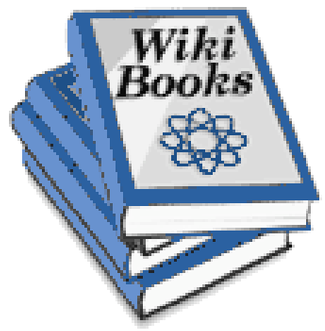
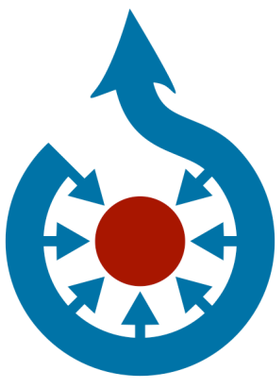







# Wikimedia architecture

Mark Bergsma <mark@wikimedia.org>
Wikimedia Foundation Inc.



# Overview


- Intro
- Our technical operations
- Global architecture
- Content Delivery Network (CDN)
- Application servers
- Persistent storage

Mark Bergsma, mark@wikimedia.org, Wikimedia Foundation Inc.

# Some figures

- The Wikimedia Foundation:
  - Was founded in June 2003, in Florida
  - Currently has 10 employees, the rest is done by volunteers
  - Yearly budget of around $1M - $2M, supported mostly through donations
  - Supports the popular Wikipedia project, but also 8 others: Wiktionary, Wikinews, Wikibooks, Wikiquote, Wikisource, Wikiversity, Wikispecies, Commons

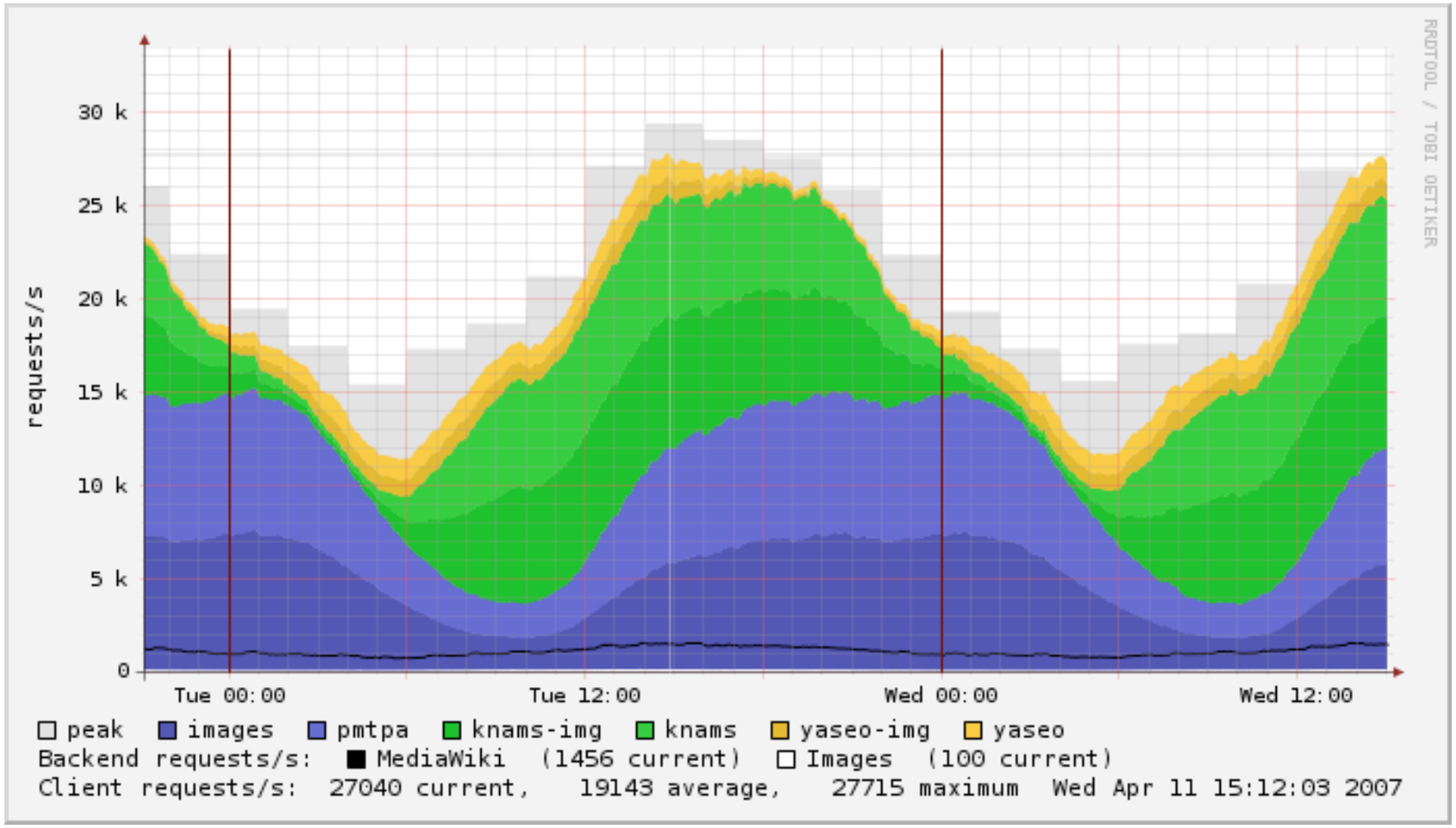
# Some figures

- Wikipedia:
  - 8 million articles spread over hundreds of language projects (english, dutch, ...)
  - 110 million revisions
  - 10th busiest site in the world (source: Alexa)
  - Exponential growth: doubling every 4-6 months in terms of visitors / traffic / servers

