Scalability and Performance

Lecture at the TU Darmstadt

Andreas Rothmann, 08.06.2015

TECHNISCHE UNIVERSITÄT DARMSTADT

.consulting .solutions .partnership

.msg

Andreas Rothmann

- Study of Electrical Engineering in Bochum
- 1997 - 2011 at sd&m
- since 2011 GB Travel and Logistics of msg systems ag
  - Consultant for several projects with performance issues
  - Head of Performance Community

Personal

born and grown up in Gelsenkirchen

married, two Cats

beeing much outdoors, geocaching, high ropes course trainer

Performance at the Microscope

```
String message = "Operation " + operation +
                 " took " + timeElapsed + " milliseconds.";
```

- The String concatenation takes slightly more time than to use `StringBuffer.append()` or `StringBuilder.append()`

```
StringBuffer message = new StringBuffer("Operation ");
       message.append(operation);
       message.append(" took ");
       message.append(timeElapsed);
       message.append(" milliseconds.");
```

- But
  - The new code is less than a millisecond faster – maybe even slower
    StringBuffer and *StringBuilder* are faster but take longer to instanciate
  - It is less maintainable
  - And: Is this the part of the code that causes the performance issues?

Just to write fast code does not make the software fast – but often causes bad maintainability

AGENDA

1. What is Performance and Why does it matter?
2. Performance in the Activities of Software Engineering
3. Architectures for Scalable & High-Performance Systems
4. Architectures for High Available Systems

Performance Comes in Different Flavors

Response Time

- How long do the users have to wait?

Throughput

- How many data sets are processed per unit of time? (Batch)

Resource Consumption

- CPU, memory, hard disk, network

Scalability

- How does the system work under higher load?
- How do faster hardware or more hardware improve the performance?

Why is Performance important?

Cost

- Low performance causes costs
  - Customer satisfaction / image damage
  - Productivity (of the users working with the system)
  - Purchase of new hardware

Missing Scalability

- **Often performance problems can not be resolved with hardware**

AGENDA

1. What is Performance and Why does it matter?
2. Performance in the Activities of Software Engineering
3. Architectures for Scalable & High-Performance Systems
4. Architectures for High Available Systems

Performance within the Activities – Specification

Requirement Engineering / Specification | **Design** | **Implementation** | **Test**

- Gather, document and cross-check the non functional requirements
- Document for each requirement
  - Frequency of function calls + number of parallel users
  - Volume of data to work on
  - Volume of data to transport
  - Maximal response time

Performance within the Activities – Design

Requirement Engineering / Specification | **Design** | **Implementation** | **Test**

- Cross-check the non functional requirements
- Design an architecture to meet the non functional requirements
- Write guidelines for the developers

Performance within the Activities – Implementation

Requirement Engineering / Specification | **Design** | **Implementation** | **Test**

- Follow the guidelines
- Keep typical performance killers in mind
- Inform the architect / designer if any (performance) problems occurring

Performance within the Activities – Test

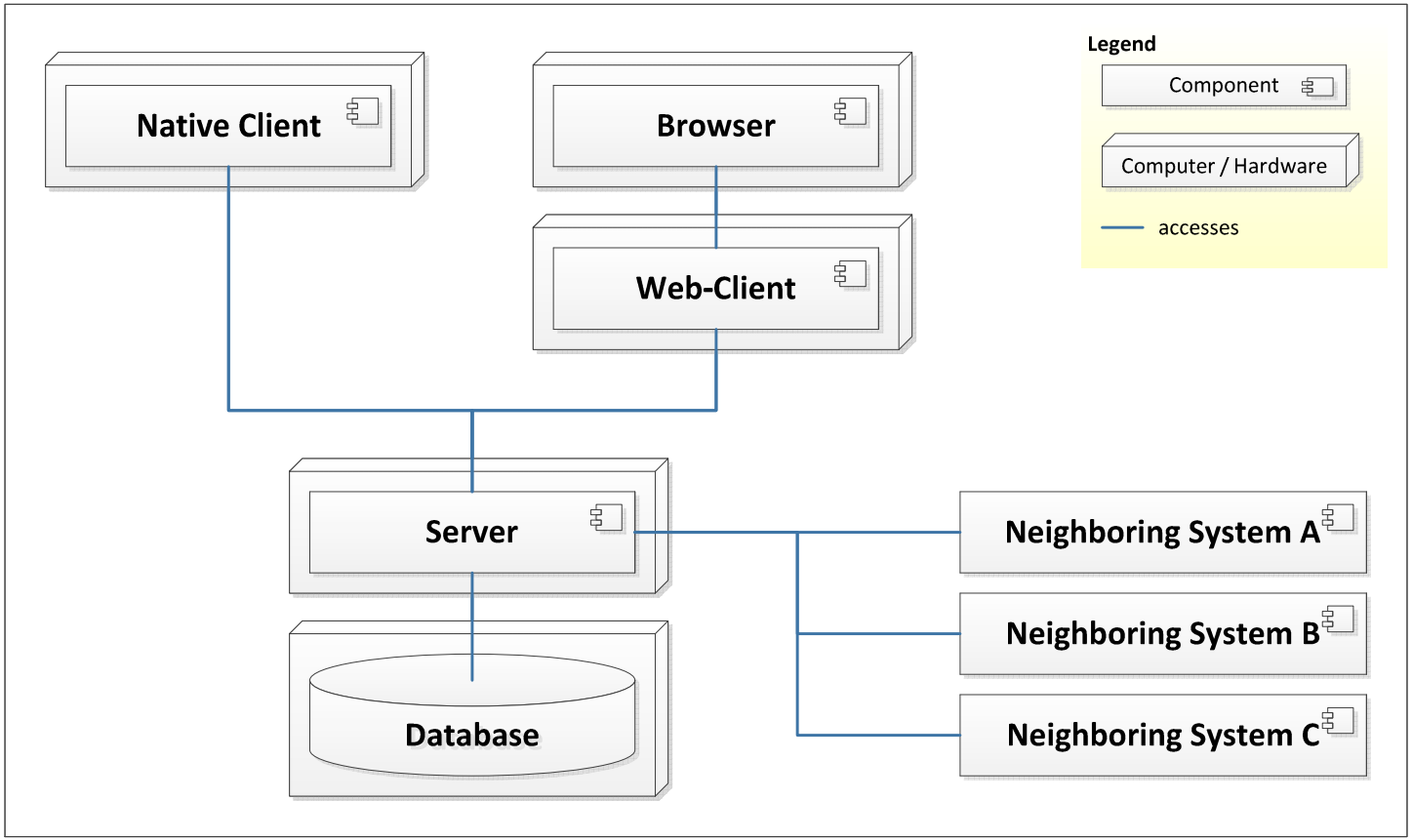
Requirement Engineering / Specification | **Design** | **Implementation** | **Test**

- Plan a performance and load test
- Do the test in a realistic environment (same hardware, data volume, load)

