (no https, 60 users, duration 12 minutes)
  - Browser based, just needs > Java 1.41 (runs on *UX*, Mac and Windows)
  - Verrrrrry flexible

Components of Proxy Sniffer

- Just two files: prxsniff.jar (8.7 MB) and prxsniff.key
- Good documentation



Components of a Test

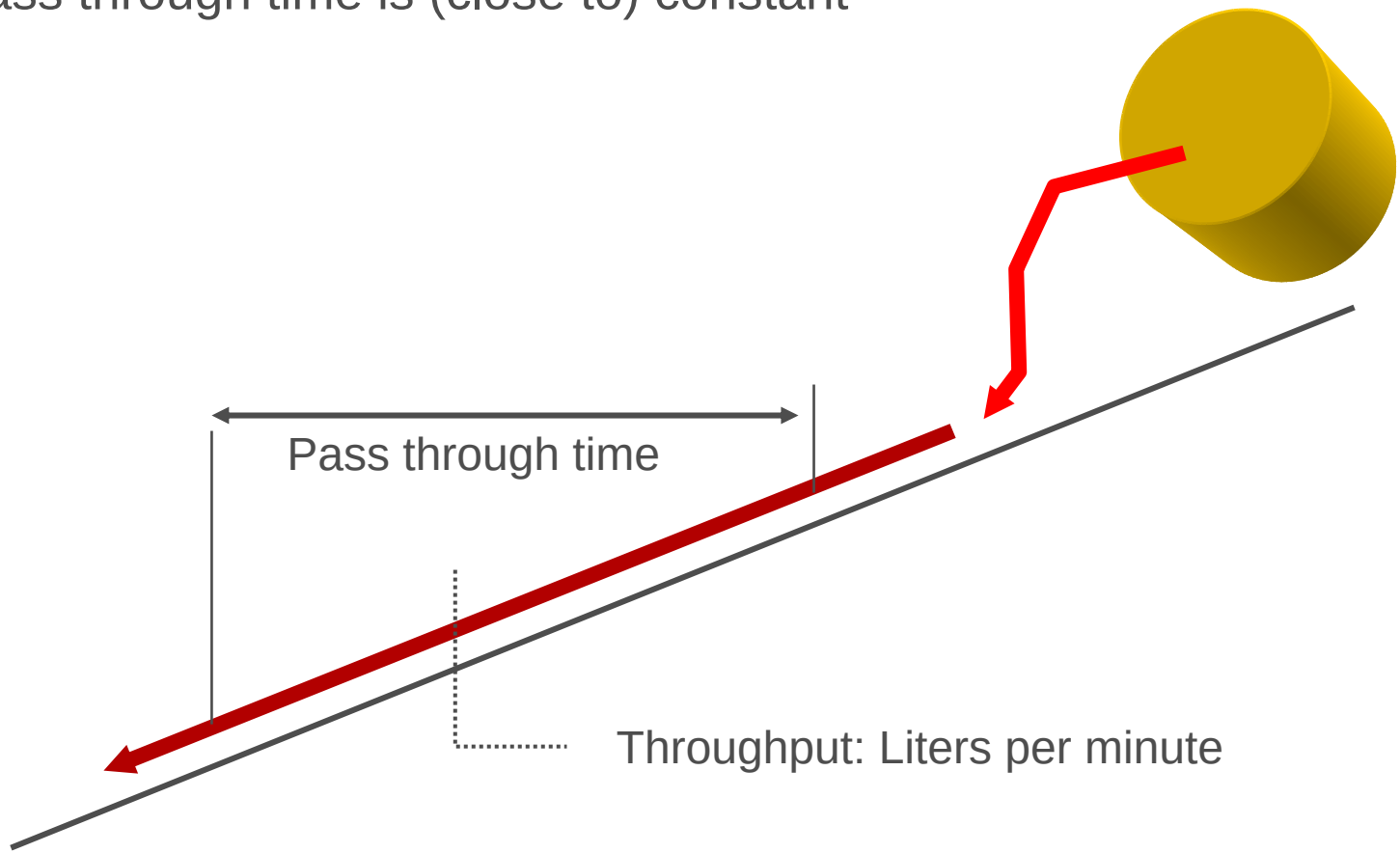


Let‘s Test

Some Theory

Linear Water Channel

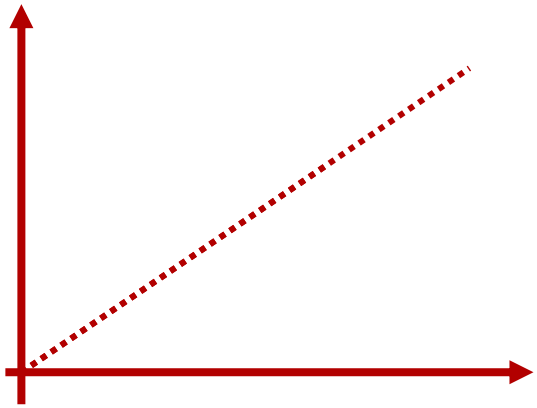
- More water == more througput
- Pass through time is (close to) constant



