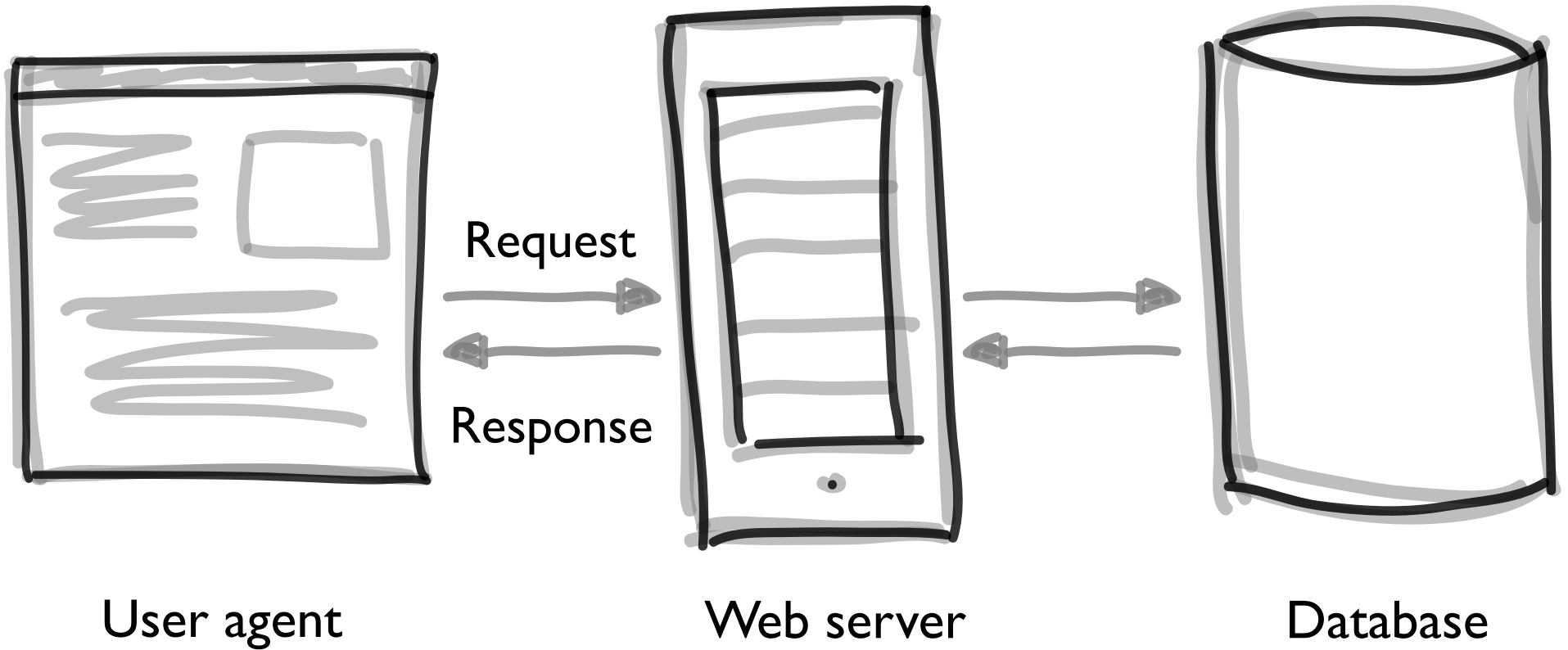


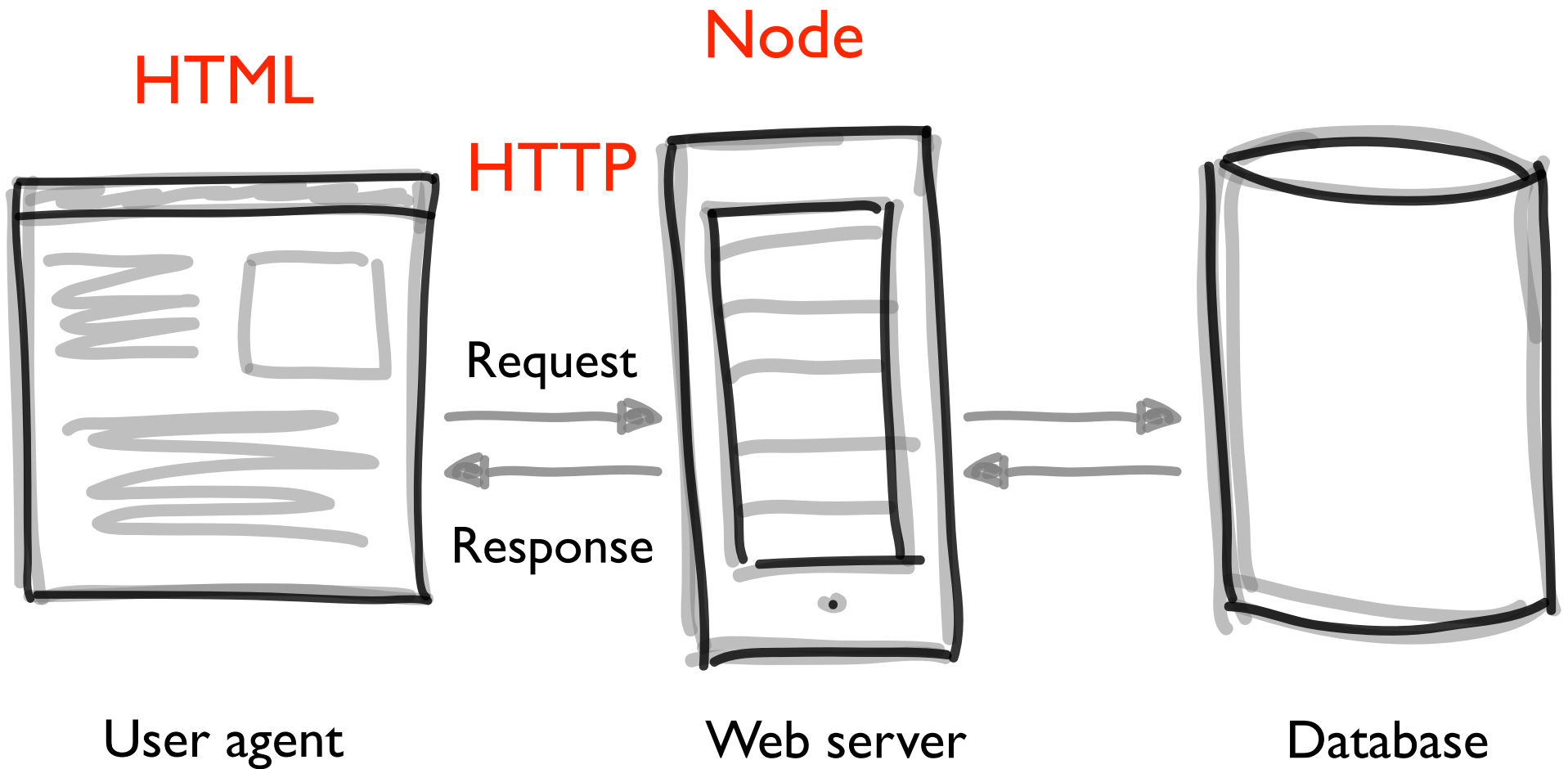
Persistence

Jonathan Sillito