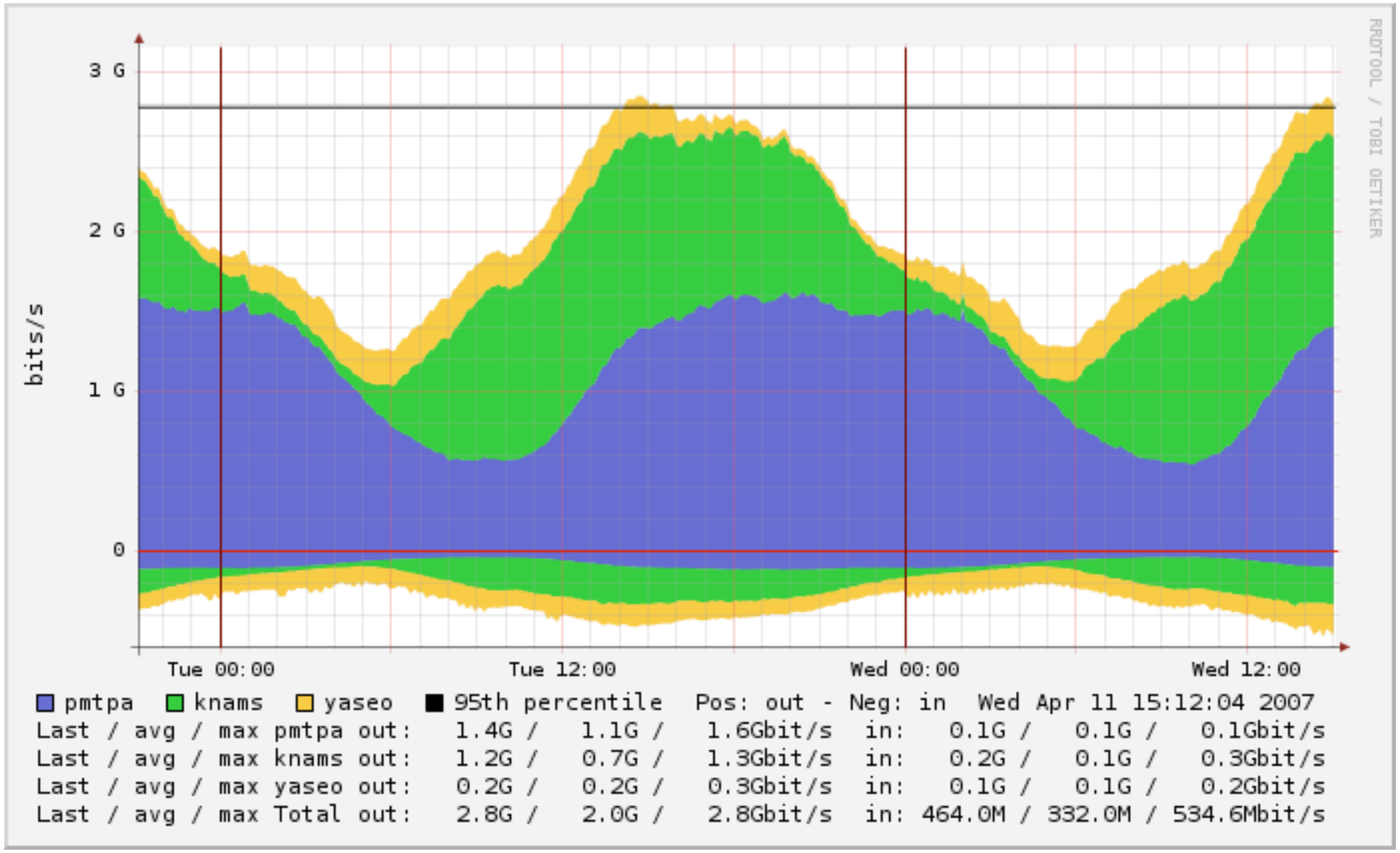
# Some technical figures

- 33 000 HTTP requests/s during peak-time
- 3.5 Gbit/s of data traffic
- 3 data centers: Tampa, Amsterdam, Seoul
- 350 servers, ranging between 1x P4 to 2x Xeon Quad-Core, 0.5 - 16 GB of memory
- 15 racks, 150 kW power capacity
- ...managed by ~ 6 people

# Pretty graphs!



# Pretty graphs!



# Our operations

- Very ad-hoc, "fire fighting mode"
  - Less so now more staff is on the pay roll, and growth is slowing down
- Technical staff spread out over the world, in the US, Europe and Australia
  - Always someone awake!
  - ...but no explicitly planned shifts

# Our operations

- Communication on IRC
  - Sometimes by mail, mailing lists or phones / VoIP
  - ...and occasionally real-life meetings (Wikimania Hacking Days, etc.)
- Documentation in a wiki
- Decisions mostly by consensus
  - Small group that knows each other very well and can get things done very efficiently

# Our operations

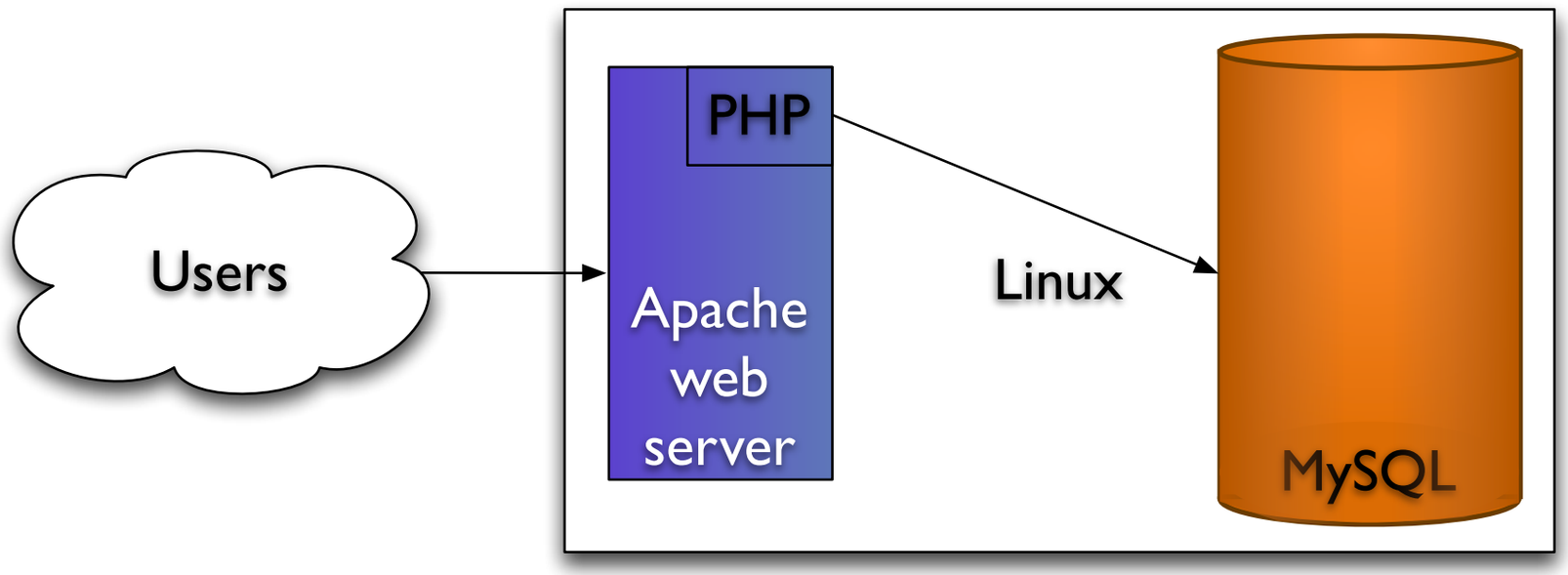
- Growth is slowing down
  - We're getting ahead of growth in terms of capacity
  - Finally time to solve structural problems, improve reliability and ease of maintenance
- Alternating periods of growth surges and quietness are nicest!
  - Keeps us sharp & inventive, and the job interesting, but gives some time to breath

