Caching

Ivan Pepelnjak (@ioshints, ip@ipSpace.net)
NIL Data Communications

Generic Principles

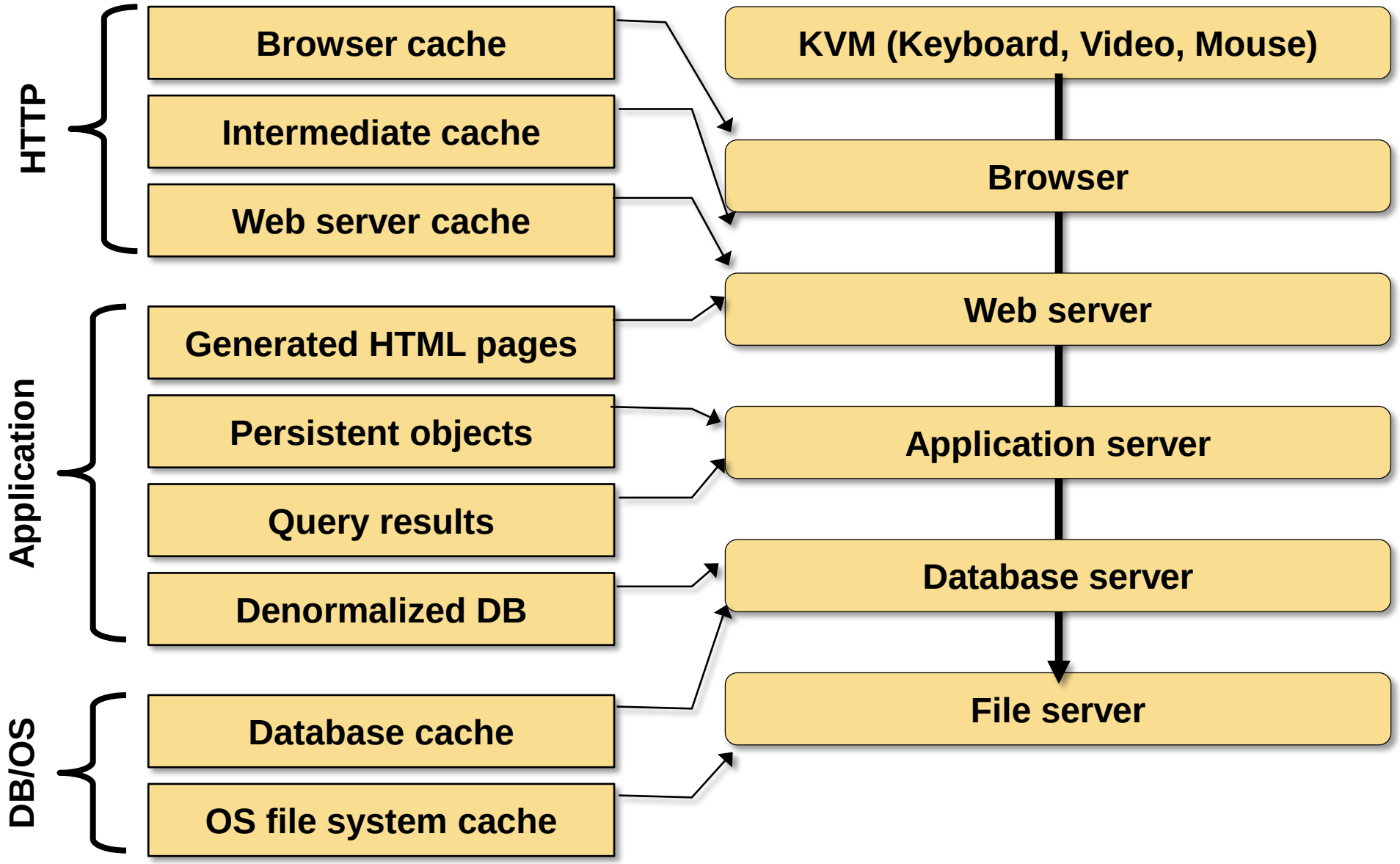
What is caching?

