#GeeCON

# What do you mean, backwards compatibility?

Trisha Gee
Java Driver Developer, 10gen
@trisha_gee

10gen | the MongoDB company

mongoDB

**Design**: translate the requirements in a specification that describes the global architecture and the functionality of the system.

http://homepages.cwi.nl/~paulk/patents/isnot/node4.html

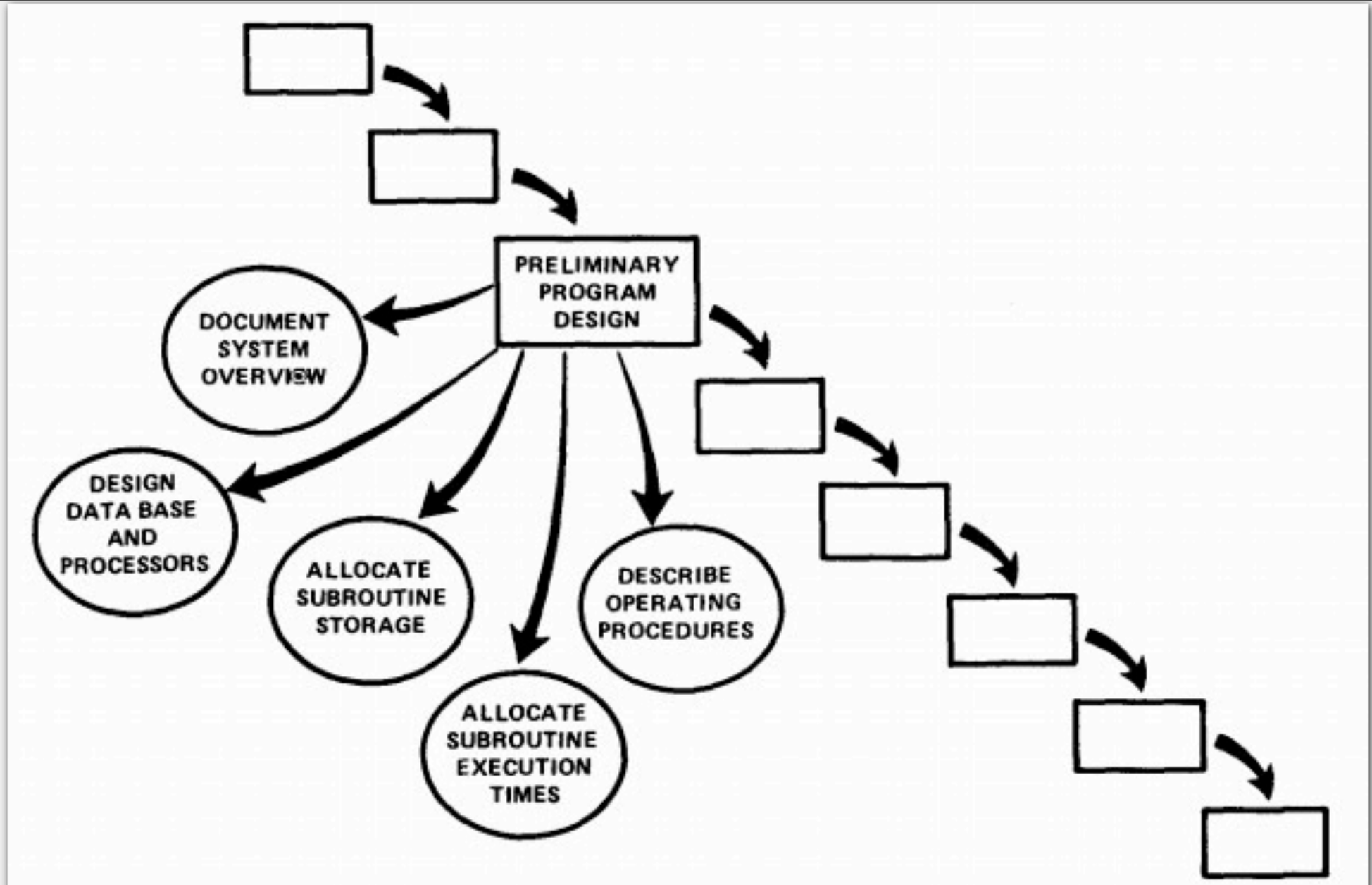



Figure 5. Step 1: Insure that a preliminary program design is complete before analysis begins.

# Managing the Development of Large Software Systems - Dr Winston Royce

http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf

<This Page Left Intentionally Blank>

# Agile Design

# Design is a Process, not a Document

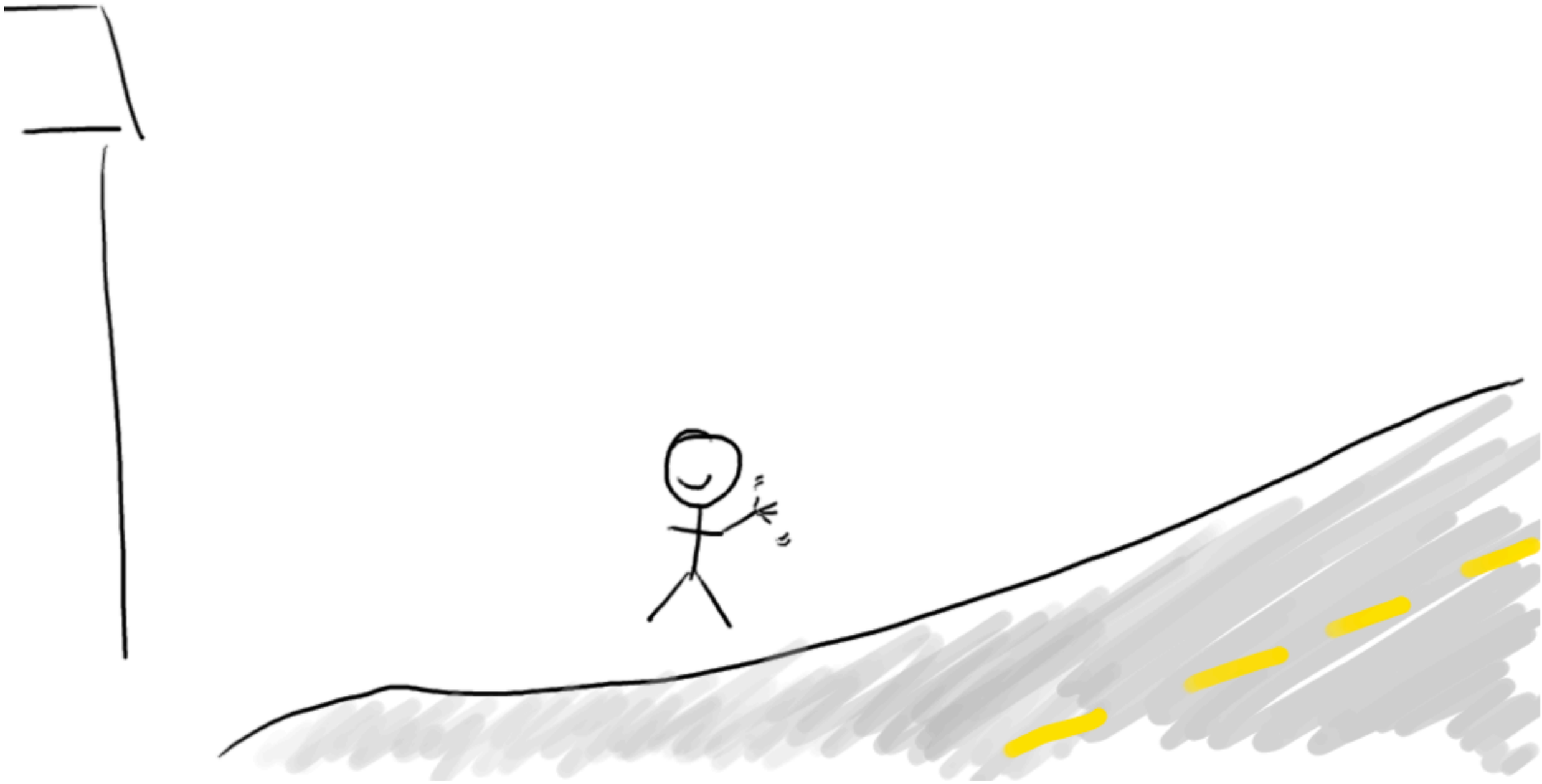
# What are you saying?

- Design is a journey, enjoy the ride
- There will be Monsters
- There will be Safe Houses
- There might not even be a destination...

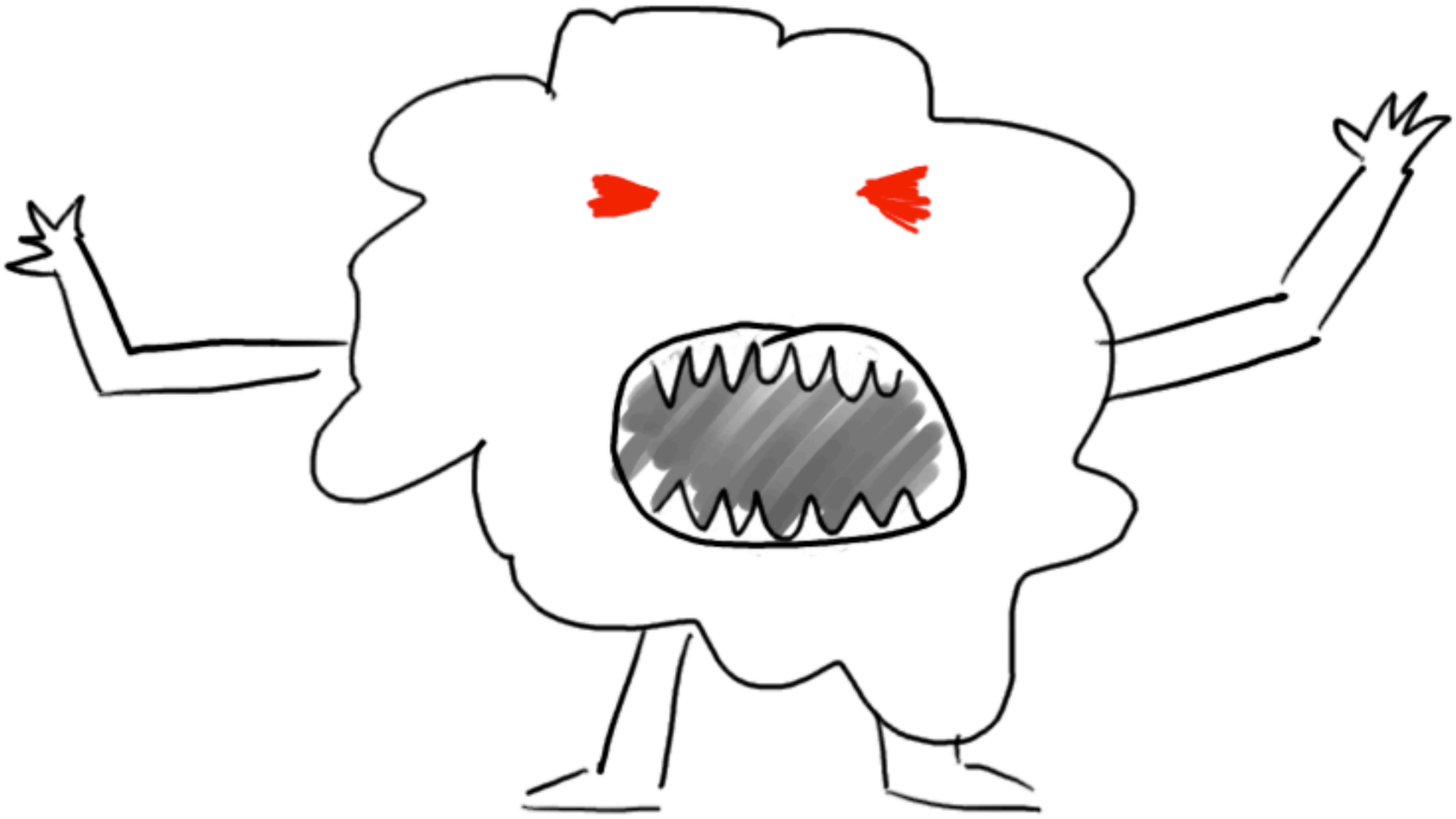
# Best Job Evar!!