# Example of a structural improvement

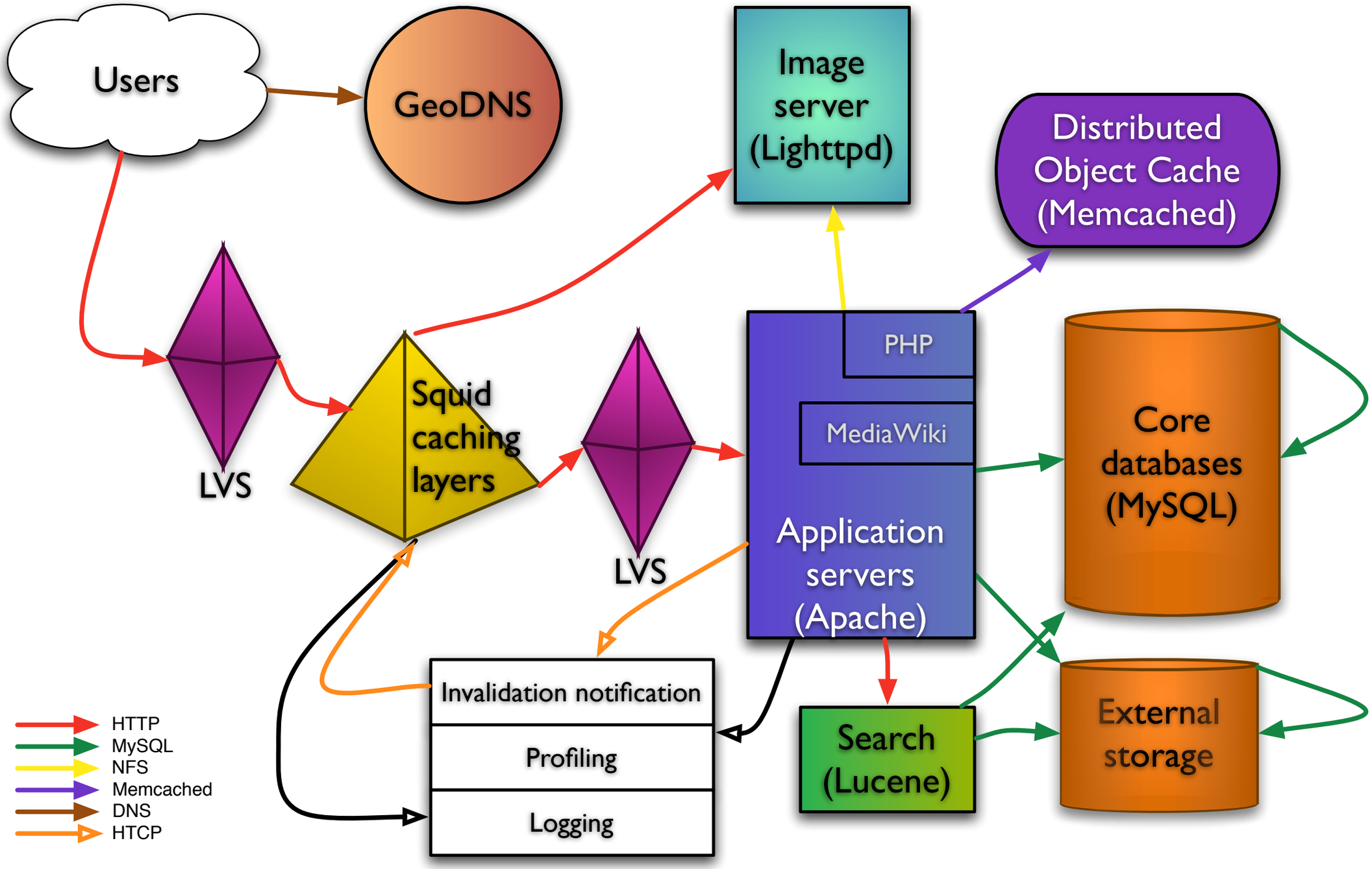
- Automated installations
  - PXE net boot
  - Ubuntu installer *preseeding*
  - Wikimedia APT repository with custom packages
- Installation of a single Squid server:

```
# apt-get install wikimedia-task-squid
```

# Architecture: LAMP...



# ...on steroids.




Mark Bergsma, mark@wikimedia.org, Wikimedia Foundation Inc.
Wikimedia architecture

# Content Distribution Network (CDN)

- 3 clusters on 3 different continents:
  - Primary cluster in Tampa, Florida
  - Secondary caching-only clusters in Amsterdam, the Netherlands and Seoul, South Korea
- Geographic Load Balancing, based on source IP of client resolver, directs clients to the nearest server cluster
  - Works by statically mapping IP addresses to countries to clusters

# Geographic Load Balancing

- Observations:
  - *Most* users use a DNS resolver relatively close to them
  - IP address to location mapping lists are available that are at least 90-95% accurate
  - Geographically close often also means close *network topology* wise
- Solution: hand out DNS answers based on the estimated location of the querying DNS resolvers

# Geographic Load Balancing

- 4 years ago, I wrote *Geobackend* for PowerDNS (for use by an IRC network)
- Geobackend reads a RBLDNS style zonefile into an efficient in-memory trie as IP map to numbers (countries)
- Uses a flat file *director map* to map numbers (countries) to DNS CNAMES
- Works pretty well for a coursely distributed set of clusters

Mark Bergsma, mark@wikimedia.org, Wikimedia Foundation Inc.
Wikimedia architecture

