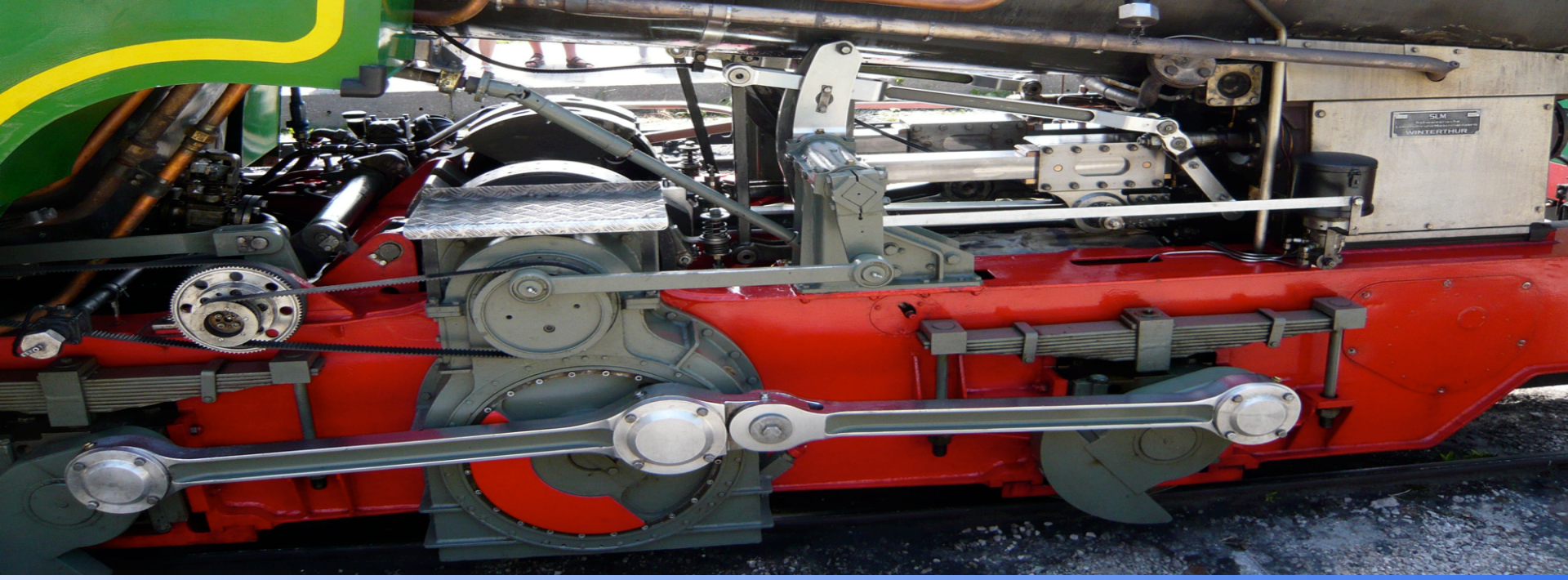


Best Practices to create High Load Websites

Beolink.org

Agenda

- Introduction
- Design and Optimizations
  - Generic Optimizations
  - Presentation Layer
  - Application Layer
  - Data Layer
- Example
- Service Operation
  - Monitoring
  - Emergency Operations

Introduction: The value of an IT Service

From an ITIL perspective, the value is composed by two components: utility (fitness for purpose) and warranty (fitness for use.)

Utility is a "[f]unctionality offered by a product or service to meet a particular need. Utility is often summarized as 'what it does'."

Warranty as "[a] promise or guarantee that a product or service will meet its agreed requirements" and as "derived from the positive effect of being available when needed, in sufficient capacity, and dependably in terms of continuity and security."

Do you have the new Amazon web site ?

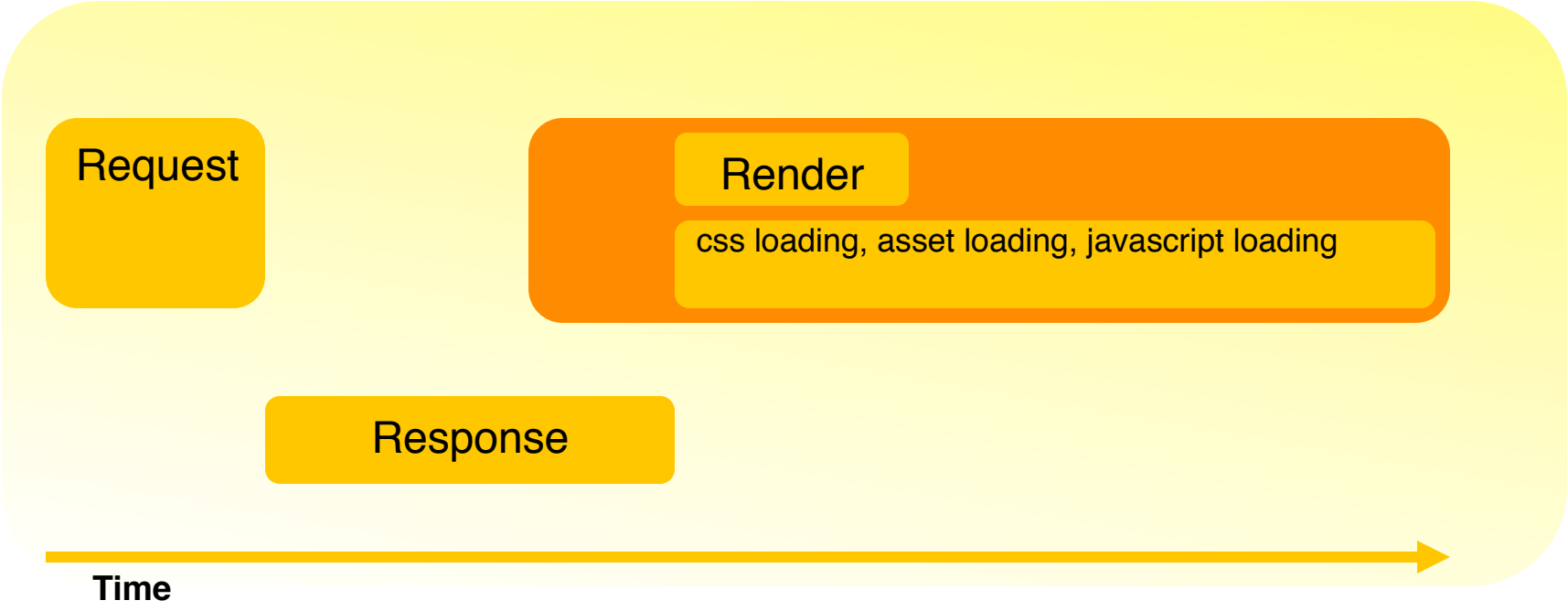
4.3 million pages/day
= 50 pages/s, I don't think so !

1000 pages/s = 86 mil pages/day
Interesting figure…

How fast is fast enough?

80% of the end-user response time is spent on the front-end.

Most of this time is tied up in downloading all the components in the page: images, stylesheets, scripts, Flash, etc. Reducing the number of components in turn reduces the number of HTTP requests required to render the page (see Netflix case studies)



Do you know your Application Architecture ?

Introduction: Architecture [ˈɑːkɪˌtɛktʃə] Beolink.org

A **software architecture** is an abstraction of the run-time elements of a software system during some phases of its operation. A system may be composed of many levels of abstraction and many phases of operation, each with its own software architecture.

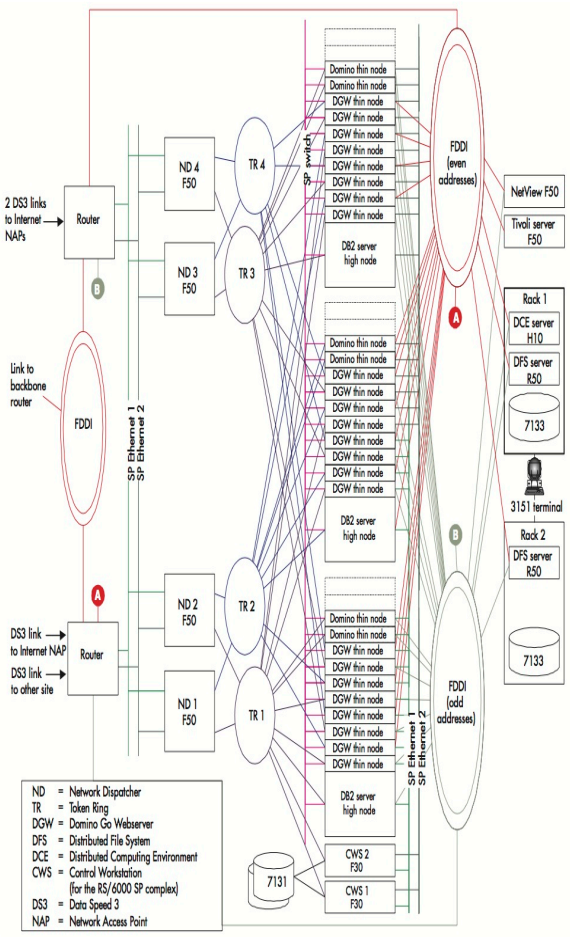
A **software architecture** is defined by a configuration of architectural elements--components, connectors, and data--constrained in their relationships in order to achieve a desired set of architectural properties.