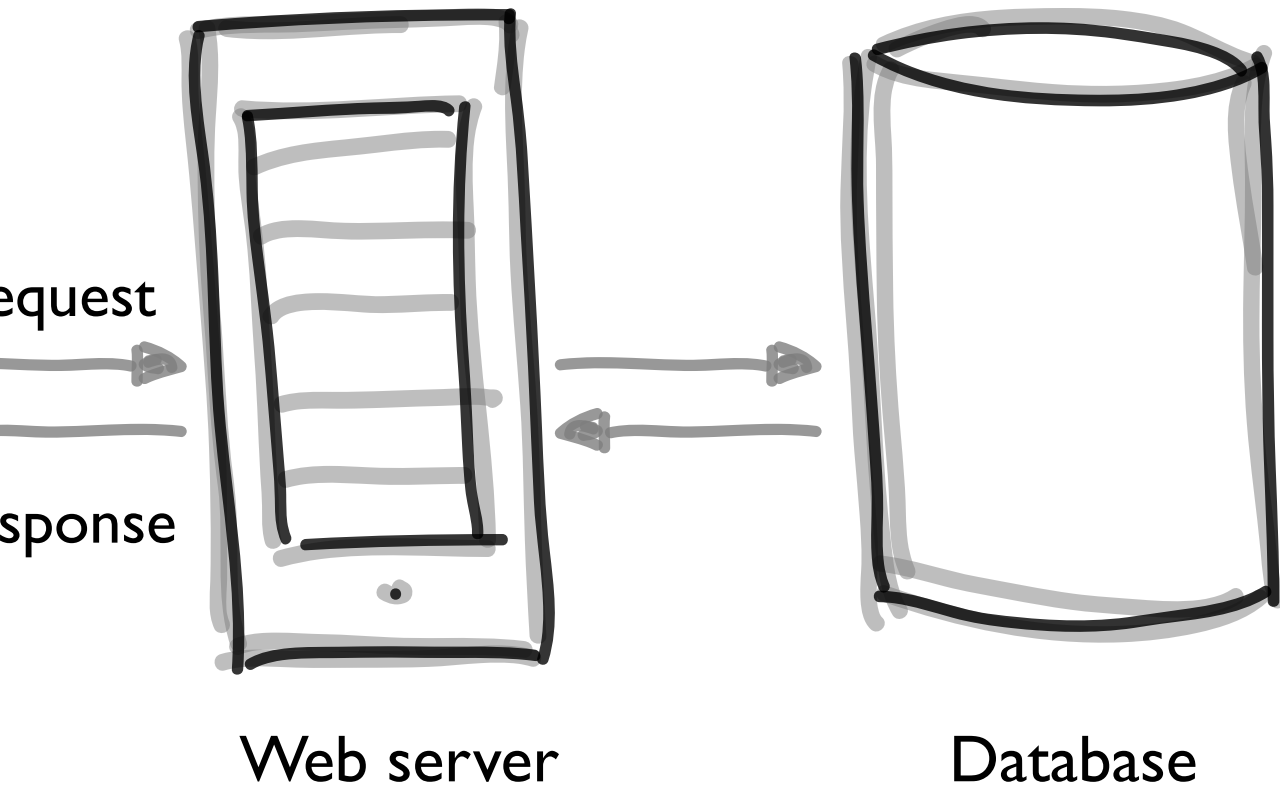
Web-based systems



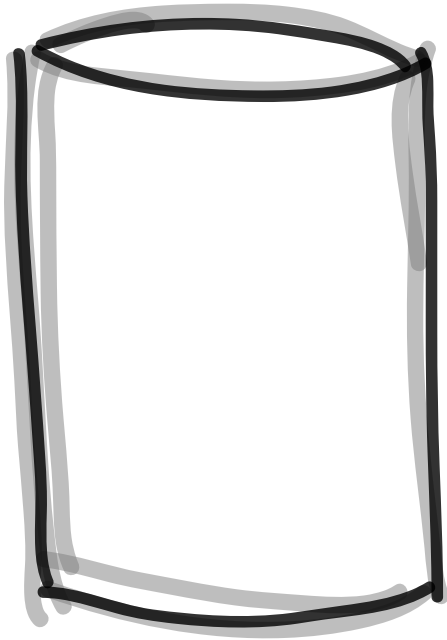
Web-based systems



Web-based systems



Persistence in Web-based Systems



Database

The question how and where to store system data? is central to engineering a web-based system.