# Squid caching

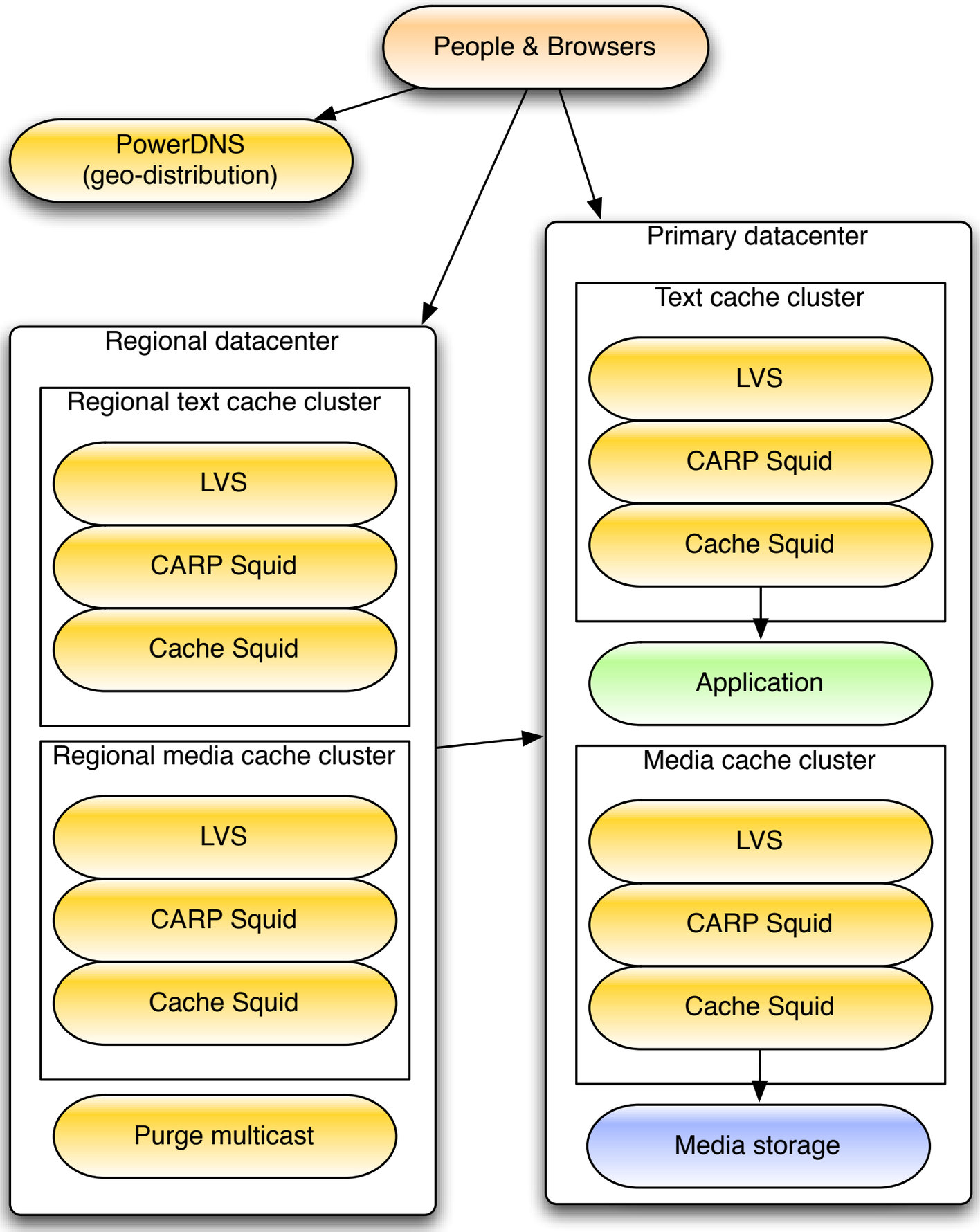
- HTTP reverse proxy caching implemented using Squid
- Split into two groups with different characteristics
  - ‘Text’ for wiki content (mostly compressed HTML pages)
  - ‘Media’ for images and other forms of relatively large static files

# Squid caching

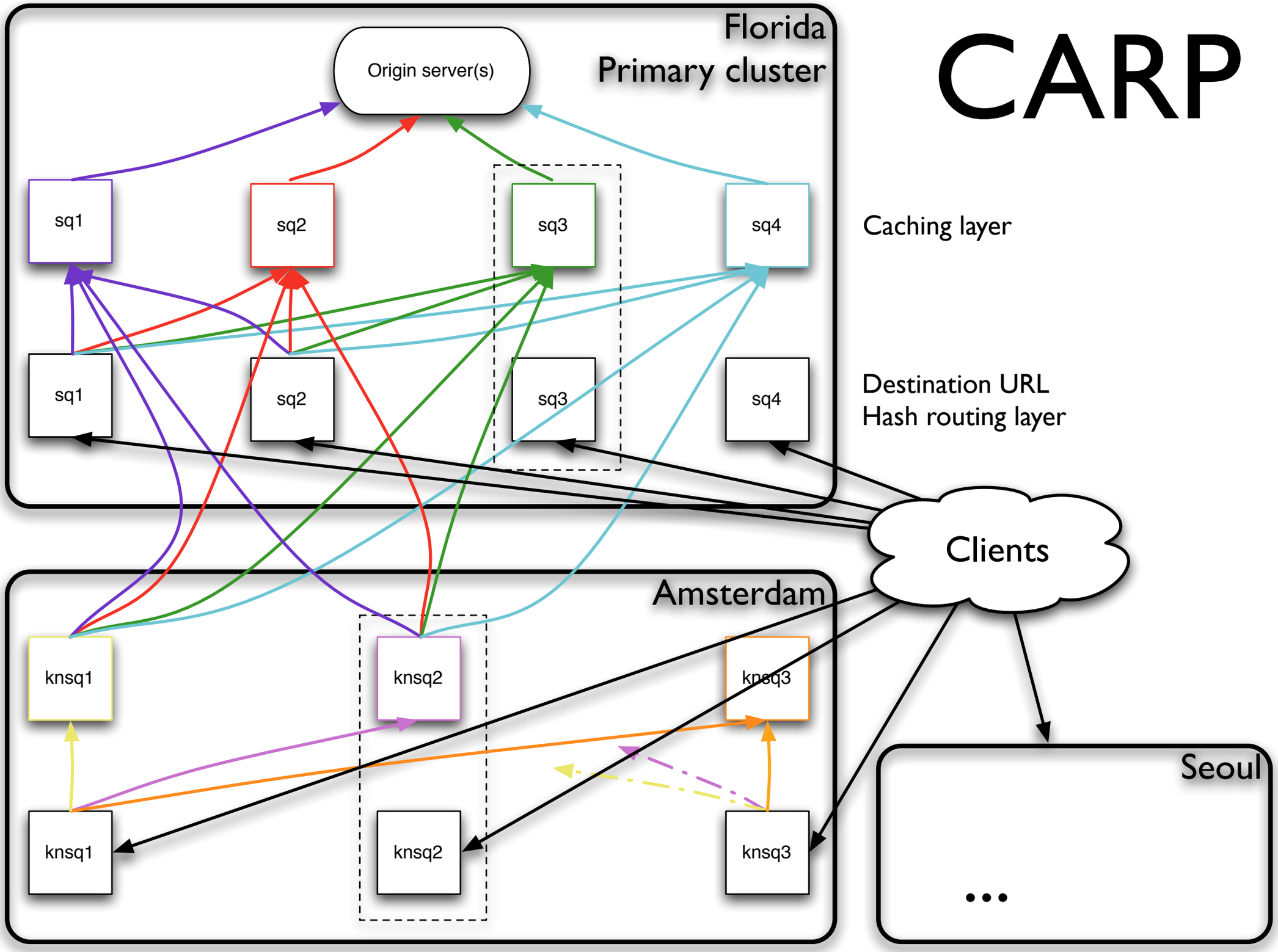
- 75 Squid servers currently, with per server:
  - ~ 1 000 HTTP requests/s, up to 2 500 under stress
  - ~ 100 - 250 Mbit/s
  - ~ 14 000 - 32 000 open connections

