Flickr : Web Services

Cal Henderson

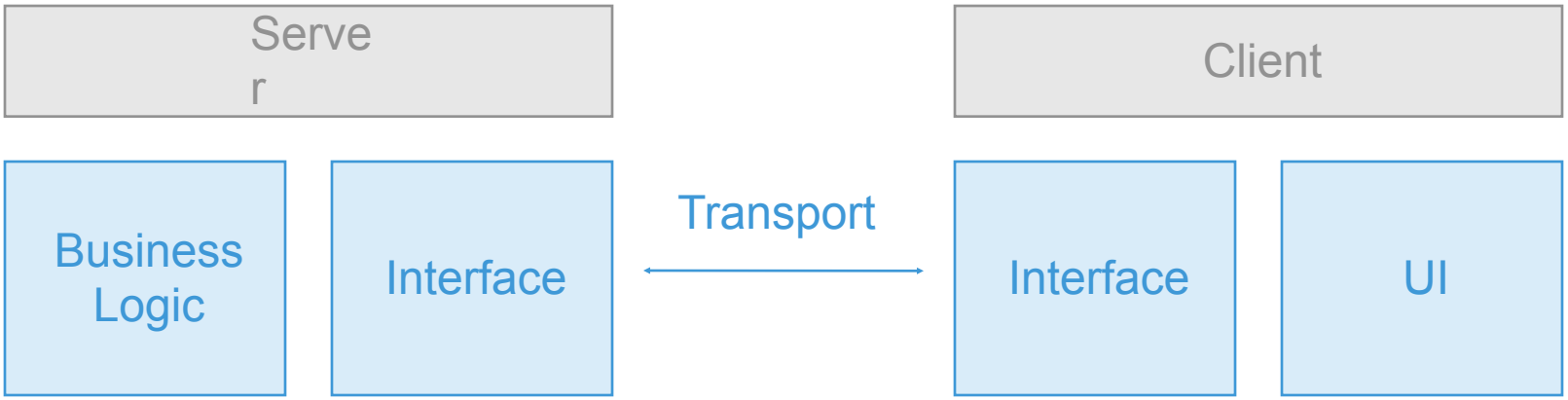
What is Flickr?

- Photo sharing website (flickr.com)
- *The* place to store digital photos
- The centre of a big distributed system
- A set of open APIs

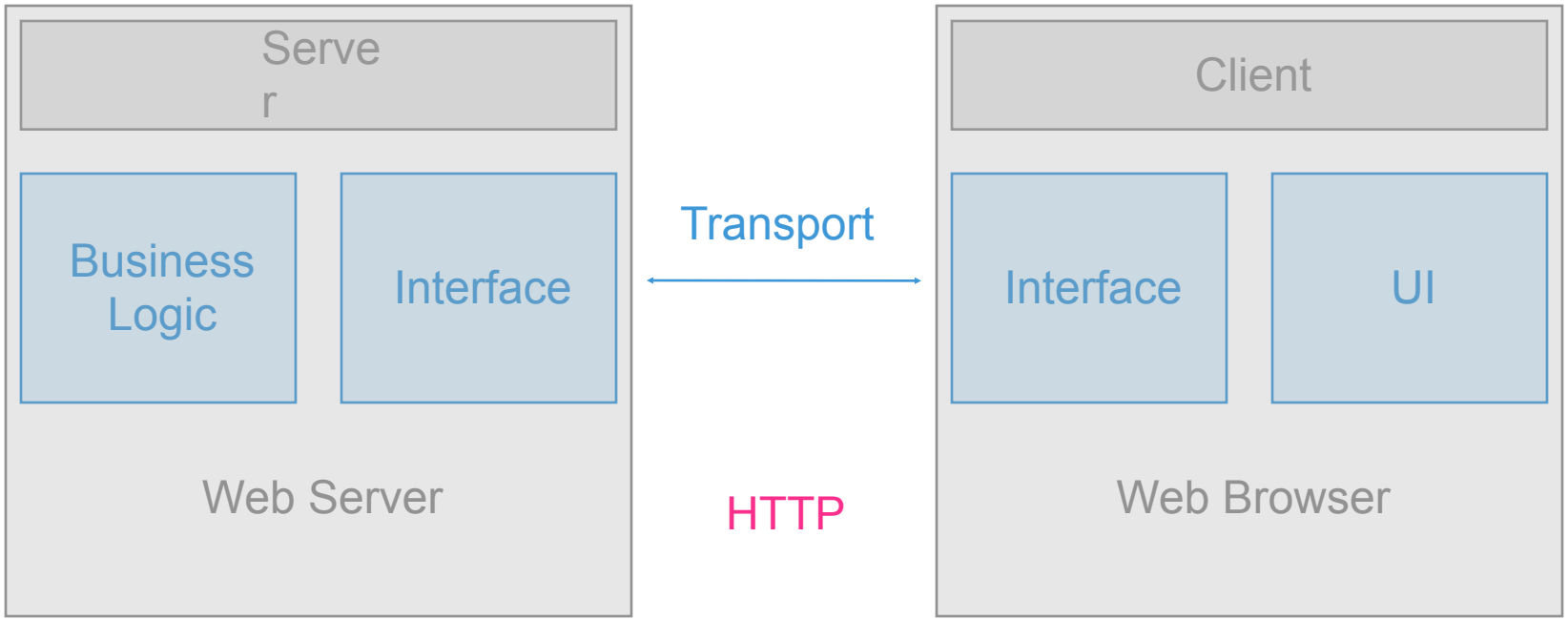
What the heck are ‘Web Services’?