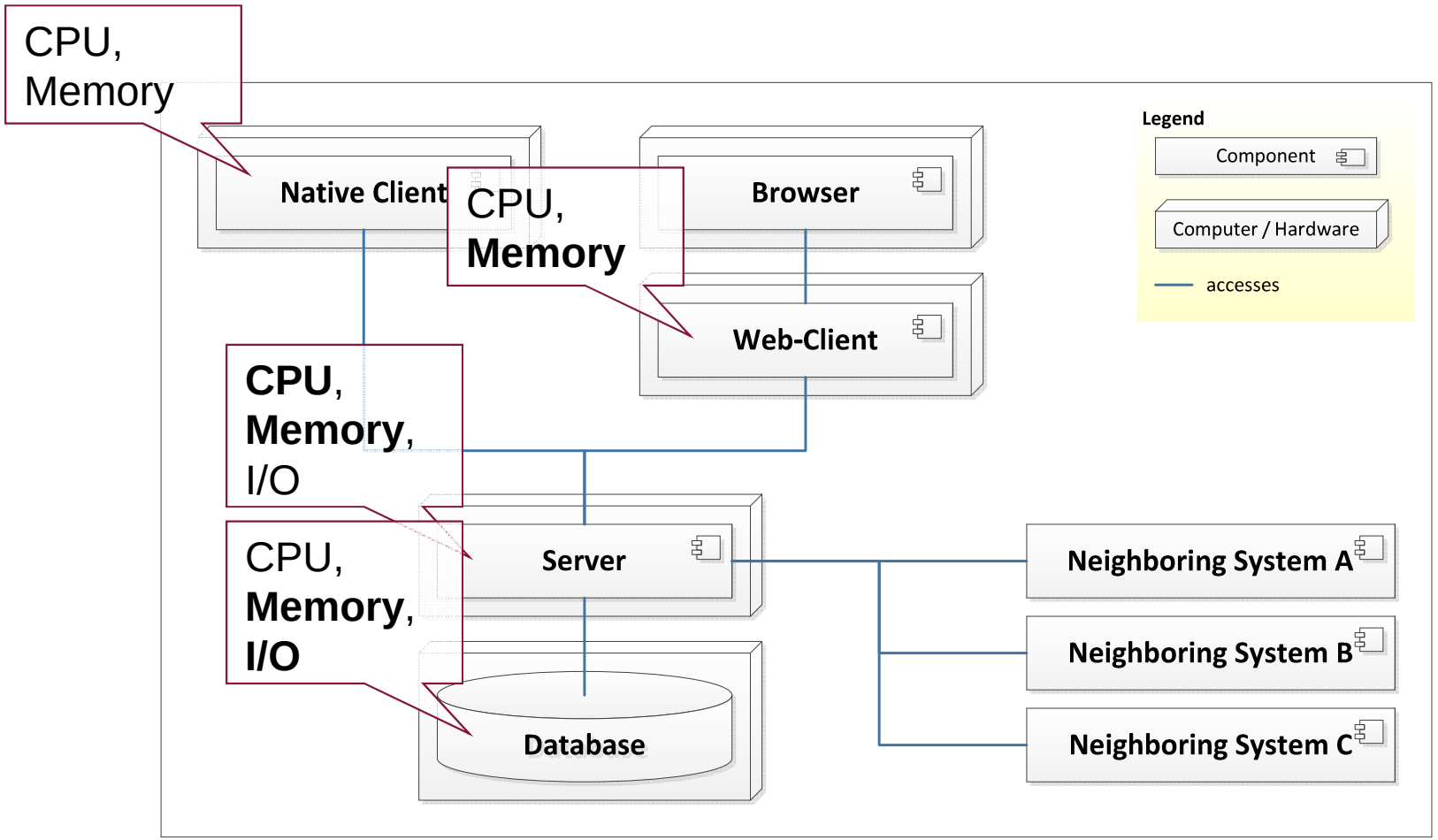
AGENDA

1. What is Performance and Why does it matter?
2. Performance in the Activities of Software Engineering
3. Architectures for Scalable & High-Performance Systems
4. Architectures for High Available Systems

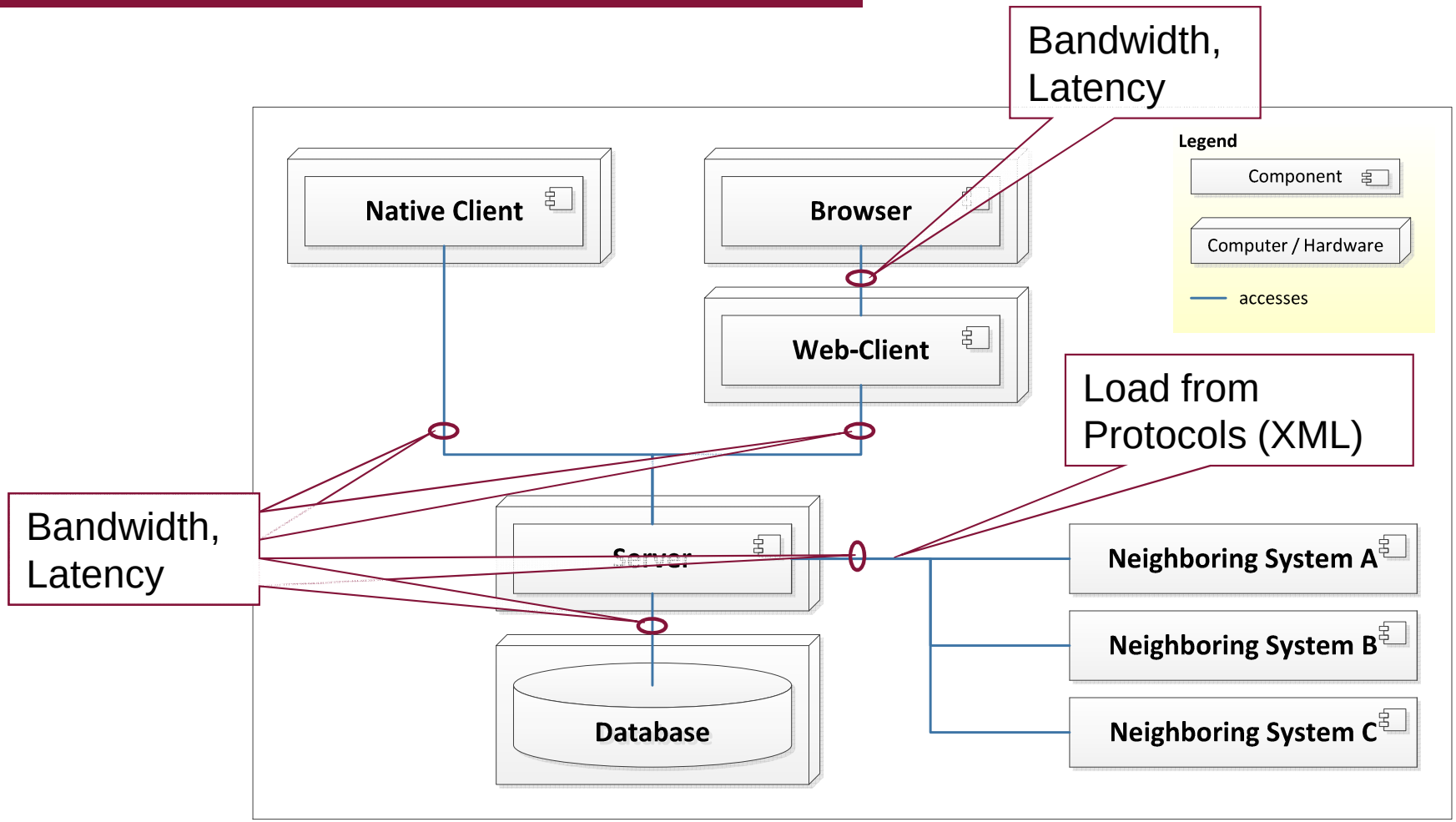
Typical Architecture for Business Information Systems



Typical Bottlenecks within the Architecture I



Typical Bottlenecks within the Architecture II



Bandwidth vs. Latency

Apollo Submarine Cable

- Between America und Europe
- 12.315 km Length
- 3,2 Terabit per Second

How long does it take to transfer one Kilobyte data via this cable?

- 1 Kilobyte → 10 Kilobit
  10 Kilobit / 3,2 Terabit / s = 10.000 bit / 3.200.000.000.000 bit / s
  = 0,000000003125 s = 3,125 x 10^{-9} s
  → 3,125 nanoseconds

Which speed does this corresponds this to?

- Speed is Distance per Time (v = s / t)
  12.315 km / 3,125 x 10^{-9} s = 3.940.800.000.000 km/s

That's more than the speed of light!

It isn't possible to calculate the runtime of a signal just from the bandwidth!

Strategies for High Performance I – Transfer Sparse Data

Low Data Volume

- Transfer just the data needed

Few Remote Calls

- No unnecessary calls
- Tailored interfaces with coarse granularity
- Cut the architectural components to reduce the number of calls