# Squid caching

- Up to 40 GB of disk caches per Squid server
- Disk seek I/O limited
  - The more disk spindles, the better!
  - Up to 4 disks per server (1U rack servers)
- 8 GB of memory, half of that used by Squid
- Hit rates: 85% for Text, 98% for Media, since the use of CARP


People & Browsers
PowerDNS
(geo-distribution)
Regional datacenter
Regional text cache cluster
LVS
CARP Squid
Cache Squid
Regional media cache cluster
LVS
CARP Squid
Cache Squid
Purge multicast
Primary datacenter
Text cache cluster
LVS
CARP Squid
Cache Squid
Application
Media cache cluster
LVS
CARP Squid
Cache Squid
Media storage

# CARP




Mark Bergsma, mark@wikimedia.org, Wikimedia Foundation Inc.
Wikimedia architecture

# Squid cache invalidation

- Wiki pages are edited at an unpredictable rate
- Only the latest revision of a page should be served at all times in order to not hinder collaboration
- Invalidation through expiry times not acceptable, explicit cache purging needs to be done
- Implemented using the UDP based HTCP protocol: on edit application servers send out a single message containing the URL to be invalidated, which is delivered over multicast to all subscribed Squid caches

Mark Bergsma, mark@wikimedia.org, Wikimedia Foundation Inc.
Wikimedia architecture

# The Wiki software


- All Wikimedia projects run on a MediaWiki platform
- Open Source software (GPL)
- Designed primarily for use by Wikipedia/Wikimedia, but also used by many outside parties
  - Arguably the most popular wiki engine out there
- Written in PHP
- Storage primarily in MySQL, other DBMSes supported
- Very scalable, very good localization

Mark Bergsma, mark@wikimedia.org, Wikimedia Foundation Inc.

# MediaWiki

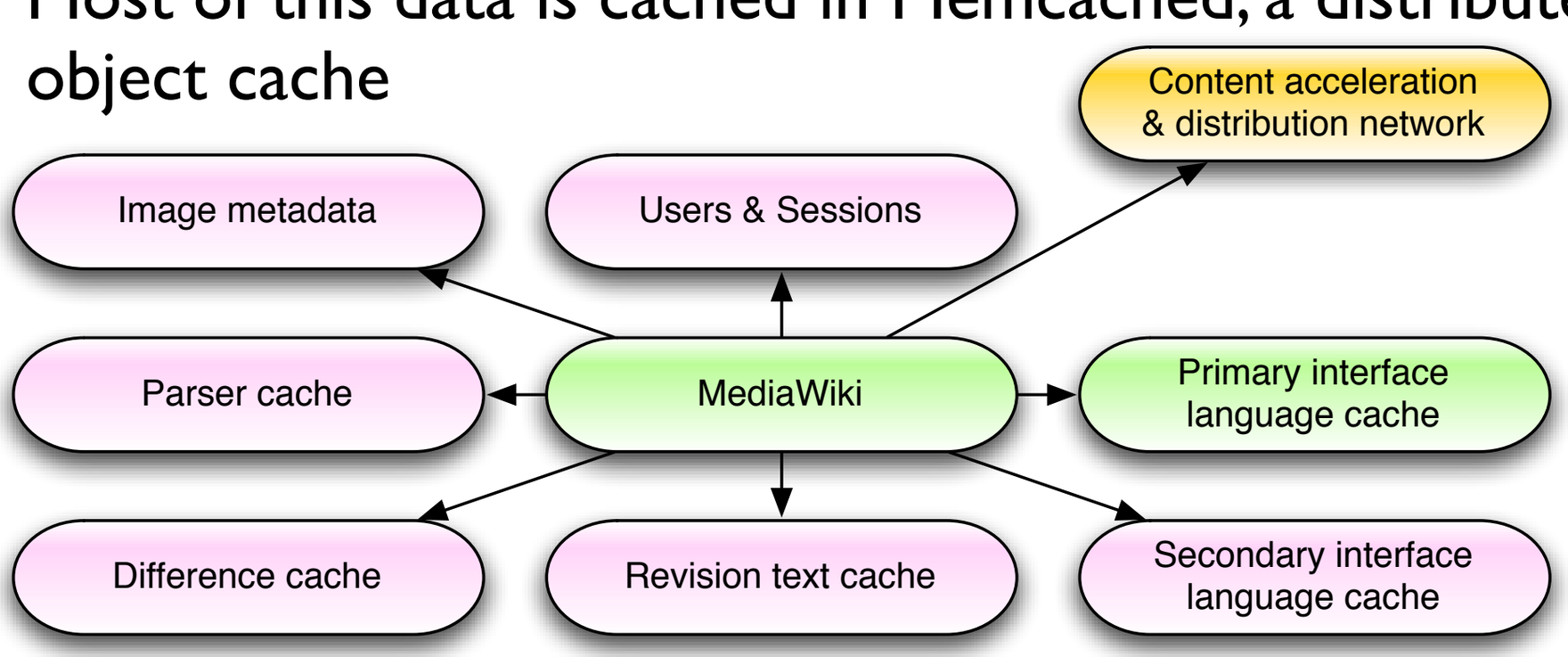
- MediaWiki in our application server platform:
  - ~ 125 servers, 40 waiting to be setup
  - MediaWiki scales well with multiple CPUs, so we buy dual quad-core servers now (8 CPU cores per box)
  - One centrally managed & synchronized software installation for hundreds of wikis
  - Hardware shared with External Storage and Memcached tasks