A **component** is an abstract unit of software instructions and internal state that provide a transformation of data via its interface.

A **connector** is an abstract mechanism that mediates communication, coordination, or cooperation among components.

Datum is an element of information that is transferred from a component, or received by a component, via a connector.

[1] Roy Filding theis



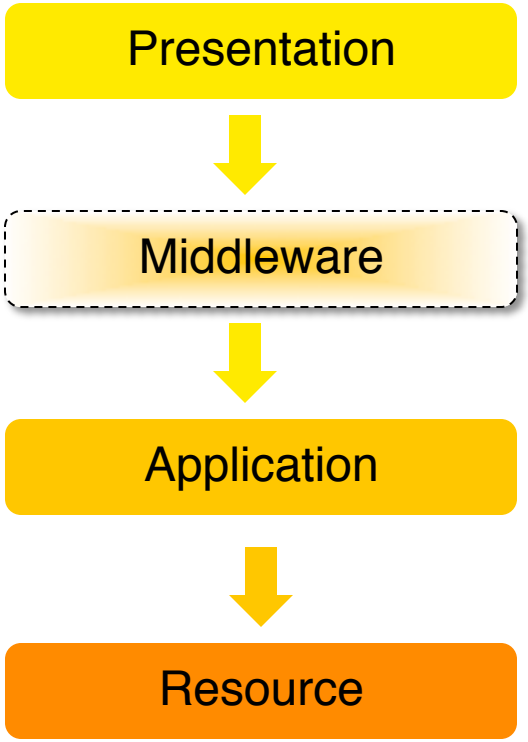
Introduction: Web Architecture

Presentation Layer supports a client, the system needs to have a presentation layer through which the user can submit operations and obtain a result.

Middleware Layer is just a level of indirection between presentation and other layers of the system. It introduces an additional layer of business logic encompassing all underlying systems.

Application layer establishes what operations can be performed over the system and how they take place. It takes care of enforcing the business rules and establishing the business processes.

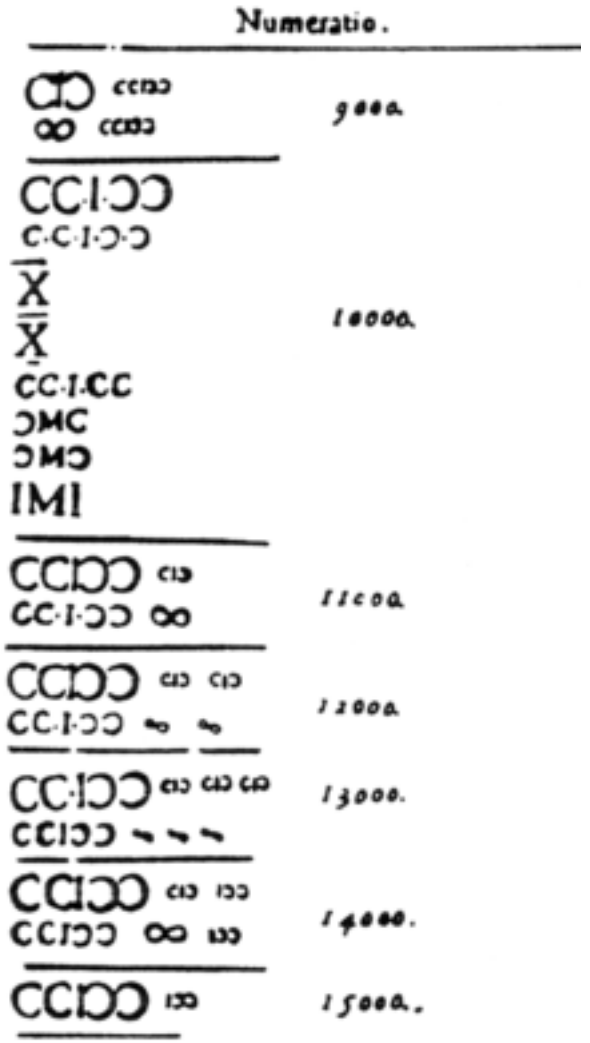
Resource (Datum) deals with the organization (storage, indexing, and retrieval) of the data necessary to support the application logic.



Design

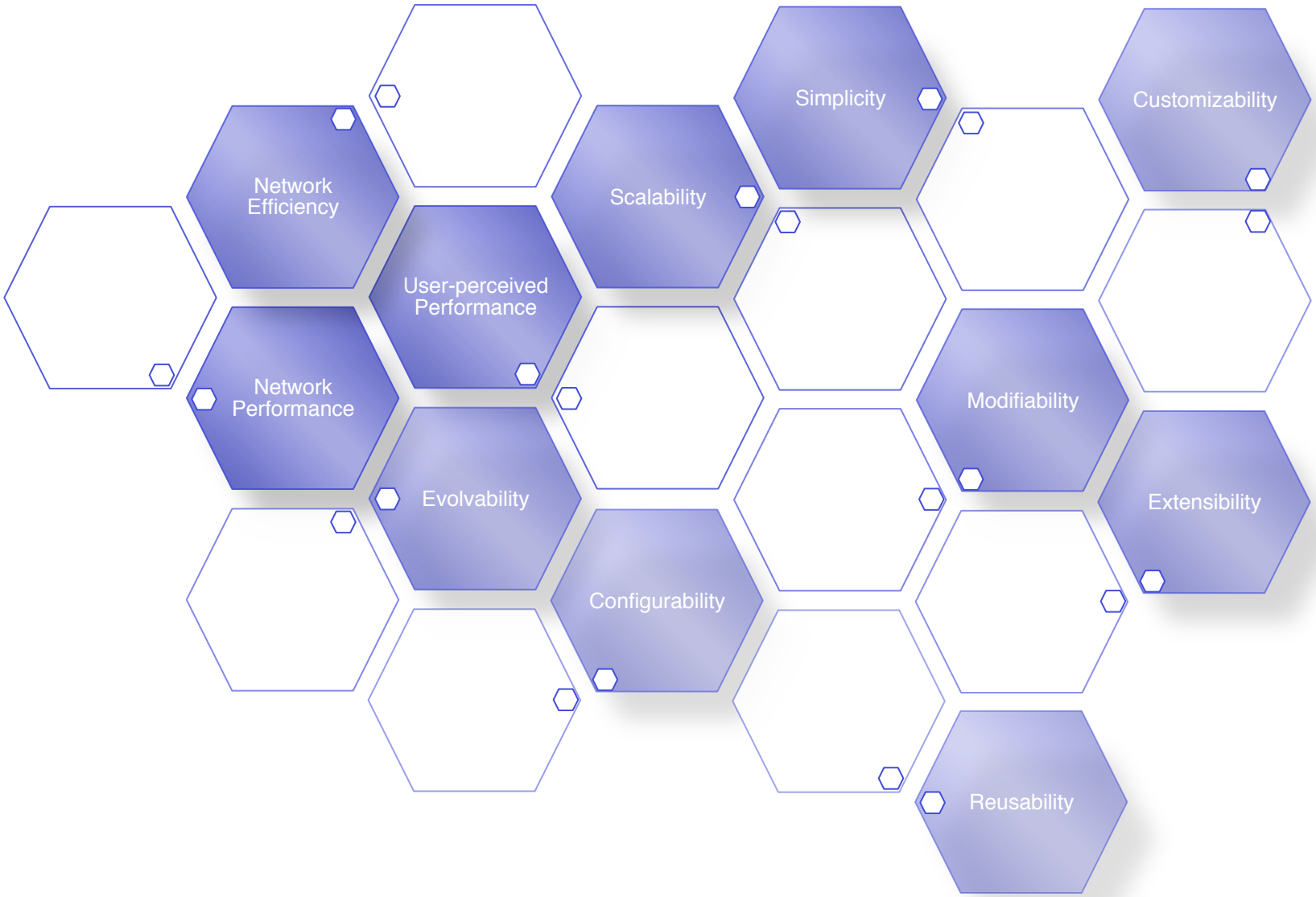
Design: Numbers

- Request per second
- Page composition
- Content type/Dimension
- Number of users
- Peaks



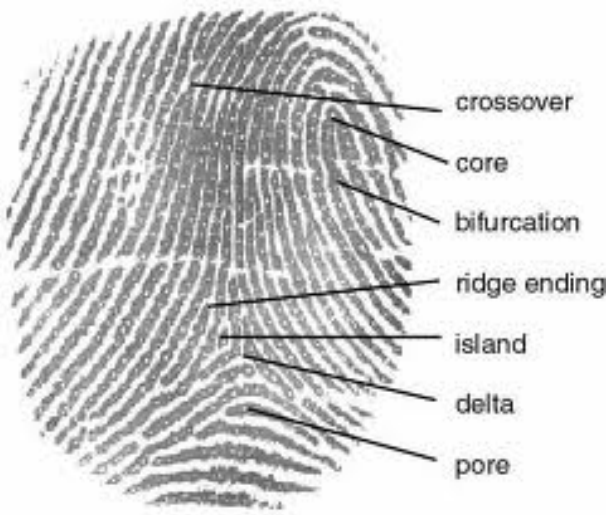
Introduction: Architectural Properties

Beolink.org



Design: Matrix

Properties	Value
Network Performance	
User-perceived Performance	
Network Efficiency	
Scalability	
Simplicity	
Modifiability	
Evolvability	
Extensibility	
Customizability	
Configurability	
Reusability	