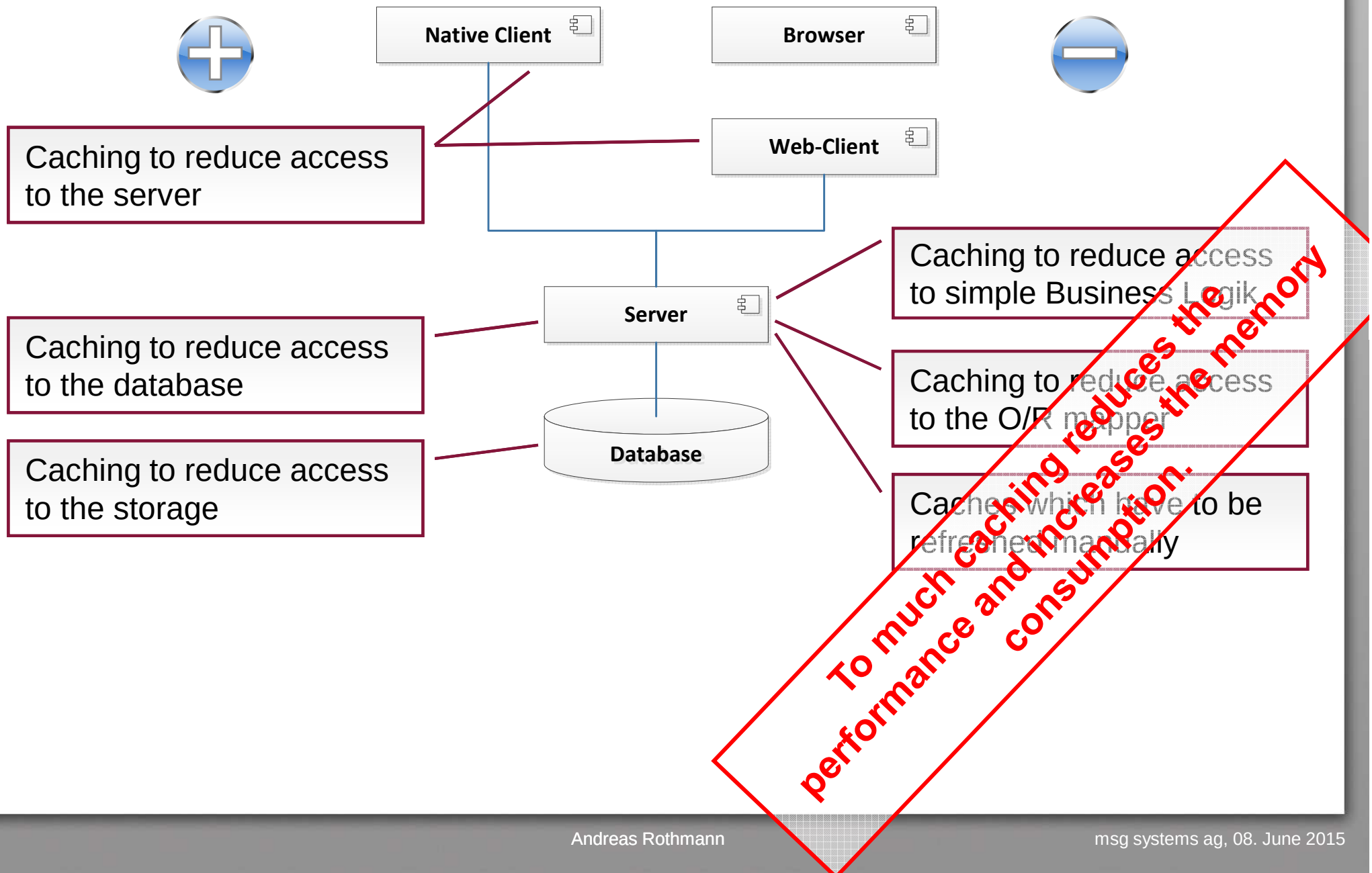
Low additional Latency

- Use protocols with low overhead

Caching

- Cache data instead of transferring it several times

Good and Bad Approaches to Caching



Strategies for High Performance II – Process Data Fast

No Unnecessary Work

- Simple – often seen the other way in projects

Efficient Algorithms

- Quicksort instead of Bubblesort
- Specialized business algorithms

Efficient Mechanisms of Programming Language

- StringBuffer / StringBuilder instead of simple String concatenation

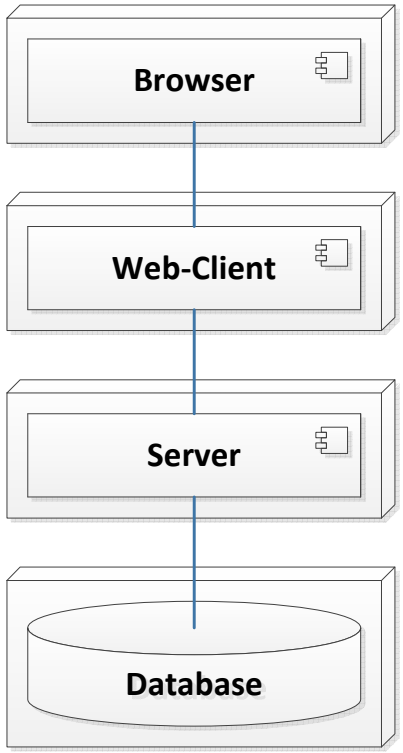
Efficient Libraries

- HashMap instead of self made container
- Specialized libraries for the current problem

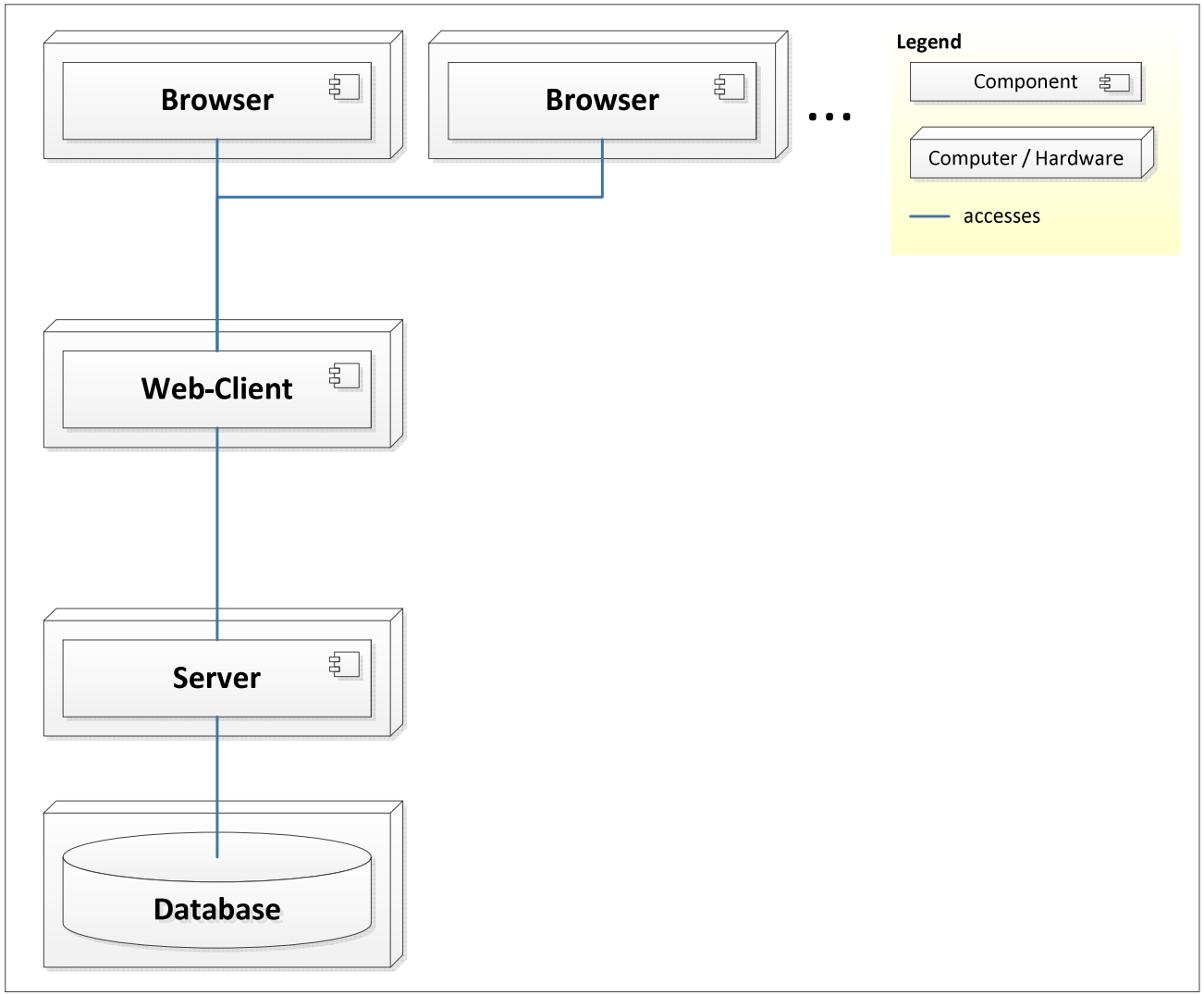
Use all of these approaches with a sense of proportion. For just three data sets Quicksort isn't faster than Bubblesort – but less maintainable

Performance by Example: A Scalable Shop for Tickets

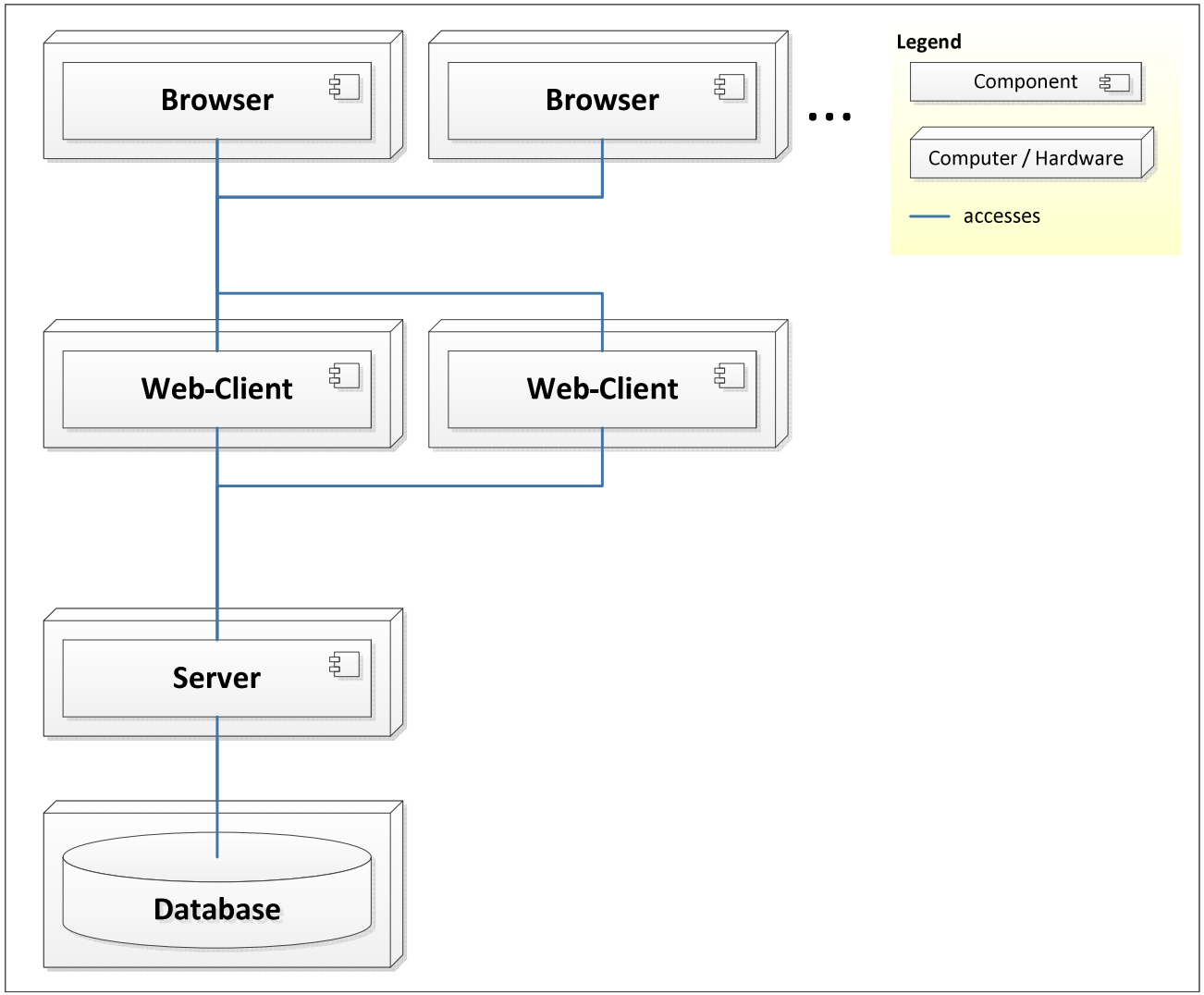
Scenario The company *Extreme Adventures* plans to sell events managed by them via the Internet. Because of the very special structure of their products there is no existing software to fulfill their needs. So they plan to buy an individual system.