# MediaWiki caching

Tell about Memcached, no dedicated slide

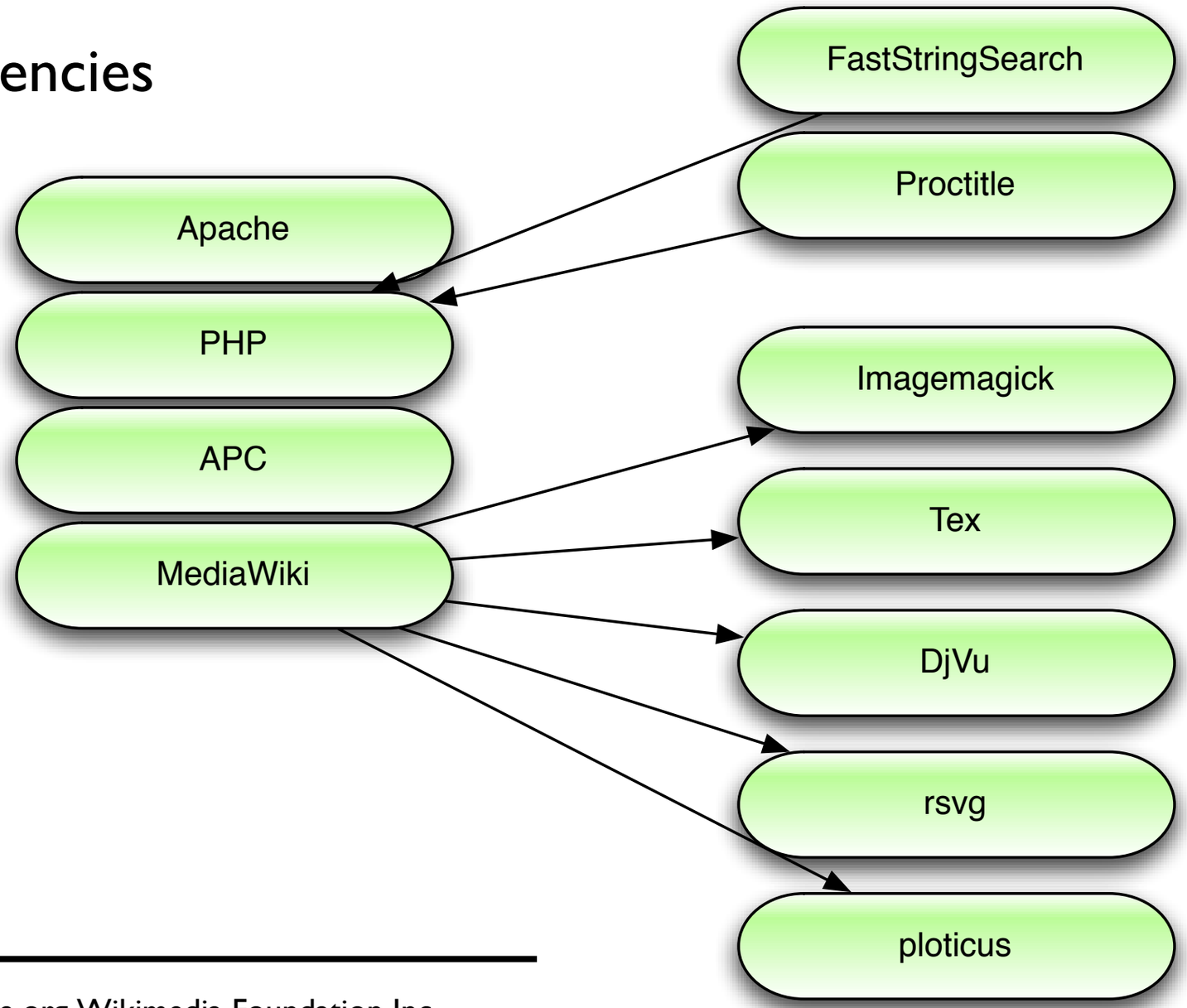
Sometimes caching costs more than recalculating or looking up at the data source... profiling!

- Caches everywhere
- Most of this data is cached in Memcached, a distributed object cache