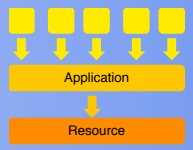


Layers

Design: Layer 0

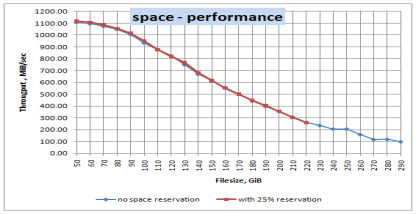
- Split the system in pieces
- I/O
- TCP/IP
- Filesystem

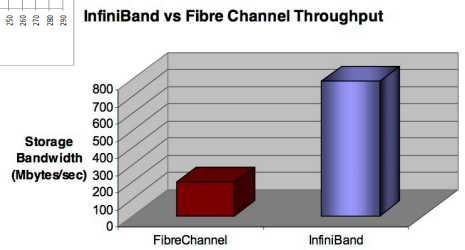
Design: I/O



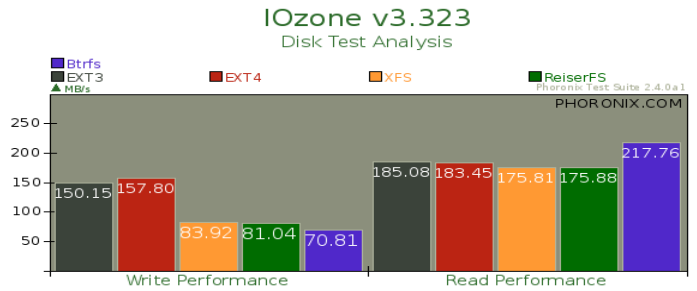
Beolink.org

- **I/O**
  - Ethernet: bonding
  - Disks: SAS/SSD/infinibend
- **Filesystem**
  - Avoid NFS
  - Different FS (for operation type)
  - Specific Options (noatime)
  - Block size
  - Journaling
- **Logging**
  - Dedicated sites
  - AMQP





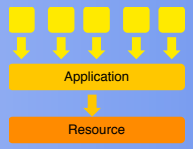
ls: cannot access nfsdir: Stale NFS file handle



Qpid™

760,000msg/sec ingress on an 8 way box or 6,000,000msg/sec OPRA messages.

Design: TCP/IP



Beolink.org

TCP/IP

net.ipv4.tcp_tw_reuse=1
net.ipv4.tcp_tw_recycle=1
net.ipv4.tcp_fin_timeout=30
net.ipv4.tcp_keepalive_time=300
/proc/sys/net/ipv4/ip_local_port_range
fs.file-max=128000
net.core.somaxconn=250000
net.ipv4.tcp_max_syn_backlog=2500
net.core.netdev_max_backlog=2500
ulimit -n 10240

500 req/sec*900 = 450.000 sockets

IP Contract

net.ipv4.netfilter.ip_conntrack_max

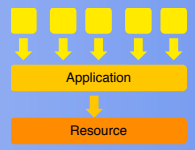
ip_conntrack: table full, dropping packet.

Firewall

Request / rate
IP ACL
Dynamic ACL (phrel)

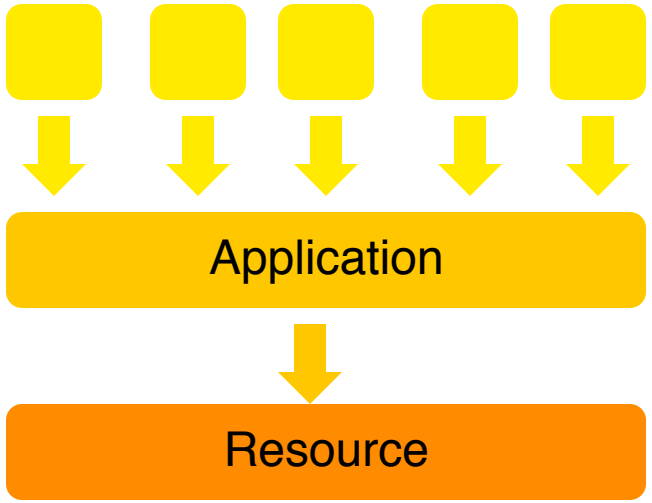
--limit rate
--limit-burst number

Design: Presentation



Beolink.org

- Yahoo Rules
- Load Distribution
- Proxy/Caching
- Web Server
- Content Delivery Network
- Split access on different Domains



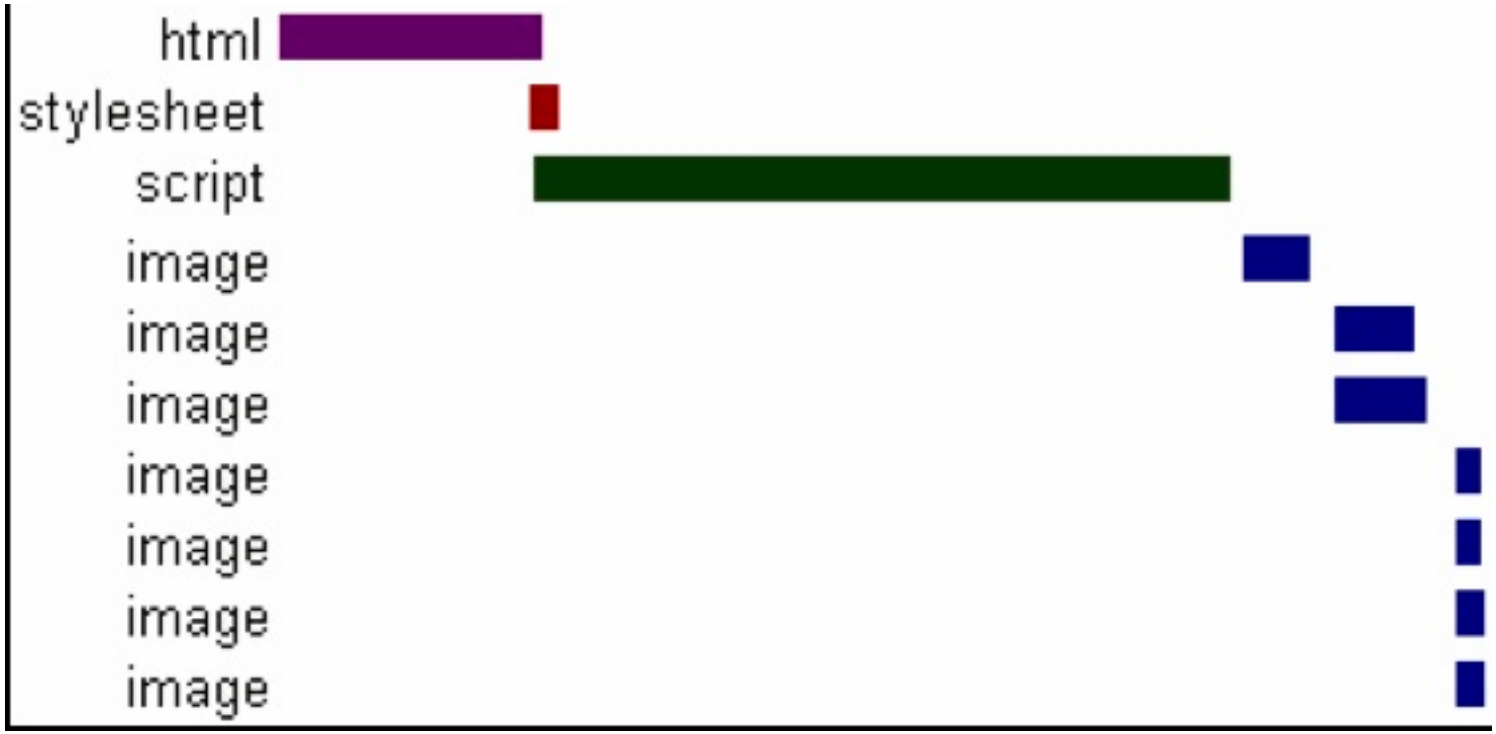
Design: Yahoo! Rules

The Exceptional Performance team (Yahoo!) has identified a number of best practices to make web pages fast. The list includes 35 best practices divided into 7 categories.