There Should Be More Than One User



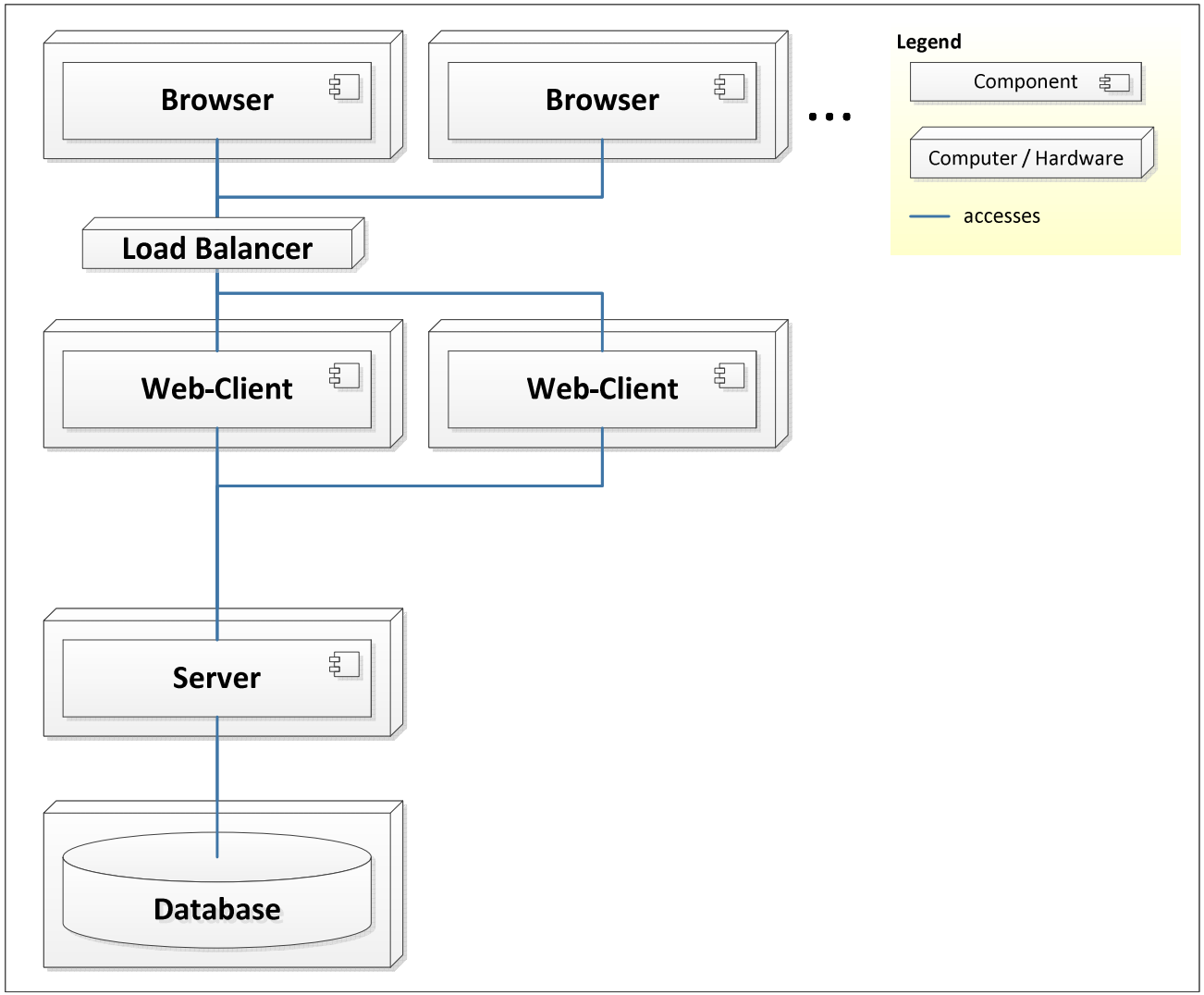
Increase the Performance of the Web-Client I



High Load at the Web-Client

- Make Web-Client redundant →The load will be dispatched
- But: How do we dispatch the load?

Increase the Performance of the Web-Client II

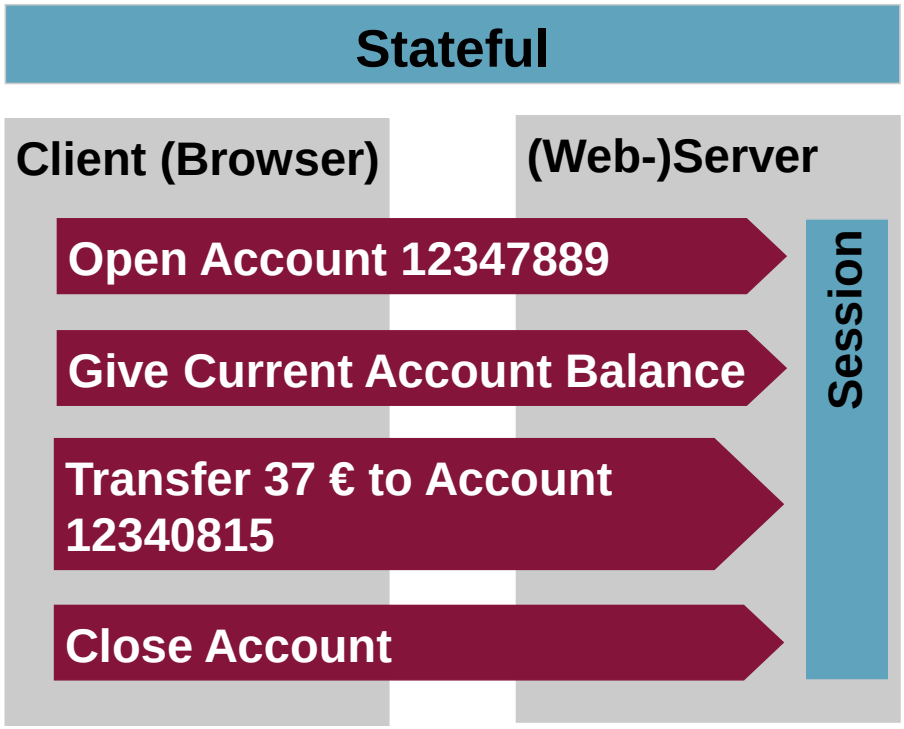


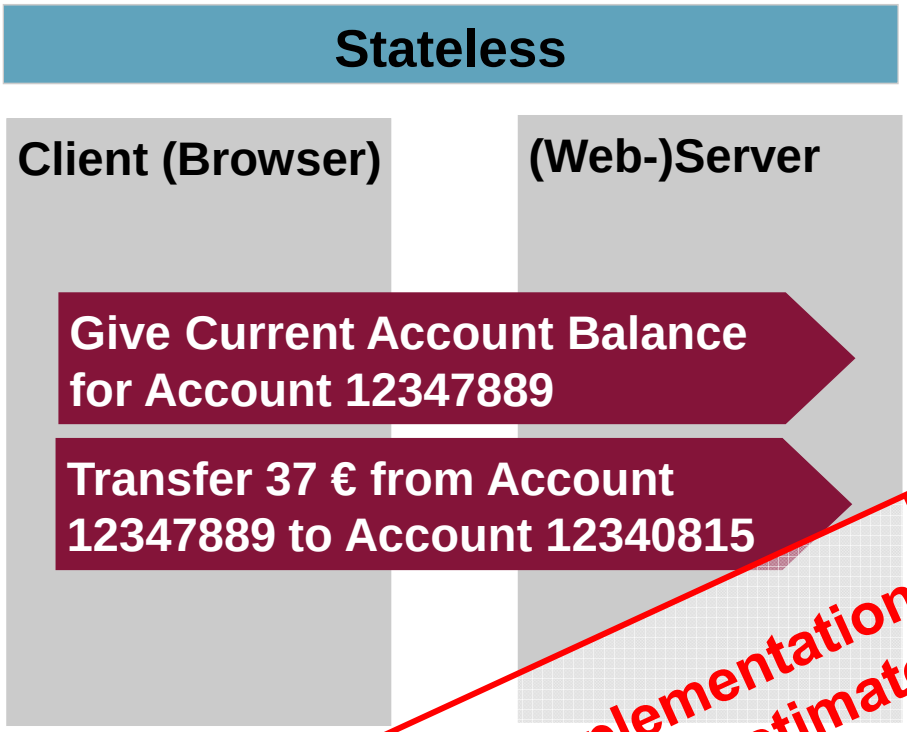
Load Balancer

- Specialized combination of hard- and software
- Configurable
  Configuration is a task for specialists and often takes some weeks
- High performance

- But: Which way do we dispatch the load?

