Databases suck for Messaging

Alexis Richardson

Oxford Geek Night May 2009
Computers were meant to get rid of this
A new kind of fail?
Solution - use a database?
Databases were meant to get rid of this too
I want to know when THIS changes
Problem - databases suck for messaging

- DATA is facts (“state”) persisted on disk
- Databases are great for storing facts and asking questions about them
- If you know what to ask them and are willing to keep asking them

- INFORMATION is always changing
- Networks are great pushing changes (“messages”) to recipients
- Databases are not optimised for this
Social applications store data
Social applications store data

And there’s terabytes of it
Social applications store data

And there’s terabytes of it

Enterprises are made of people too
So: the same issues arise
I want to know when THIS changes
Email doesn’t scale
Information gets old - real quick
But current information is valuable
Example: Flickr

- Flickr is a vast database of social objects
- Filtered by interest and relationships
- So - tell me what’s currently relevant
- Without sending me more emails....
Polling sucks

On monday friendfeed polled flickr nearly 3 million times for 45000 users, only 6K of whom were logged in. Architectural mismatch. #oscon08

7:47 PM Jul 23rd, 2008 from twhirl

timoreilly
Tim O'Reilly
What’s going on here? We are trapped in the database world view

That is, the subscriber sits in a loop and asks repeatedly,

"Anything new yet?"
"Anything new yet?"
"Anything new yet?"

The server (if it's smart enough to) has to check and reply,

"No, you have the most recent version."
"No, you have the most recent version."
"No, you have the most recent version."

Example due to Brett Slatkin and Brad Fitzpatrick at Google
How do we defeat this evil?
Can we apply the Hollywood Principle?
Can we apply the Hollywood Principle?

Hint: phone calls and SMS don’t travel through databases
Email push is direct

alexis  to:blaine  “blah”  blaine
Email push is direct

alexis → to: blaine
  “blah”
  “blah”
  “blah”
  “blah”

blaine
Polling is just reverse spam

alexis

"?"

"?"

"?"

"?"

blaine
Publishing to a queue ("topic") takes the spam burden off the receiver.
And you can use pubsub and queues in all sorts of ways
Databases are not meant to do Pubsub

SELECT * FROM queue WHERE done = 0 ORDER BY created LIMIT 1
A true story

most people would just create a simple "queue" table

and: SELECT * FROM queue WHERE done = 0 ORDER BY created LIMIT 1

“concurrency issues on that thing now - inserting into the queue occasionally takes longer than doing the task that needs to be executed synchronously”

“middle management did not want to have "new third party software" because it would be too much Operations to learn and manage”

so they decided for the time being a MySQL based queue would be sufficient (only few million messages/day) and implemented it in PHP/MySQL resulting in lots of dev hours for implementation / testing, and more hours because of performance issues ..

so i think we have spent about the same amount of time as we would to get a descent thing up and running, but now we're stuck with a lame and unscalable mysql database machines that is dressed as a message broker
Flickr from a Pubsub point of view

- Flickr = people publishing to a vast set of streams (photostreams)
- Users express interest through subscription
- I don’t need to see everything - only changes on what I follow

This seems better - but what’s missing?
- I am still trapped by the database world view.
- I still poll for changes (that’s what RSS does)
- I want the PUSH that email gave me, without the spam.....
Characteristics of PUBSUB

- A means of authenticated communication (network transport)
  - eg HTTP, OAuth

- An addressable place to publish to
  - Usually a topic, feed, endpoint or address.

- A way to name, enrol, share, and discover these addressable places
  - For example “blaine@twitter.com” or - TBD!

- A way to deliver and ack delivery (or “take responsibility”)

- The above is in fact a distributed object system
How do you solve a problem like Flickr/Twitter/...?

<table>
<thead>
<tr>
<th></th>
<th>Database</th>
<th>Pubsub/MQ</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>Objects serialised as rows in tables</td>
<td>Messages organised into streams</td>
</tr>
<tr>
<td><strong>Interest</strong></td>
<td>Filter by query</td>
<td>Durable “follow”</td>
</tr>
<tr>
<td><strong>Notification</strong></td>
<td>Pull (polling sucks!)</td>
<td>Push</td>
</tr>
<tr>
<td><strong>Buffering</strong></td>
<td>Put = add row to table</td>
<td>PUT and GET are symmetric</td>
</tr>
<tr>
<td><strong>Scale</strong></td>
<td>Overheads tend to grow indefinitely</td>
<td>Data flows out to destinations</td>
</tr>
</tbody>
</table>

Notes:
- Pull: data is requested from the source
- Push: data is sent to the destination
- Buffering (non-idempotent) refers to operations that may not be idempotent when executed multiple times.
- Scale: overheads tend to grow indefinitely, data flows out to destinations.
You might need messaging if ... you need to monitor data feeds

(CC) Kishore Nagarigari
You might need messaging if ... you need a message delivered responsibly
You might need messaging if ... you need things done in order
You might need messaging if ... you are using the cloud
RabbitMQ is a messaging server that just works!

Im in yr serverz, queueing yr messagez

Photo credit: http://flickr.com/photos/53366513@N00/67046506/