10gen | the MongoDB company

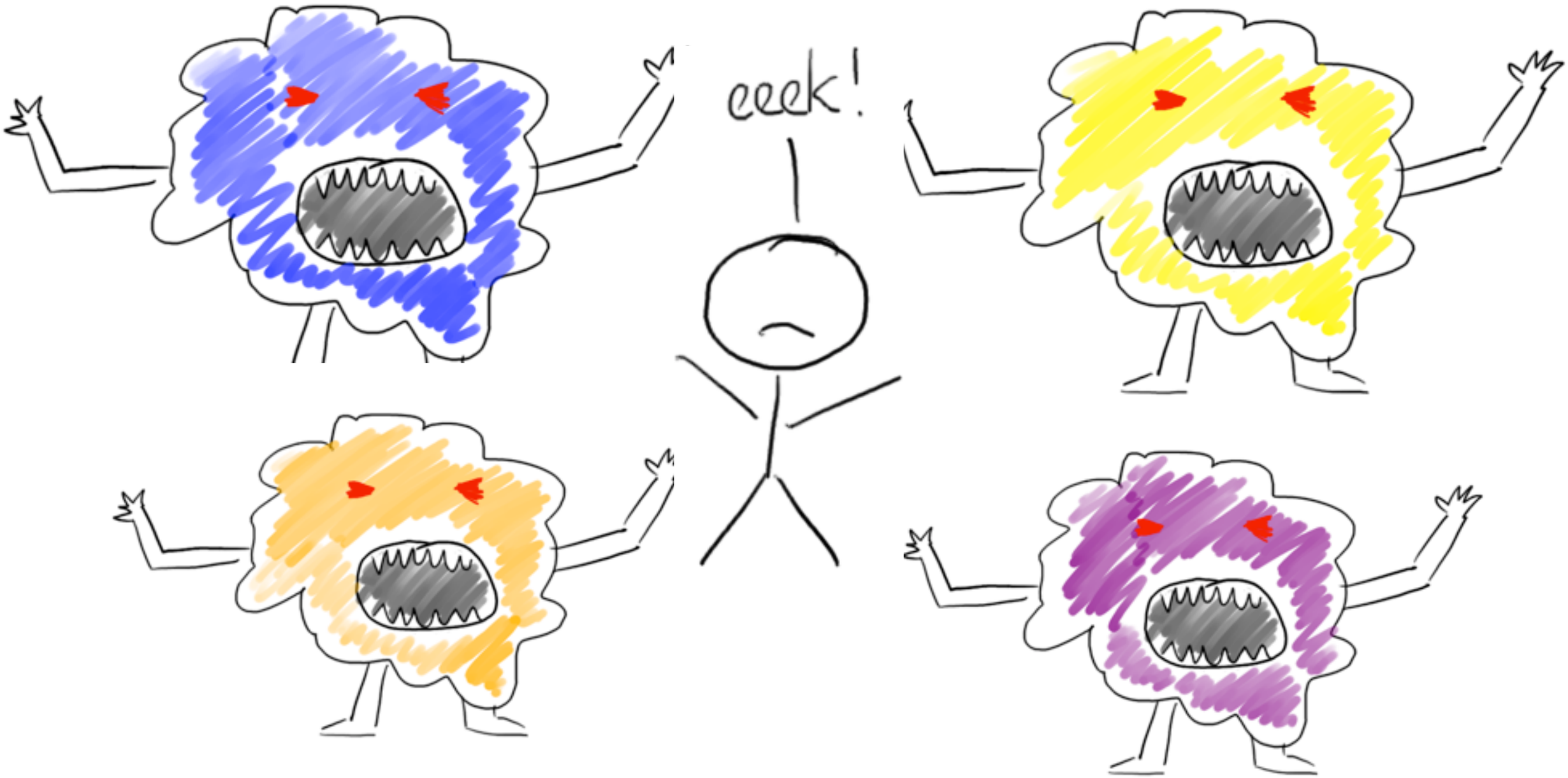
mongoDB



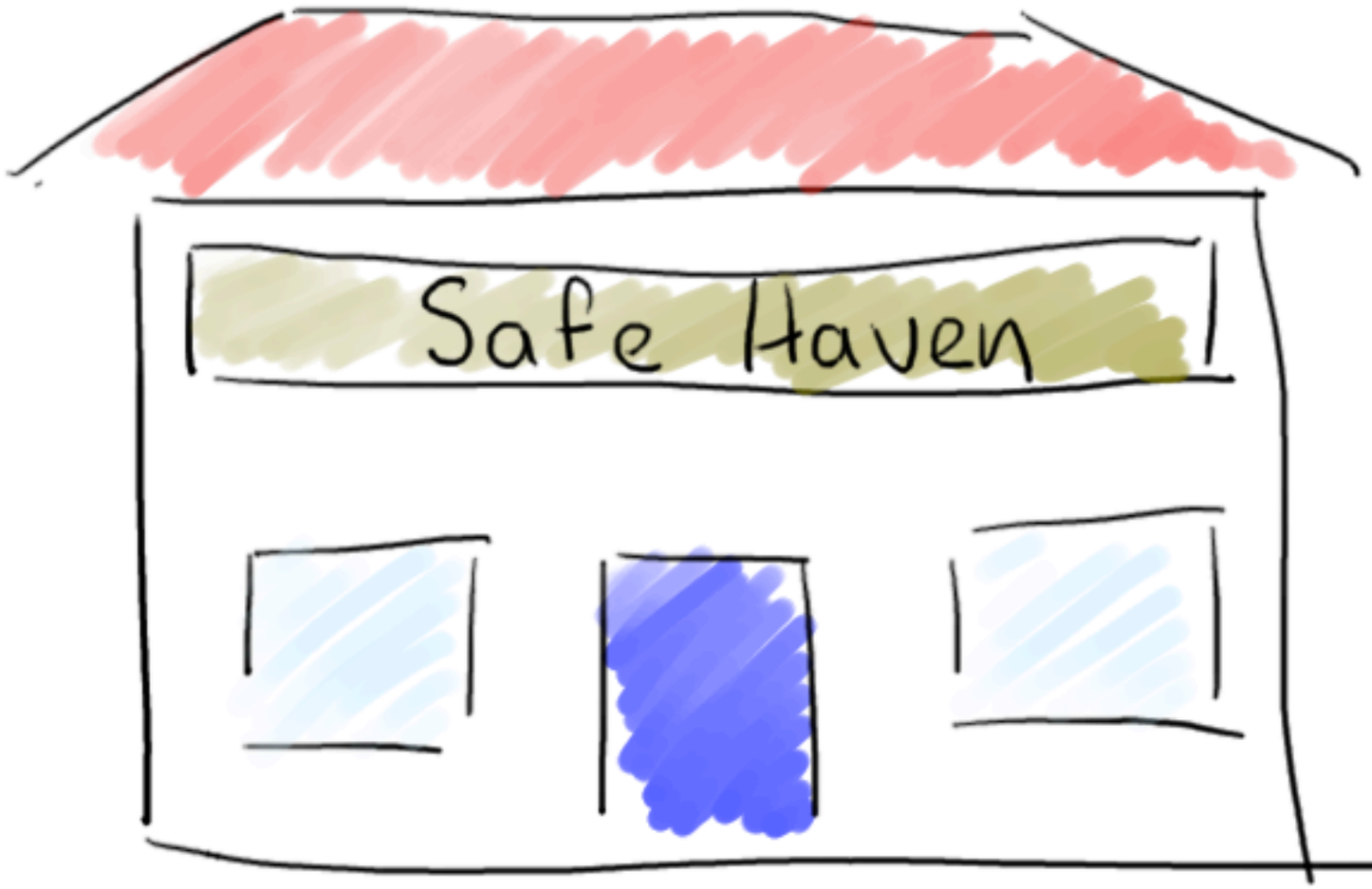
# We’re off!



# Backward Compatibility




# Lots of unknowns




# Design Goals

# Java Driver 3.0 Goals/Design



File Edit View Insert Format Tools Table Help Last edit was made on December 13, 2012 by jeff.yemin

Normal text | Arial | 11

a. Currently there are no integration tests that cover mongos failover, authentication, replica set failover, etc.
b. Distinguish between unit tests and integration tests
    i. There's quite a bit now that that can be tested without a running Mongo:
        1. ReadPreference, WriteConcern, QueryBuilder, etc.

9. Judicious use of interfaces
    a. Examples: MongoClient, MongoDatabase, MongoCollection. This will make it easier to create multiple implementations that have no relationship in a class hierarchy, and allow mocking for those who thinks that it's a good idea
10. Exception hierarchy
    a. Example: Distinguish between exceptions generated due to an error from the server and an error on the client. They should be types for each.
11. Prefer immutability
    a. Example: MongoOptions
12. Replace non-constant static fields with either
    a. static methods
        i. ReadPreference.PRIMARY => ReadPreference.primary()
    b. fields of other classes
        i. BSON._encoding/_decodingHooks become fields of MongoClient
13. Prefer SocketChannel to Socket and use direct ByteBuffer

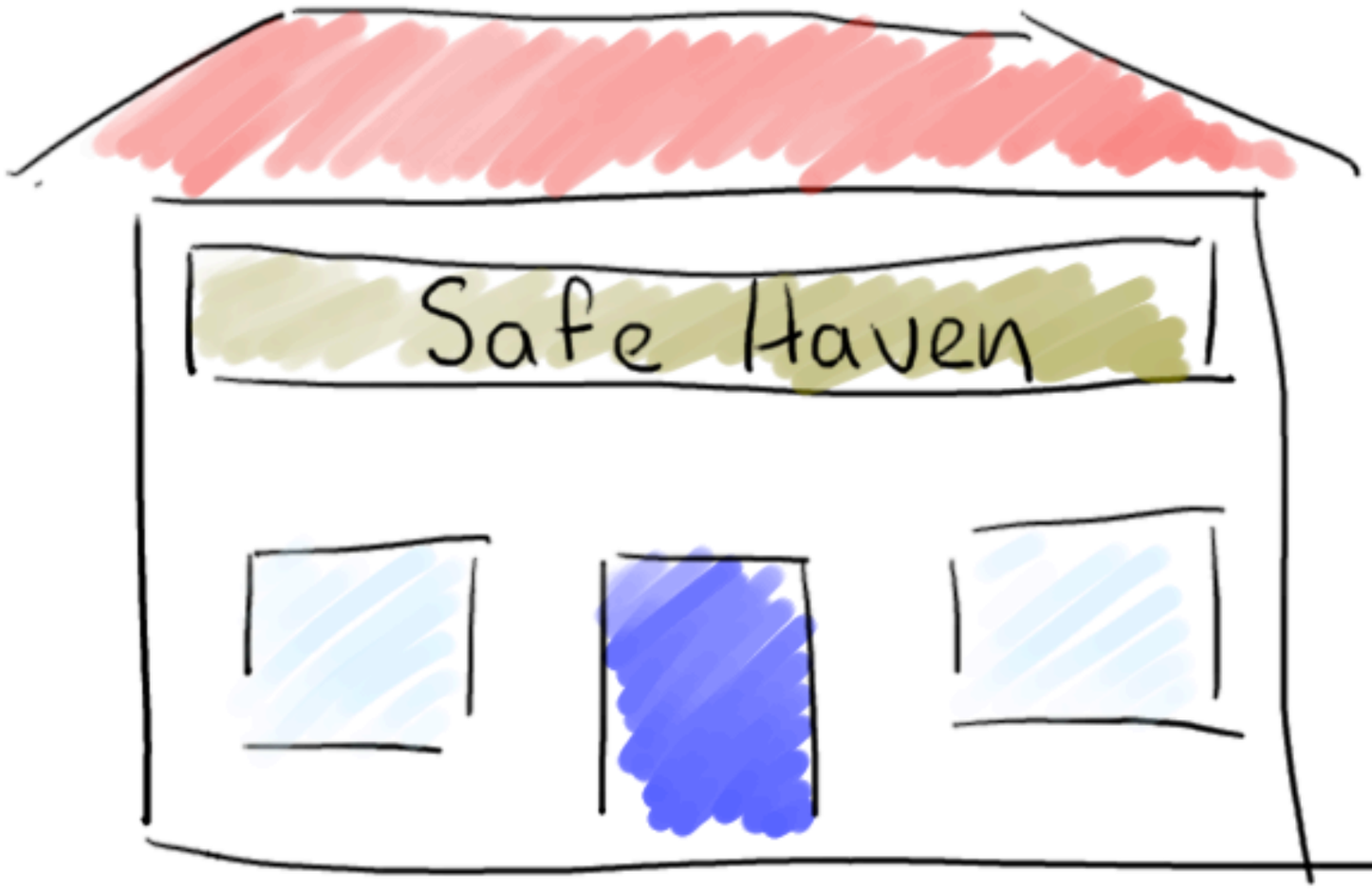
# Yes, it's a document

something we can add in later in a minor release.