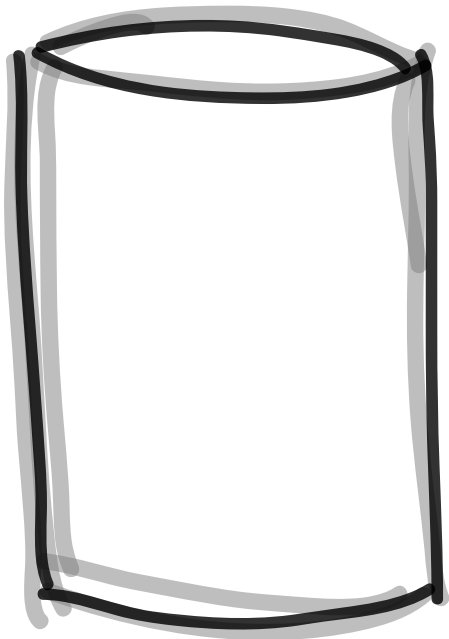
Persistence and Twitter

As of several months ago, twitter deals with about 70 million tweets per day or 800 tweets per second. At 200 Bytes/tweet, this works out to about 12 GB per day of tweet data and 8 TB of data overall.

They need to be stored in a way that allows them to be stored and delivered in realtime.

- MySQL database for basic storage
- Snowflake: a system for generating unique IDs for tweets (can do 10k/s)
- Flockdb: a distributed graph database (Justin Bieber has 5.1 million followers)
- Hosebird: streaming API for delivering tweets

Some Options for Persisting Data



SQL database

SQLite

KeyValue or Document store

VMWare's Redis

Special purpose

Google's BigTable (tabular data)

Command reference - Redis


http://redis.io/commands#hash

redis

Commands Clients Documentation Community Download Issues

All Keys Strings **Hashes** Lists Sets Sorted Sets Pub/Sub Transactions Connection Server

HDEL key field Delete a hash field	HLEN key Get the number of fields in a hash
HEXISTS key field Determine if a hash field exists	HMGET key field [field ...] Get the values of all the given hash fields
HGET key field Get the value of a hash field	HMSET key field value [field value ...] Set multiple hash fields to multiple values
HGETALL key Get all the fields and values in a hash	HSET key field value Set the string value of a hash field
HINCRBY key field increment Increment the integer value of a hash field by the given number	HSETNX key field value Set the value of a hash field, only if the field does not exist
HKEYS key Get all the fields in a hash	HVALS key Get all the values in a hash



Redis

SQL (& RDBMS)

Relational Database Management System

A database management system based on the relational model

- Data is stored in tables with columns of attributes
- Relationships among data is stored in the tables

Creating a new table in SQL

Primary key uniquely identifies each row in a table

AUTOINCREMENT means that the value will be assigned as +1

of the previous value

```
create table users (  
  id INTEGER PRIMARY KEY AUTOINCREMENT,  
  name TEXT,  
  email TEXT UNIQUE,  
  password TEXT,  
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP);
```

UNIQUE means that the value must be unique across all rows.

Columns (or attributes) are listed as <name> <type> pairs (plus other constraints).

Inserting Rows in SQL

```
insert into users (name,email,password) values  
( 'Sally', 'sally@example.com', '9000765345678' );
```

Relationships In Data?

Assumes there is a users table
and an issues table.

```
create table comments (  
  id INTEGER PRIMARY KEY AUTOINCREMENT,  
  user_id INTEGER,  
  issue_id INTEGER,  
  content TEXT,  
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP);
```

Relationships In Data?

How relationships are modelled will depend on the nature of the relationship.

- One to many (e.g., comments to users)
- Many to many (e.g., tags to comments)

Selecting data from a table

```
select * from users where email='sally@example.com';
```

SQLite

SQLite

With SQLite there is no server, just a file that is accessed through the command line or through modules.

There is no user/permissions management.

This command (creates and) opens a database called tasks, stored in a file called “tasks.db”.

```
% sqlite3 tasks.db
SQLite version 3.6.12
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite>
```