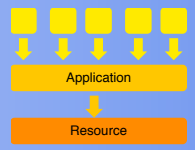
25 % without modifying infrastructure

Content	Server	Cookie	CSS	Javascript	Images	Mobile
•Make Fewer HTTP Req •Make Ajax Cacheable •Reduce the number of DOM •…	•Use get for Ajax Req •Flush Buffer Early •…	•Reduce Cookie Size •Use Cookie-Gree Domains •…	•Put Stylesheets at Top •Avoid CSS exp •Avoid Filters •…	•Put scripts at Botton •Make javascript and CSS ext •Minify •…	•Make favicon.ico small and Cacheable •Optimize •Do not Scale Image in HTML •…	•Keep components under 25 kb •Pack components into a multipart document

Example

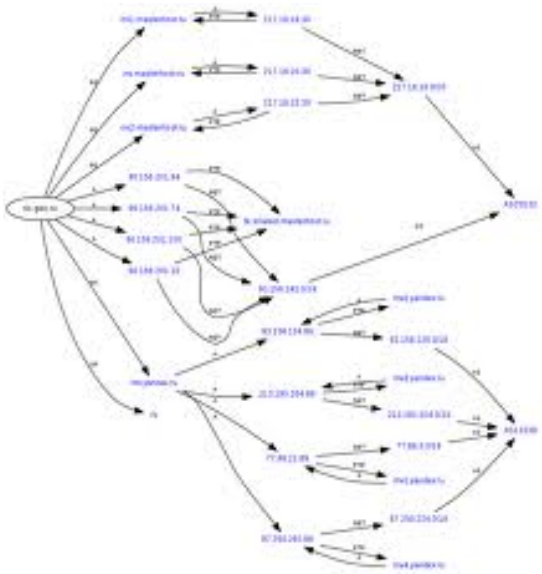

html
stylesheet
script
image
image
image
image
image
image
image

Design: Load Distribution



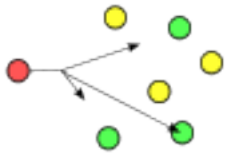
Beolink.org

- **DNS**
  - Low TTL
  - GEO IP
  - Round Robin
  - More than one IP
  - Response base on system load
  - Split Components across Domains

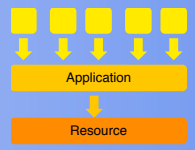


- **Load Balancer**
  - TCP/IP
  - Layer 7
  - SSL

- **Anycast**
  - Up to 32 systems per IP
  - High availability on WAN

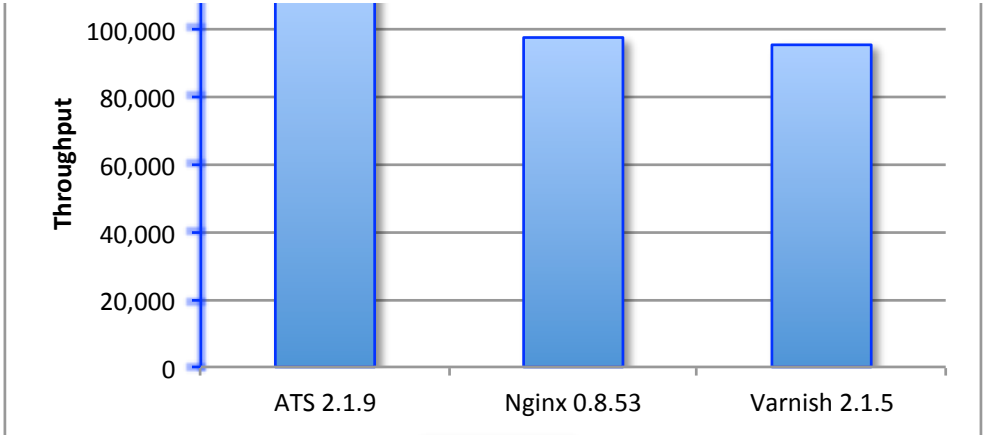


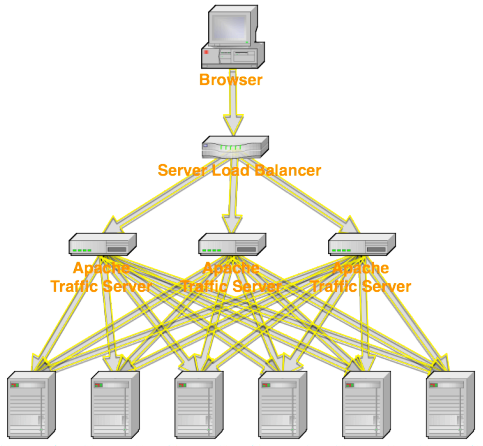
Design: Proxy/Caching



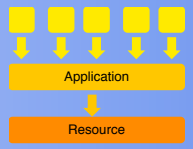
Beolink.org

- Configure ETags
- Add expiration or Cache-control Header
- Extension modules
- Reverse Proxy base on url or domains
- Redirect on business logic (Middleware)





Design: WebServer



Beolink.org

- VirtualHost with dedicated IP
- Compress content
- Process Model
  - Number of Process
  - Number of clients
  - Spare..
- KeepAlive and KeepAliveTimeout
  The KeepAlive directive allows multiple requests to be sent over the same TCP connection. It changes model from Request to User

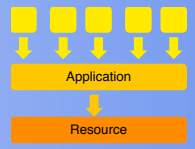
		Gzip		Deflate	
	Size	Size	Savings	Size	Savings
Script	3.3K	1.1K	67%	1.1K	66%
Script	39.7K	14.5K	64%	16.6K	58%
Stylesheet	1.0K	0.4K	56%	0.5K	52%
Stylesheet	14.1K	3.7K	73%	4.7K	67%

Apache optimization

Remove unneeded modules
Set AllowOverride to None
Avoid FollowSymLinks and SymLinksIfOwnerMatch
Avoid content negotiation (Multiview)
MaxClients =(Total Memory - Operating System Memory) / Size Per Apache process.

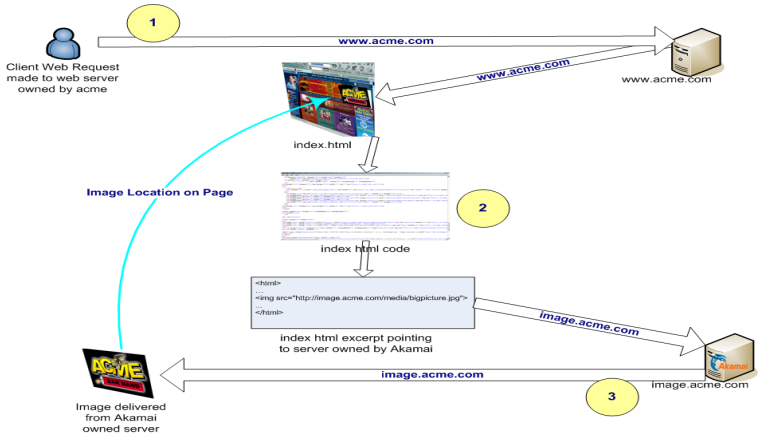
MinSpareServers, MaxSpareServers, and StartServers: Apache can spawn a maximum of 32 child processes per second

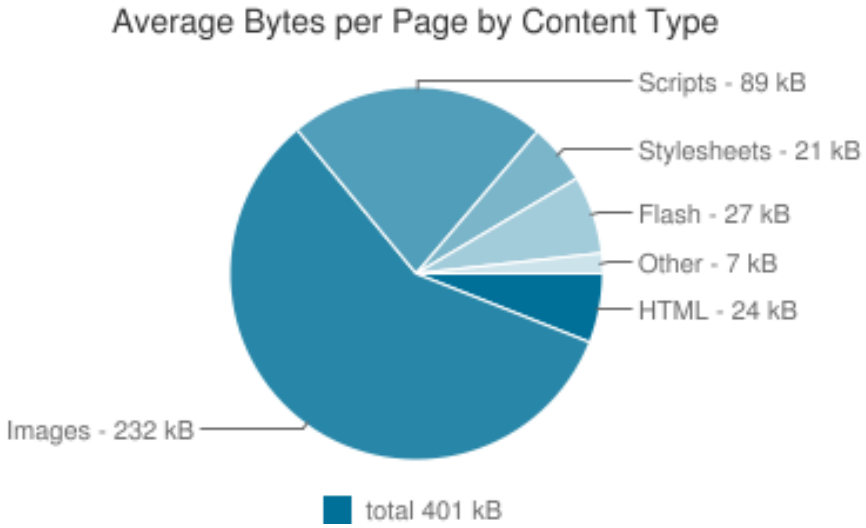
Design: Content Delivery Network



Beolink.org

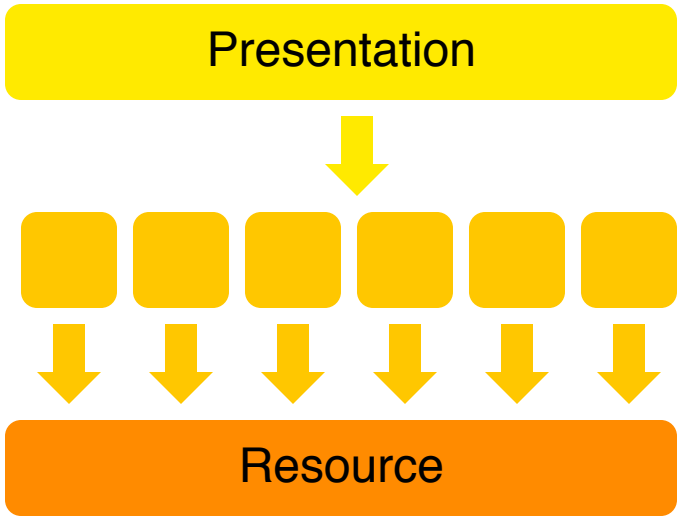
A content delivery network or content distribution network (CDN) is a system of computers containing copies of data placed at various nodes of a network.