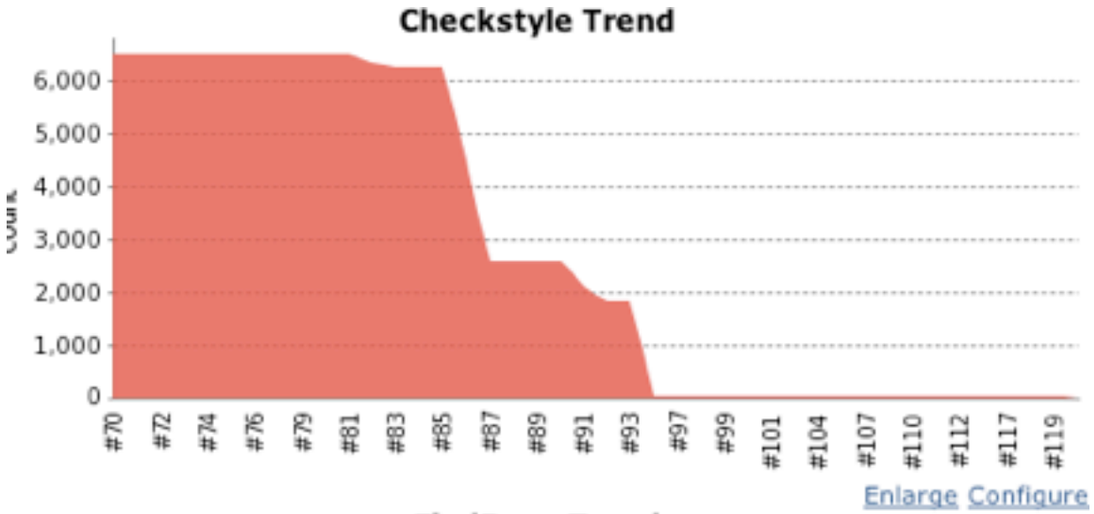
# Design Goals

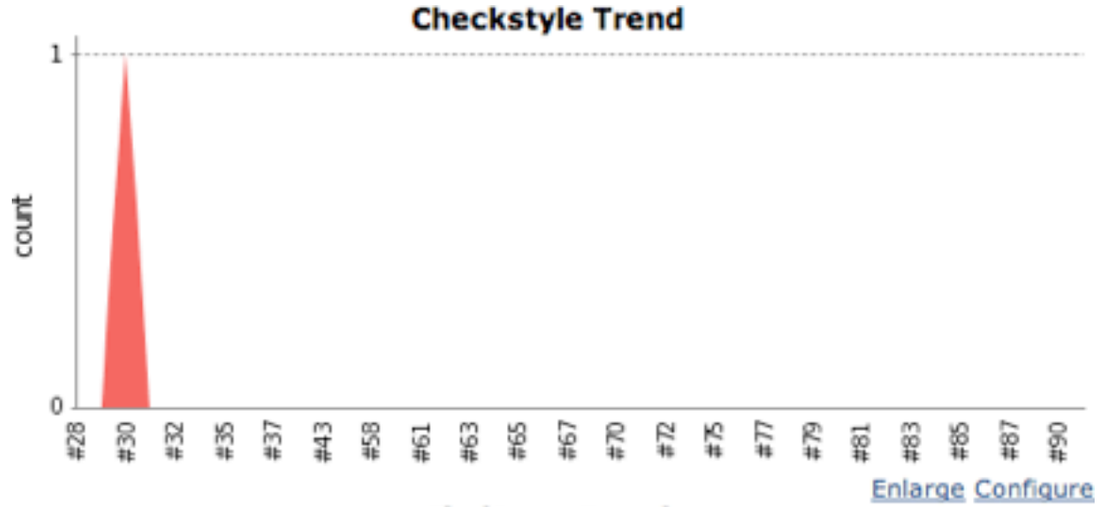
- Consistency
- Cleaner design
- Intuitive API
- Sane Exception handling
- Test friendly
- Backwards compatible