# MediaWiki dependencies

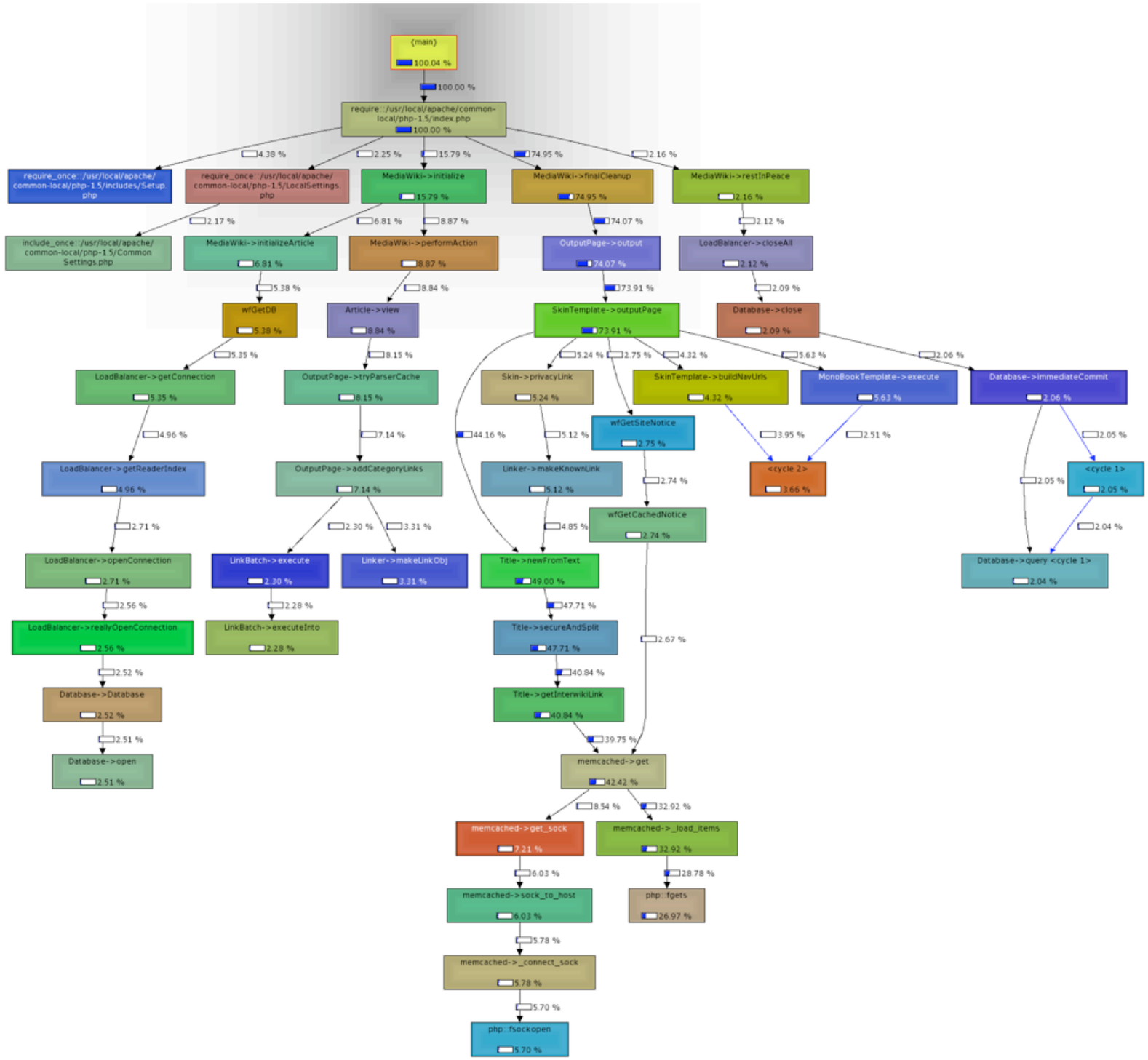
- A lot of dependencies in our setup



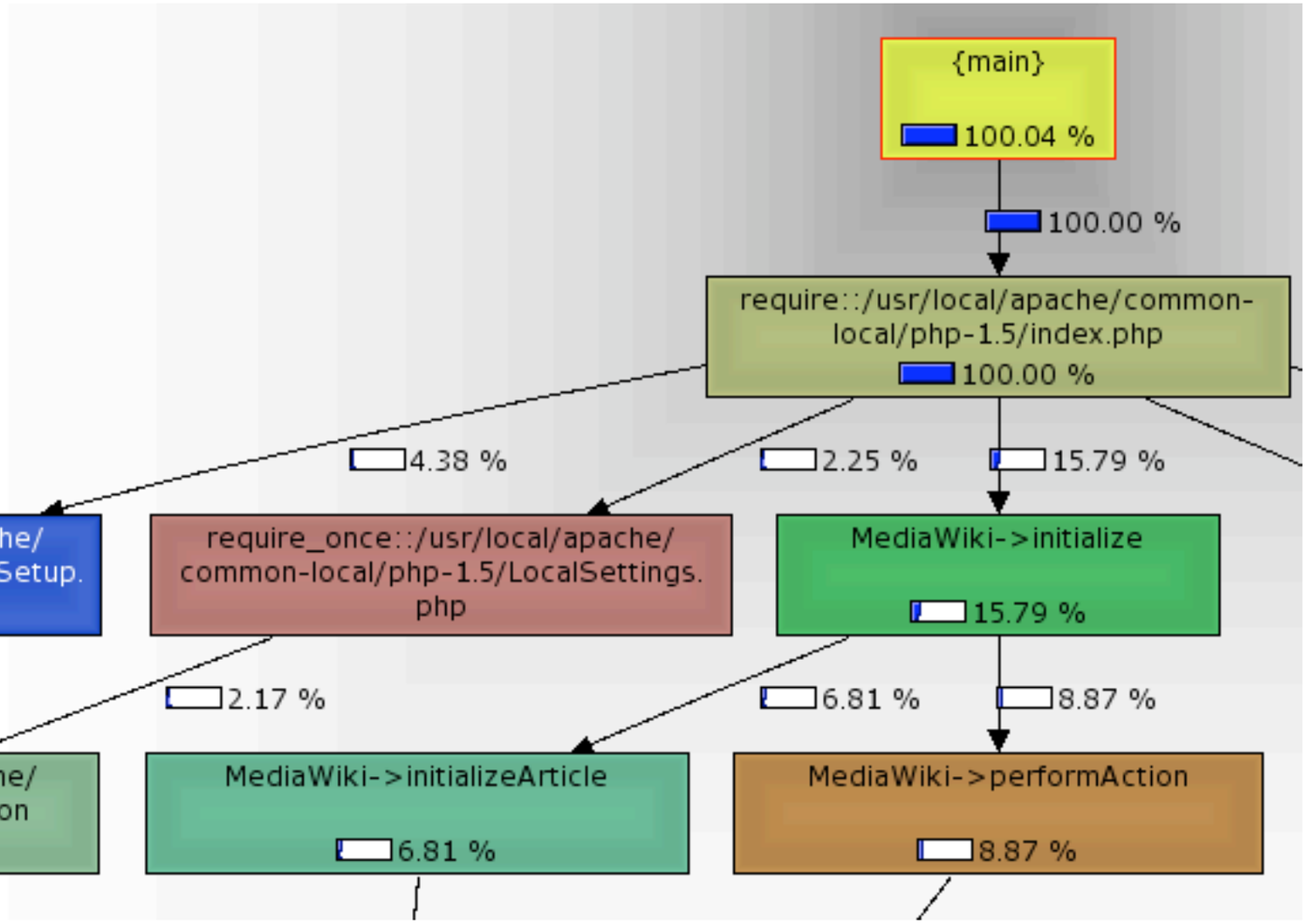
# MediaWiki optimization

- We try to optimize by...
  - not doing anything stupid
  - avoiding expensive algorithms, database queries, etc.
  - caching every result that is expensive and has temporal locality of reference
  - focusing on the hot spots in the code (profiling!)
- If a MediaWiki feature is too expensive, it doesn't get enabled on Wikipedia