Source: David Fischer

Unlimited Linear Model

Throughput:
Liters per minute



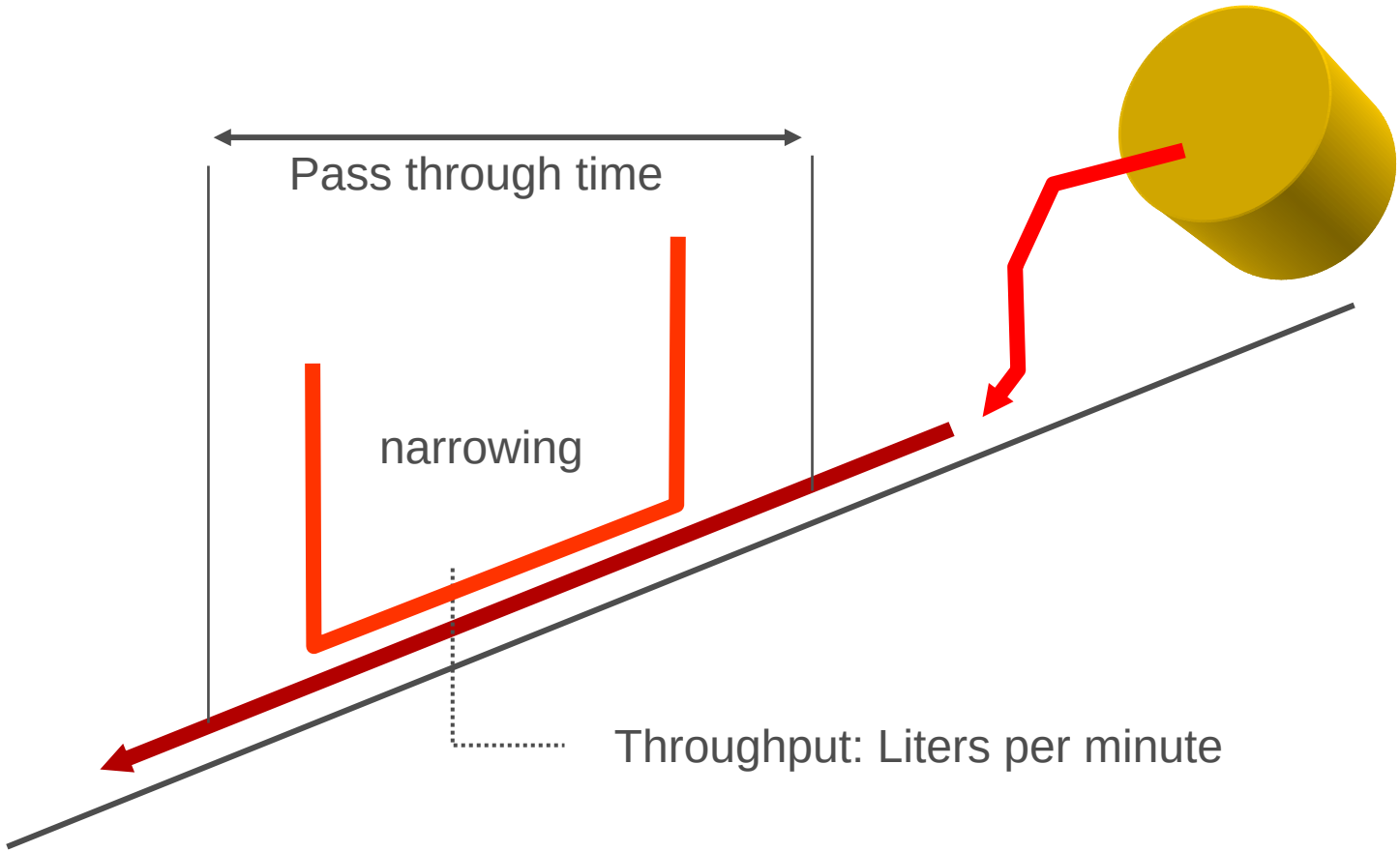
Water per minute

Pass through time

Water per minute

