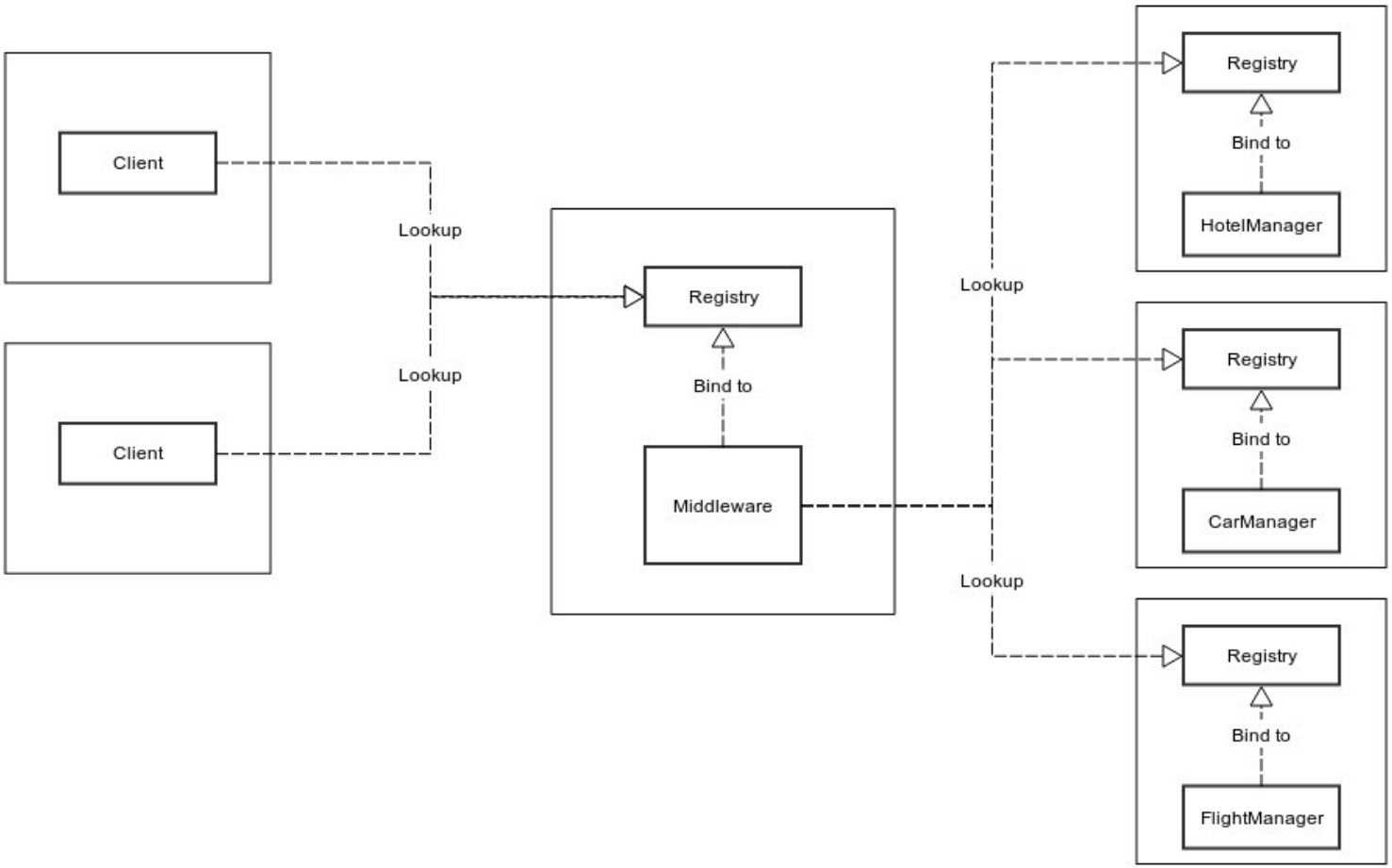
Outline

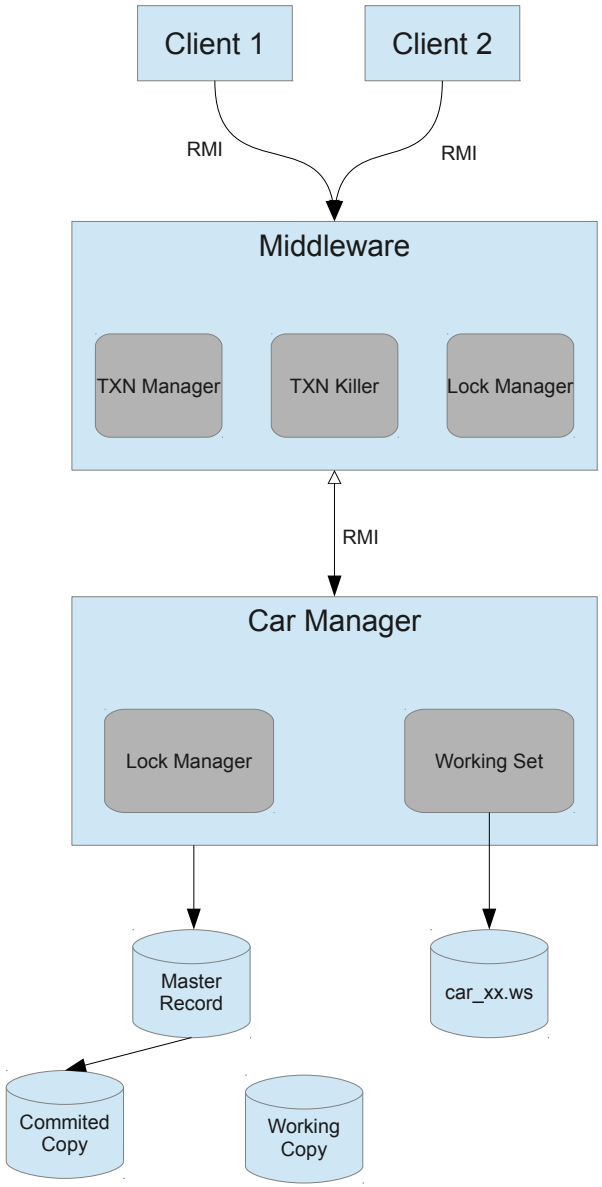
Overall architecture

RMI Architecture



(Small difference if a resource manager is in recovery mode)

General architecture



Implementation details

Persistence & Shadowing

/tmp/Group5/ contains master records and databases.

- *flightdb.0, flightdb.1*: copies of the flight[1] data items
- *flightdb.mr*: points to the commited version

When a resource manager is started, if the master record and the file it points to exist, data is loaded from disk.

At every commit, data is saved to the file not pointed to by the master record and `committed := 1 - committed`

[1] Mutatis mutandis for other resources

Working Set

Three hash tables:

- XID $\rightarrow$ Vector$\langle$Command$\rangle$
- XID $\rightarrow$ Vector$\langle$Item Description$\rangle$
- Item Description $\rightarrow$ Modified Item

Writes are stored in the first table and applied to the copy.
Reads are done against the modified item.
To commit: apply commands
To abort: do nothing

Saved to disk before sending vote.