Stateless Architectures



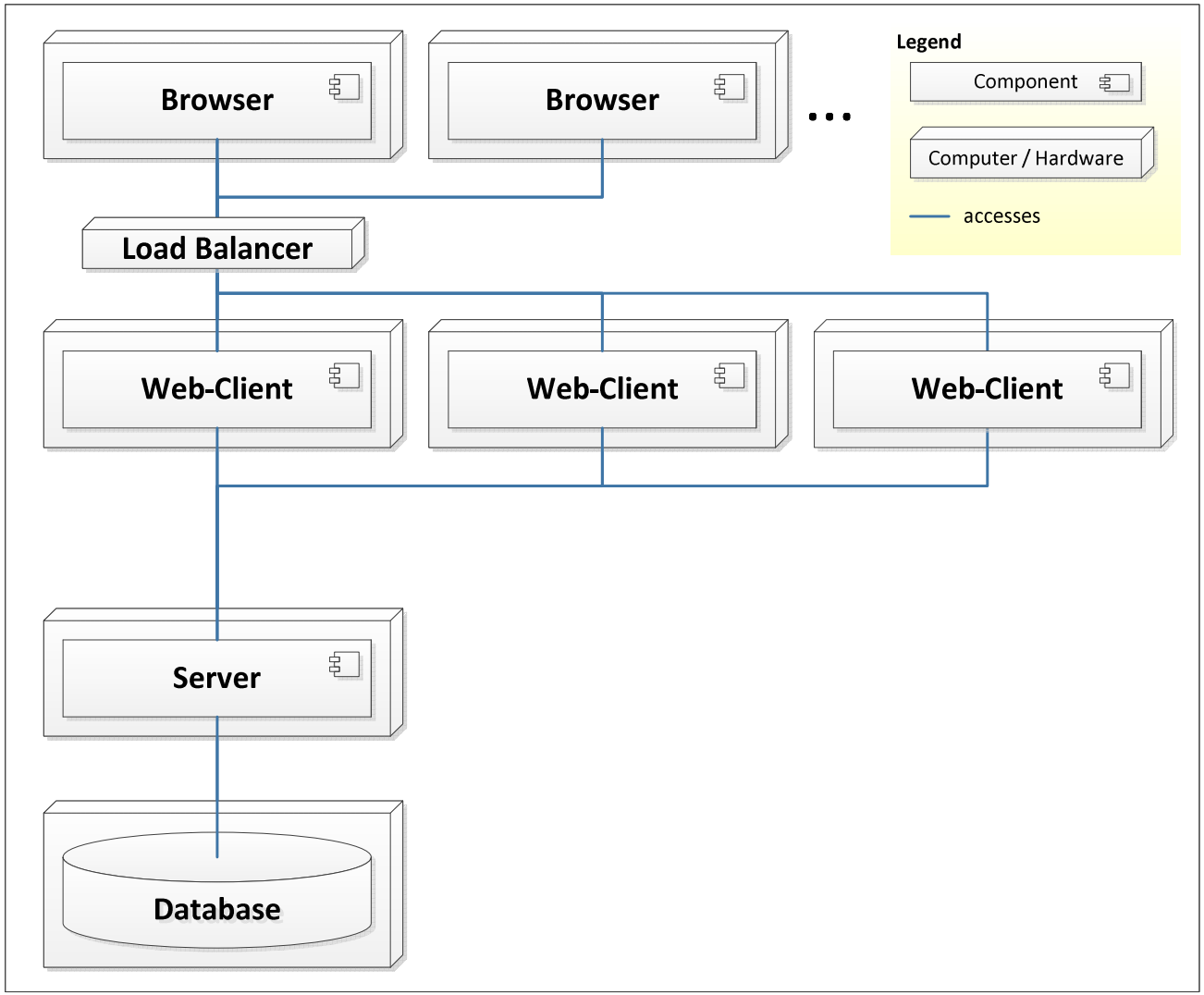


Advantages of Stateless Architectures

- Server doesn't need to hold the state
  - ➔ Less memory consumption
  - ➔ Each server can work on the request
    - ➔ Additional servers can be used to serve more users (high Scalability)

Thread safe and stateless implementation are quite complex – do not underestimate!

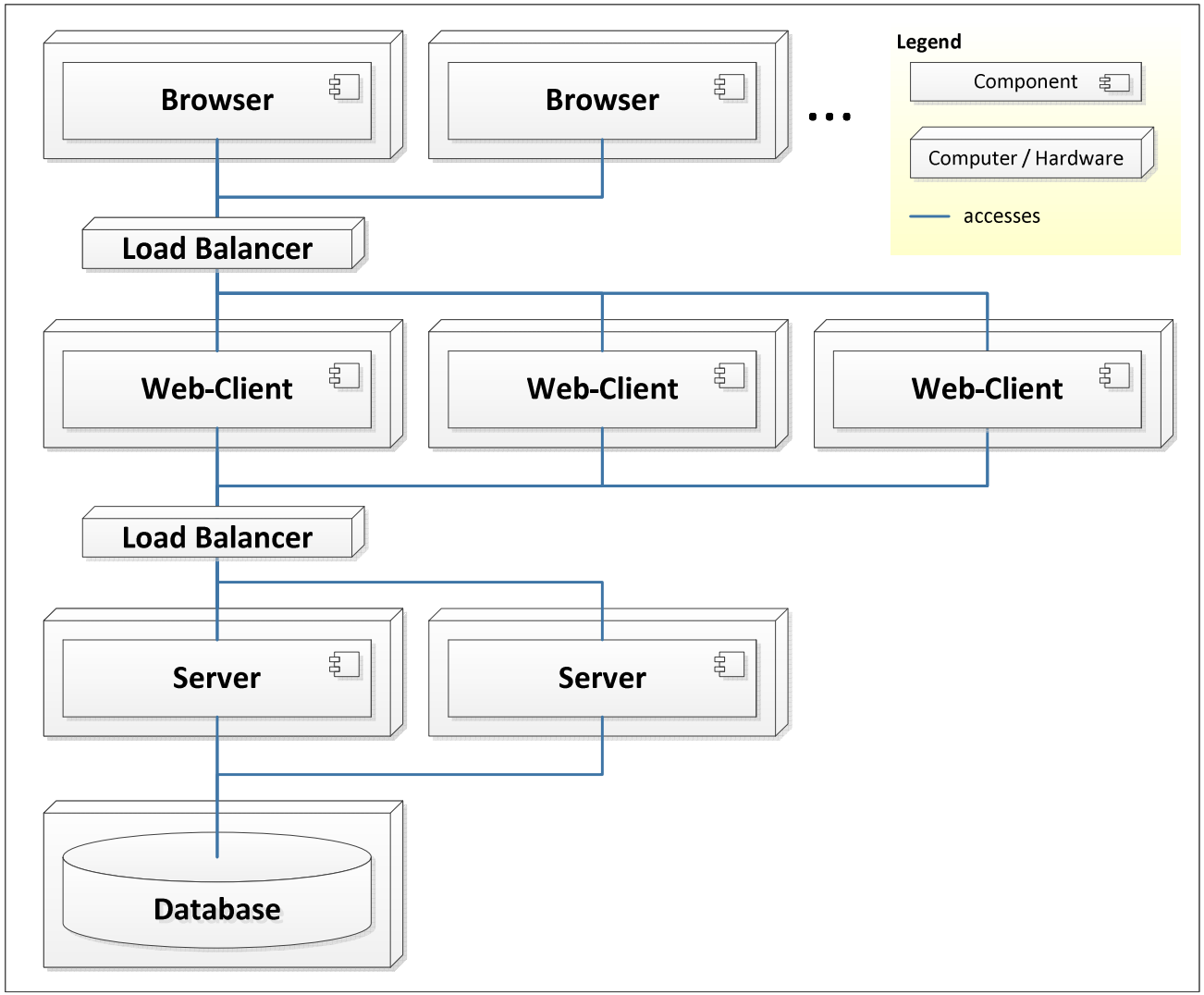
Increase the Performance of the Web-Client further



Simple

- Additional Web-Clients are easy to add.