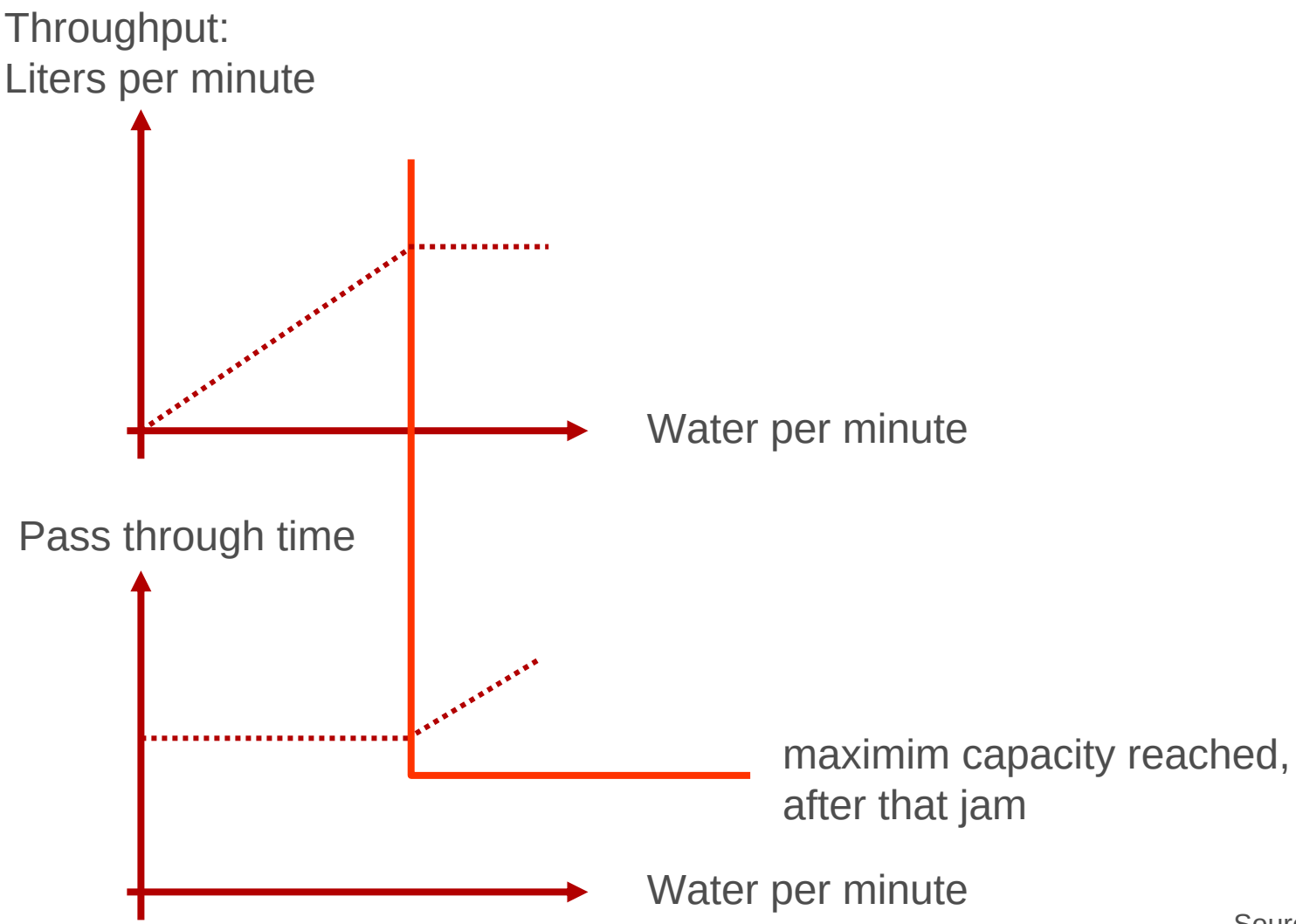
Source: David Fischer

Limited Water Channel