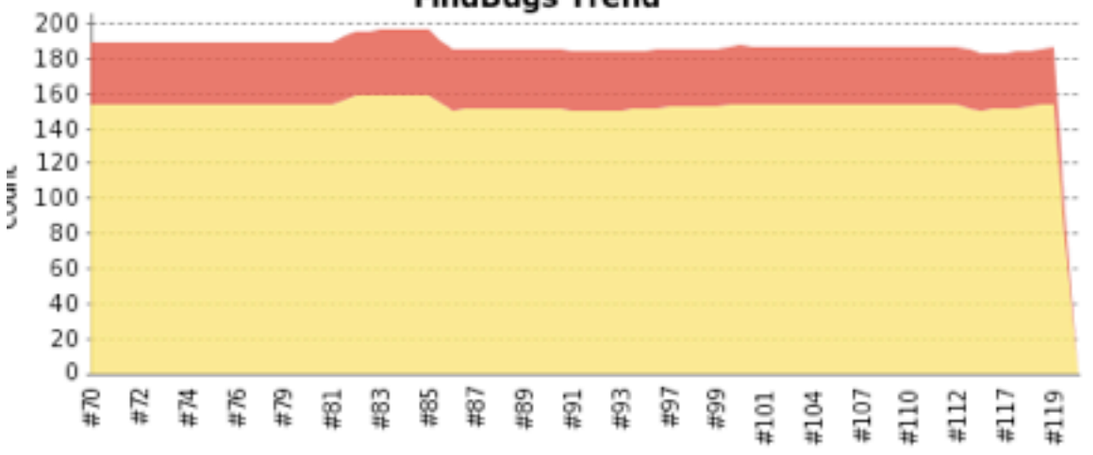
# Lack of consistency

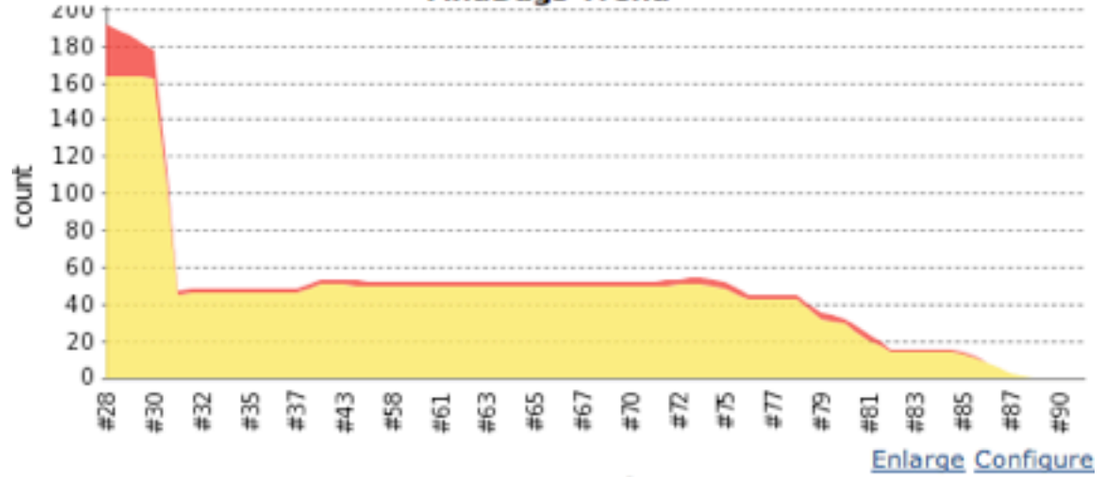


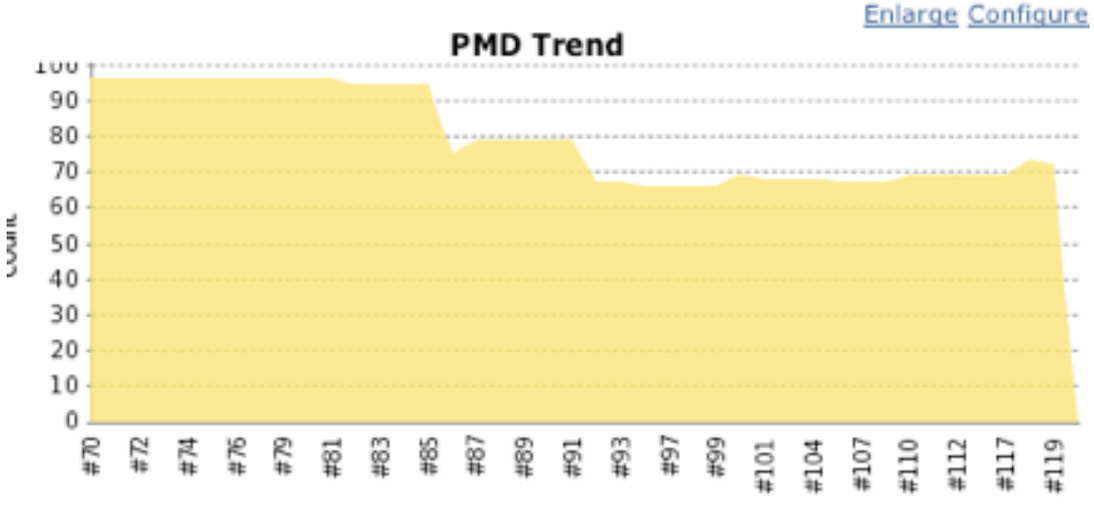

# Coding Standards

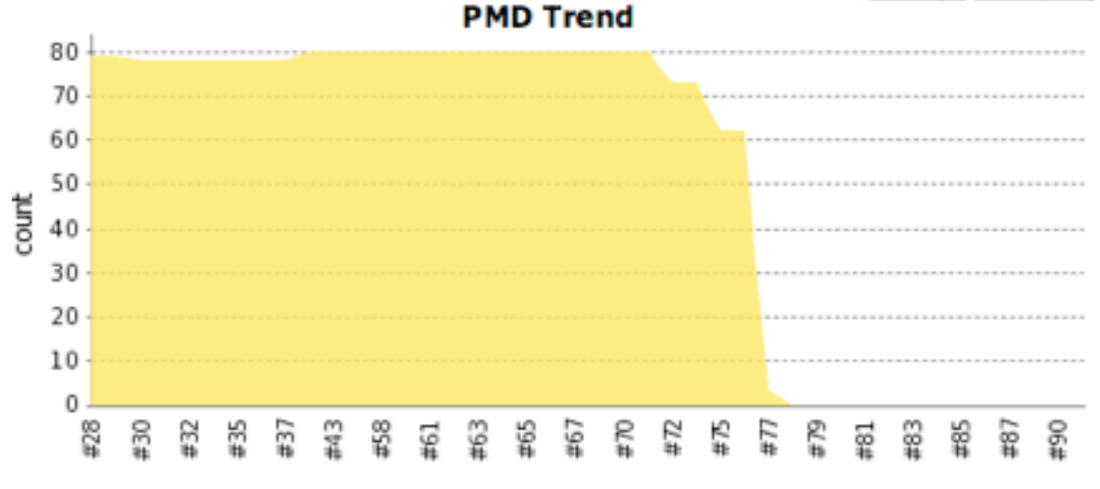


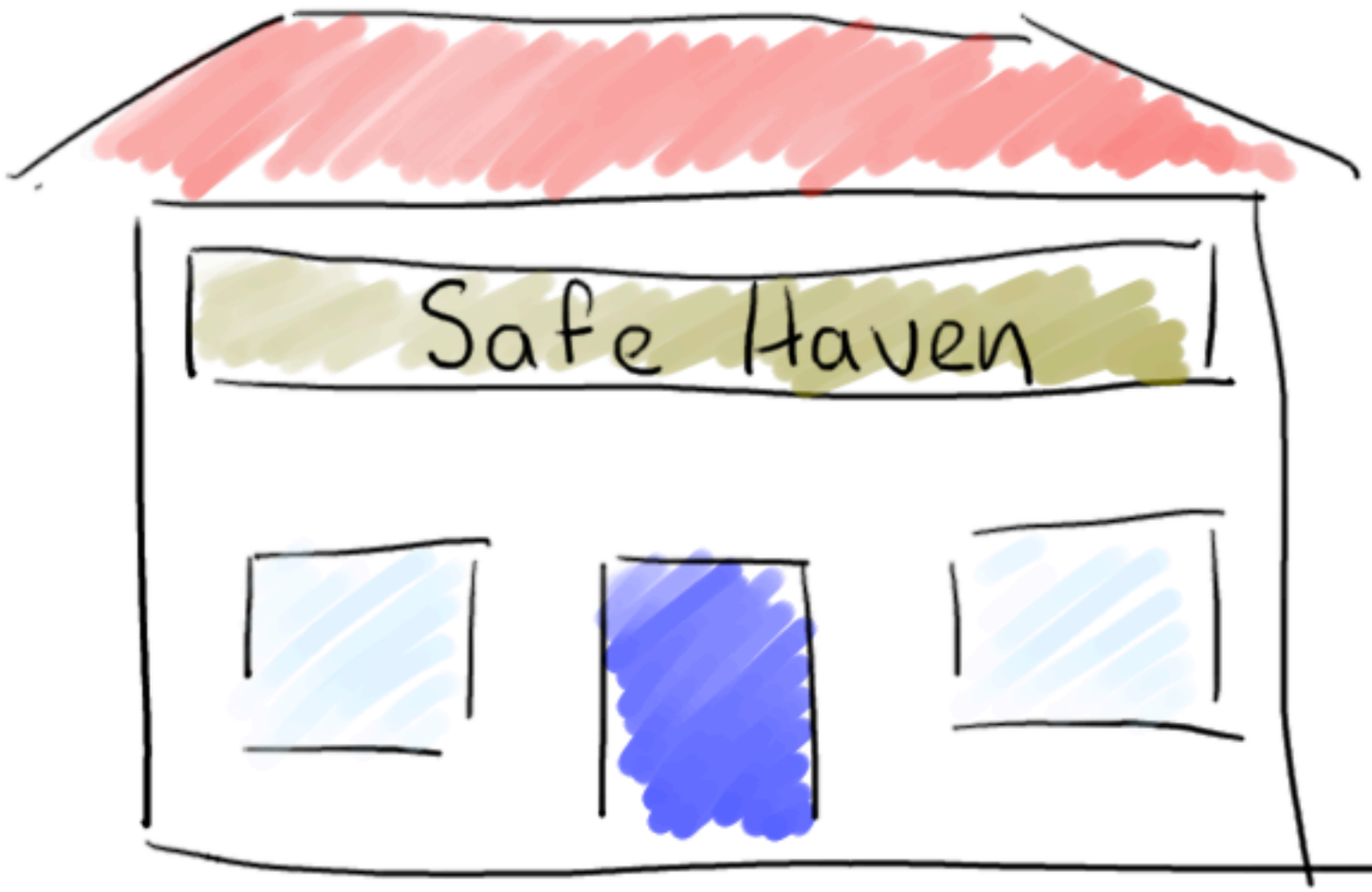

# Zero Analysis Errors

# No more arguments

# Design Goals

- ✓**Consistency**
- Cleaner design
- Intuitive API
- Sane Exception handling
- Test friendly
- Backwards compatible

# Users




# Identify Our Users

# Three Types Of Users

1. Java Developers
2. ODMs / other drivers / third parties
3. Contributors

# Java Developers

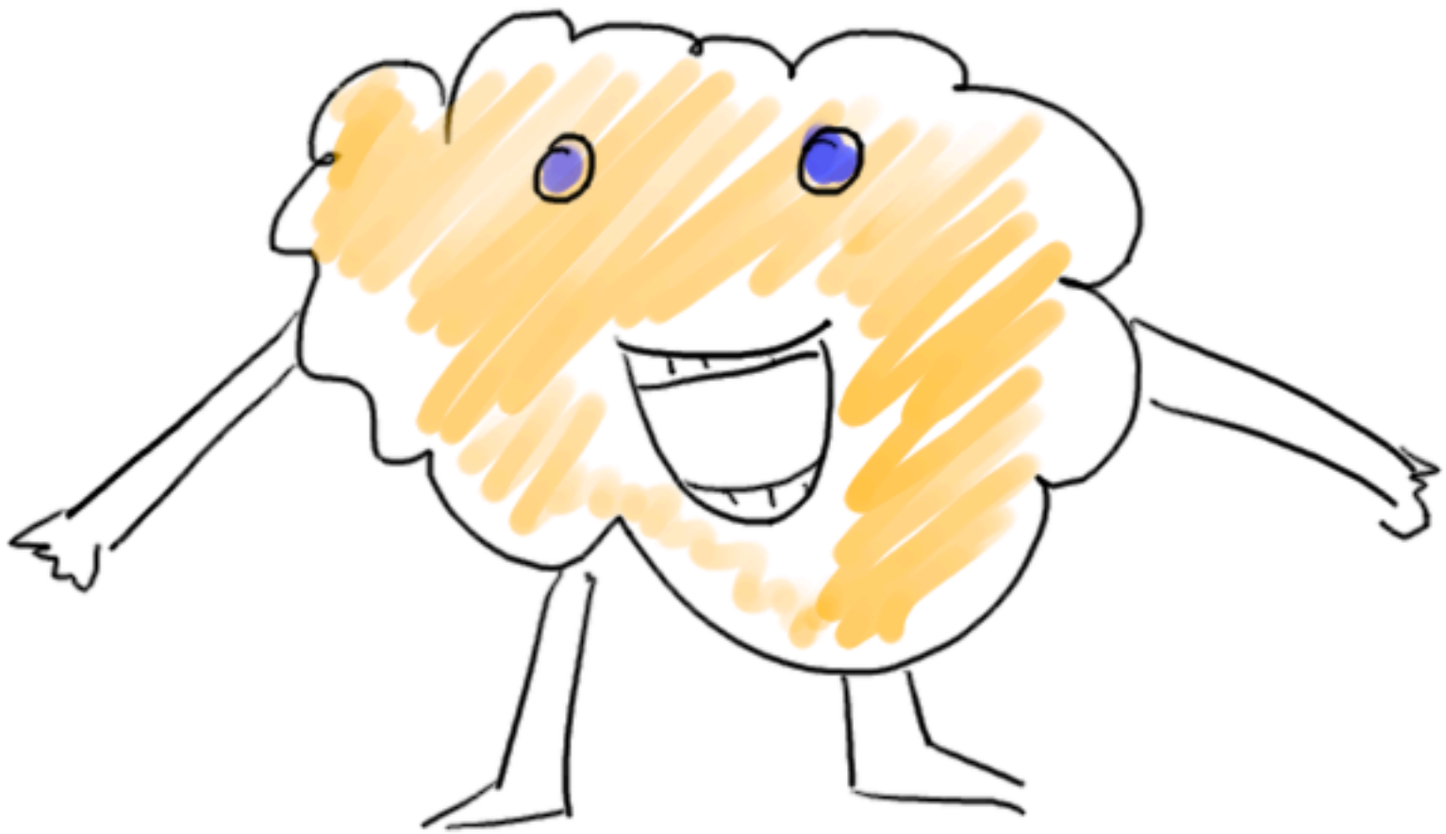
- Consistency
- Cleaner design
- **Intuitive API**
- **Sane Exception handling**
- **Test friendly**
- **Backwards compatible**

# Third Party Libraries

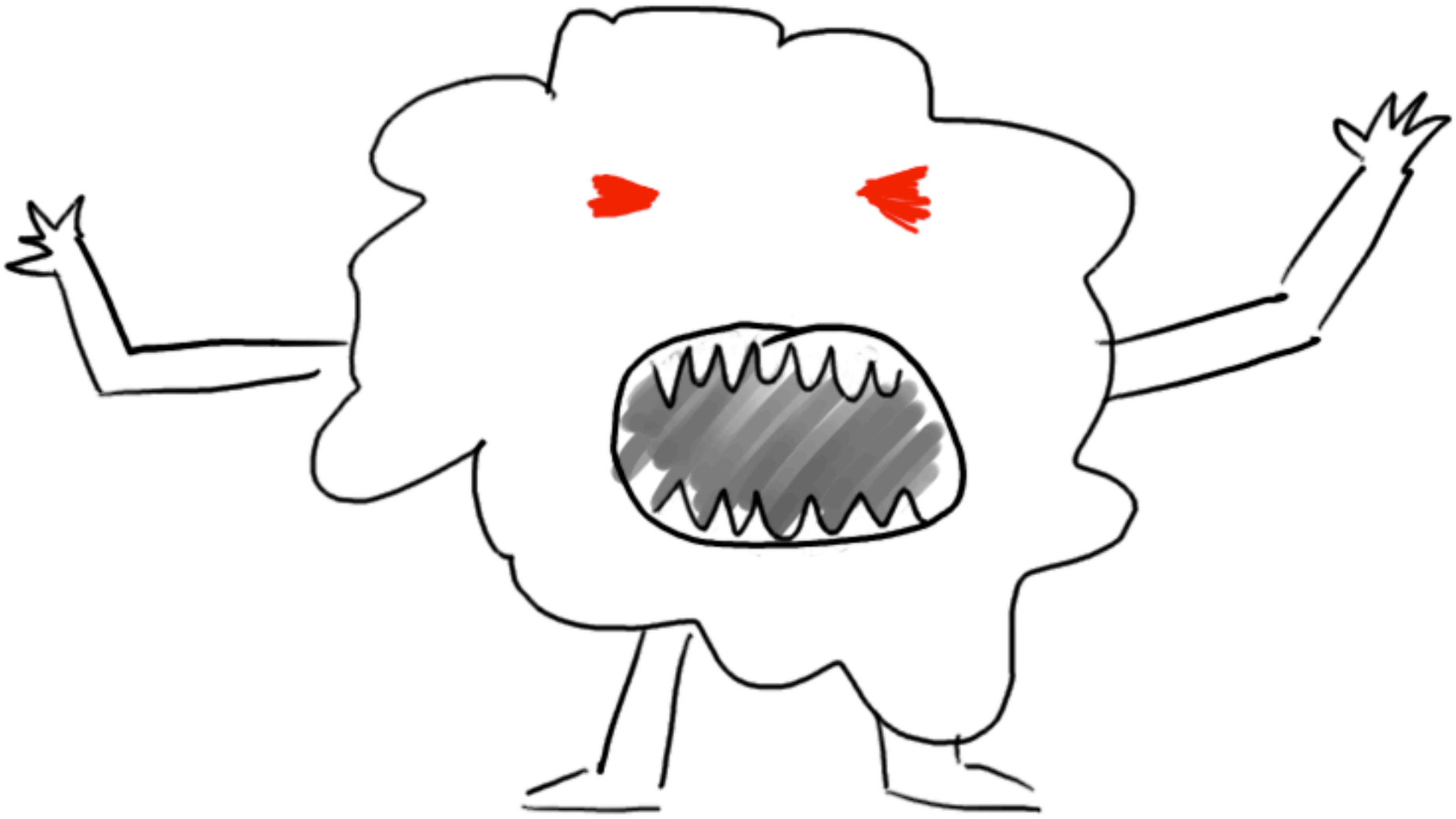
- **Consistency**
- **Cleaner design**
- Intuitive API
- Sane Exception handling
- **Test friendly**
- **Backwards compatible**