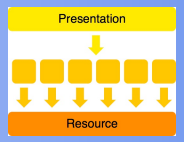


- Preload
- Access Control

- Distribution
- Accelerator
- Session and Cookies
- More instance on same system



Design: Distribution

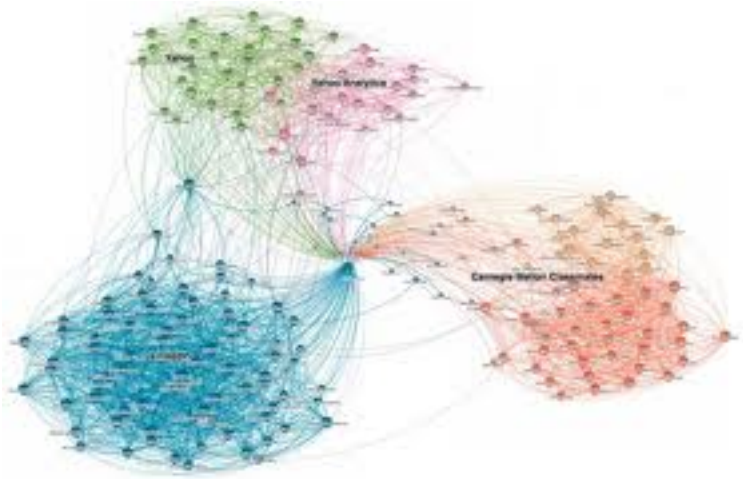


Beolink.org

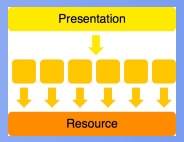
- **Shared**
  All components have all the functions (round robin)

- **Function/resource**
  Components are grouped by function/resource

- **User**
  Components are divided by cluster of users



Design: Accelerator

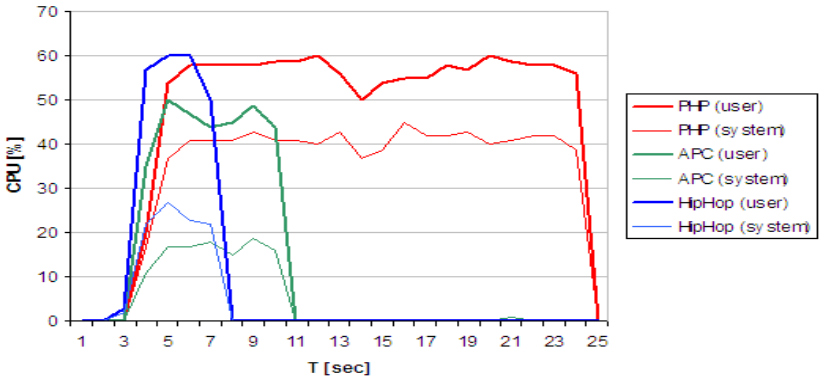


Beolink.org

❑ PHP Accelerator

	PHP	with APC	%	with eA	%
Tot sec	175.62	33.21	528.83%	29.18	601.85%
Req/sec	5.69	30.11	529.17%	34.26	602.11%
ms per req	175.62	33.21	528.83%	29.19	601.71%

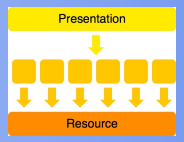
HIPHOP



❑ Python

Framework	Transaction rate [1/sec]
Go http	2063
Twister	2020
Web.go	1753
Tornado	1662
Tornado+nginx	1364
Web.py+gevent	888
Web.py+gunicorn	538
Web.py+CheryPy	304
Web.py+flup+nginx	211

Design: Modular Application Logic



Beolink.org

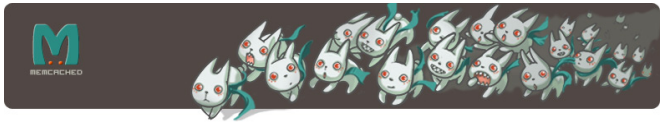
Cookies

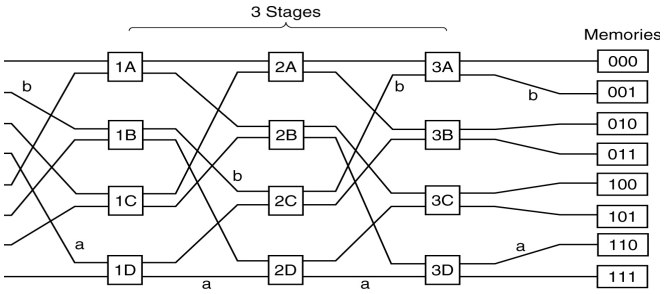
- Size
- Encryption (key rotation)

Cookie Size	Time	Delta
0 bytes	78 ms	0 ms
500 bytes	79 ms	+1 ms
1000 bytes	94 ms	+16 ms
1500 bytes	109 ms	+31 ms
2000 bytes	125 ms	+47 ms
2500 bytes	141 ms	+63 ms
3000 bytes	156 ms	+78 ms

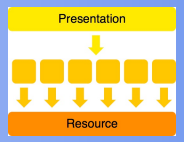
Session

- Sticky session->table in memory
- Round Robin->memcached
- Two levels (NUMA)





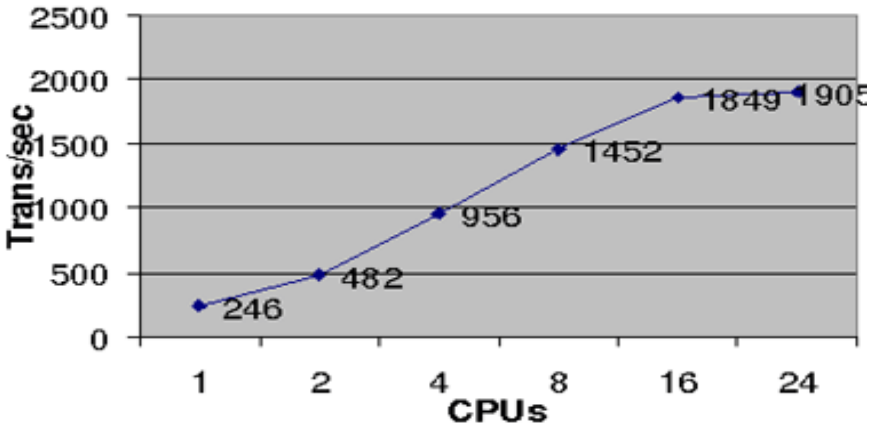
Design: More instance



Beolink.org

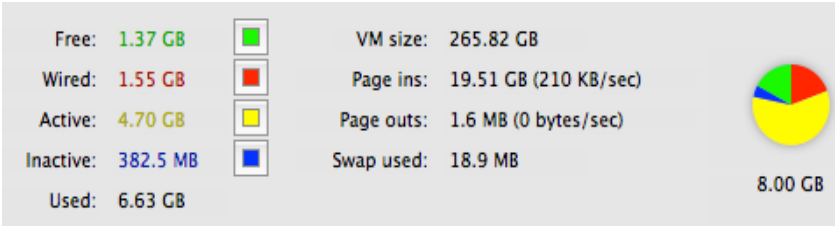
Better CPU Usage

Many applications do not support parallelization, then it is not possible to implement a scale up approach (deploying the applications on larger servers)

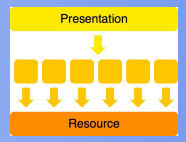


Better Memory Usage

Many applications still use 32 bits or have fixed internal data structure

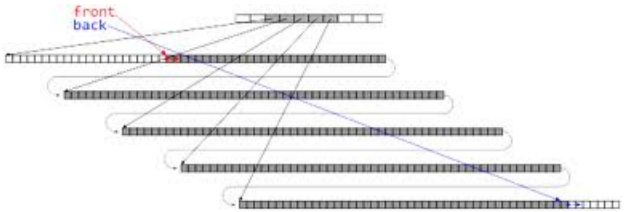


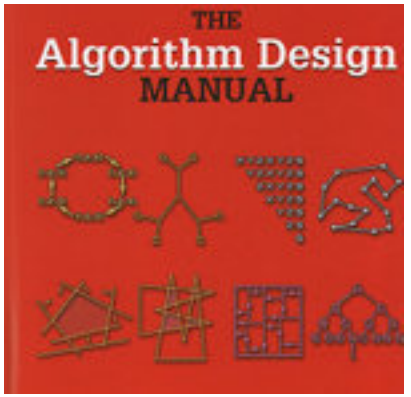
Design: Modular Application Logic



Beolink.org

- **Choose the correct algorithms and data structures**
  - dqueue vs list, hash vs trees, locks vs read/write locks, bloom filter

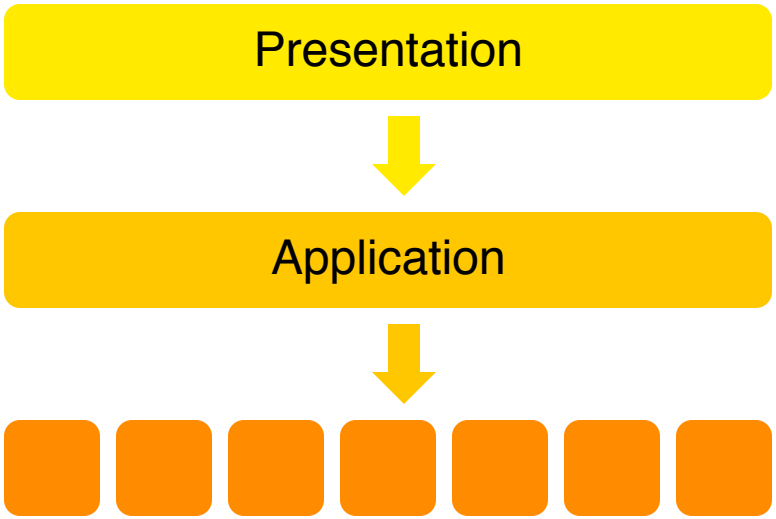




- **Memory allocation**
  - Reuse memory, stack vs heap, tcmalloc
- **Make fewer system calls**
  - Larger writes and reads

