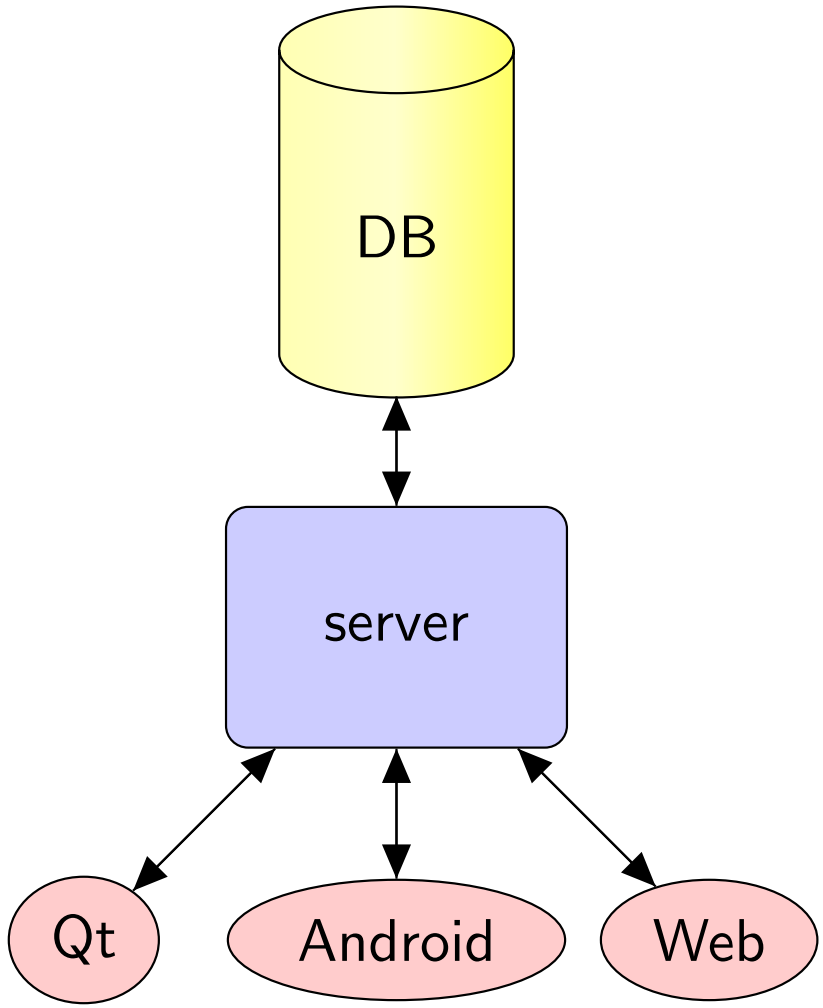
Coexist: A database framework for Android and Web clients