- The future of the Internet!!!1
- Really just buzzwords

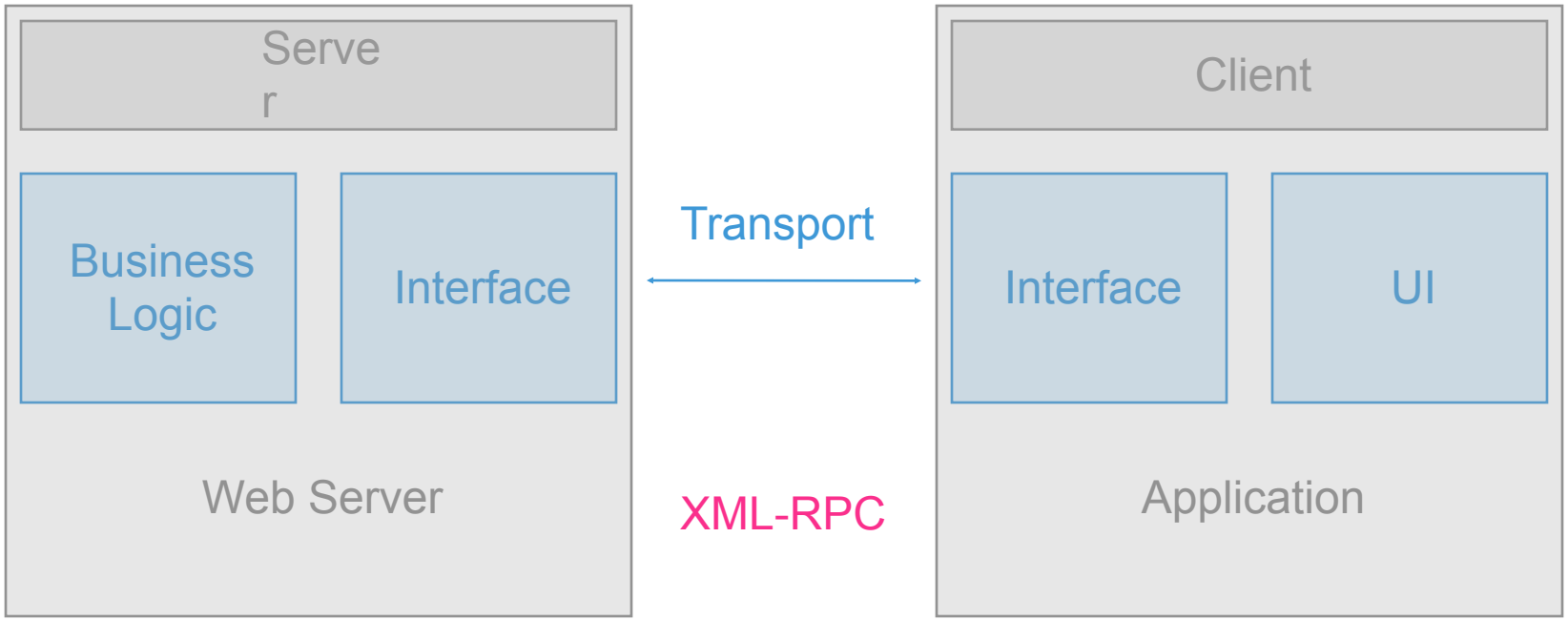
Web services in a nutshell



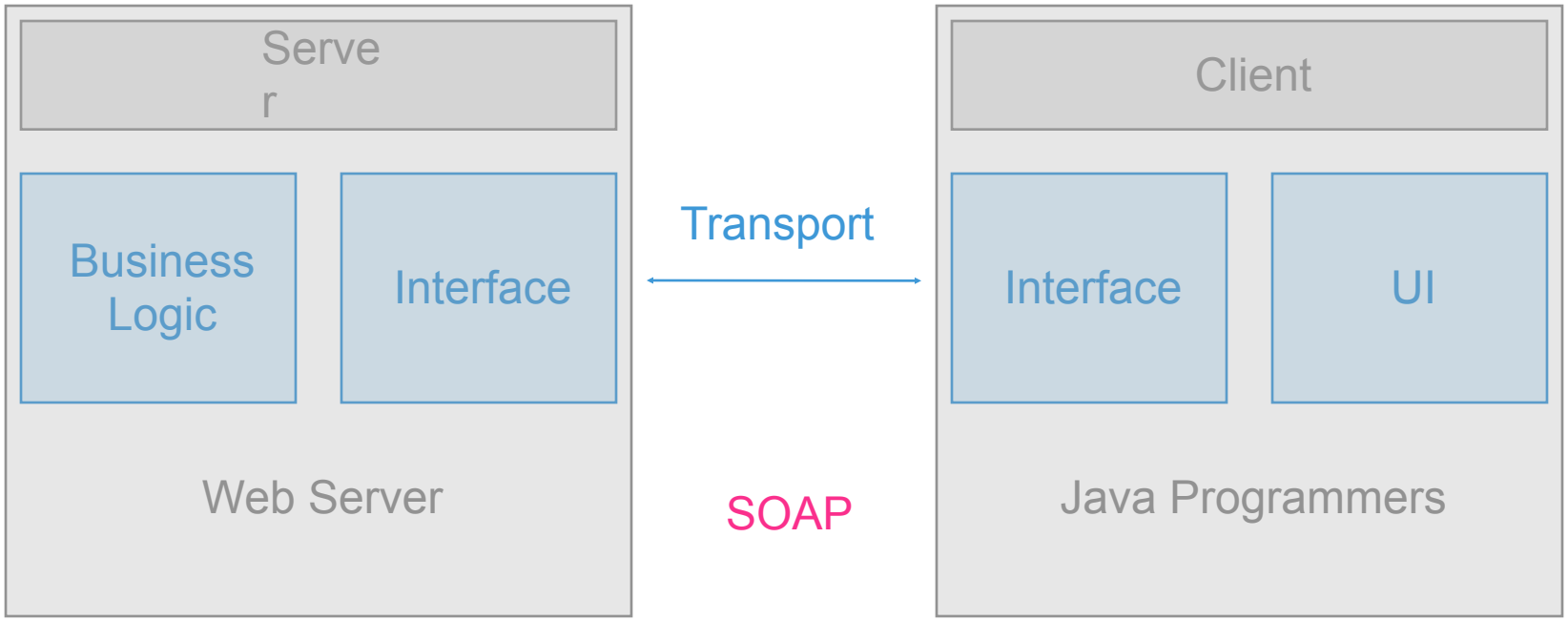