# http://noc.wikimedia.org/cgi-bin/report.py



# MediaWiki profiling




Mark Bergsma, mark@wikimedia.org, Wikimedia Foundation Inc.
Wikimedia architecture

# Persistent data

- Persistent data is stored in the following ways:
  - Metadata, such as article revision history, article relations (links, categories etc.), user accounts and settings are stored in the *core databases*
  - Actual revision *text* is stored as blobs in *External storage*
  - Static (uploaded) files, such as images, are stored separately on the image server - metadata (size, type, etc.) is cached in the core database and object caches

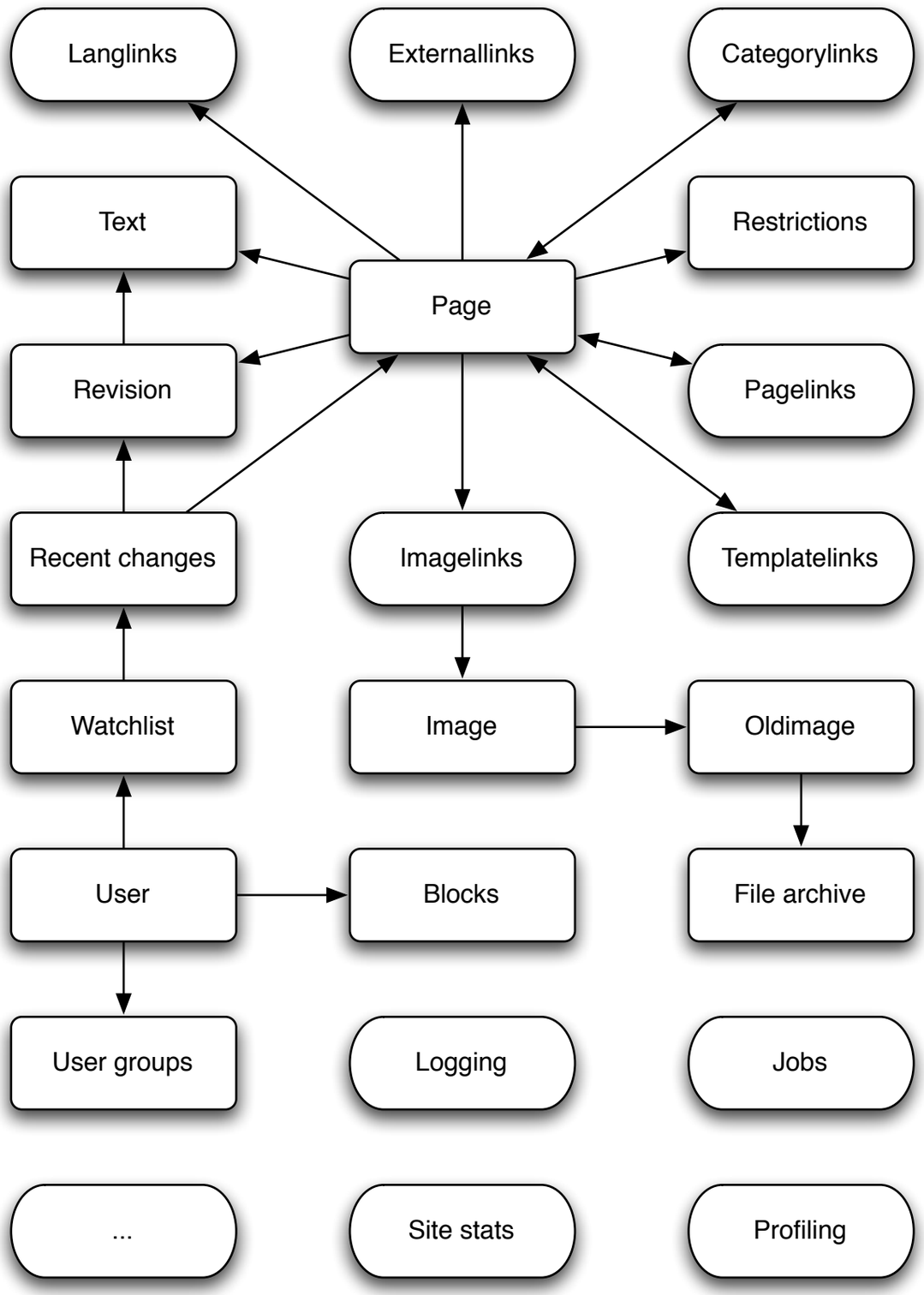
# Core databases

- Separate database per wiki (*not* separate server!)
- One master, many replicated slaves
- Read operations are load balanced over the slaves, write operations go to the master
    - The master is used for some read operations in case the slaves are not yet up to date (*lagged*)
- Runs on ~15 DB servers with 4 - 16 GB of memory, 6x 73 - 146 GB disks and 2 CPUs each

# Core database scaling

- Scaling by:
  - Separating read and write operations (master/slave)
  - Separating some expensive operations from cheap and more frequent operations (query groups)
  - Separating big, popular wikis from smaller wikis
- Improves caching: temporal and spatial locality of reference and reduces the data set size per server

# Core database schema



# External Storage

- Article text is stored on separate data storage clusters
  - Simple append-only *blob storage*
- Saves space on expensive and busy core databases for largely unused data
- Allows use of spare resources on application servers (2x 250-500 GB per server)
- Currently replicated clusters of 3 MySQL hosts are used; this might change in the future for better manageability

# Text compression

- All revisions of all articles are stored
  - Every Wikipedia article version since day 1 is available
- Many articles have hundreds, thousands or tens of thousands of revisions
- Most revisions differ only slightly from previous revisions
- Therefore subsequent revisions of an article are *concatenated* and then compressed
  - Achieving very high compression ratios of up to 100x

# Media storage

- Currently:
  - 1 Image server containing 1.3 TB of data, 4 million files
  - Overloaded serving just 100 requests/s
  - Media uploads and deletions over NFS, mounted on all application servers
  - Not scalable
  - Backups take ~ 2 weeks

# Media storage

- New API between media storage server and application servers, based on HTTP
  - Methods *store*, *publish*, *delete* and *generate thumbnail*
- New file / directory layout structure, using *content hashes* for file names
  - Files with the same name/URL will have the same content, no invalidation necessary
- Migration to some distributed, replicated setup

# Thumbnail generation

- `stat()` on each request is too expensive, so assume every file exists
- If a thumbnail doesn't exist, ask the application servers to render it