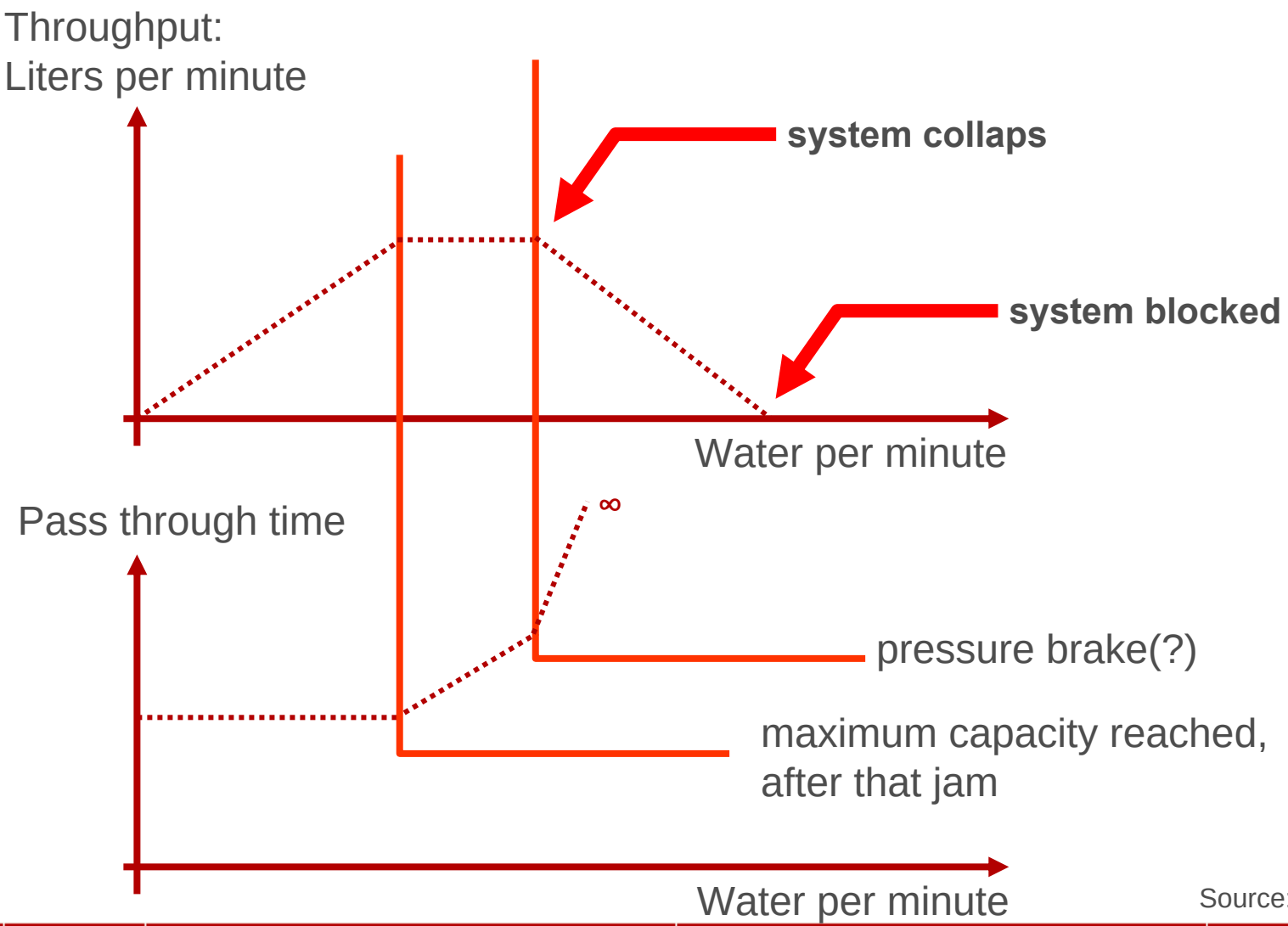
Source: David Fischer

Limited Linear Model



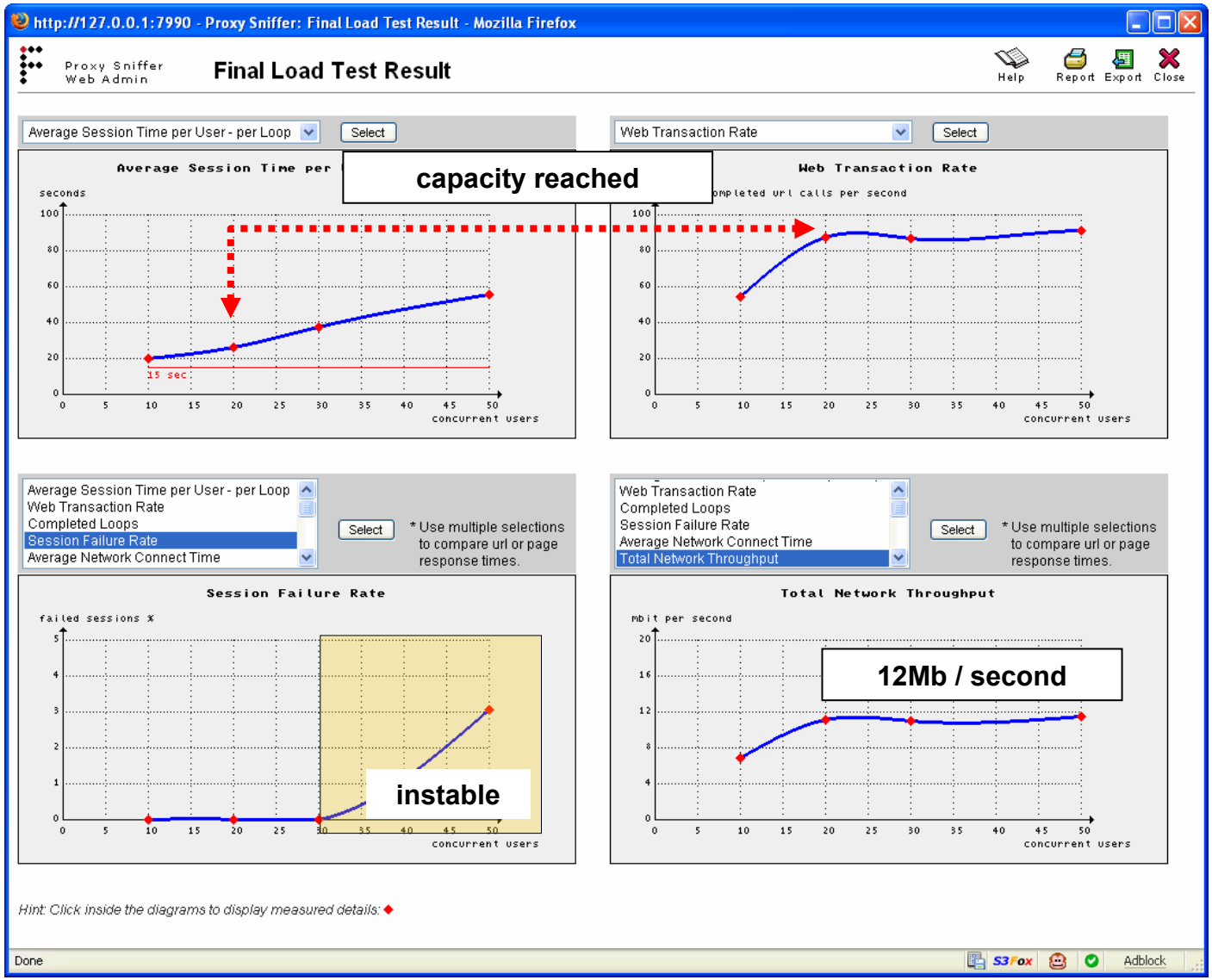