# Contributors

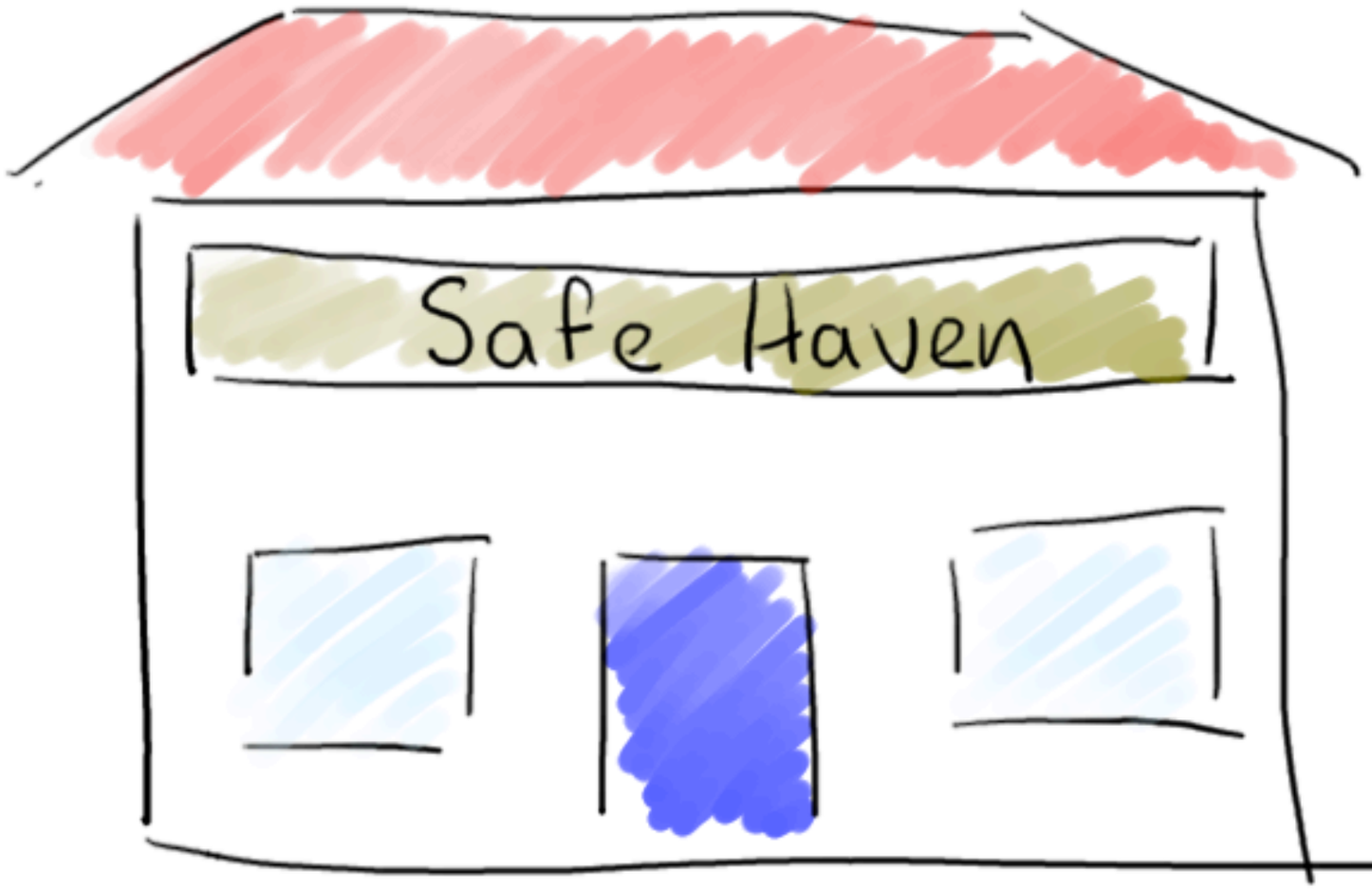
- **Consistency**
- **Cleaner design**
- Intuitive API
- **Sane Exception handling**
- **Test friendly**
- Backwards compatible