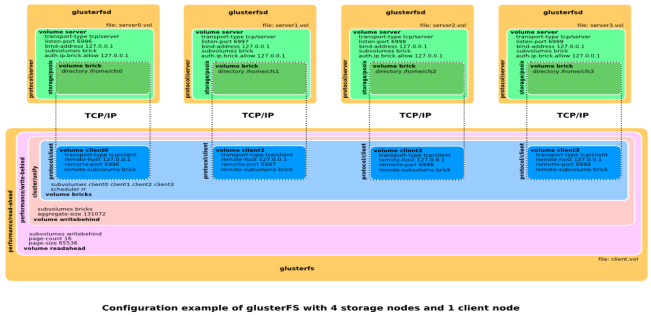
Design: Data Layer

- **Filesystem**
  - Distributed
  - Replication
- **Database**
  - Partitioning
  - Replication
  - NoSQL
- **Hierarchical Storage**
  - Directory Server
  - JSR-170/230



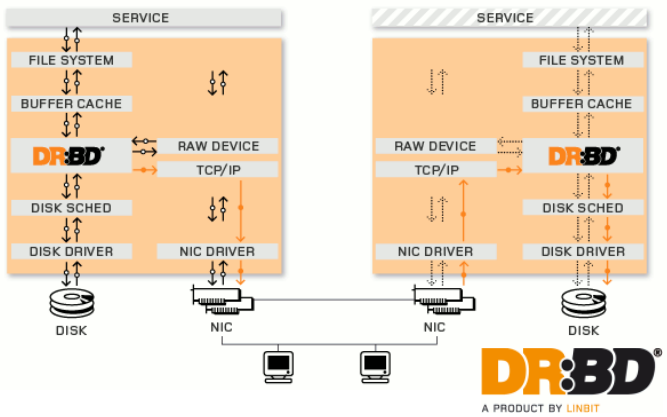
Design: Filesystem

- **Distributed Filesystem**
  - Local copy/cache
  - Parallel



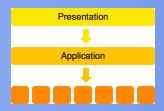
Configuration example of glusterFS with 4 storage nodes and 1 client node

- **Replication**
  - rsync+inotify
  - DRDB



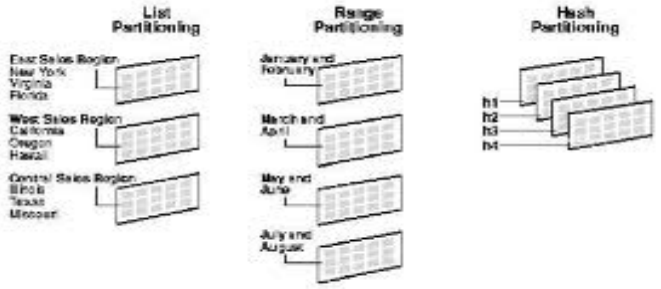
Do not use file system as a
COMMUNICATION PROTOCOL !

Design: Database



Beolink.org

- **Partitioning**
  - Small table
  - Many systems

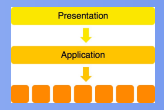


- **Replication**
  - Separation btw Read and Write
  - Different Index on different system

Tungsten Replicator

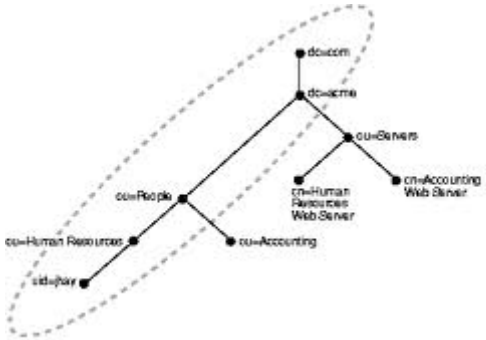
- **NoSQL**
  - Key=value
  - No schema

Design: Hierarchical Storage

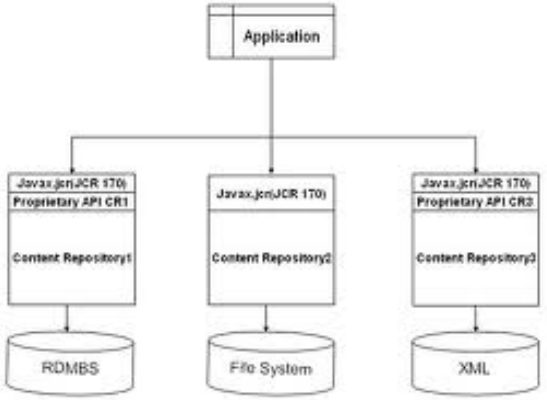


Beolink.org

- **Directory Server**
  - Split in domain
  - Multi Master
  - Right Index
  - Avoid COS



- **JSR**
  - Distribution



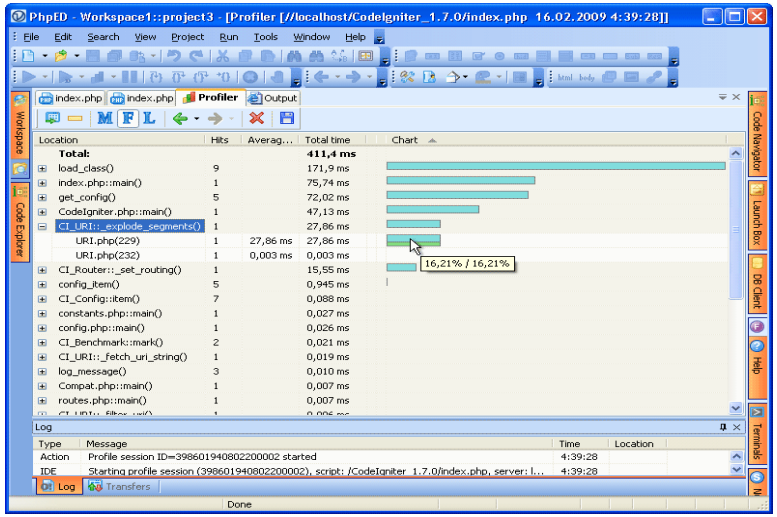
Do not use Directory Servers as a
A STANDARD DATABASE !

Design: Profiling

The 20% of the code is responsible for 80% of the results (time)

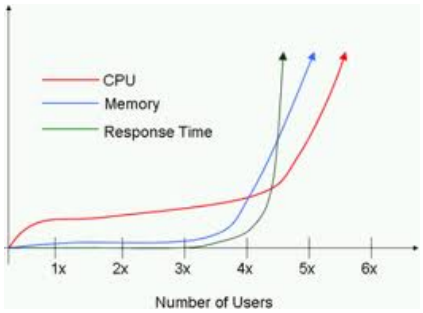
- Find bottleneck before production
- Dynamic program analysis
- Show frequency of called functions
- Show usage of lines in code
- Show duration of function calls

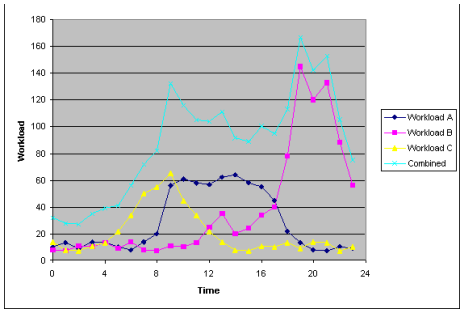
Cmd line: top, htop, vmstat, dstat, strace
Statistical: Oprofile, google profile
Instrumentiing: valgrind's callgrind, gprof