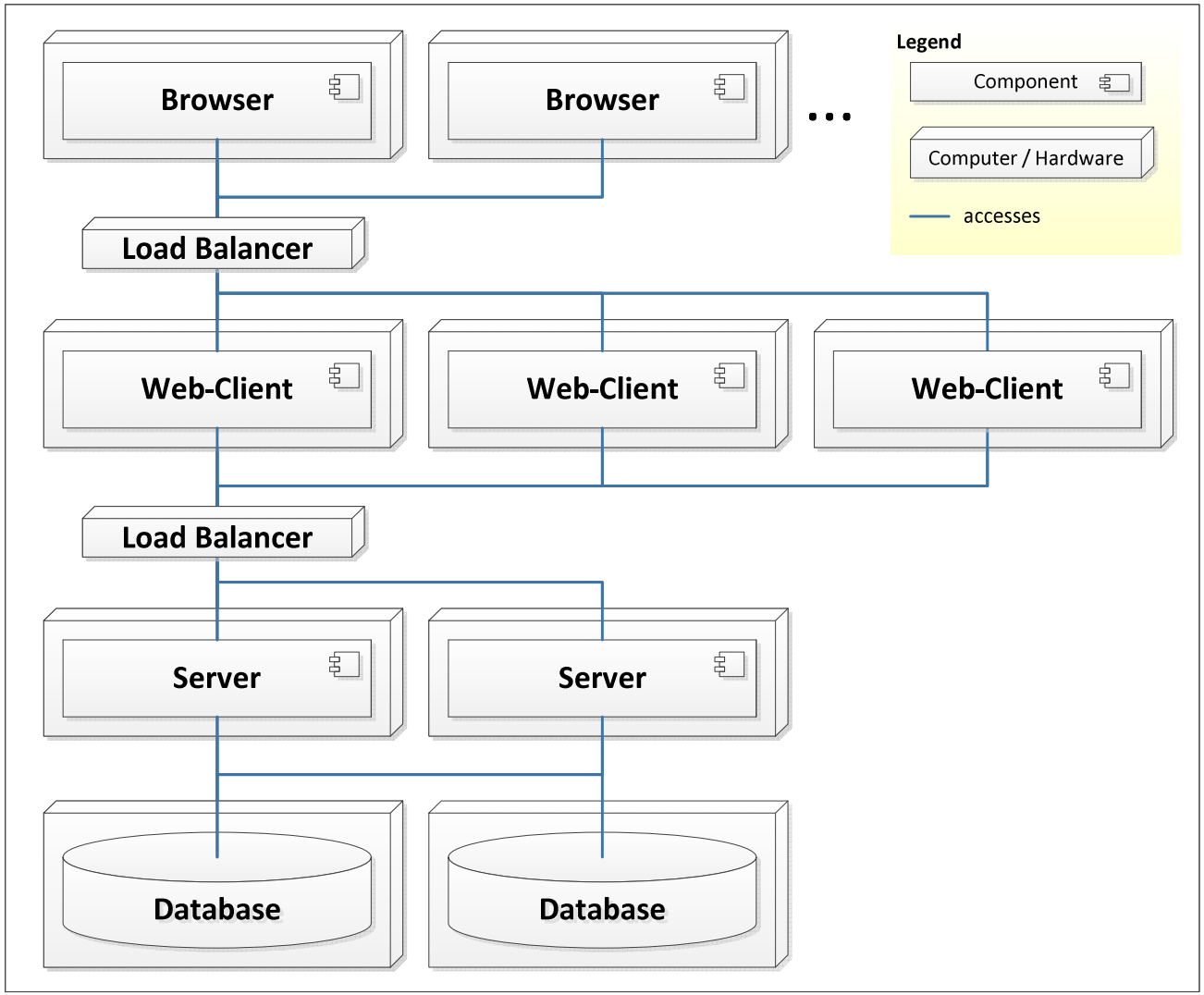
Increase the Performance of the Server



Analogous Web-Client

- The next layer will be made redundant just like the Web-Client.
- Heads up: The configuration of the load balancers is completely different from that for the Web-Clients.
- As an alternative for a load balancer most JEE containers offer software based approaches for load balancing.

Increase the Performance of the Database



DB does own Load Balancing

- All modern Database Management Systems (DBMS, just Database) do offer the ability for clustering. Here no load balancer is needed.
- But: Even though vendors of big relational databases propagate clustered databases these don't scale well.

Why Redundancy isn't that Easy

Problem parallelizable?

- Till now: No parallelizing but "just" put users on different resources
- How to we dispatch the users/requests?
- For further reflection: How would we sort in parallel?

Session Handling

- Can Web-Server 2 deal with getting session from Web-Server 1?
- Alternative: How do we bind sessions to a server?

Increased Failure Probability

- More components that could fail

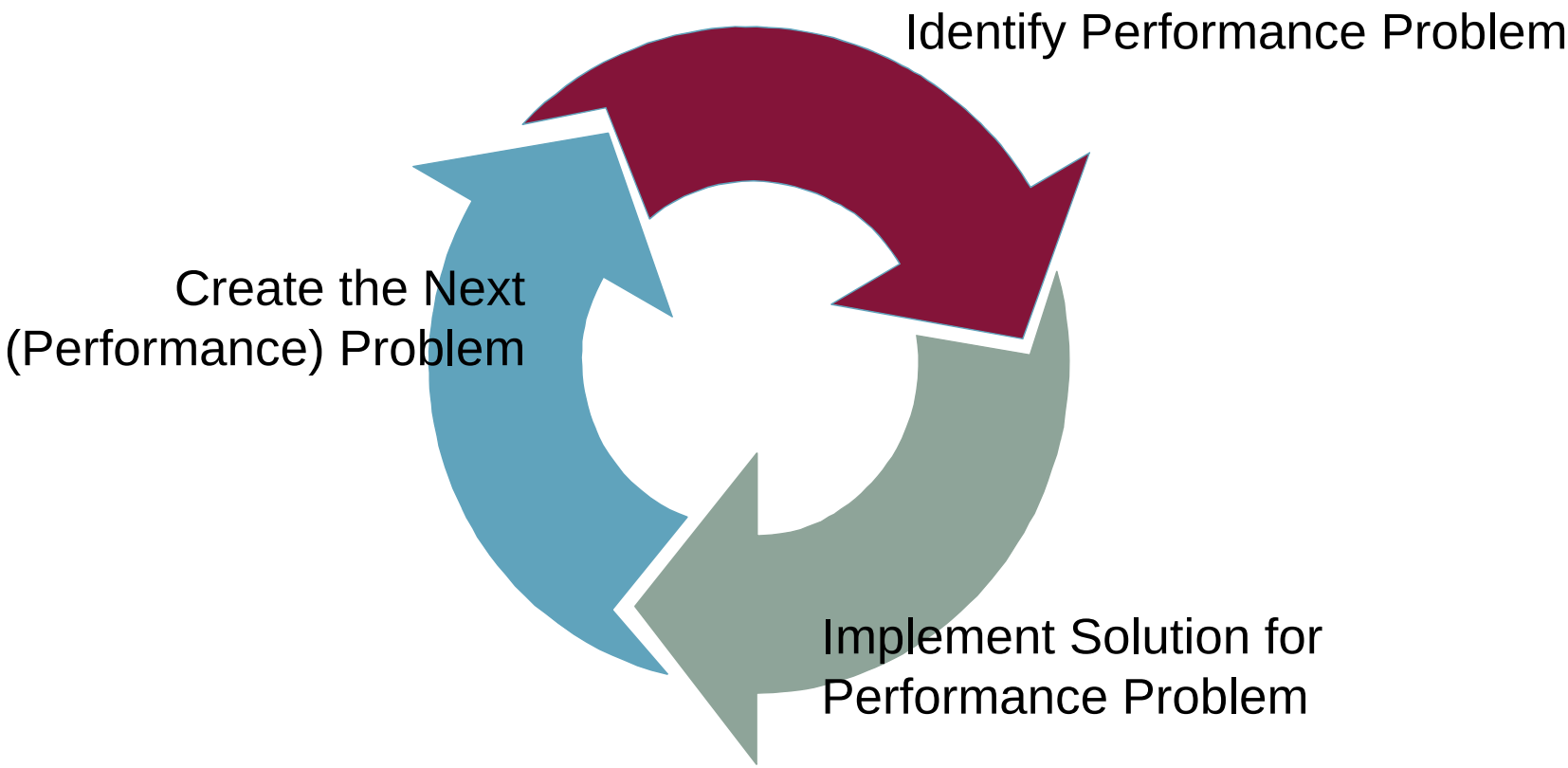
Additional Complexity

- How are the caches of Server 1 and Server 2 synchronized?
- What happens when two users try to book the last event ticket?
- How is the system updated?

…

- …

Typical Performance Improvement Cycle



Strategies for Performance III – Prepare for Redundancy

Prepare for Redundancy

- Take redundancy into account at the beginning – even if you don't need it then
- But: Don't spent (much) effort in doing so

Avoid Bottlenecks

- Design the architecture in a way that there are no Bottlenecks
  ➔Extreme difficult task

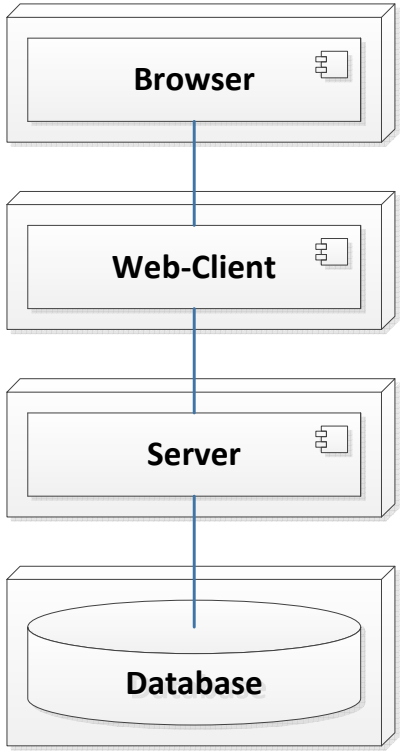
AGENDA

1. What is Performance and Why does it matter?
2. Performance in the Activities of Software Engineering
3. Architectures for Scalable & High-Performance Systems
4. Architectures for High Available Systems

Availability by Example: A Highly Available Shop

Scenario The company *Extreme Adventures* is very successful. With just few customers they earn much money. Because of that the extensions for more performance were never implemented.