- Storing data closer to consumers
- Reduces bandwidth and latency
- Also: storing cooked data

Where can you cache?

- Anywhere in the application stack
- Wherever it makes most sense

Application Architecture Stack



Generic Cache Issues

- Expiration, expiration with revalidation, or invalidation
- Object lifetimes
- Approximate or eventual correctness

Other trick:

- Know how to fake data (counters)
- Use AJAX to update writer's pages
- Delay updating readers' pages

HTTP Caching

HTTP Caching

Caching model:

- Multi-level cache hierarchy
- Shared and private caches

Expiration model:

- Server-specified expiration (**Expires** header or **max-age** directive in **Cache-Control** header)
- Heuristic expiration (use **Date, Age** and **Last-Modified** headers)

Validation model:

- Needs **Last-Modified** and/or **Etag** headers
- Strong and weak **Etag** validators

HTTP Caching Headers

Use Cache-Control header only!

- Applies to requests and responses
- What is cacheable: **public**, **private**, **no-cache** (optionally with fields) and **no-store**
- Expiration: **max-age** and **s-maxage**
- Revalidation policy: **must-revalidate** or **proxy-revalidate**

HTTP Caching Headers

Validation

- Send request with **If-Match** and/or **If-Modified-Since**
- Response: new copy of the data or 304 status code
- Warning: HTTP uses weird date format

Revalidation

- Send request with cache-validating conditional and **max-age=0**

Reloading

- Send request with **Cache-Control: no-cache, max-age=0**

Example: No Caching

```
GET / HTTP/1.1
Host: www.google.si
Connection: keep-alive
```

```
HTTP/1.1 200 OK
Cache-Control: private, max-age=0
Content-Encoding: gzip
Content-Type: text/html; charset=UTF-8
Date: Mon, 18 Feb 2013 08:43:55 GMT
Expires: -1
```

Response will not be stored in public or private caches

Example: Explicit Caching

```
GET /images/icons/product/chrome-48.png HTTP/1.1
Host: www.google.si
Connection: keep-alive
```

```
HTTP/1.1 200 OK
Age: 18260
Cache-Control: public, max-age=691200
Content-Length: 1834
Content-Type: image/png
Date: Mon, 18 Feb 2013 03:39:35 GMT
Expires: Tue, 26 Feb 2013 03:39:35 GMT
Last-Modified: Mon, 02 Apr 2012 02:13:37 GMT
```

Response is cached, not checked until it expires

Example: Conditional GET After Cache Expiration

```
GET /gb/images/k1_a31af7ac.png HTTP/1.1
Host: ssl.gstatic.com
Connection: keep-alive
If-Modified-Since: Thu, 22 Nov 2012 05:50:39 GMT
```

```
HTTP/1.1 304 Not Modified
Age: 309278
Date: Thu, 14 Feb 2013 18:49:17 GMT
Expires: Fri, 22 Feb 2013 18:49:17 GMT
```

Request sent after local cache expires, response extends expiration date

Example: Request Fresh Content

```
GET /safebrowsing/rd/… HTTP/1.1
Host: safebrowsing-cache.google.com
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
```

```
HTTP/1.1 200 OK
Content-Type: application/vnd.google.safebrowsing-chunk
Date: Mon, 18 Feb 2013 07:43:25 GMT
Content-Length: 7164
Cache-Control: public,max-age=172800
Age: 3946
```

Fresh content requested with *Cache-Control: no-cache*

Example: Etag Used as File Checksum

```
GET /flowers/Liliaceae/…JPG HTTP/1.1
Host: www.zaplana.net
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