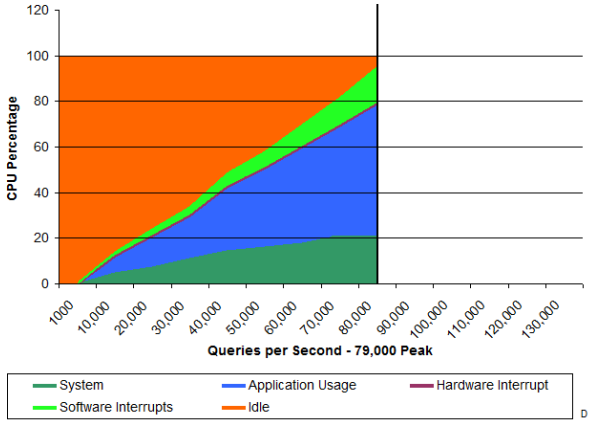


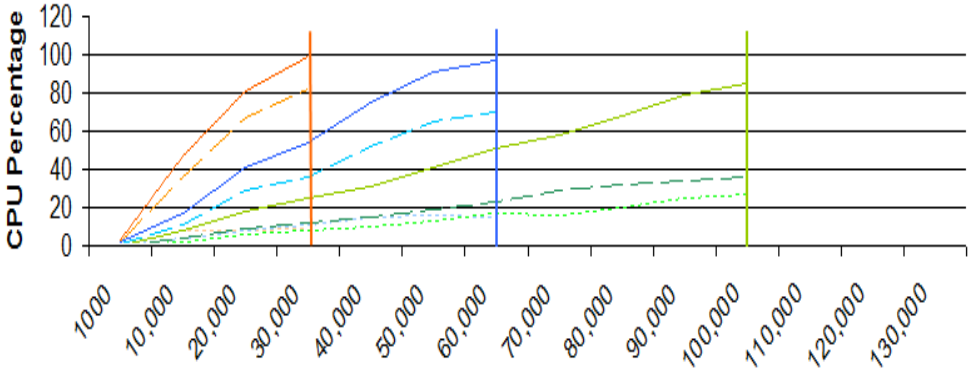
Design: Capacity

- Find relation btw application tasks and resources usage
- Find max Load







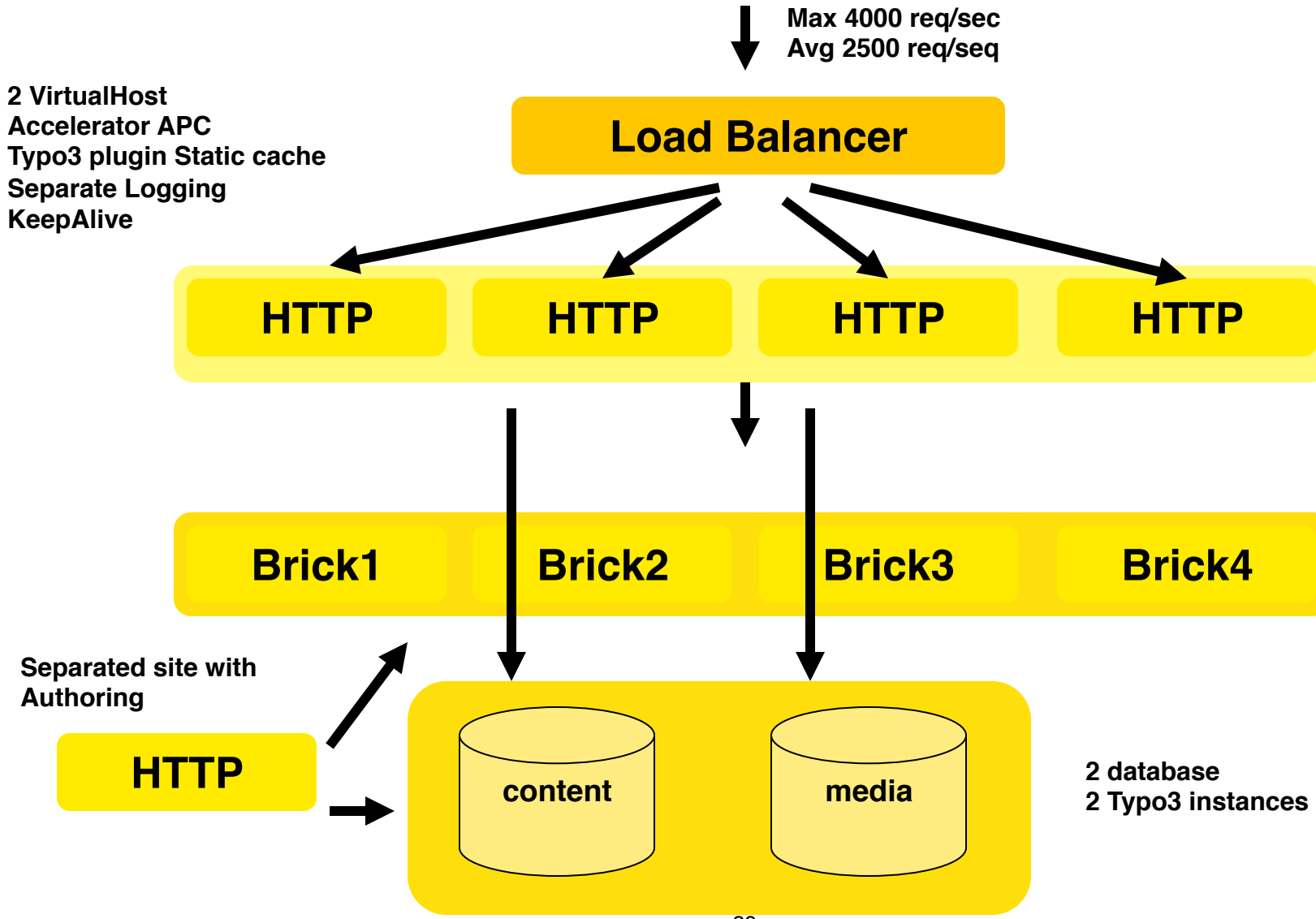


Design: Matrix

Properties	Replication	Load Balancing	…
Network Performance	o		
User-perceived Performance	+		
Network Efficiency	-		
Scalability	++		
Simplicity	-		
Modifiability	o		
Evolvability	o		
Extensibility	o		
Customizability	o		
Configurability	-		
Reusability	+		

Example

Design: CMS


Max 4000 req/sec
Avg 2500 req/seq
2 VirtualHost
Accelerator APC
Typo3 plugin Static cache
Separate Logging
KeepAlive
Load Balancer
HTTP
HTTP
HTTP
HTTP
Brick1
Brick2
Brick3
Brick4
Separated site with
Authoring
HTTP
content
media
2 database
2 Typo3 instances

Example: Something goes wrong ? Beolink.org

load averages: 534.93 281.26 1.26

Something goes wrong ...

```
$loops = 150 $steps = 200
if (is_file($this->resource)) {
        $this->sysLog('Waiting for a different process to release the lock');
        $i = 0;
        while ($i<$this->loops) {
                $i++; usleep($this->step*1000);
                clearstatcache();
                if (!is_file($this->resource)) {
                                // Lock became free, leave the loop
                                $this->sysLog('Different process released the lock');
                                $noWait = false;
                                break;
                }
        $noWait = true;
        }
        if (($this->filepointer = touch($this->resource)) == false) {
                 throw new Exception('Lock file could not be created');
        }
...
```

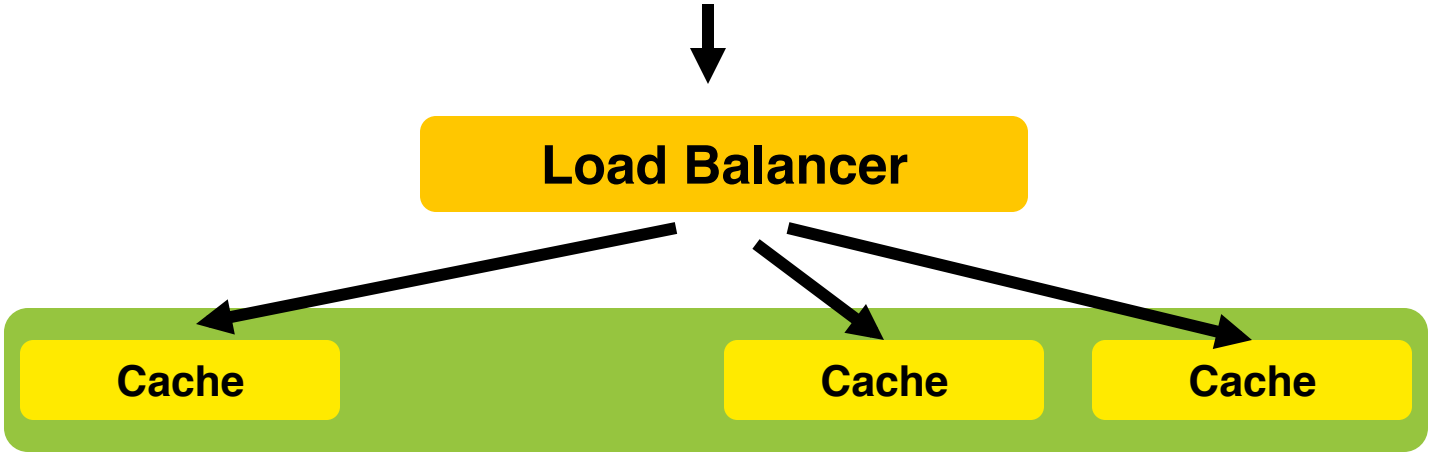
500 req/sec * 30 sec

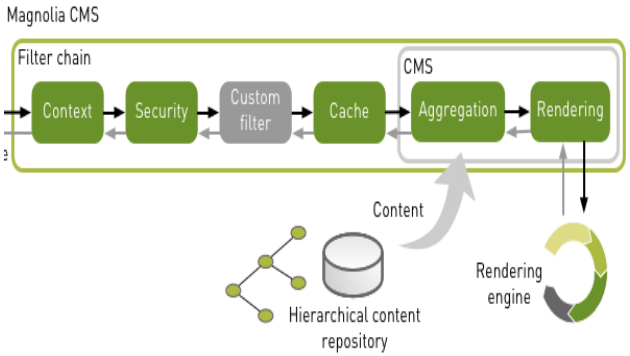
15000 Sockets
15000/req_per_proc Processes
15000 DB connections !!!