Source: David Fischer

Non Linear System

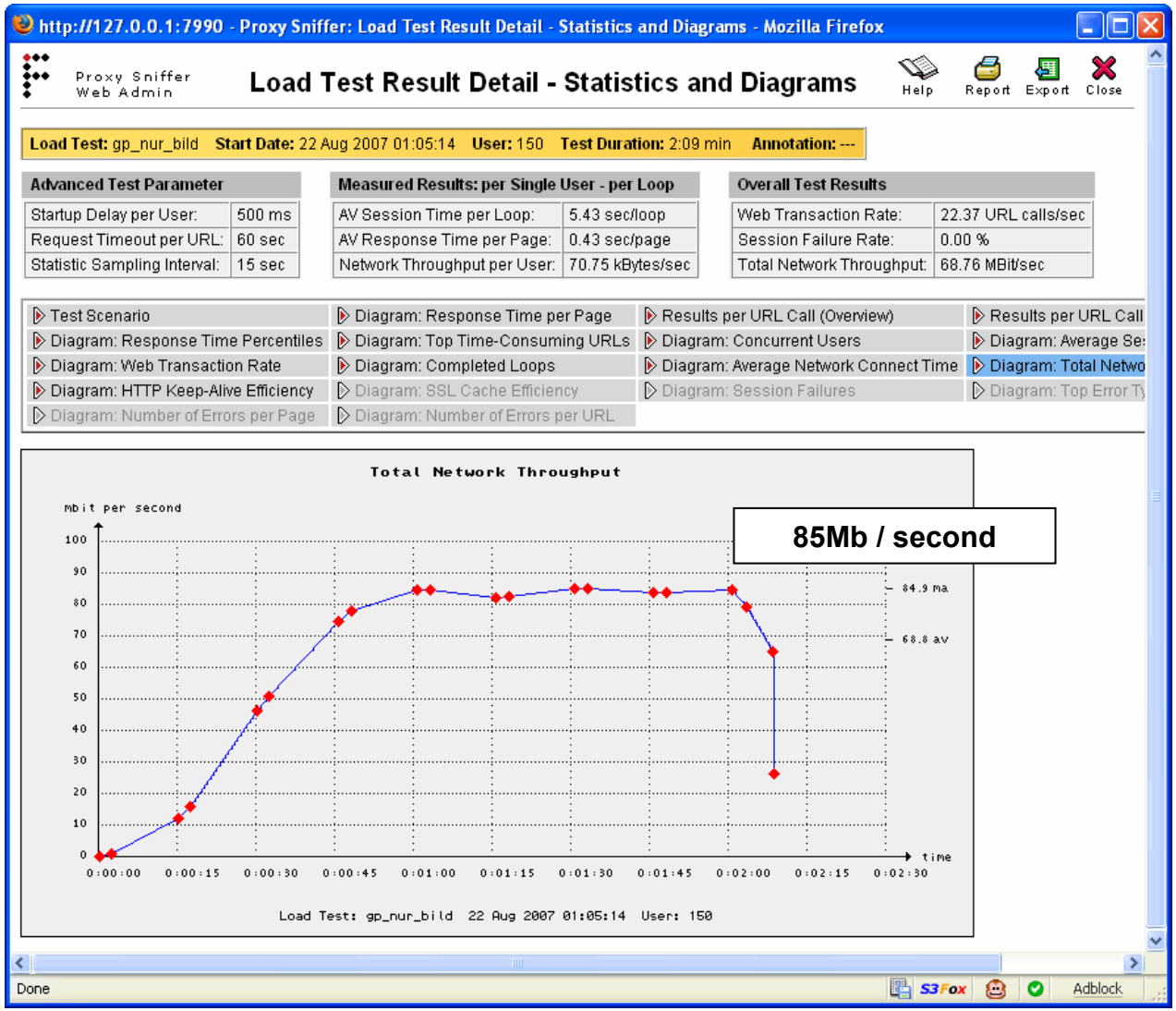