But now the company is facing another problem: A failure of the shop system would cause a loss of sales. But even worse: The reputation of the company as very reliable would be damaged. Therefore the availability of the shop system should be increased to 99,999 % (5 minutes downtime per year).



Availability – Measuring Unit

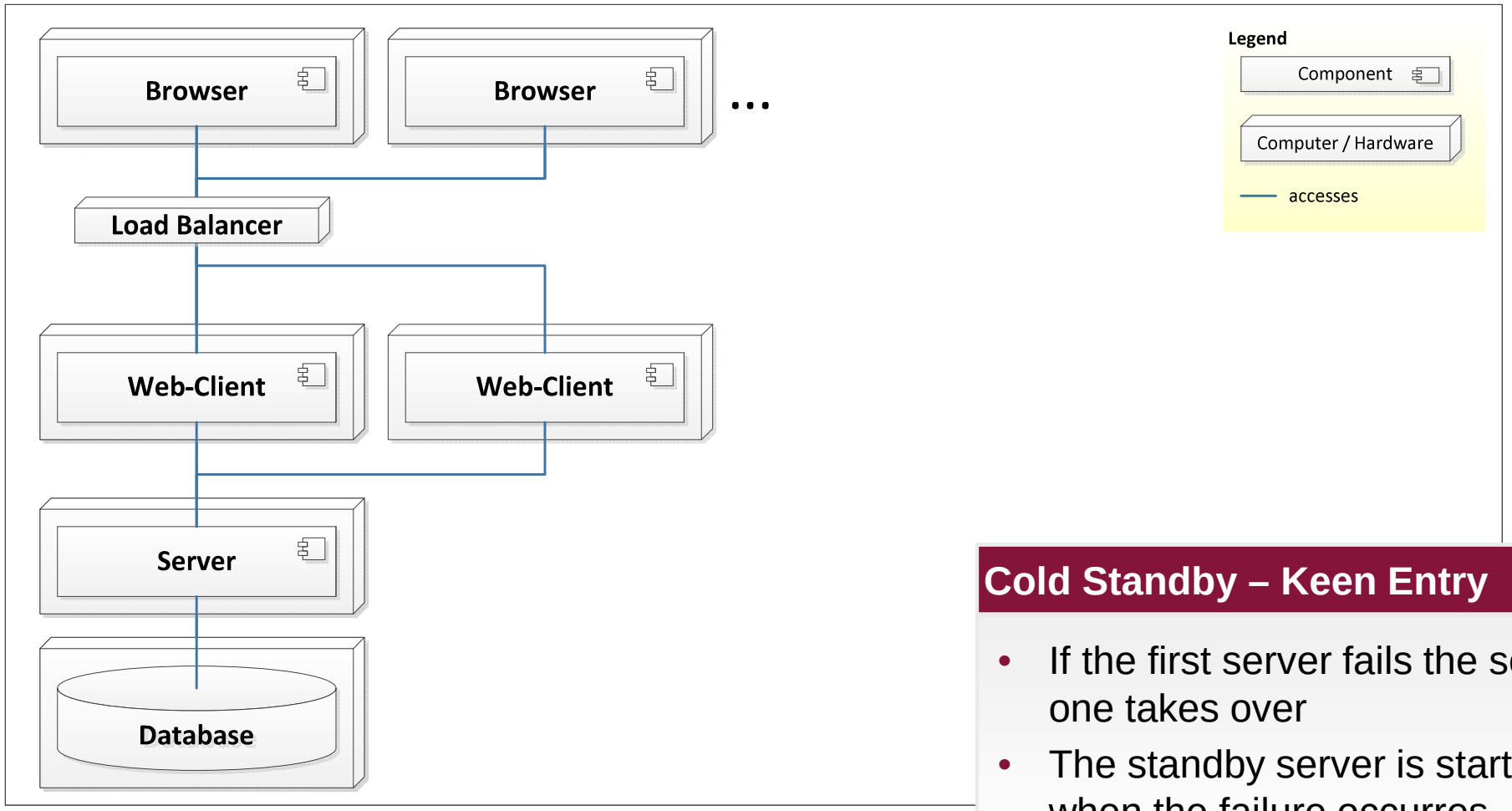
Availability The ratio of (a) the total time a functional unit is capable of being used during a given interval to (b) the length of the interval.

$$A = \frac{E[\text{Uptime}]}{E[\text{Uptime}] + E[\text{Downtime}]}$$

Availability	Term	Maximal Downtime per Year
99 %		~ 87 hours
99,9 %	3 nines	~ 9 hours
99,99 %	4 nines	~ 53 minutes
99,999 %	5 nines	~ 5 minutes

High Availability

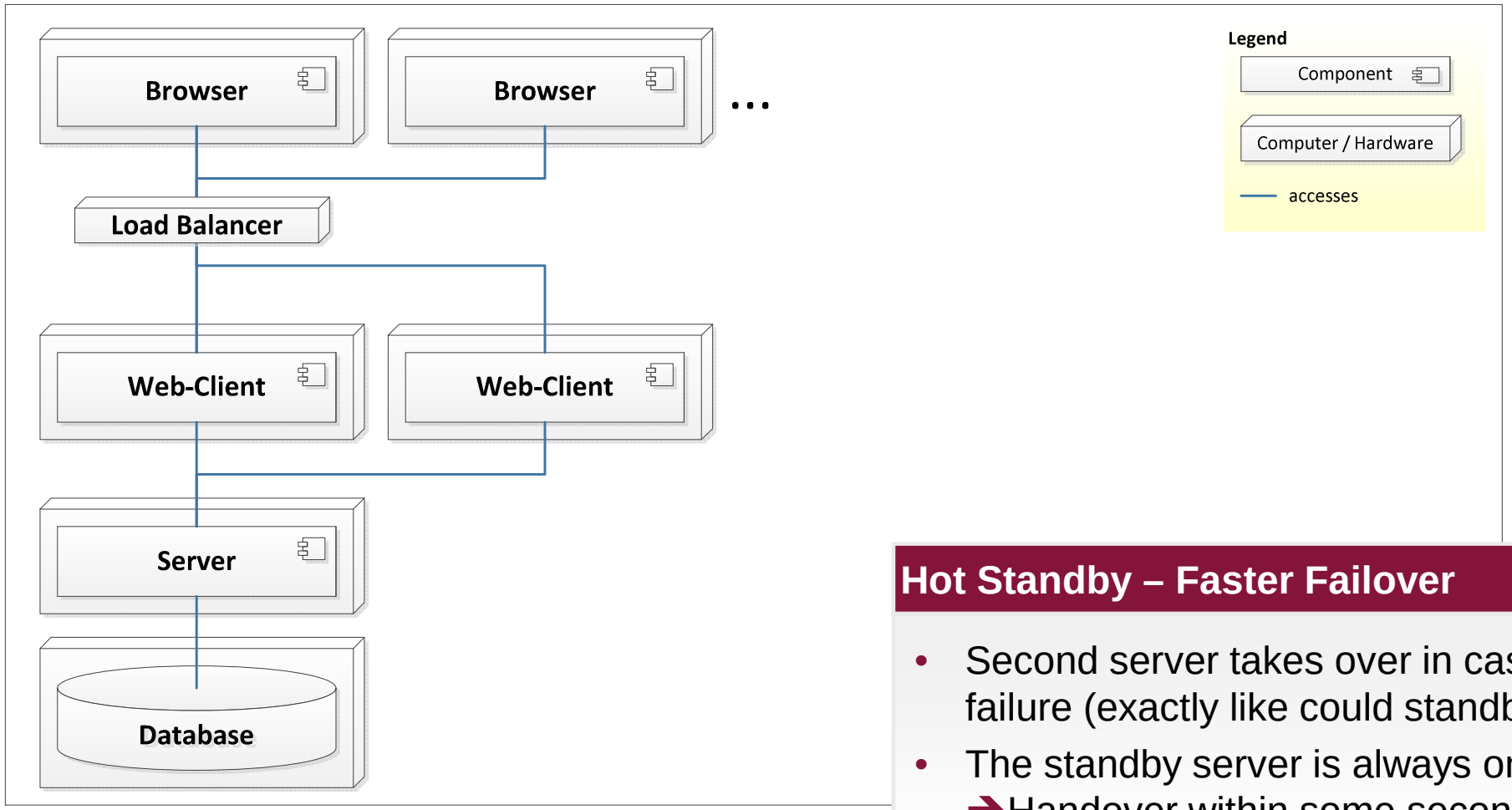
Cold Standby – a Second Server Is Available



Cold Standby – Keen Entry

- If the first server fails the second one takes over
- The standby server is started when the failure occurres