Anthony Naddeo
adv. Sudarshan Chawathe

February 14, 2013

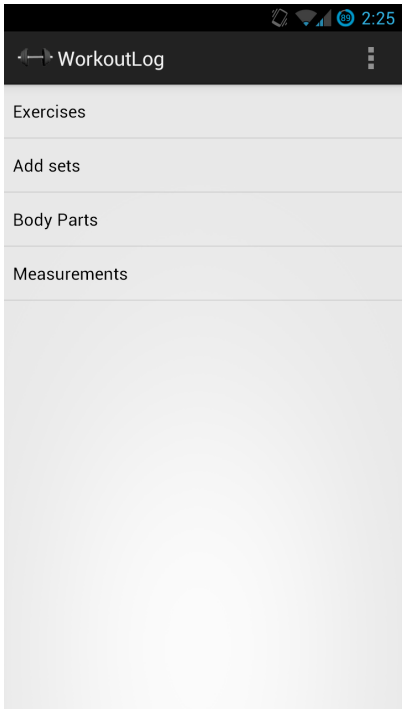
Quick and codeless data management

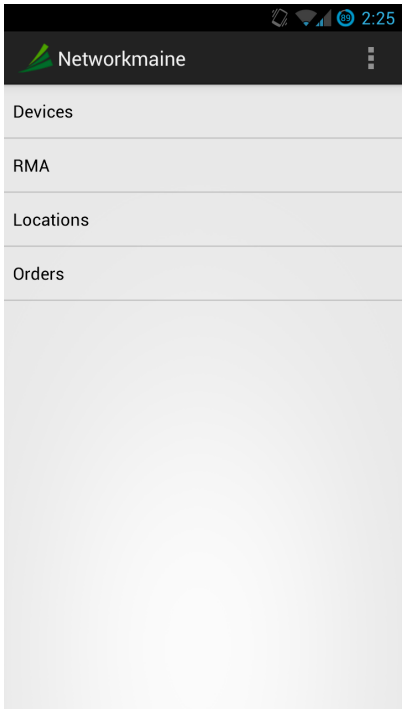
Coexist is a framework that allows Android and web clients to perform CRUD operations on a shared database by filling out configuration files.



Why is this useful?

- There is a large overlap between different CRUD focused applications. Having to write a second application to do the same thing as the first, with different data, can be avoided.
- Enabling mobile (or arbitrary) client types is common enough that it should be done automatically.





How does the database work?

- The database needs to be mirror-able

How to support mirroring

We make use of *metacolumns*: typical SQL columns that are used for some application level purpose.

id	name	year	*mod_ts*
1	Anthony	4	*0*
2	Lucas	4	*10*
3	Steven	3	*0*
4	Ryan	4	*10*

- The `mod_ts` column contains the most recent update on that tuple
- Allows the server to give clients missing information when supplied with most recent client side `mod_ts`

Motivation

- The simplest solution.
- No dependencies.

How does the server work?

- The server is designed as an API.

Synchronization protocol and API

There are two simple RESTful function that enable synchronization

- `/api/sync` : Request all tuples on the server that are not present on the client.
- `/api/schema` : Request the SQL that builds the database.

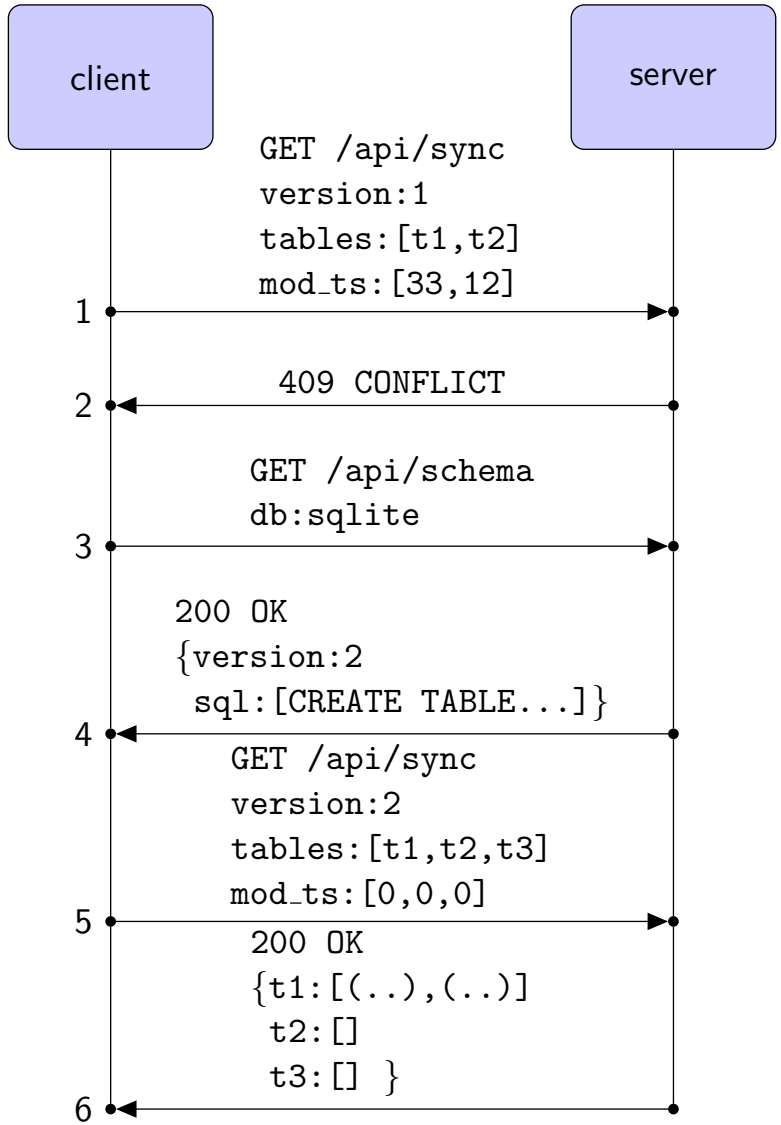
Normal `GET` requests can include a representative signature instead. T and M are the concatenation of table names and maximum `mod_ts`.