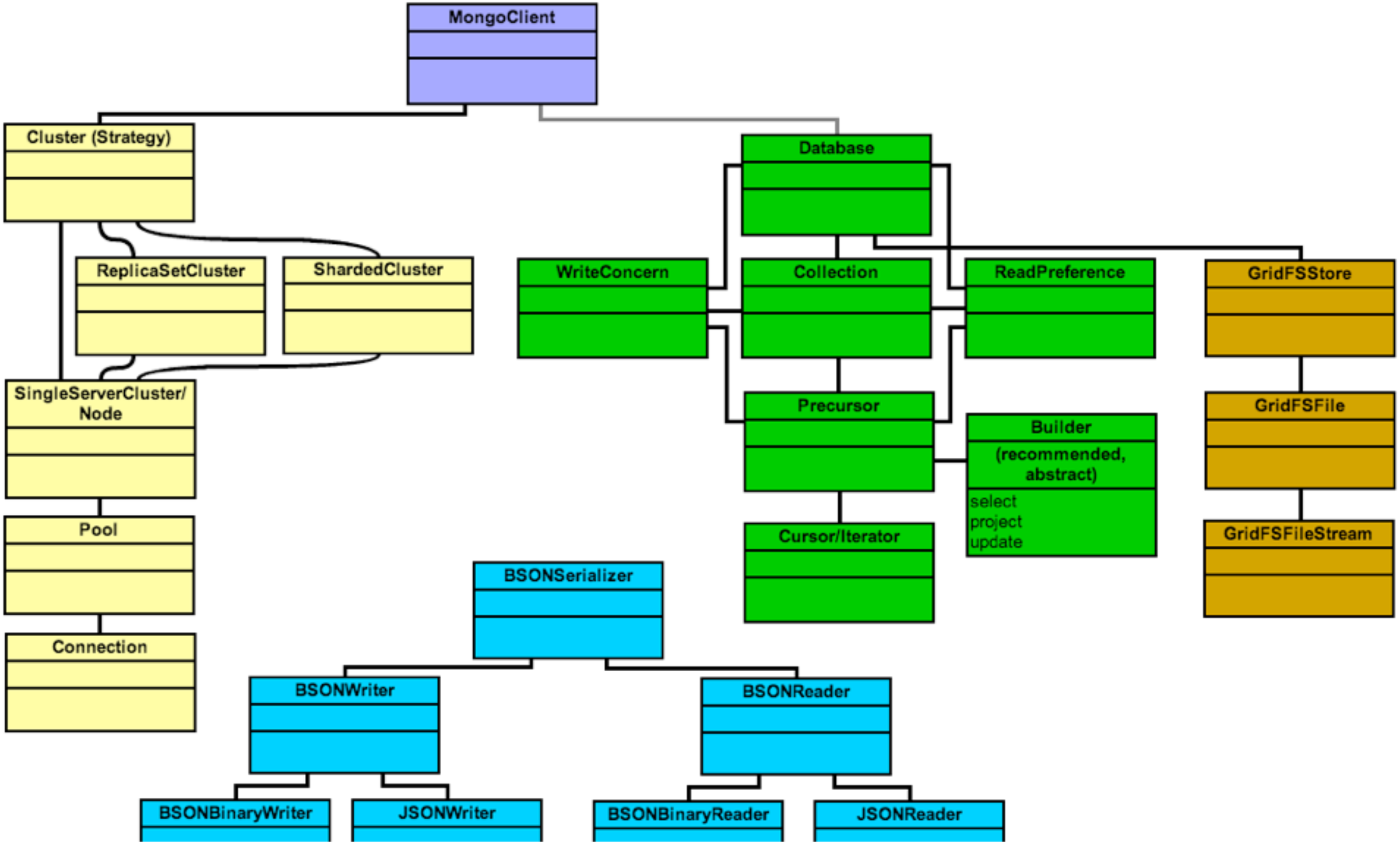
# Users are our friends



# Backward Compatibility



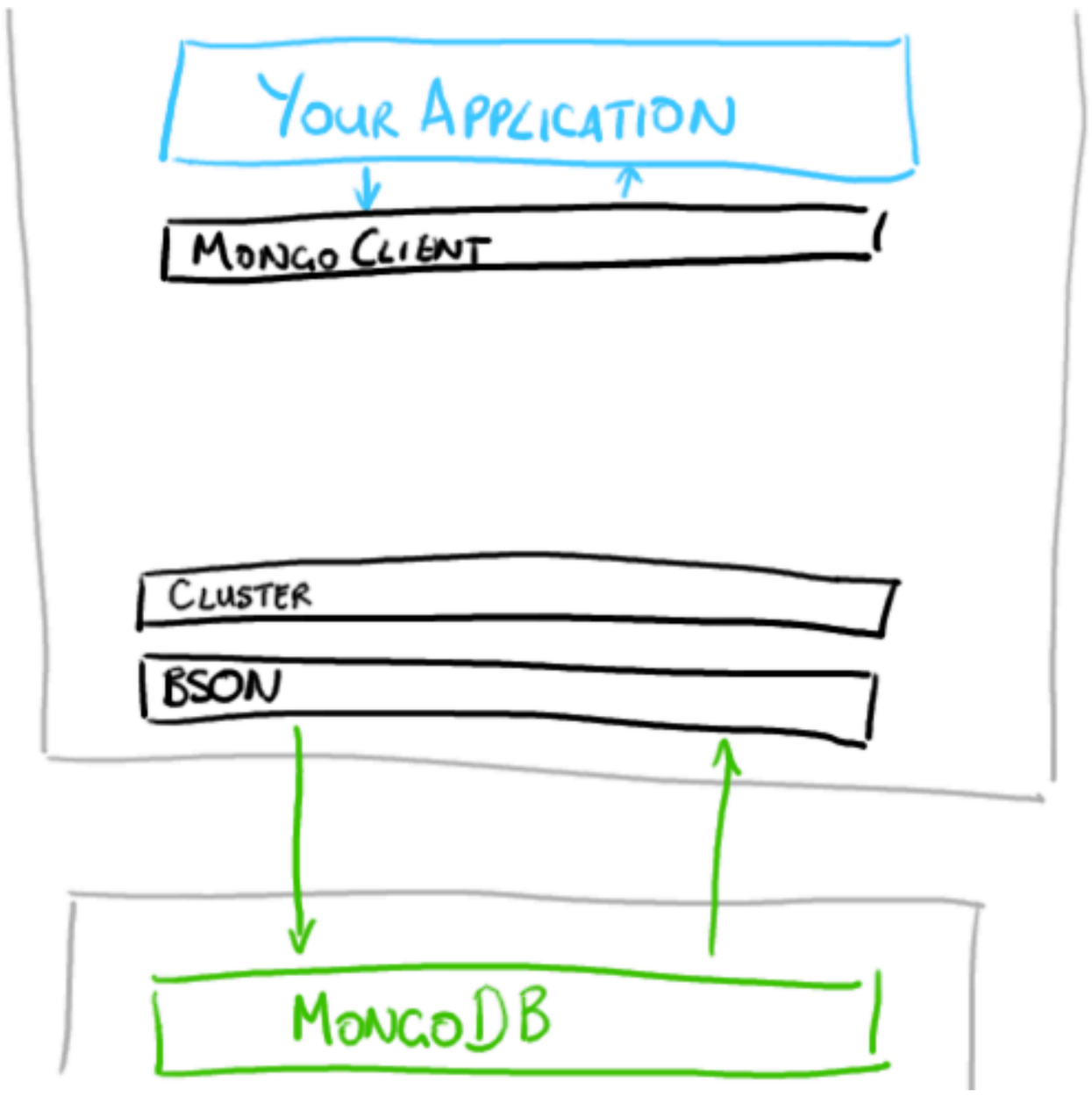

# Architecture



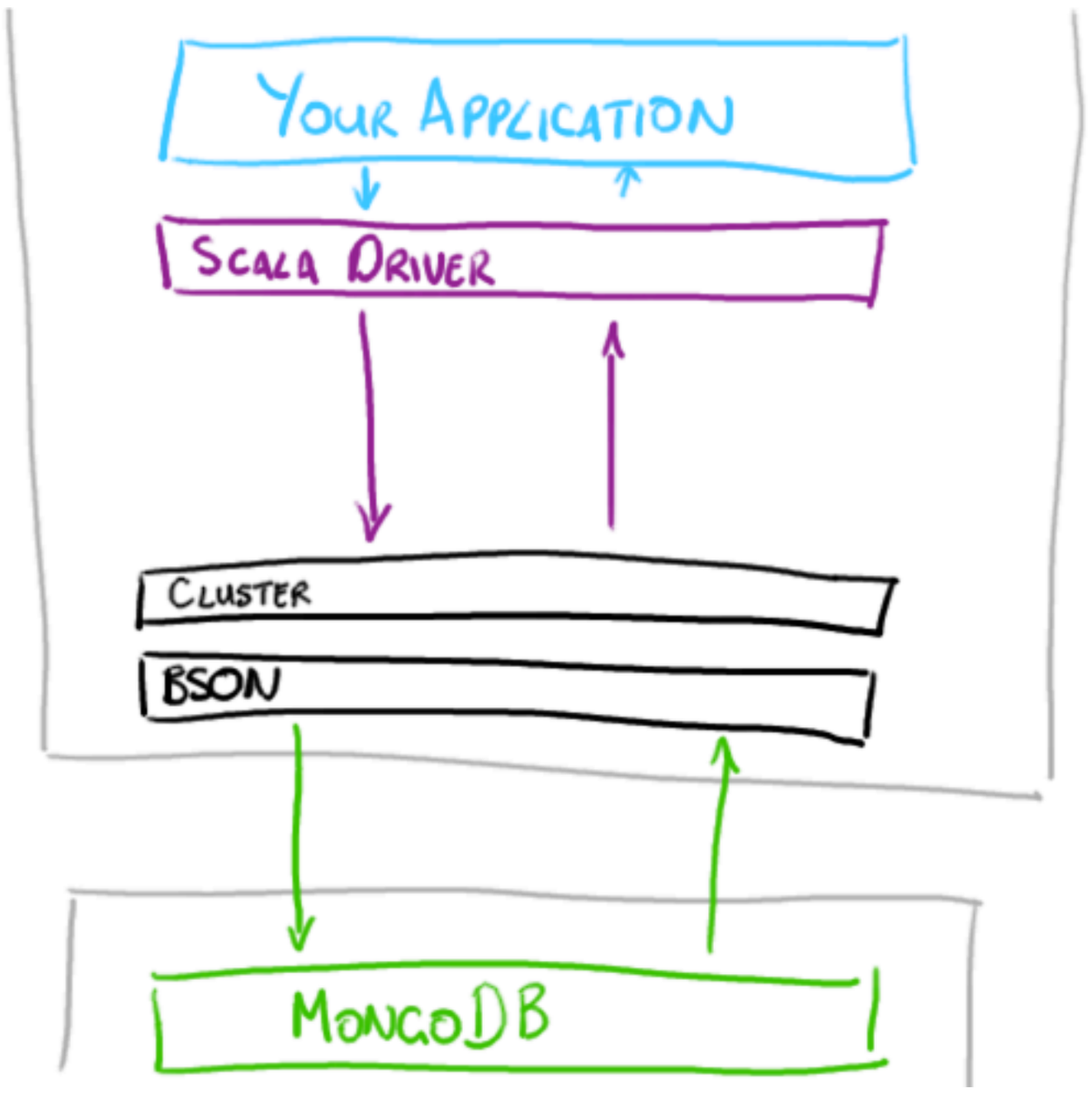

# UML, yuk!

create and

gliffy




# High Level Architecture



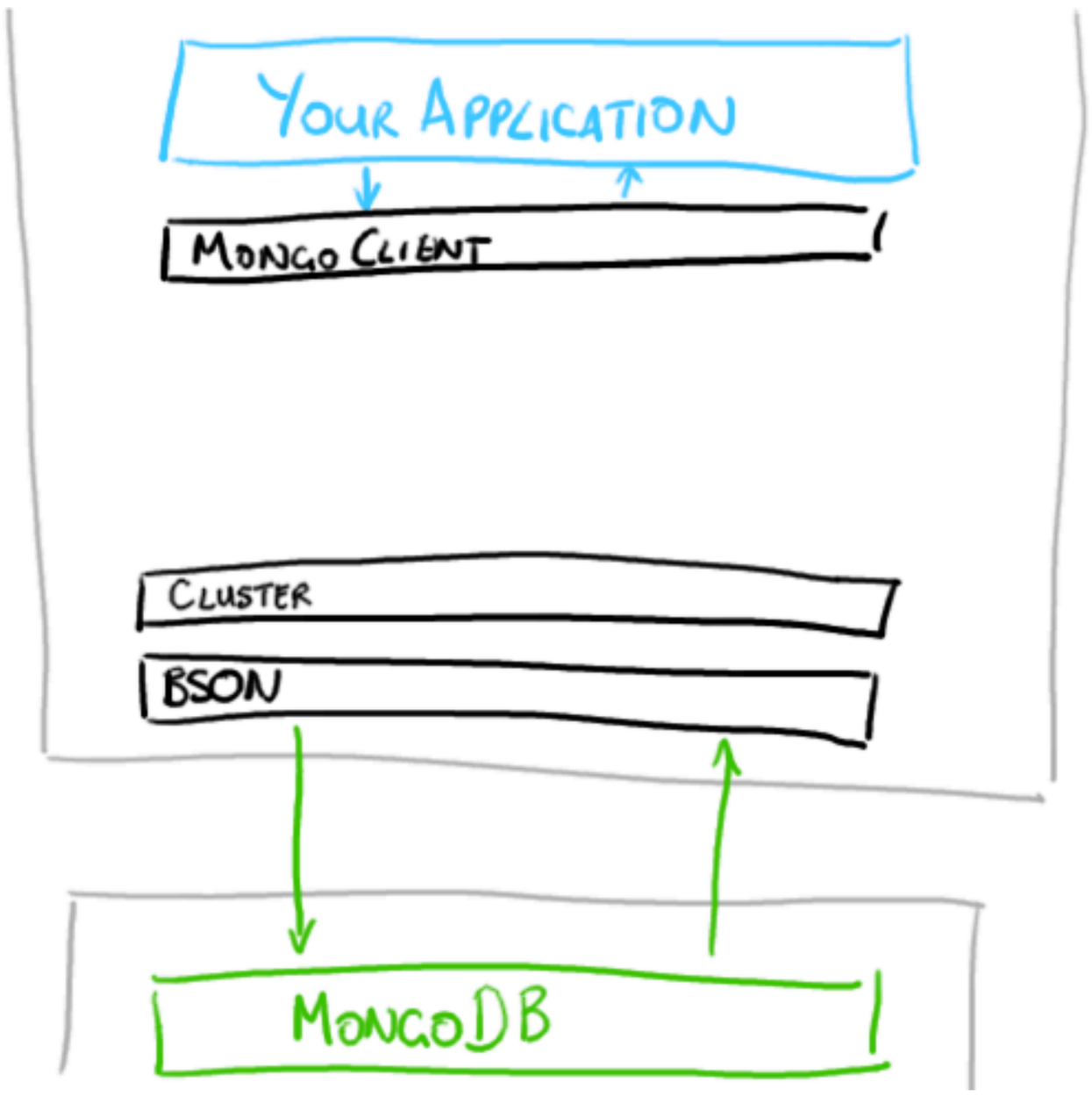

# Scala Driver

# Design Goals

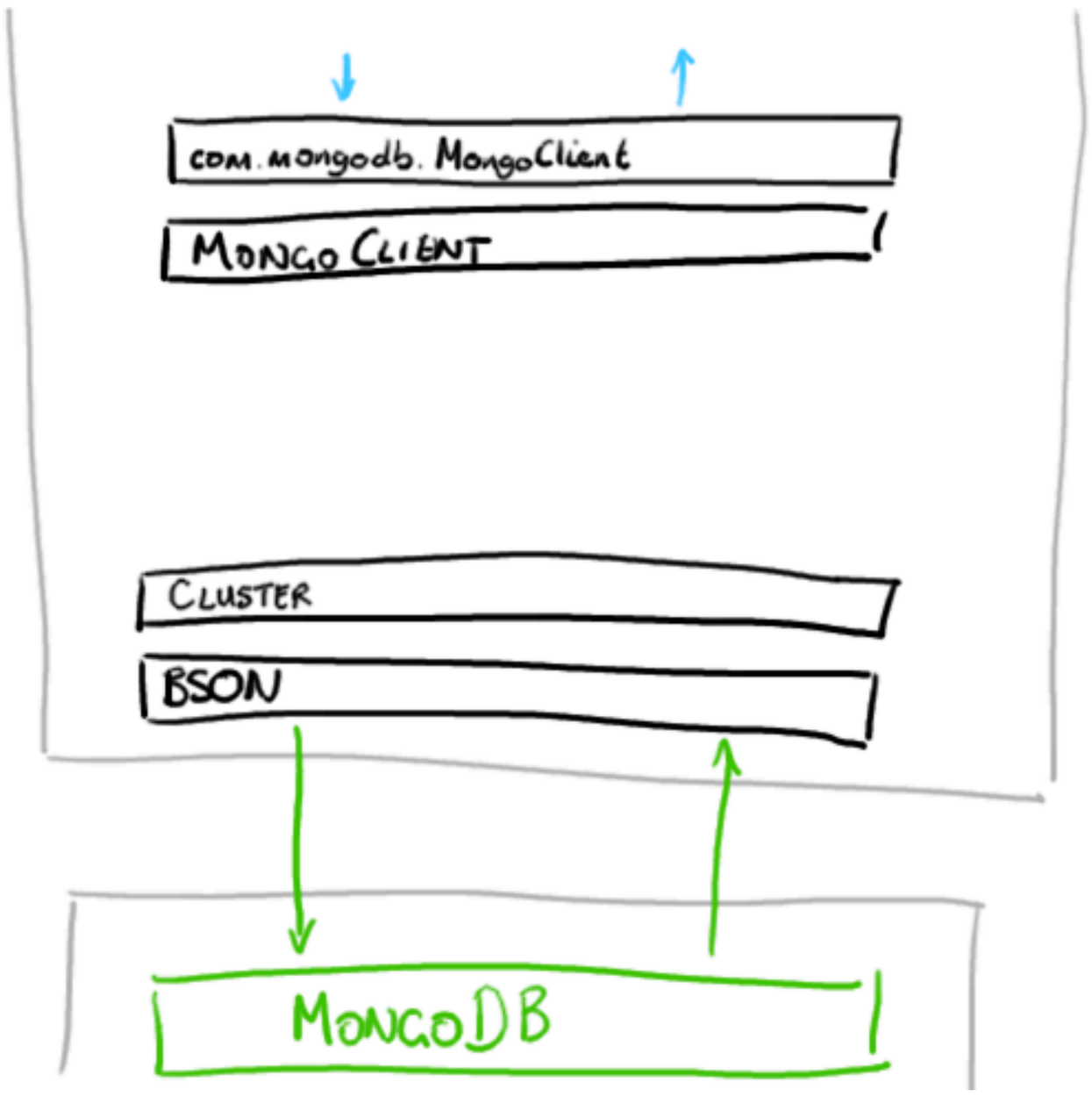
- Consistency
- ✓**Cleaner design**
- Intuitive API
- Sane Exception handling
- Test friendly
- Backwards compatible

