CouchDB at the BBC

• In production
• Key Value Store - not a MapReduce, nor a document database*
• Operations more important than features
• A small - but important - feature in a much larger infrastructure
the platforms

• Until recently, two internet platforms

• news.bbc.co.uk

• www.bbc.co.uk

• Both are essentially static
the platforms

- There is now a third
- Forge
  - PHP - Zend
  - Lots of memcached
  - Java - Spring - Tomcat
Forge needs a key-value store

- Not everything needs to be ACID
- Replication of MySQL is not easy
- Scaling of MySQL isn’t too easy either
- ... but we do have MySQL for those times when it’s still appropriate

- NoSQL
Our problems

• Some of the most important considerations ...
Our problems: Don’t know what we’re doing

http://www.flickr.com/photos/plunkmasterknows/357836855
Our problems: Don’t know how we’ll be used

http://www.flickr.com/photos/48116183@N00/536340489/
Our problems: Can’t have EVERYONE trying to figure this out

http://www.flickr.com/photos/not_words_but_in_things/356632727
Our problems: expandability is a MUST
A typical setup?
Large data sets
Operational sharding
Developers’ contract
Ops expandability
Views
Active-active DCs
Uses of the KV store

• Homepage - 40M users’ preferences
• /spaces* - preferences
• iPlayer - 2-3M playlists
• LabUK* experiments

• These are all in the works, coming in the next 3-6 months ... or have had their identities changed
Why the KV API?

• API stability
• Contract with the developer community
• Developer usage constraints
• Business-level access control
• Monitoring and alerting
• Expandability
Why CouchDB?

- Append-only file
- Good IO to disk
- Graceful performance degradation under load
- Consistent memory usage
- CouchDB + Erlang: OS manageability
Benchmarks
Benchmarks
Hardware: our CouchDB servers

- 2 x Quad core Intel Xeon
- 2 GHz
- 16 GB RAM
- 4 x 10k SAS RAID 5
- 0.5 TB usable HDD
Disk IO bound

- The hardware is our limitation
- `beam.smp` works well in our setup
- CPU interrupts, load spread quite evenly
What next?

• Relying on hardware redundancy - not software :-(
• Talk to the Meebo folks (worried about Twisted)
• Better understanding of compaction, near continuous replication and conflicts

• https://monitor.forge.bbc.co.uk/zport/dmd/Reports/Multi-Graph%20Reports/KV%20store
Thank you
BBC Feeds Hub

26th June 09
CHALLENGE
“The number of new projects across the BBC starting to use feeds in creative ways is growing very quickly - just think of spaghetti... on a massive scale.”
DAB Live Text:
Man Utd 1 – 0 Tottenham

Mobile:
Man U 1 – Spurs 0

Web:
Manchester United 1 – 0 Tottenham Hotspur
MODEL
Messages from various sources are collated, transformed and routed to their destinations.

**Sources:**
- Live text
- Now playing
- On air now

**Destinations:**
- Push Feeds
- Ejabberd
- Xmpp client
- Other

**Network Components:**
- Feeds Hub
- RabbitMQ

BBC Audio & Music Interactive and Mobile

LShift
Messages arrive at SOURCES and arrive at DESTINATIONS via a PIPELINE.
PIPHELINES are connected COMPONENTS expressing business rules ...

which can all be wired together arbitrarily.
GATEWAYS are the relays to outside by which messages arrive at SOURCES and depart from DESTINATIONS.

e.g., HTTP POST

e.g., XMPP PubSub
FEEDS HUB
ARCHITECTURE
RabbitMQ underneath; let others do what they're good at, e.g., ejabberd, RabbitHub
Model items are stored as documents in CouchDB

Model items refer to PLUGINS – eg “XMPP Gateway”, “Regular expression replacer”

Plugins are just programs
The ORCHESTRATOR manages a process for each model item.
It wires everything together with AMQP exchanges and queues.
Streams can be interleaved, duplicated, and load-balanced using the messaging layer.
Harnesses mediate between the orchestrator and the plugins’ programming language environments.
Processes are organised into a hierarchy and can be restarted individually by the orchestrator.
Open Source

• Feeds Hub
  • Prototype release scheduled 6th July

• RabbitMQ
  • www.rabbitmq.com
  • http://groups.google.com/group/rabbitmq-discuss