Web services in a nutshell



Web services in a nutshell



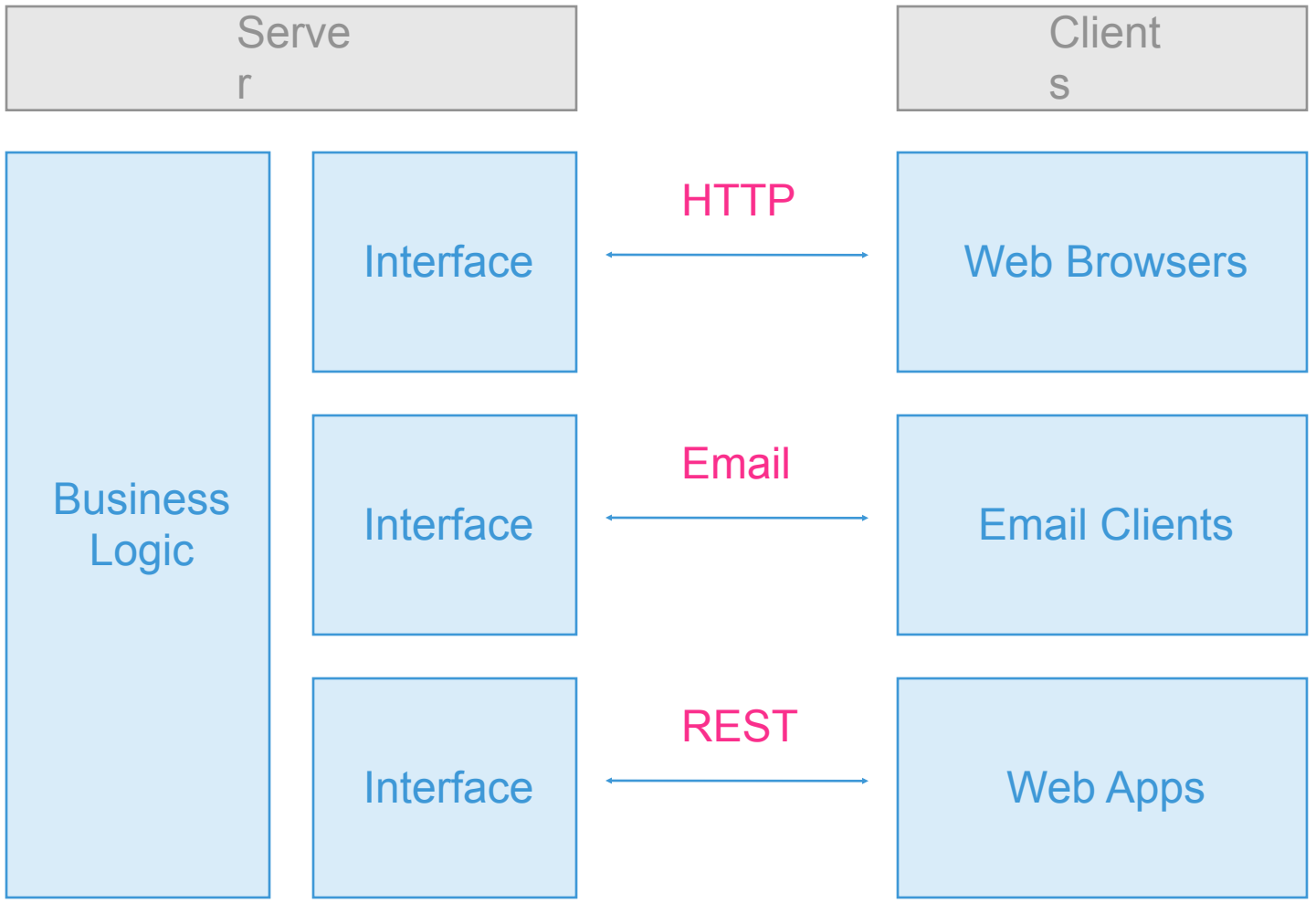
Web services in a nutshell



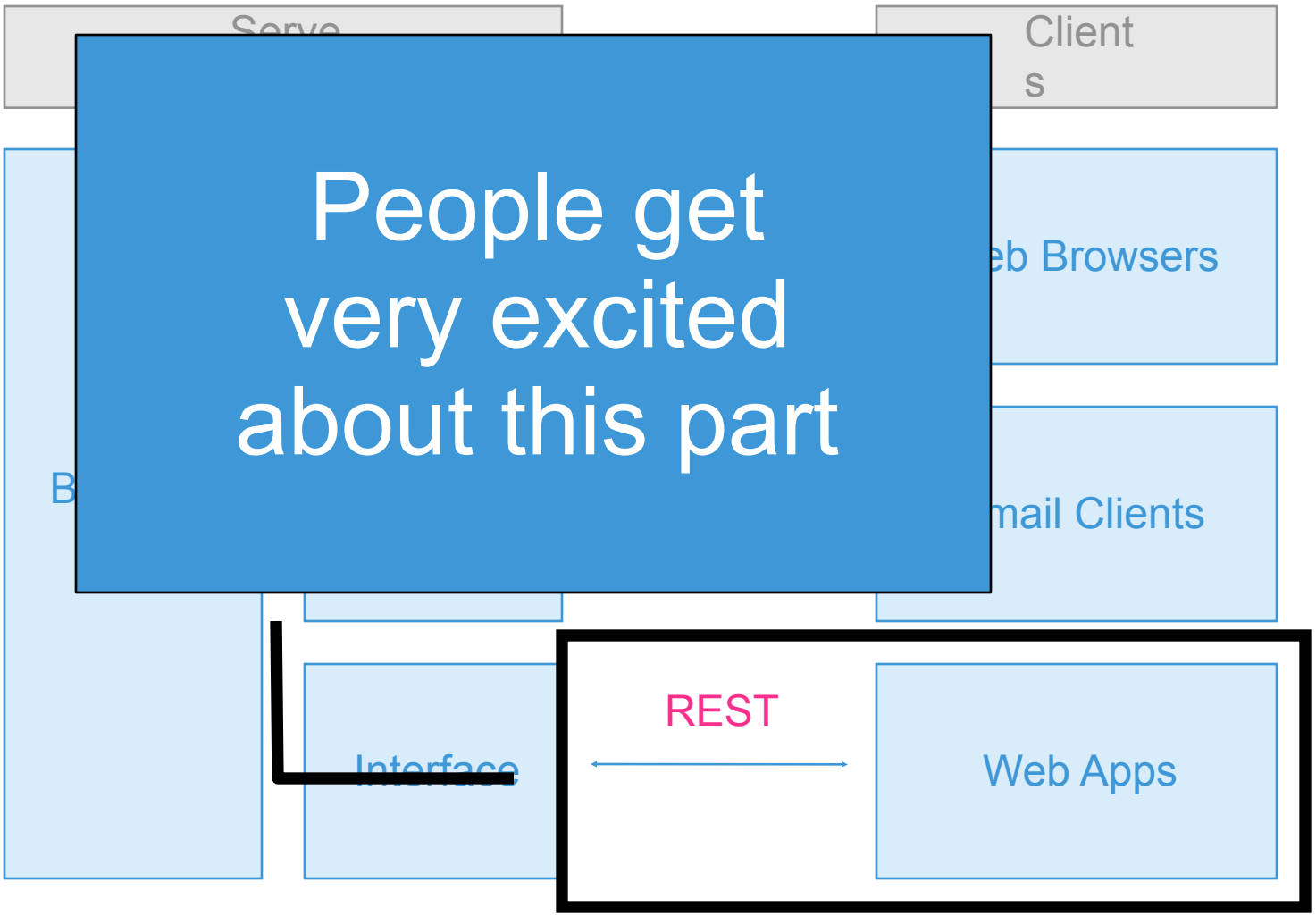
Why should I care?