```
HTTP/1.1 200 OK
Content-Length: 7174
Content-Type: image/jpeg
Last-Modified: Sat, 13 Jul 2002 19:24:02 GMT
Etag: "36bb38d8a22ac21:dbed2"
Server: Microsoft-IIS/6.0
```

File checksum in Etag, no explicit expiration date

Estimating Expiration Times

Compute content age:

- Initial: *Age* header or *now* - *Date* header
- Add time since last validation

Compute freshness:

- *Max-Age* header
- *Expires* - *Date* headers
- 10% of *(now – Last-Modified* header) – recommendation implementations are browser-dependent
- Some caches might store responses with no *Last-Modified* header

Revalidate if freshness < age

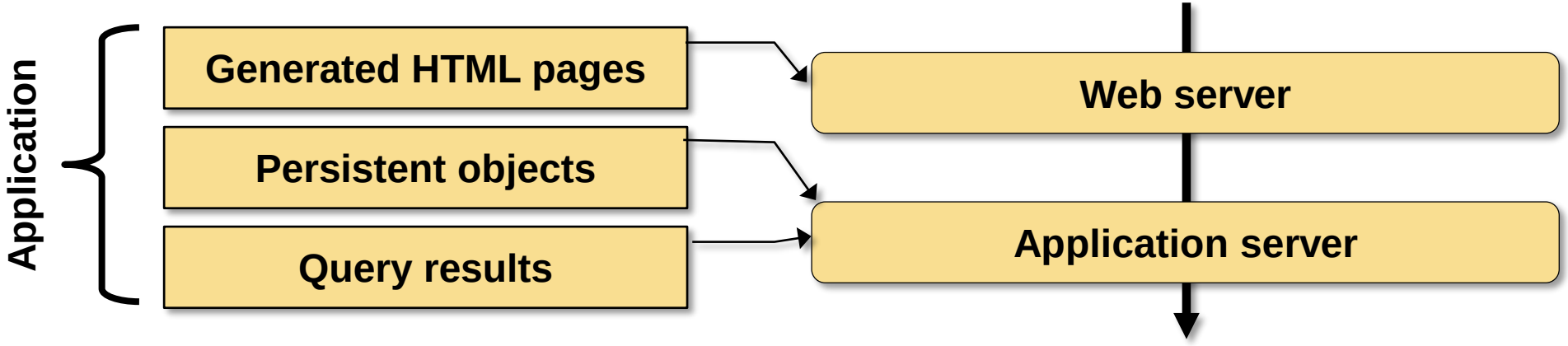
Example: Reload with Etag

```
GET /flowers/Liliaceae/…JPG HTTP/1.1
Host: www.zaplana.net
Accept: image/png,image/*;q=0.8,*/*;q=0.5
If-Modified-Since: Sat, 13 Jul 2002 19:24:02 GMT
If-None-Match: "36bb38d8a22ac21:dbed2”
Cache-Control: max-age=0
```