Source: David Fischer

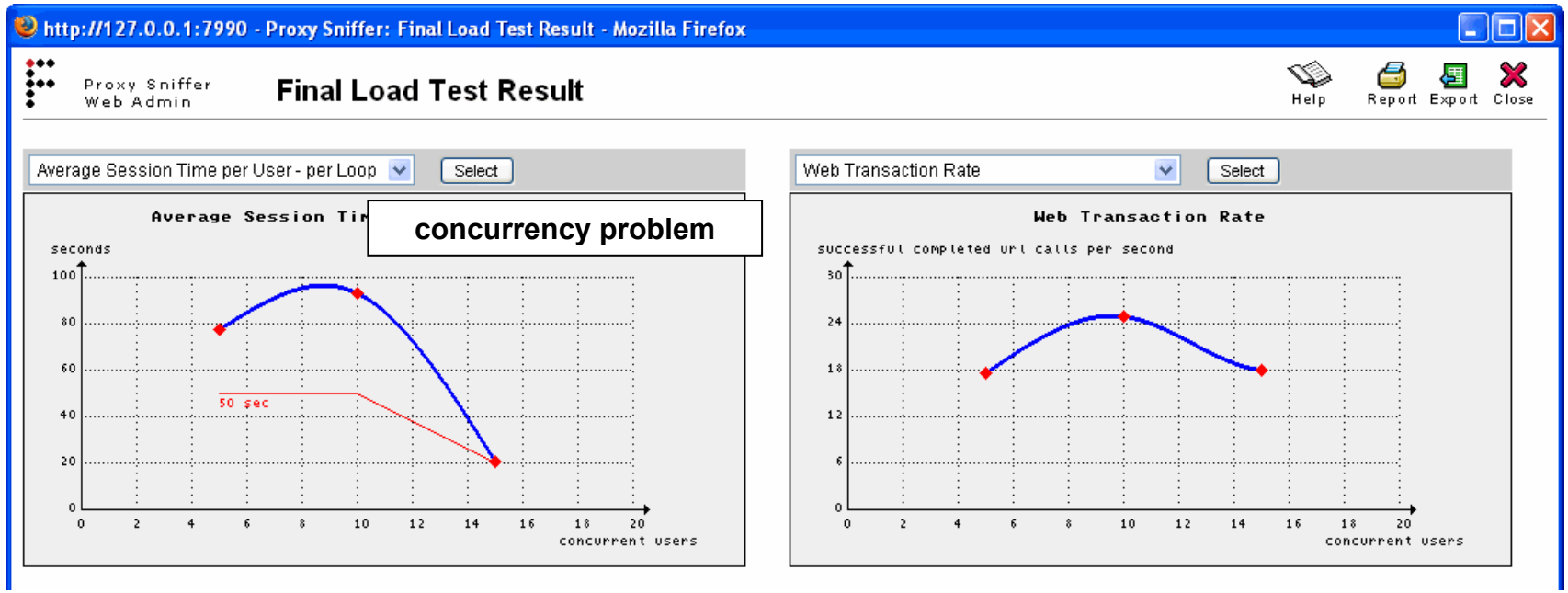
Looks like this ;-)