- You can avoid code reuse
- While offering multiple services…

Web services



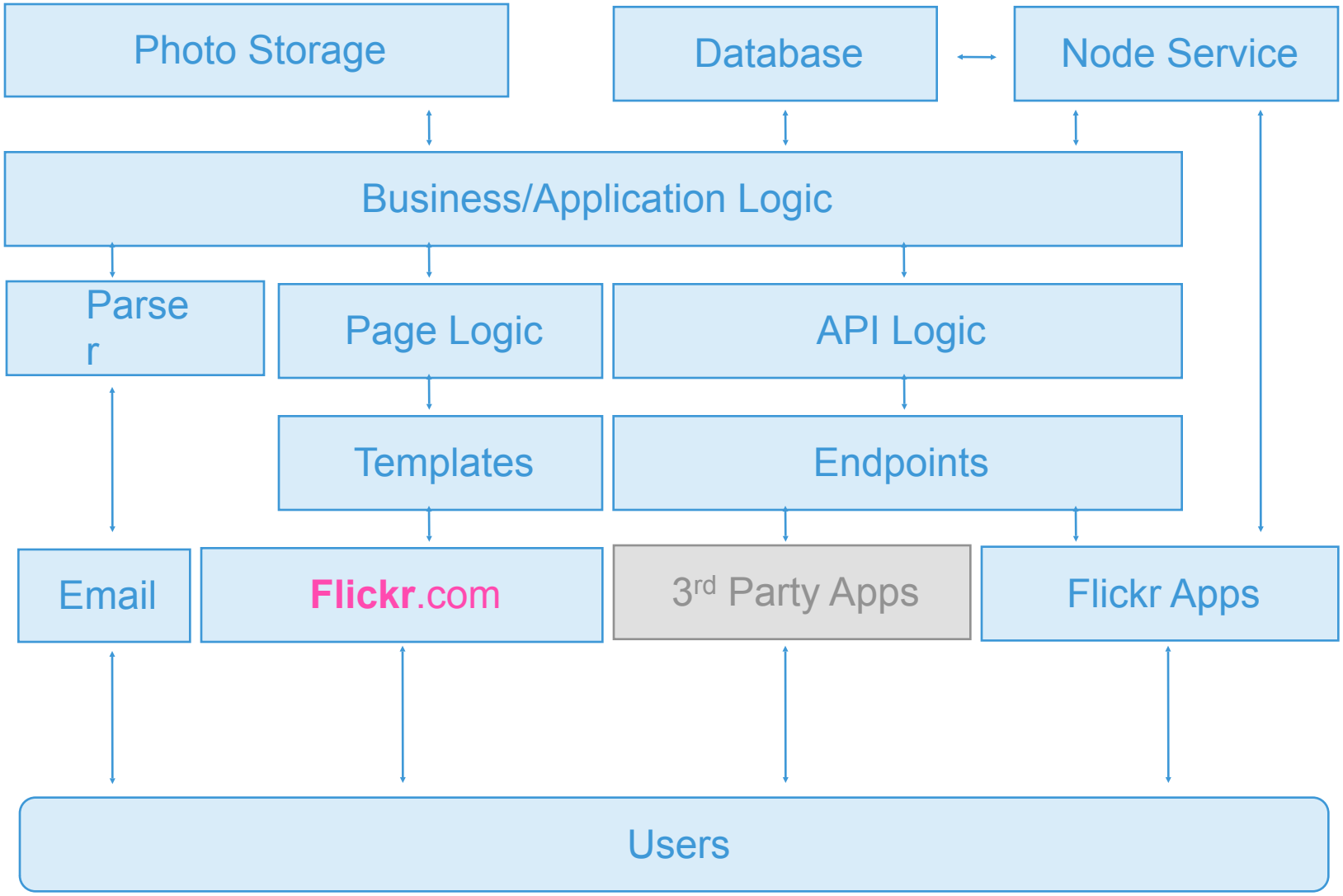
Web services



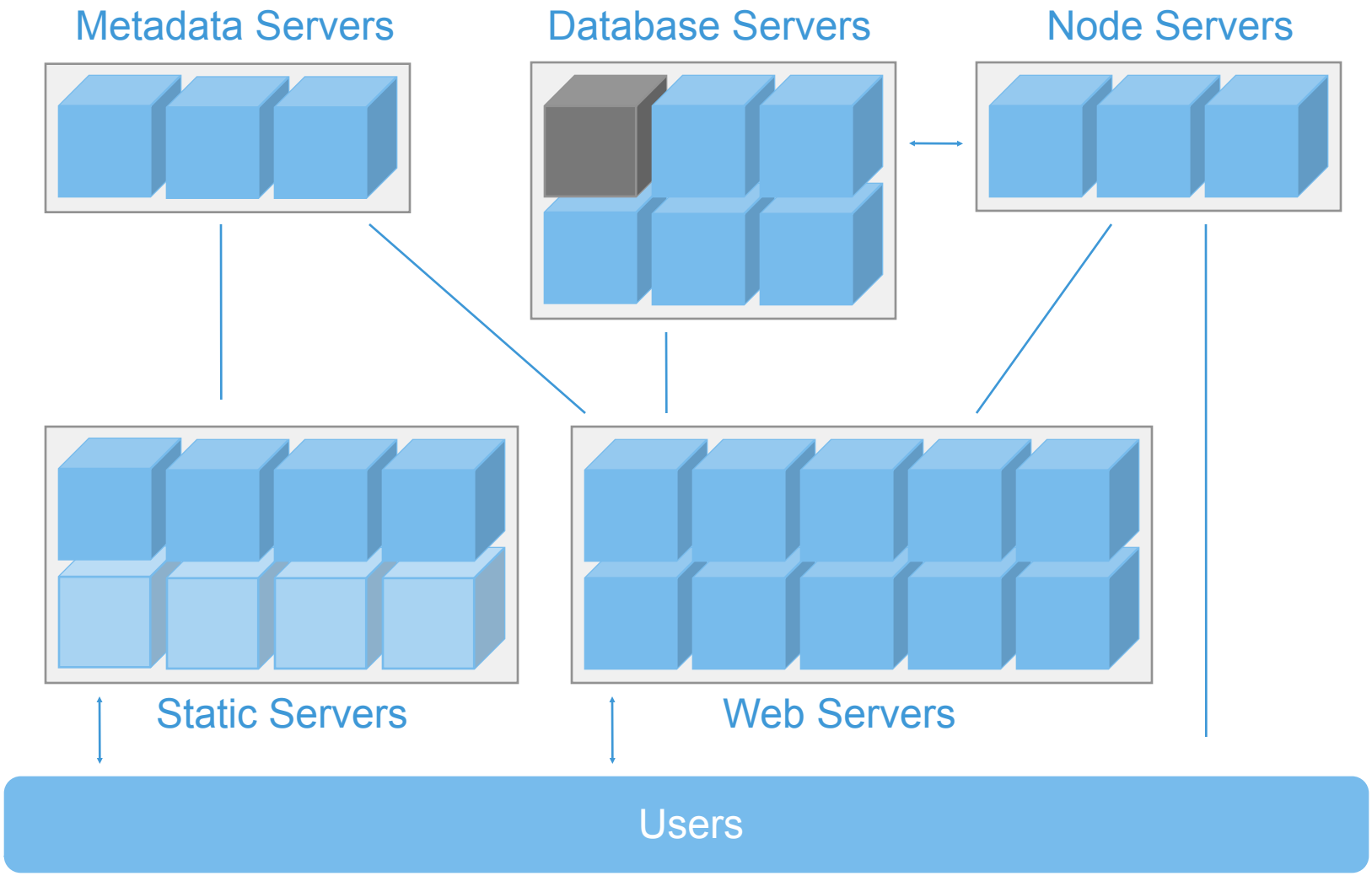
Ok, I get that bit

- Give me a real example!
- Aren’t you supposed to be talking about Flickr?

Flickr's Logical Architecture



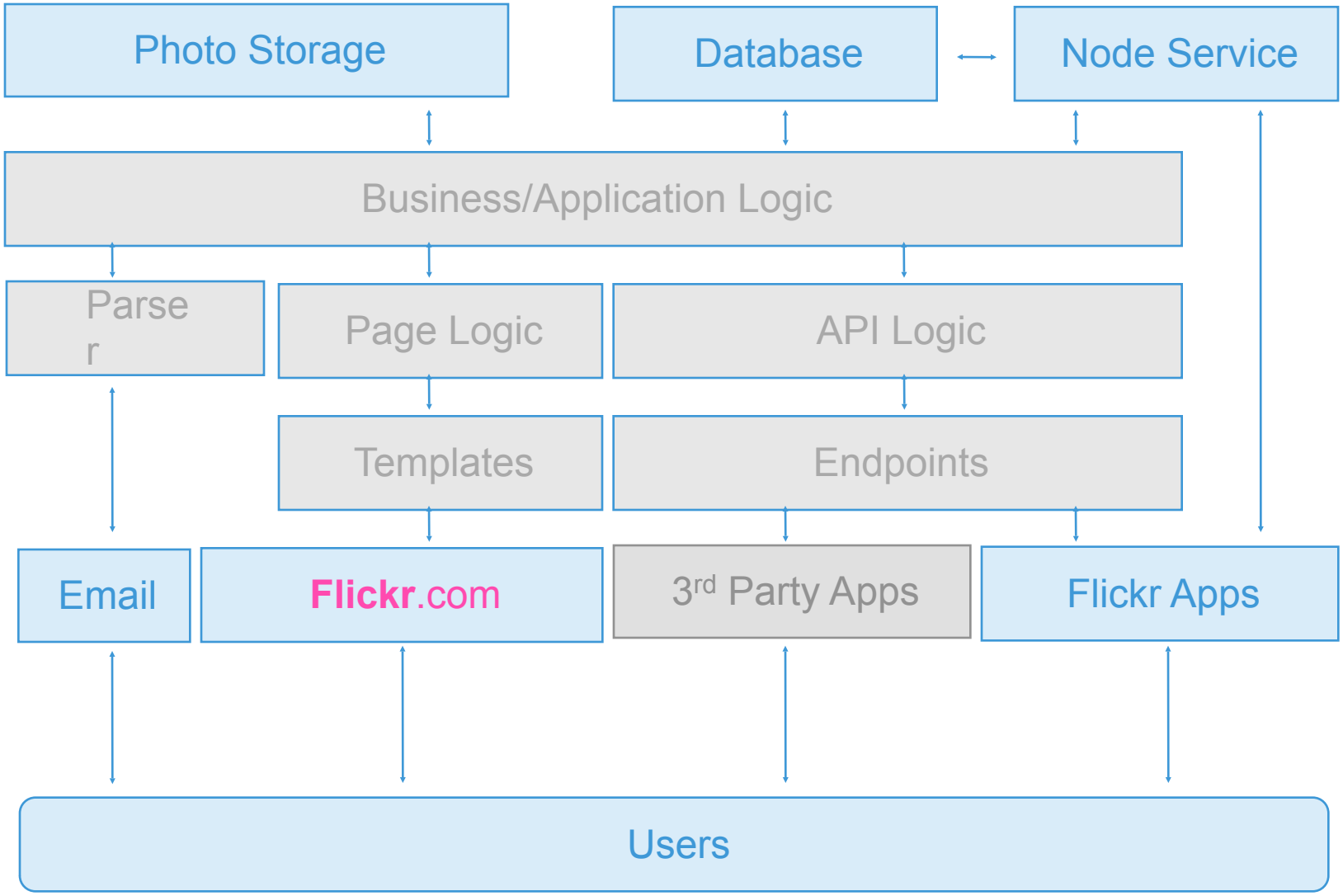
Flickr's Physical Architecture



But seriously…

- We only care about PHP!
- So where does Flickr use it?

PHP is at the core of Flickr



Ok, ok – what besides PHP?

- Smarty for templating
- PEAR for XML and Email parsing
- Java for…
  - Controlling ImageMagick (image processing)
  - Storage metadata
  - The node service
- MySQL (4.0 / InnoDb)
- Perl for deployment & testing tools
- Apache 2, Redhat, etc. etc.