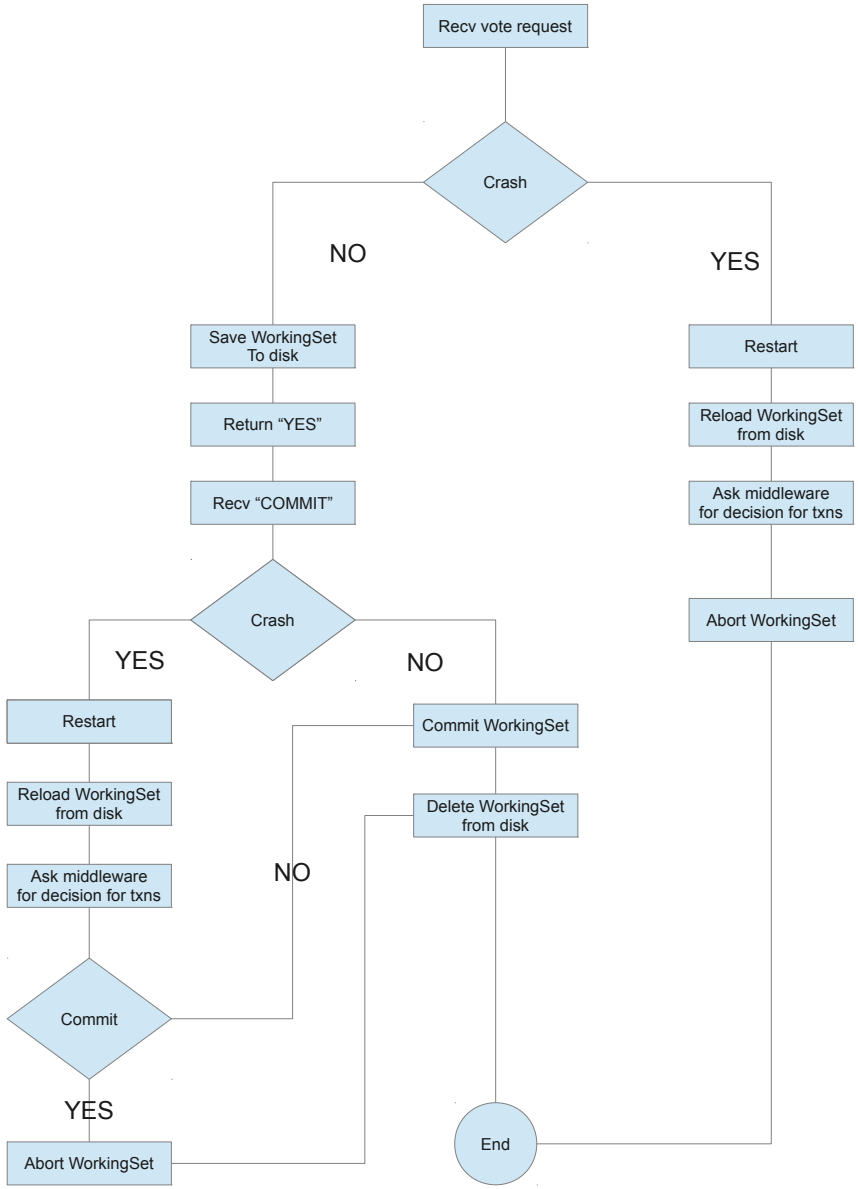
Transaction Manager

Four hash tables:

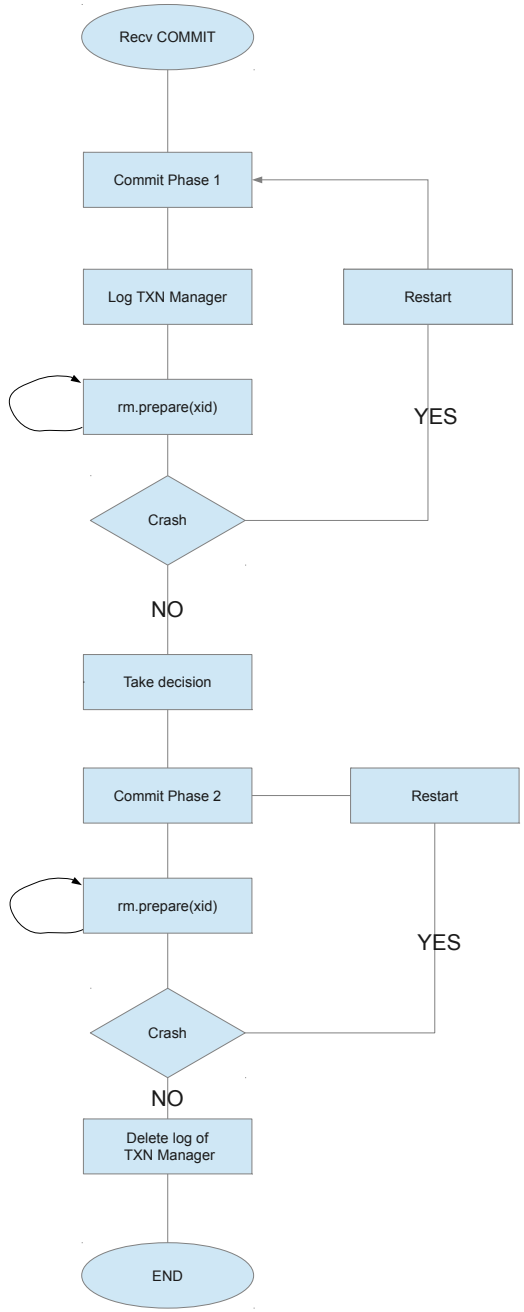
- XID $\rightarrow$ (NOTCOMMITTED $\|$ PHASE1 $\|$ PHASE2)
- XID $\rightarrow$ Vector$\langle$Involved RM$\rangle$
- XID $\rightarrow$ TTL
- XID $\rightarrow$ (COMMIT $\|$ ABORT)