- $sha1(version + T + M + username + sha1(password))$

Sample client - server dialog

Here is a sample conversation between a server and a client who has an out of date schema while attempting to re-sync.

note: The application will not require updates for database changes.



Motivation

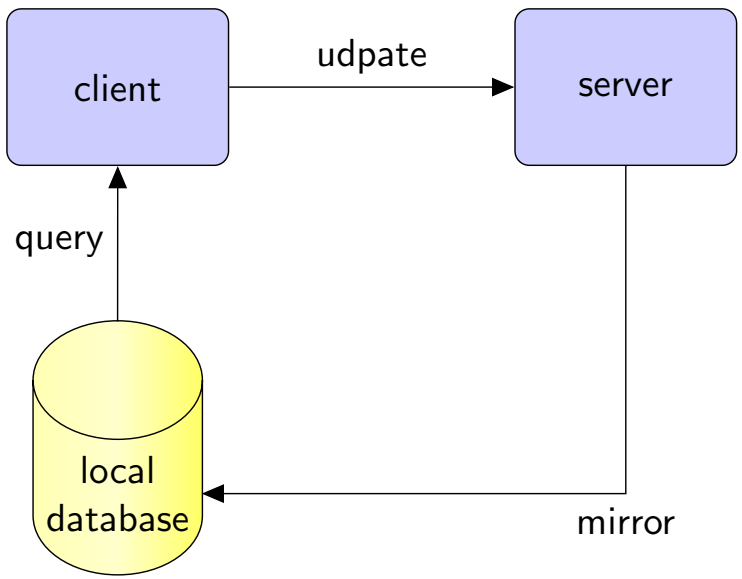
- Simple.
- A uniform way for accessing the database is needed to allow arbitrary clients to interface independently.
- Signatures can be calculated by any client and don't have any [unreasonable] dependencies.

How does the client work?

How does the mobile client work.

The entire database is mirrored locally.

- Only dependent on network for posting changes
- Faster query times than Window or Cache could ever reach
- Simple to understand



How does the client know what to request?


```
[
{"tag":"CAPS05654",
"serial":"00020300105608 4",
"model":"TCH-17",
"contract number":"none",
"hostname":"USR-Orono1",
"description":"
    TotalControl17"
},
{"tag":"CAPS05407",
"serial":"72729663",
"model":"7206VXR",
"contract number":"none",
"hostname":"GW-UMF",
"description":"6.SlotChassis
    "
}
]
```

```
create table Exercises(
  exercise VARCHAR(30)
      PRIMARY KEY,
  mod_ts DATETIME , --
      metacolumn
  deleted INTEGER  --
      metacolumn
);

create table Sets(
  name VARCHAR(30),
  exercise VARCHAR(30),
  set_num INTEGER ,
  reps_done INTEGER ,
  weight INTEGER ,
  date_done DATE,
  mod_ts DATETIME, --
      metacolumn
  deleted INTEGER --
      metacolumn
);
```

Motivation

- Coexist was designed with mobile clients in mind, and offline-readability as a feature.
- All queries can be performed locally and avoid network bottleneck.
- After initial sync, only missing rows are ever downloaded.

How does deployment work?

Tentatively, a conf file must be filled out. I have made a tool that will download and compile the latest version of Coexist and its clients according to this.

```
; Android stuff
name=Workoutlog
image=logo.png
notification=notification.png
package=com.domain
api=http://domain.com:/api/

; Server stuff
version=1
user=
pass=
db=
host=localhost
dbms=mysql

create_dir=ui
schema_dir=sql
```

What needs to be done?

- Add more client types (Qt, Desktop Java etc.)
- Support more client features (i.e. Barcode scanning on mobile clients.)
- Create tools to automate the configuration.

I would like to thank

My advisor, Sudarshan Chawathe