# Design Goals

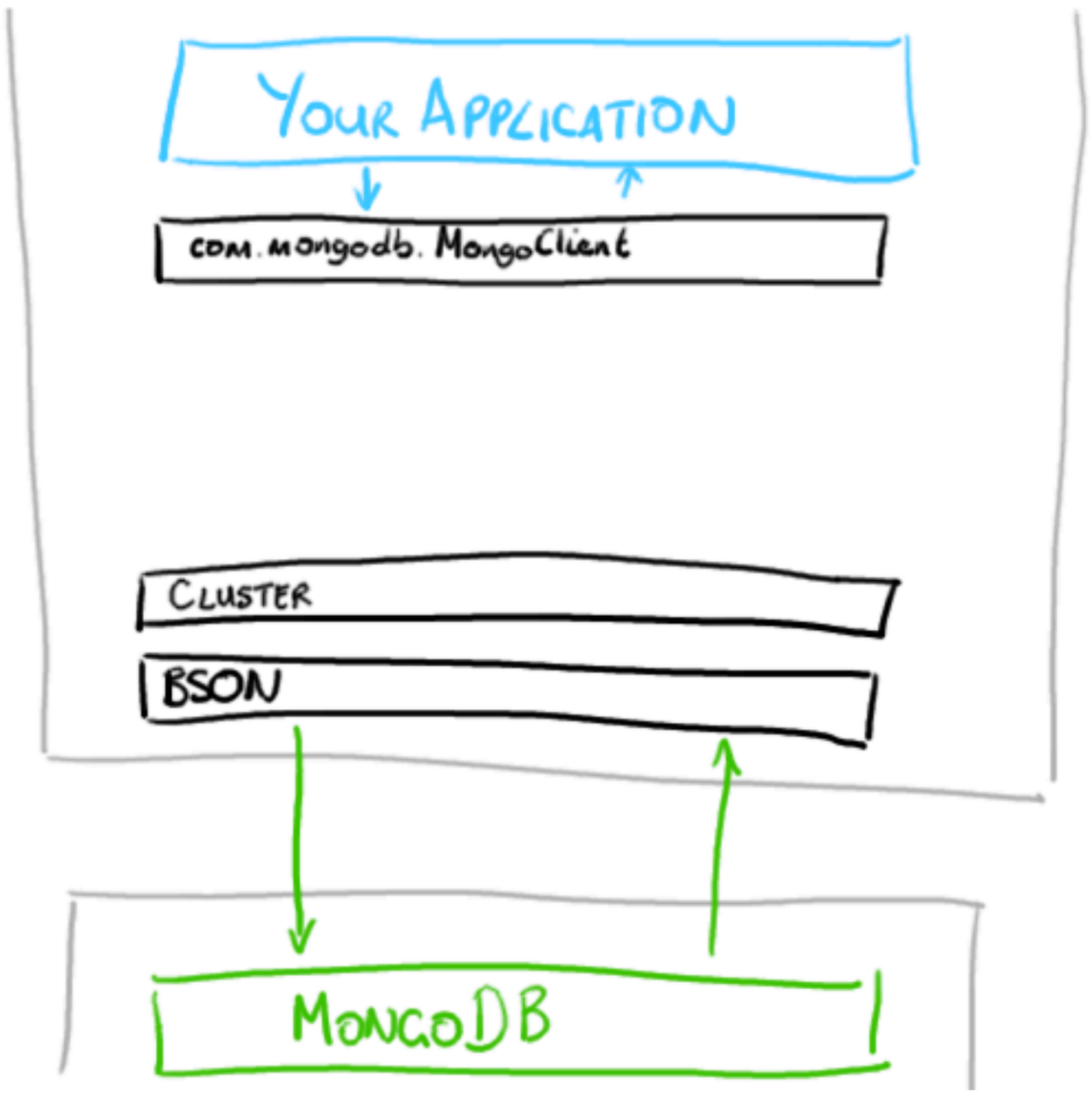
- Consistency
- Cleaner design
- Intuitive API
- Sane Exception handling
- Test friendly
- **Backwards compatible**



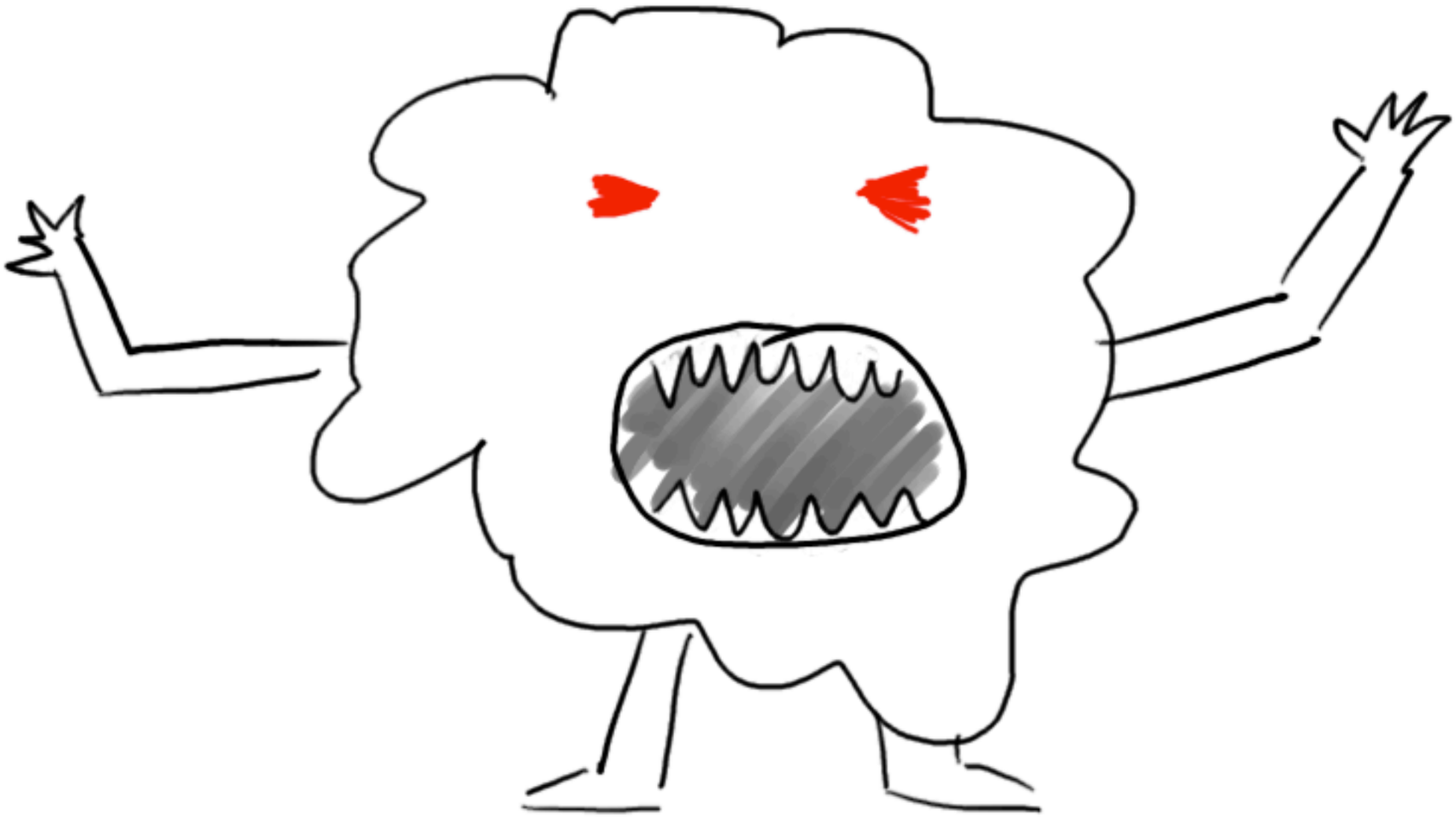

# High Level Architecture



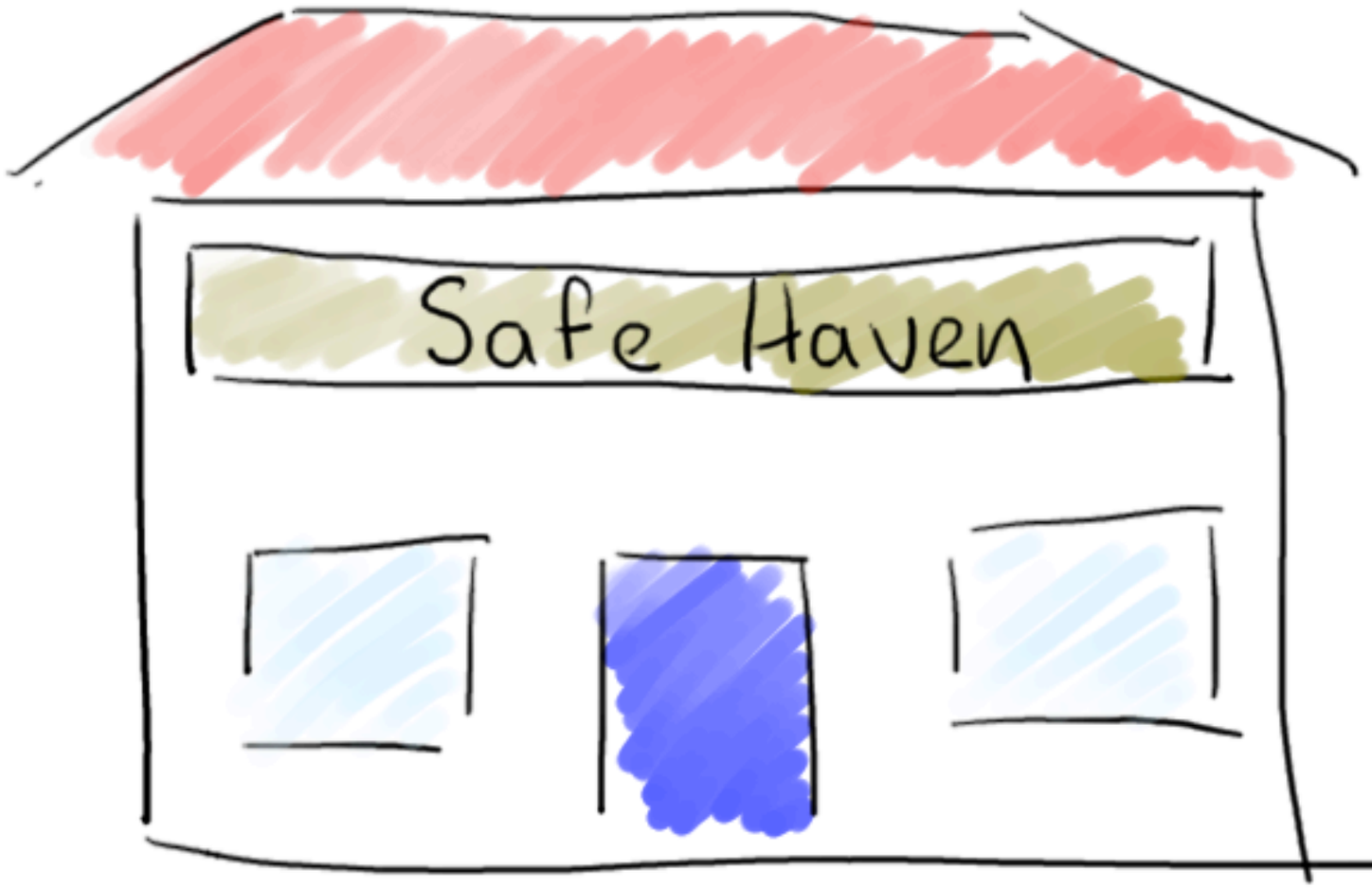

# Option 1: Wrapping




# Option 2: Connecting



# Backward Compatibility?




# Tests Pass

We win!