Medium sized application

- Small team (3 programmers until recently)
  - 1 PHP, 1 Flash/DHTML, 1 Java
- >60,000 lines of PHP code
  - >80 smarty extensions
- >60,000 lines of templates
- >250,000 users
- >3,500,000 photos
- >50,000,000 page views per month
- Growing fast
  - Like, ***really*** fast
    - So these stats are out of date by now

Thinking outside the web app

- Services
  - Atom/RSS/RDF Feeds
  - APIs
    - SOAP
    - XML-RPC
    - REST
    - We love PEAR::XML::Tree

More services

- Email interface
  - Postfix
  - PHP
  - PEAR::Mail::mimeDecode
- FTP
- Uploading API
- Authentication API
- Unicode
  - (Not really a service, but common to all Flickr services)

Even more services

- Real time application
  - The “node service”
- ‘Cool’ flash apps
  - Which use the REST APIs
- Blogging APIs
  - Blogger API (1 & 2)
  - Metaweblog API
  - Atom
  - LiveJournal

APIs are simple!

- Modeled on XML-RPC (sort of)
- Method calls with XML responses
- Named arguments (key/name pairs)
  - Tricky in WebServices.framework on Mac OS X
- SOAP, XML-RPC and REST are just transports
- PHP endpoints mean we can use the same application logic as the website
  - Endpoints talk to the business logic using PHP function calls
    - Essentially a really fast transport

XML isn't simple :(

- PHP 4 doesn't have good a XML parser
  - PHP 5 is new and scares me
    - (and it wasn't out when we started)
- Expat is cool though (PEAR::XML::Parser)
- Why doesn't PEAR have XPath?
  - Because PEAR is stupid!
  - PHP 4 sucks!
  - Actually, PHPXPath rocks
    - http://phpxpath.sourceforge.net/

Creating API methods

- Stateless method-call APIs are easy to extend
  - They don't affect each other
- Adding a method requires no knowledge of the transport
  - We just get passed arguments and return XML
  - The transport layer hides all that junk
- Adding a method once makes it available to all the interfaces
- Self documenting – method dispatch requires a list of methods
  - Because everyone hates writing documentation

Red-Hot Unicode Action

- UTF-8 pages
- CJKV support
- It's ***really*** cool

どんなカメラをお使いでしょう？

日本語flickr / Discuss

Current Discussion

group photo pool
31 Aug '04, 11.45pm PDT

contactsってどう やって

kazenotayori says:

現行機種:
撮影後はどうしていますか:
次に買うのは何？:

みたいな感じで・・・
Posted at 11:35pm, 18 August 2004 PDT (Permalink)

kazenotayori says:

現行機種:
CASIO QV-2000UX 2.1Mega Pix
撮影後はどうしていますか:
Photoshopしてから、リサイズ
次に買うのは何？:
やっぱり、D70でしょうか。
Posted 2 weeks ago. (Permalink)

Unicode for all

- It's ***really*** easy
  - Don't need PHP support
  - Don't need MySQL support
  - Just need the right HTTP headers
  - UTF-8 is 7-bit transparent
    - Just don't mess with high characters
      - Don't use HtmlEntities()!
        - » Or |escape in Smarty
- But bear in mind…
  - JavaScript has patchy Unicode support
  - People using your APIs might be stupid
    - Some of them ARE stupid, guaranteed

Scaling the beast

- Why PHP is great
- MySQL scaling
- Search scaling
- Horizontal scaling

But first…

- Why do we need to scale?
  - There are a lot of people on the Internet
  - They all want to use our “web services”
  - Whether they know it yet or not

Why PHP is great

- Stateless
  - We can bounce people around servers
  - Everything is stored in the database
  - Even the smarty cache
  - “Shared nothing”
    - (so long as we avoid PHP sessions)
- But what this really means…
  - …is we just have to deal with scaling elsewhere

A MySQL Scaling Haiku

- Database server slow
- Load of over two hundred
- Replication wins!

MySQL Replication

- But it only gives you more SELECT's
- Else you need to partition vertically
- Re-architecting sucks :(

Looking at usage

- But really, we SELECT much more than anything else
  - A snapshot says
    - SELECT's 44m
    - INSERT's 1.3m
    - UPDATE's 1.7m
    - DELETE's 0.3m
- 19 SELECT's for each IUD

Replication is really cool

- A bunch of slave servers handle all the SELECT's
- A single master handles IUD's
- We can scale horizontally, at least for a while.

Searching

- A simple text search
- We were using RLIKE
- Then switched to LIKE
- Then disabled it all together

FULLTEXT Indexes

- FULLTEXT saves the day!
- But they’re only supported on MyISAM tables
- And we use InnoDb for locking
- We’re doomed :(

But wait!

- Partial replication saves the day

- Replicate the portion of the database we want to search
- But change the table types on the slave to MyISAM
- It can keep up because it's only handling IUD's on a couple of tables
- And we can reduce the IUD's with a little bit of vertical partitioning