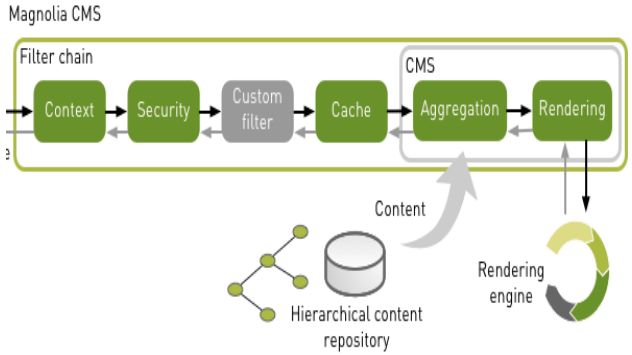
ALL the processes are in READY STATE

The content could be locked forever

Design: CMS



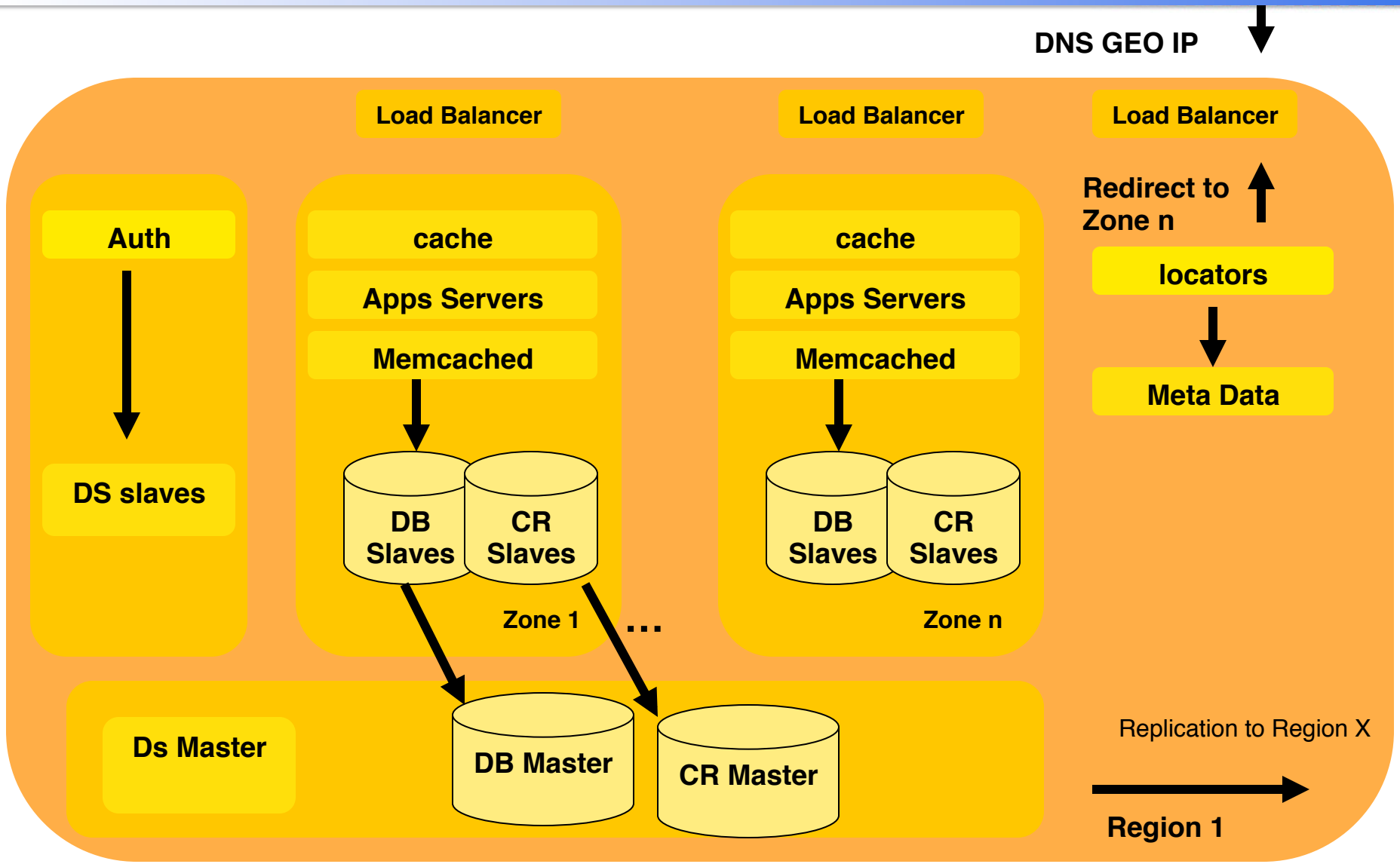




...

Master/Author

Design: Enterprise



Service Operation

- **Internal**
  - System
  - Internal state
  - Traffic
- **External**
  - User experience, time spent for each component (components time loading, rendering, execution,…)
- **Alarm**
  - Define key performance Indicator on System Capacity

System

Requests stats (per second, per IP, …)
Concurrent Users
System load (cpu, memory,disk I/O,..)
Number of Processes

APPS

Memory per instance
Number of elements in Data structure
Communication Timeout

Database

Connection
Query time
Query per Table
Top query

SO: The red button

- **Improve capacity**
  - Remove controls
  - Increase number of systems
  - Dynamic to Static
  - Increase TTL of cache
  - Async Operation
  - Operations Queue

- **Disconnection**
  - Exclusion of Cluster of users
  - Exclusion IP list
  - Bandwidth reduction
  - Web site Sections closure

7 Lessons Learned

(so far)

Partitioning Algorithms !

(application driver)

Kill your Web Designer !

(one shot)

Never repeat !

(caching)

Share nothing !

(local content)

Warm up!

(empty cache)

Asynchronous

(queue)

Measure, Measure, Measure, Measure, Measure, Measure!

Cloud ?

Things must change!

- Web UI for users, affiliates, marketing, operations
- Agile machine management is part of the API
- Scale up -and down
- Live upgrade of running system
- Persistence with key-value stores
- A Petabyte filesystem is part of the application
- MapReduce jobs close the loop
- Developers deploy to the cloud to test

Original work:
http://svn.apache.org/repos/asf/labs/clouds/apache_cloud_computing_edition.pdf

XVIII European AFS meeting 2011

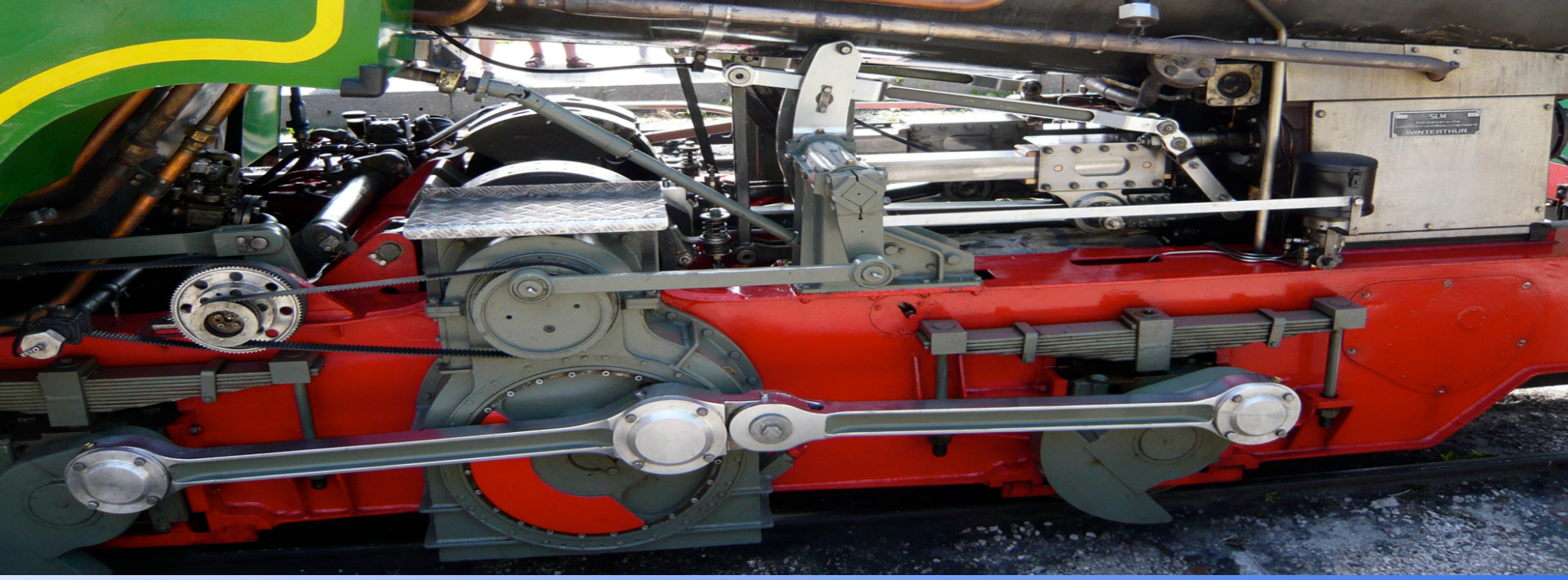
HAMBURG – GERMANY

4-7 October

Who should attend:

- Everyone interested in deploying a globally accessible file system
- Everyone interested in learning more about real world usage of Kerberos authentication in single realm and federated single sign-on environments
- Everyone who wants to share their knowledge and experience with other members of the AFS and Kerberos communities
- Everyone who wants to find out the latest developments affecting AFS and Kerberos

More Info: http://www.openafs.org/



Thank you

manfred@freemails.ch

Beolink.org