```
HTTP/1.1 304 Not Modified
Last-Modified: Sat, 13 Jul 2002 19:24:02 GMT
Accept-Ranges: bytes
Etag: "36bb38d8a22ac21:dbed2"
```

Application Caching

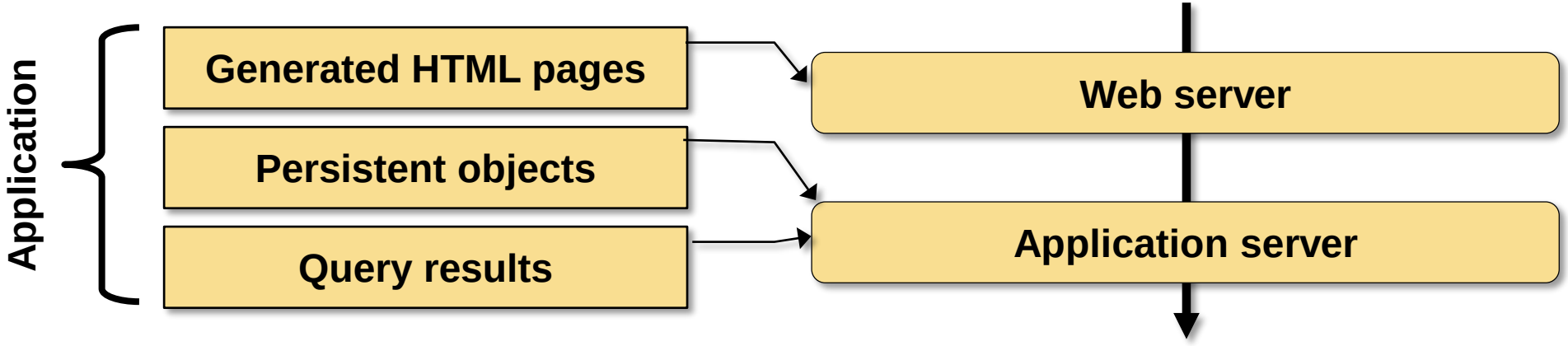
Application Caching



Use the simplest possible caching store:

- Disk files (Apache mod_cache and mod_disk) or reverse proxy (Squid, Varnish)
- APC object store
- Distributed in-memory cache (memcached)
- Non-transactional DB (MongoDB)
- Regular DB (long-lived objects only)

Application Caching: Invalidation or Expiration?



- Expiration is simpler, invalidation is more effective
- Database triggers for low-volume transactions
- Keystore and application-side updates for high-volume
- Use expiration if you don’t trust your programmers
- Err on the side of consistency (purge cache before commit)

Case Studies

Static Content

Challenge

- Minimize the number of HTTP requests

More details

- Download static content only once ➔ HOW?
- Use intermediate caches ➔ HOW?
- Invalidate when content changes ➔ HOW?

Static Content

Solution

- Very large expiration times specified in web server configuration
- Cache-Control set to *public*
- Use CDN
- Use hosted JavaScript libraries (ex: Google Code)

Script-generated content (ex: thumbnails)

- Make resources static ➔ use unique URL per resource
- Set *Expires* and *Cache-Control* headers
- Use local web server cache or reverse proxy

Refreshing Static Content

Invalidate static content by changing the URL

Options

- http://host/path/to/content.css?rev=x
  Might not work with all caches
- http://host/path/to/content.css/revnum
  Might not work with all servers

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" type="text/css"
    href="/css/wiki20.css?207" />
...
```

Dynamic Content – Mostly Reads

Environment

- Dynamic script-generated content
- Many reads, few writes

Examples

- Blogs, wikis, content management systems
- Pages with comments/feedback

Challenge

- Minimize server load

Dynamic Content – Reasonable Consistency

Solution

- Scripts set reasonable max-age
- Last-Modified set to current date/time
- Expires could be used instead of max-age

Caching improvements

- Use HTTP status code 303 on POSTs (invalidates document in **Location** header)
- Use caching in web server or proxy server ➔
  Client requests are not hitting server-side scripts

Dynamic Content – Better Consistency

Caching in reverse proxy

- Scripts set very long expiration times
- Local cache adds reasonably low max-age directive
- Content cached by clients/intermediate caches expires before content in local cache
- HTTP status code 303 on POSTs invalidates document in local cache
- Option: prevent caching in intermediate caches (set *Cache-Control: private*)

Dynamic Content – Logged-In Users

Challenge: Display different content to logged-in users

Option A:

- Use reasonably low max-age
- Use session-ID in URL

Option B:

- Use low max-age, **must-revalidate** and **Etag**
- Server scripts compare session cookie with **If-Match**

Always serve user-specific data with Cache-Control: private

Dynamic Content – Few Updates, Hard to Generate

Option A: cache HTML

- Must invalidate cache on every design change

Option B: cache cooked data

Generic design decisions:

- Caching or storing?
- Memory or DB?
- Invalidation policy
- Handling race conditions

Dynamic Content – Updated Stream/Timeline

- Cache the baseline page
- Generate stream content with AJAX
- Periodically update the cached image of baseline page

Questions?