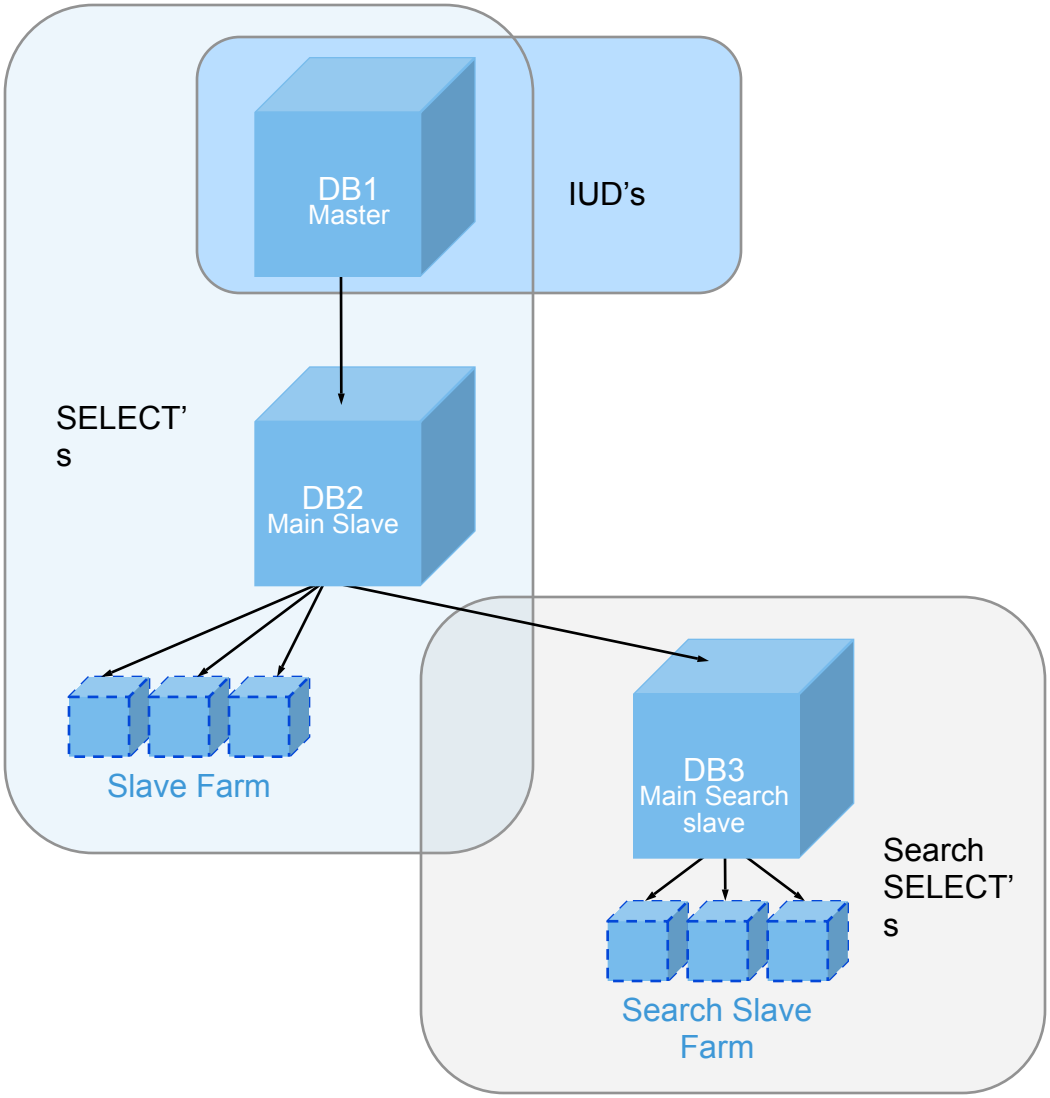
JOIN's are slow

- "Normalised data is for sissies"
- Erm,
- "Selective de-normalisation can be a big win"

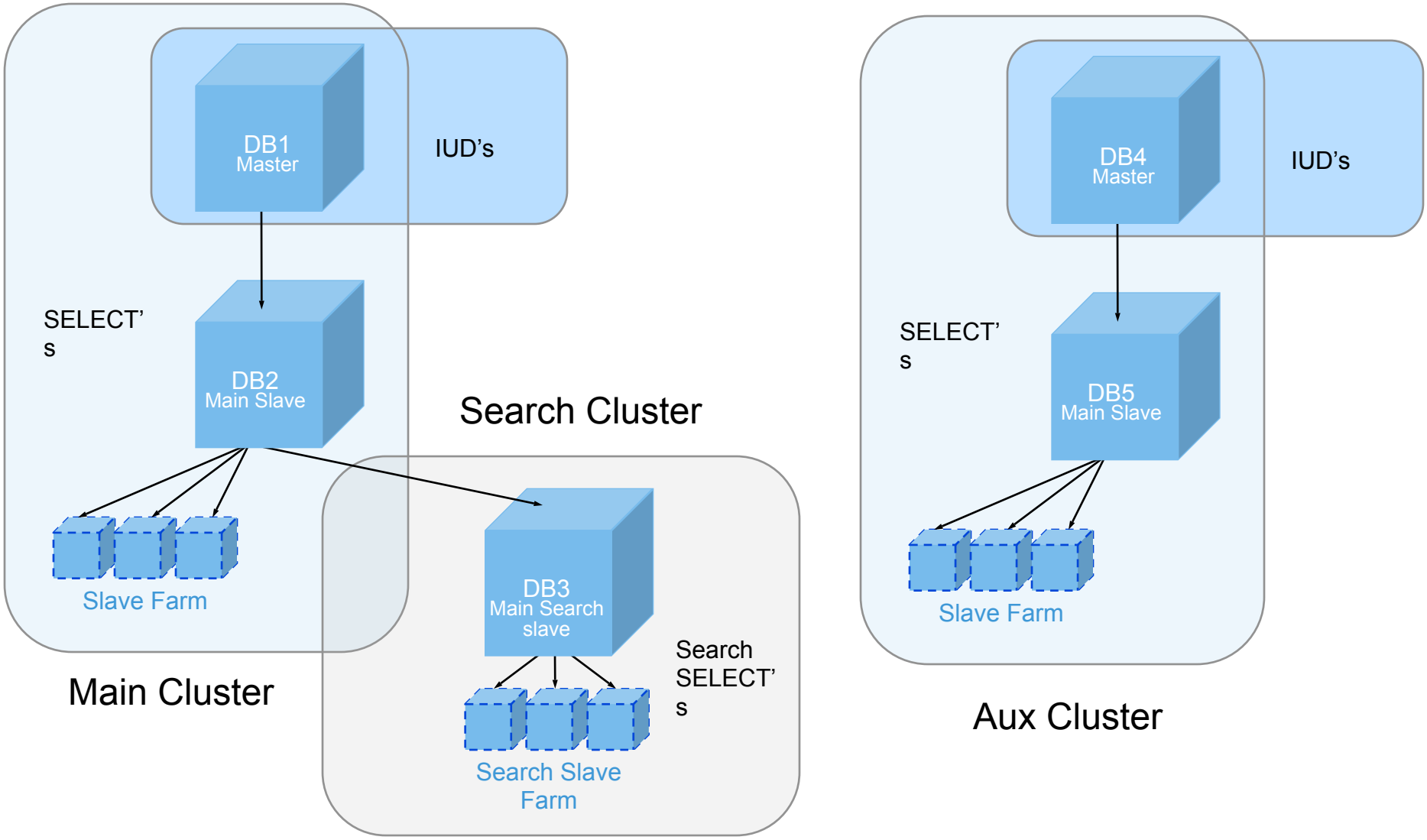
- Keep multiple copies of data around
- Makes searching faster
- Have to ensure consistency in the application logic

- For instance, have a concat'd field containing a bunch of child-row data, just for searching.

Our current setup



Our current, current setup



Horizontal scaling

- At the core of our design
- Just add hardware!
- Inexpensive
- Not exponential
- Avoid redesigns/re-architectures

Talking to the Node Service

- Just another service with an API
  - But just internal at the moment
- Everyone speaks XML (badly)
- Just TCP/IP - fsockopen()
- We're issuing commands, not requesting data, so we don't bother to parse the response
  - Just substring search for `state="ok"`
  - This only works for a simple protocol

Still talking to the Node Service

- Don’t rely on it!
  - Check the connection was established
  - Use a connection timeout
  - Use an IO timeout!