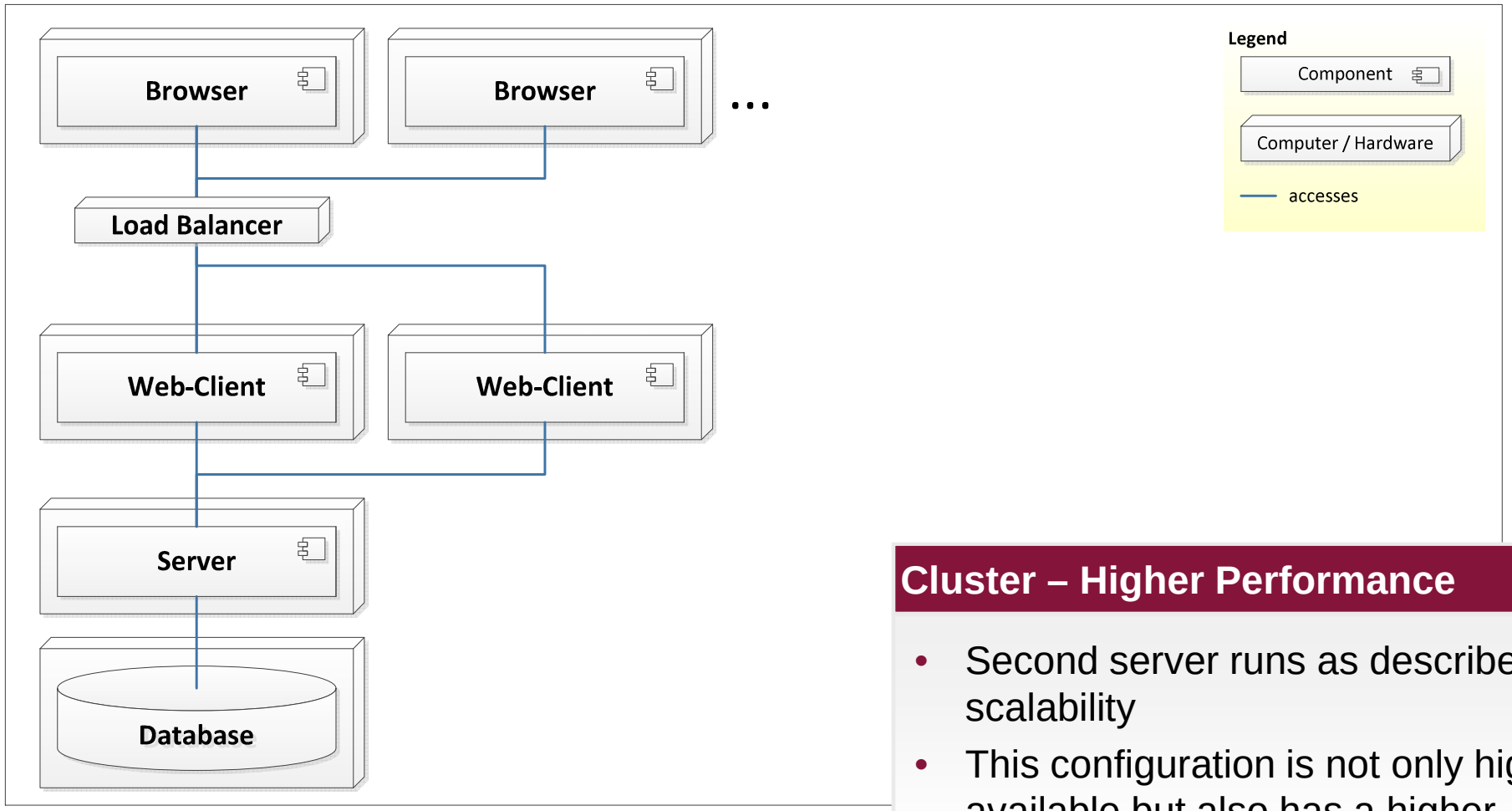
Hot Standby – the Second Server Is Always Online



Hot Standby – Faster Failover

- Second server takes over in case of failure (exactly like could standby)
- The standby server is always online
  ➔Handover within some seconds

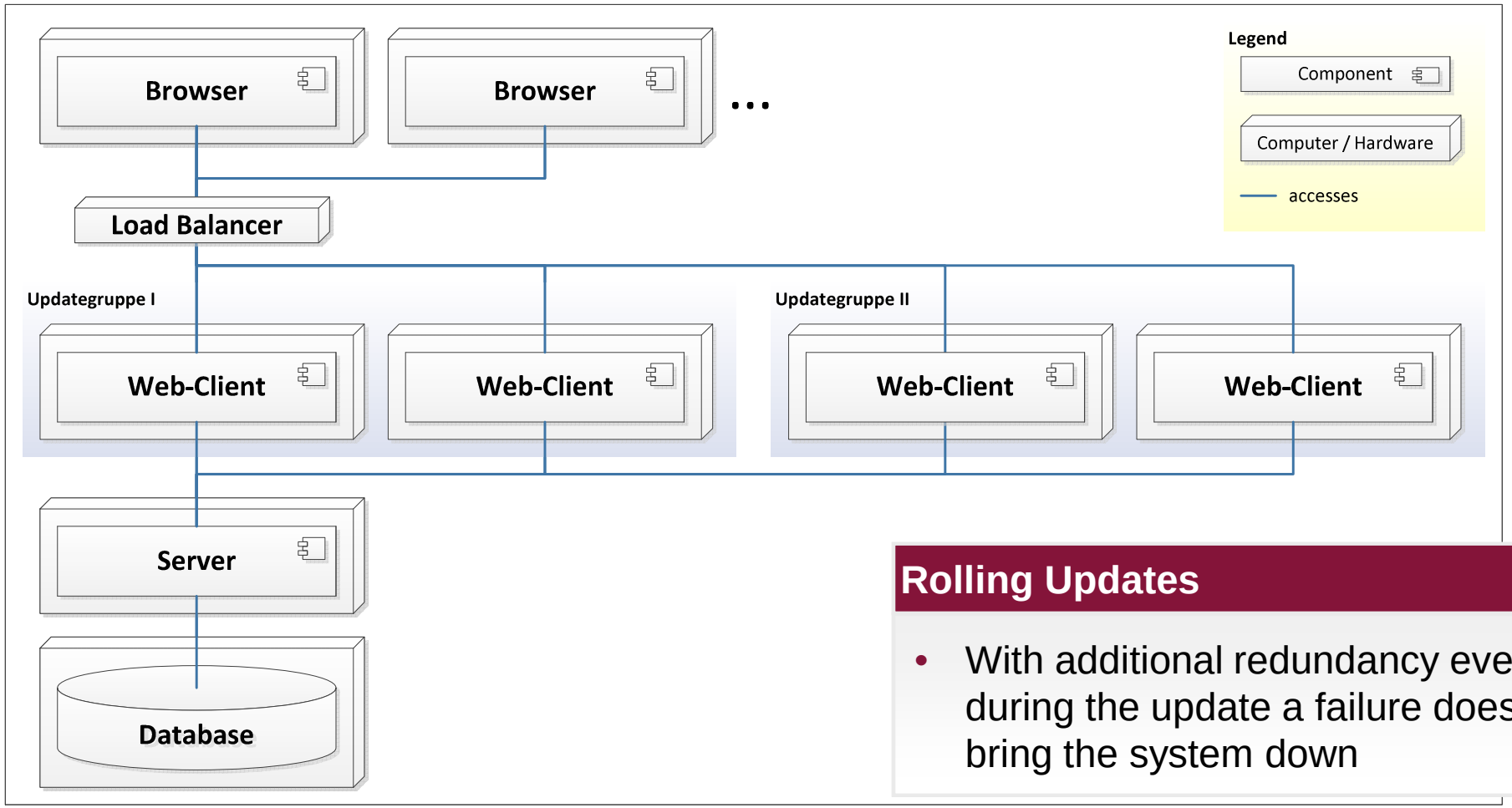
Cluster – the Second Server Takes a Part of the Load



Cluster – Higher Performance

- Second server runs as described for scalability
- This configuration is not only higher available but also has a higher performance

Rolling Updates – When the Server Fails During An Update



Rolling Updates

- With additional redundancy even during the update a failure doesn't bring the system down

Why Redundancy isn't that Easy II

High Complexity

- Instead of one server now four (!) servers
- Complex dependencies + proceedings (Rolling Update)
  ➔ Probability for human failure increases

Systematically not Taken into Account

- Operator error
- Software error

➔ Both are more likely to happen that a hardware failure

Technically not Taken into Account

- „cleaner trips on the network cable "
- Connection to the Internet
- How to detect a fallen out system?
- Failure of one system causes the next one to fail → domino effect
- blackout, fire, flooding, earthquake, …

Summary and Perspective

1. What is Performance and Why does it matter?
2. Performance in the Activities of Software Engineering
3. Architectures for Scalable & High-Performance Systems
4. Architectures for High Available Systems
5. Performance Measurement
6. Performance Analysis of Existing Systems

Thank you for your attention.

Andreas Rothmann
Principal IT Consultant
Max-Planck-Straße 40
50354 Hürth
Telefon: +49 2233 9721 6311
andreas.rothmann@msg-systems.com

www.msg-systems.com

.consulting .solutions .partnership

.msg