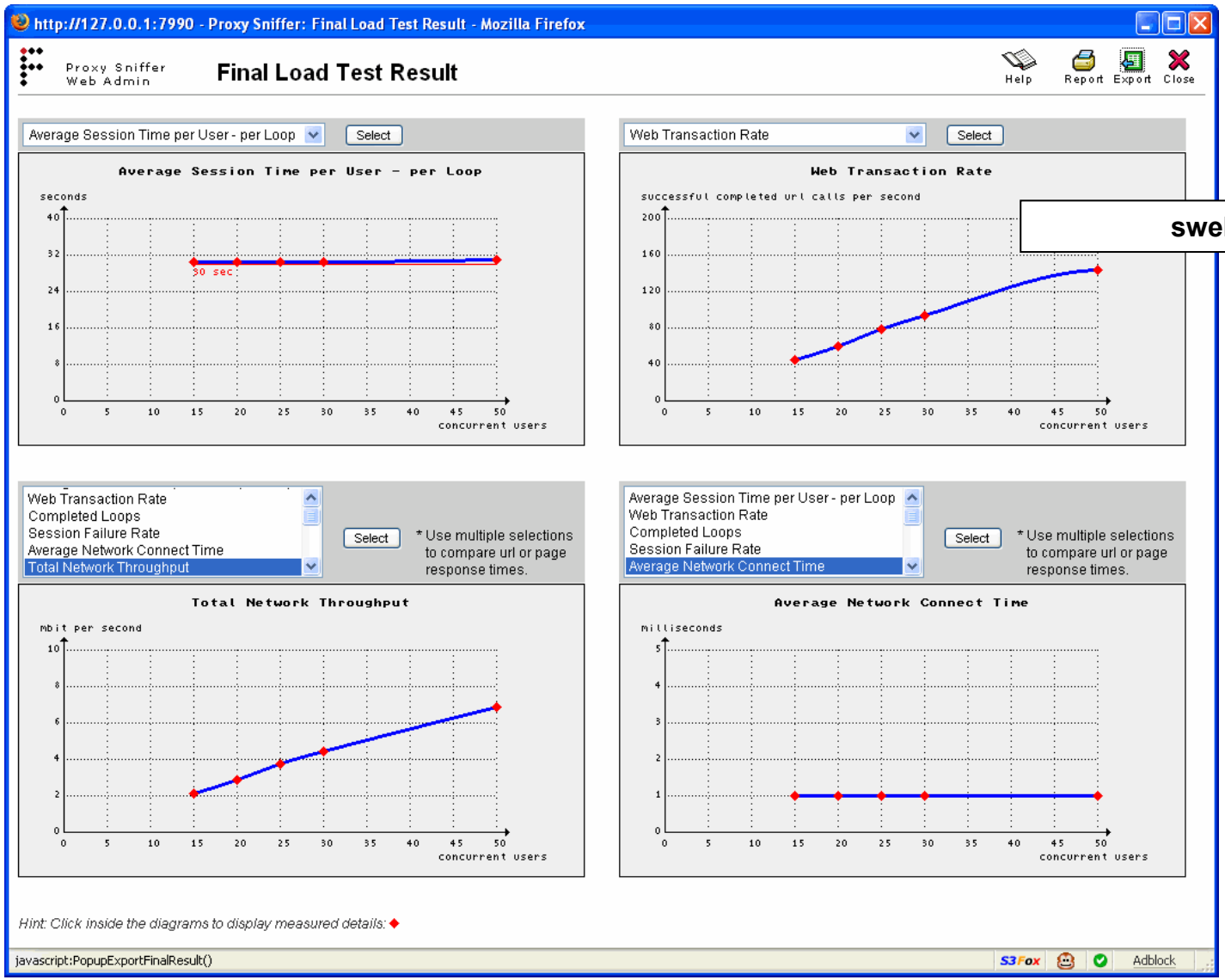
Looks like this ;-)



Looks like this ;-)



Looks like this ;-)



What to test?

- **Really though** is the correct scenario
  - Look at historic logfiles / stats (plus safety margin)
  - Ask domain experts
  - Take the user stories (“follow the money”)
- What to go for
  - Bandwith → Fetch one large item (again and again)
  - # Transactions → Short test cycles with 5, 10, 20, 40... users to find out where the systems levels (or decreases or crashes)
  - Long runner → Execute a slow test during hours
- What’s nice to know too
  - Start system under load
  - Stop system under load
  - Degradation (i.e. cluster node shutdown)

Watch our for...

- Realistic scenario
  - request mix
  - think time
  - read vs write (POST requests)
  - caching
- Get 100% CPU of the tested system first
- Look at the database (slow queries)
- Look at the HTTP response content/headers of the test
- Does system exectute the test deliver the load?
- Error logs

The Usual Suspects

» Network (Pipe, Router, Firewall)	< 1%
» Load Balancer	5 %
» Reverse Proxy (Product)	5 %
» Reverse Proxy (Configuration)	10 %
» OS configuration (TCP/IP Stack)	5 %
» Application Framework	10 %
» Configuration of Application	20 %
» Programming/code of Application	40 %
» Database	5%

Source: David Fischer

Summary

Summary

- LoadTest MUST be part of every project plan
- Product owner must define what's needed
- Do not test/optimize performance early in the dev cycle
- Test as close to the application as you can get
  - and when only you find issues → change view
- Percentile statistics are OK (90% of requests fit)
- Scenarios are though to get
- Validate test results....

Webtuesday Zürich

Load Testing of Web Applications

namics

juerg.stuker@namics.com

http://blog.namics.com