RSS / Atom / RDF

- Different formats
  - (all quite bad)
- We're generating a lot of different feeds
- Abstract the difference away using templates
- No good way to do private feeds. Why is nobody working on this? (WSSE maybe?)
  - Most of the feed readers (including bloglines.com) support basic HTTP Auth
    - Easy to implement in PHP
      - We love PHP
        - » It's great!

Receiving email

- We want users to be able to email photos to Flickr
- Get postfix to pipe each mail to a PHP script
- Parse the mail and find any photos

- Cellular phone companies hate you
- Lots of mailers are retarded
  - Photos as text/plain attachments
  - Segments out of order
  - No mime types
  - UUEncoded and mime-less

Processing email

- PEAR to the rescue
- Mail::mime_decode
  - With some patches
    - UUEncoding
    - Relax the address atom parser
- We need to convert character sets
  - ICONV loves you

Upload via FTP

- PHP isn't so great at being a daemon
  - PHP4, I mean. Maybe PHP 5 is great
- Leaks memory like a sieve
- No (easy) threads
- Java to the rescue
- Java just acts as an FTPd and passes all uploaded files to PHP for processing
  - This isn't actually public
- Not my idea
  - Bricolage does this I think. Maybe Zope?

Blogs

- Why does everyone loves blogs so much?
- Only a few APIs really
  - Blogger
  - Metaweblog
  - Blogger2
  - Movable Type
  - Atom
  - Live Journal

It’s all broken

- Lots of blog software has broken interfaces
- It’s a support nightmare
- Manila is tricky
- But it all works, more or less
- Abstracted in the application logic
- We just call `blogs_post_message();`
- And so can you, via the API

Back to those APIs

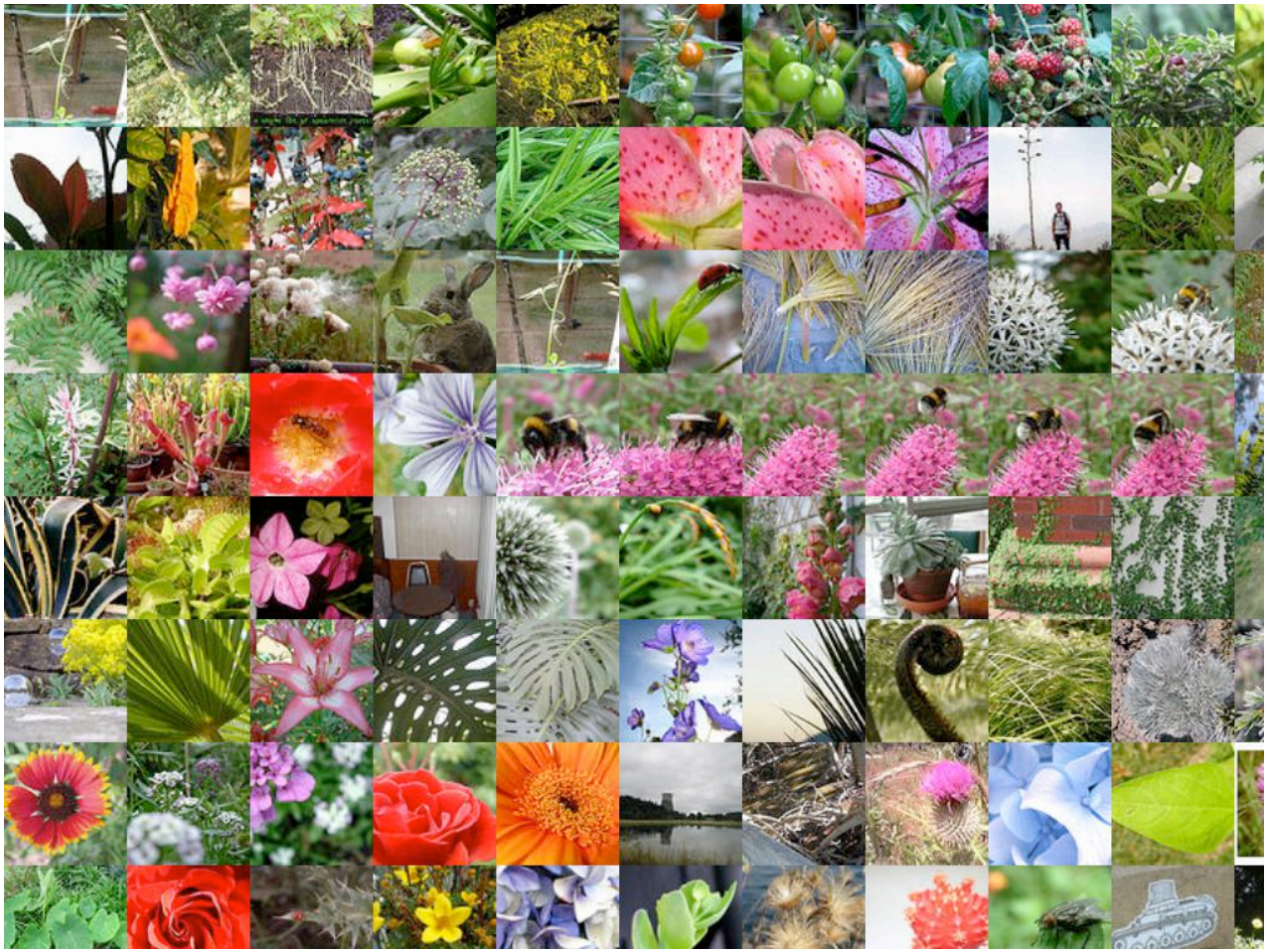
- We opened up the Flickr APIs a few months ago

- Programmers mainly build tools for other programmers
- We now have Perl, python, PHP, ActionScript, XMLHTTP, .NET, Objective-C, C++, C and Ruby interface libraries

- But also a few actual applications

Flickr Rainbow

Tag Wallpaper



iPhoto Plugin

- We developed a Mac uploader
- But it wasn’t great
- A user developed an iPhoto plugin
- It *was* great

- APIs encourage people to do your work for you

Flickr Carnivore

- Uses Carnivore PE
  - Sniffs AIM traffic (amongst others) from the local net
- Calculates the most popular words of the moment
- Uses the Flickr API to display photos of those words
- It’s like a really invasive zeitgeist

Flickr Tivo

- A Tivo app which uses Flickr photos
- Just Type in some tags
- And your TV becomes a “digital picture frame”

So what next?

- Even more scaling
- PHP 5?
- MySQL 5?
  - or NDB?
- Taking over the world

Flickr : Web Services

Cal Henderson

These slides are online

http://ludicorp.com/flickr/

Any **Questions?**