In charge of 2PC
Logged to disk during 2PC for recovery

Two-Phase Commit (Participant view)



Two-Phase Commit (Middleware view)



Using the system

Makefile system

The different components are started with a set of *make* rules:

Make rule	Description
make compile	(Re)compile the whole system
make runcar [2]	Start a rmiregistry instance and the car manager
make recovercar	Restart the car manager, providing link to middleware
make runserver	Start a rmiregistry instance and the middleware
make recoverserver	Restart the middleware in recovery mode
make runclient	Start a client and connect to the middleware

[2] Idem for flight and hotel managers

Crash commands

To crash the middleware or a resource manager, extra commands have been added to the client.

crash,⟨condition⟩,⟨manager⟩

Examples:

```
# Crash the car manager after
# saving the working set
> crash,P_A_SAVEWS,car

# Crash the transaction manager after
# all vote requests have been sent
> crash,C_A_ALLREPLY,tm
```

Demo

Demo

Persistence

```
$ make populate
$ ls /tmp/Group5

> shutdown

# Restart severs

> start
> querycar,xxx,city999
```

Demo

Shadowing

```
$ ls /tmp/Group5  # Only .mr and .1 files

> start
> itinerary,xxx,11,11,city11,true,true
> commit,xxx

$ ls /tmp/Group5  # Now with .0 files
```

Demo

Crash car before saving working set

```
> crash,P_B_SAVEWS,car
> start
> itinerary,xxx,22,22,city22,true,true
> commit,xxx   # Boom

$ make recovercar

> start
> querycar,xxx,city22  # 1000 cars
> abort,xxx
```

Demo

Crash flight after saving working set

```
> crash,P_A_SAVEWS,flight
> start
> itinerary,xxx,33,33,city33,true,true
> commit,xxx   # Boom

$ ls /tmp/Group5/*.ws
$ make recoverflight

> start
> queryflight,xxx,33  # 1000 seats
> abort,xxx
```

Demo

Crash hotel after sending vote

```
> crash,P_A_COMMITRECV,hotel
> start
> itinerary,xxx,44,44,city44,true,true
> commit,xxx   # Boom

$ make recoverhotel

> start
> queryroom,xxx,city44  # 999 rooms!
> abort,xxx
```

Demo

Crash middleware during phase 1

```
> crash,C_A_ALLREPLY,tm
> start
> itinerary,xxx,55,55,city55,true,true
> commit,xxx   # Boom

$ make recoverserver

> start
> querycustomer,xxx,55  # Everything's there!
> abort,xxx
```

Demo

Crash middleware during phase 2

```
> crash,C_A_ALLCOMMIT,tm
> start
> itinerary,xxx,66,66,city66,true,true
> commit,xxx   # Boom

$ make recoverserver

> start
> querycustomer,xxx,66  # Everything's there!
> abort,xxx
```