# Design Goals

- Consistency
- Cleaner design
- Intuitive API
- Sane Exception handling
- Test friendly

✓**Backwards compatible**

# Not Dead Yet...

# The Public API

# Design Goals

- Consistency
- Cleaner design
- **Intuitive API**
- Sane Exception handling
- Test friendly
- Backwards compatible

# Caveats

- It won't look like this
- Haven't decided consistent names yet
- Need something that suits all drivers

# Find

# Find

```
collection.find(query).skip(1000).limit(100);
```

# Find

```
collection.find(query).skip(1000).limit(100);

collection.find(query).skip(1000).limit(100);
```

# Find

# Find

```
collection.find(query).skip(1000).limit(100);

collection.find(query).skip(1000).limit(100);

collection.find(query, fields);
```

```
collection.find
```

```
find(DBObject ref)                          DBCursor
find()                                      DBCursor
find(DBObject ref, DBObject key…            DBCursor
find(DBObject query, DBObject f…            DBCursor
find(DBObject query, DBObject f…            DBCursor
findAndModify(DBObject query, D…            DBObject
findOne()                                   DBObject
findAndModify(DBObject query, D…            DBObject
findAndModify(DBObject query, D…            DBObject
findAndRemove(DBObject query)               DBObject
findOne(DBObject o)                         DBObject
Use ⇧⌘⏎ to syntactically correct your code after completing (balance parentheses etc.)
```

# Which One?

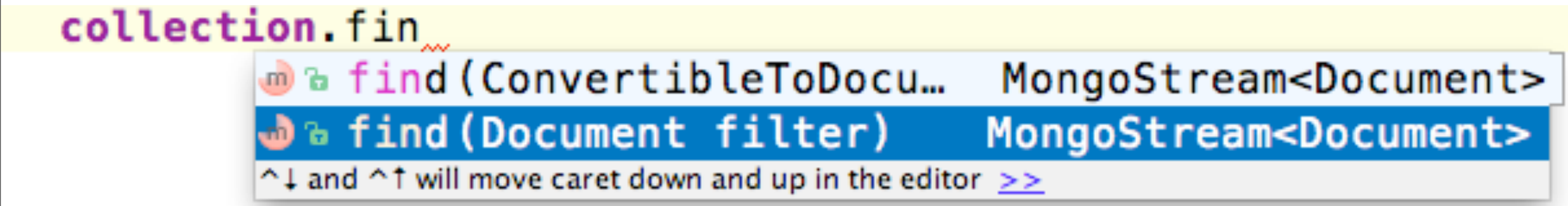
# Find

# Find

```
collection.find(query).skip(1000).limit(100);

collection.find(query).skip(1000).limit(100);

collection.find(query, fields);
```

# Find

```
collection.find(query).skip(1000).limit(100);

collection.find(query).skip(1000).limit(100);


collection.find(query, fields);

collection.find(query).select(fields);
```




# Fewer Decisions

```
collection.find(filter).
```

```
getCriteria()                                  MongoFind
limit(int limit)                               MongoStream<Document>
noLimit()                                      MongoStream<Document>
readPreference(ReadP…                          MongoStream<Document>
select(ConvertibleTo…                          MongoStream<Document>
select(Document selector)                      MongoStream<Document>
skip(int skip)                                 MongoStream<Document>
sort(ConvertibleToDo…                          MongoStream<Document>
sort(Document sortCr…                          MongoStream<Document>
tail()                                         MongoStream<Document>
Press ^. to choose the selected (or first) suggestion and insert a dot afterwards >>
```

# “Cmd + space” friendly

# Find

```
collection.find(query).skip(1000).limit(100);

collection.find(query).skip(1000).limit(100);


collection.find(query, fields);

collection.find(query).select(fields);
```

# Remove

# Remove

```
collection.remove(query);
```

# Remove

```
collection.remove(query);

collection.find(query).remove();
```

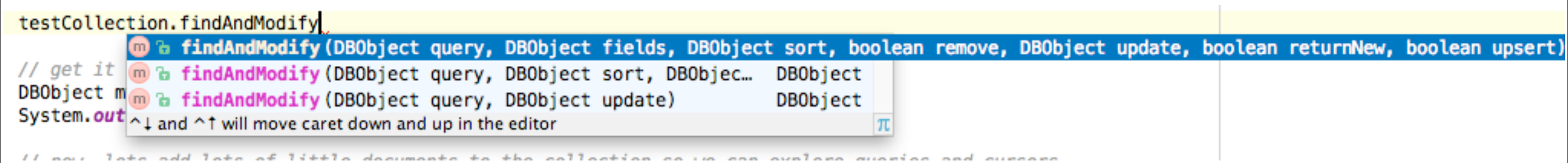
# Find and Modify

# Find and Modify

```
collection.findAndModify(query, update);
```

# Find and Modify

```
collection.findAndModify(query, update);

collection.find(query).updateOneAndGet(update);
```


testCollection.findAndModify
findAndModify(DBObject query, DBObject fields, DBObject sort, boolean remove, DBObject update, boolean returnNew, boolean upsert)
findAndModify(DBObject query, DBObject sort, DBObjec… DBObject
findAndModify(DBObject query, DBObject update) DBObject
^↓ and ^↑ will move caret down and up in the editor
// get it
DBObject m
System.out

# They hate me!

# Find and Modify

# Find and Modify

```
collection.findAndModify(query, update);

collection.find(query)
          .updateOneAndGet(update);

collection.findAndModify(query,
                         fields,
                         sort,
                         false,
                         update,
                         true,
                         false);
```

# Find and Modify

```
collection.findAndModify(query, update);

collection.find(query)
          .updateOneAndGet(update);

collection.findAndModify(query,
                         fields,
                         sort,
                         false,
                         update,
                         true,
                         false);

collection.find(query)
          .sort(sort)
          .updateOneAndGet(update);
```

# Find and Modify

# Find and Modify

```
collection.findAndModify(query, update);

collection.find(query)
          .updateOneAndGet(update);

collection.findAndModify(query,
                         fields,
                         sort,
                         false,
                         update,
                         true,
                         false);

collection.find(query)
          .sort(sort)
          .updateOneAndGet(update);
```

# Find and Modify

```
collection.findAndModify(query, update);

collection.find(query)
          .updateOneAndGet(update);

collection.findAndModify(query,
                         fields,
                         sort,
                         false,
                         update,
                         true,
                         false);

collection.find(query)
          .sort(sort)
          .updateOneAndGet(update);

collection.find(query)
          .sort(sort)
          .getOneAndUpdate(update);
```

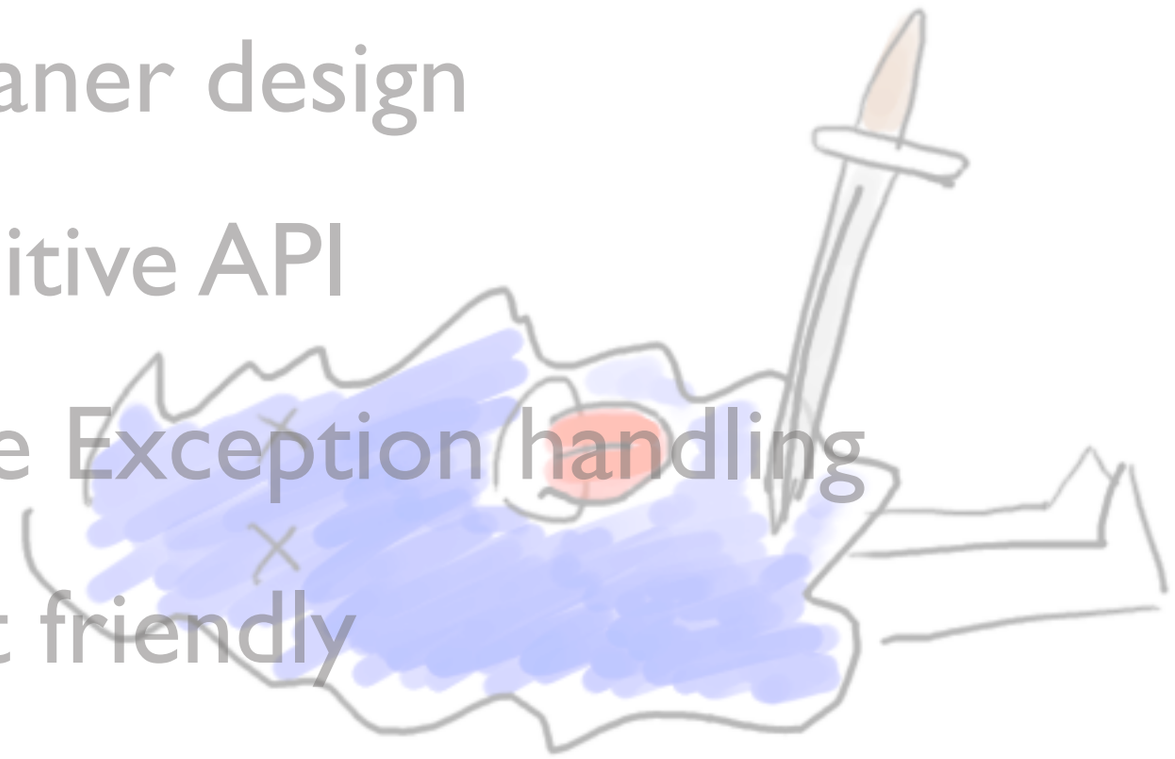
# Lack of consistency

# Consistency at last

```
collection.find(query).limit(10);

collection.find(query).limit(10).remove();


collection.find(query).sort(sortCriteria).getOne();

collection.find(query).sort(sortCriteria).remove();

collection.find(query).sort(sortCriteria).count();
```

## ✓Consistency

- Cleaner design
- Intuitive API
- Sane Exception handling
- Test friendly
- Backwards compatible

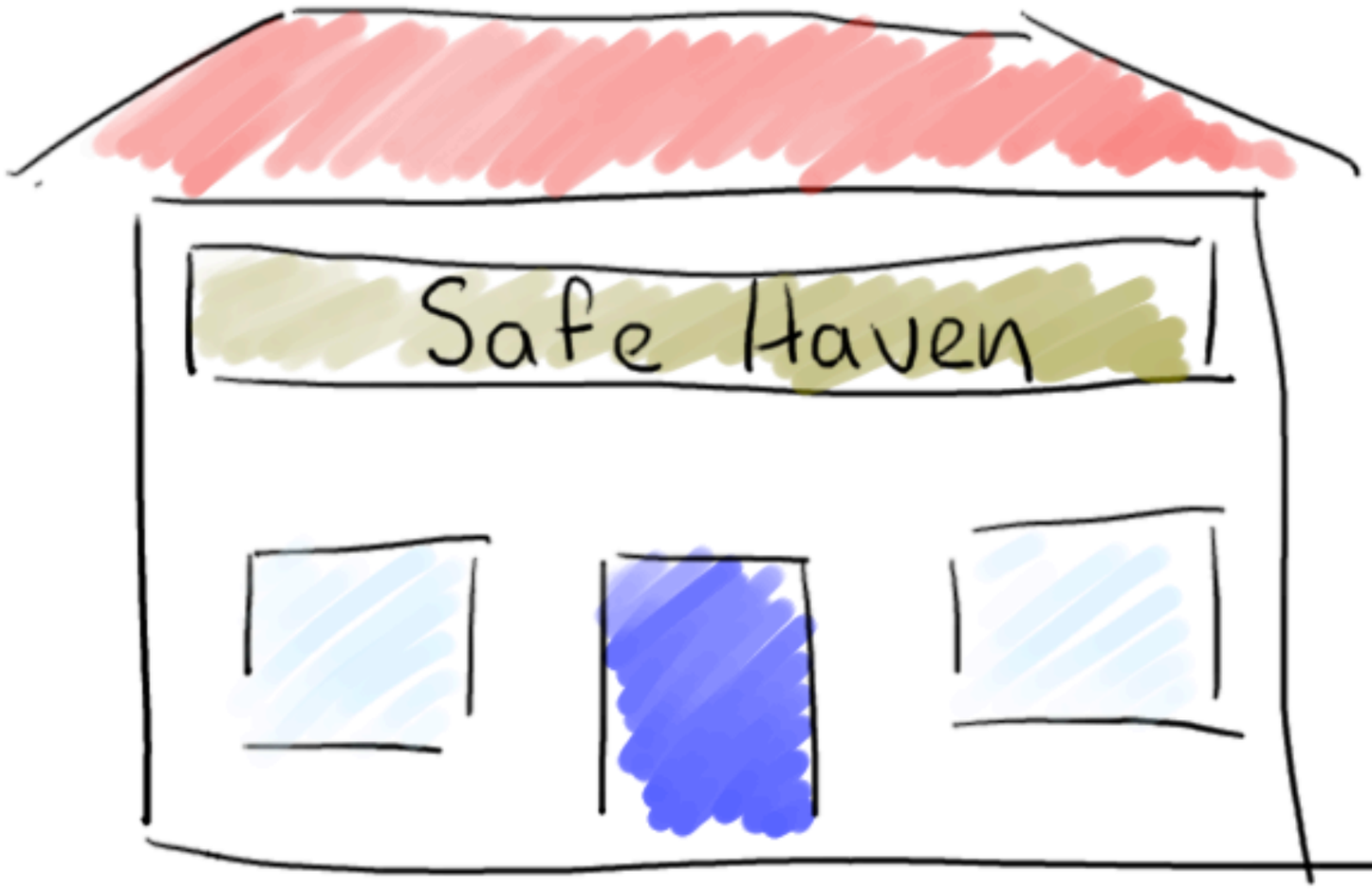


# Muerto del todo

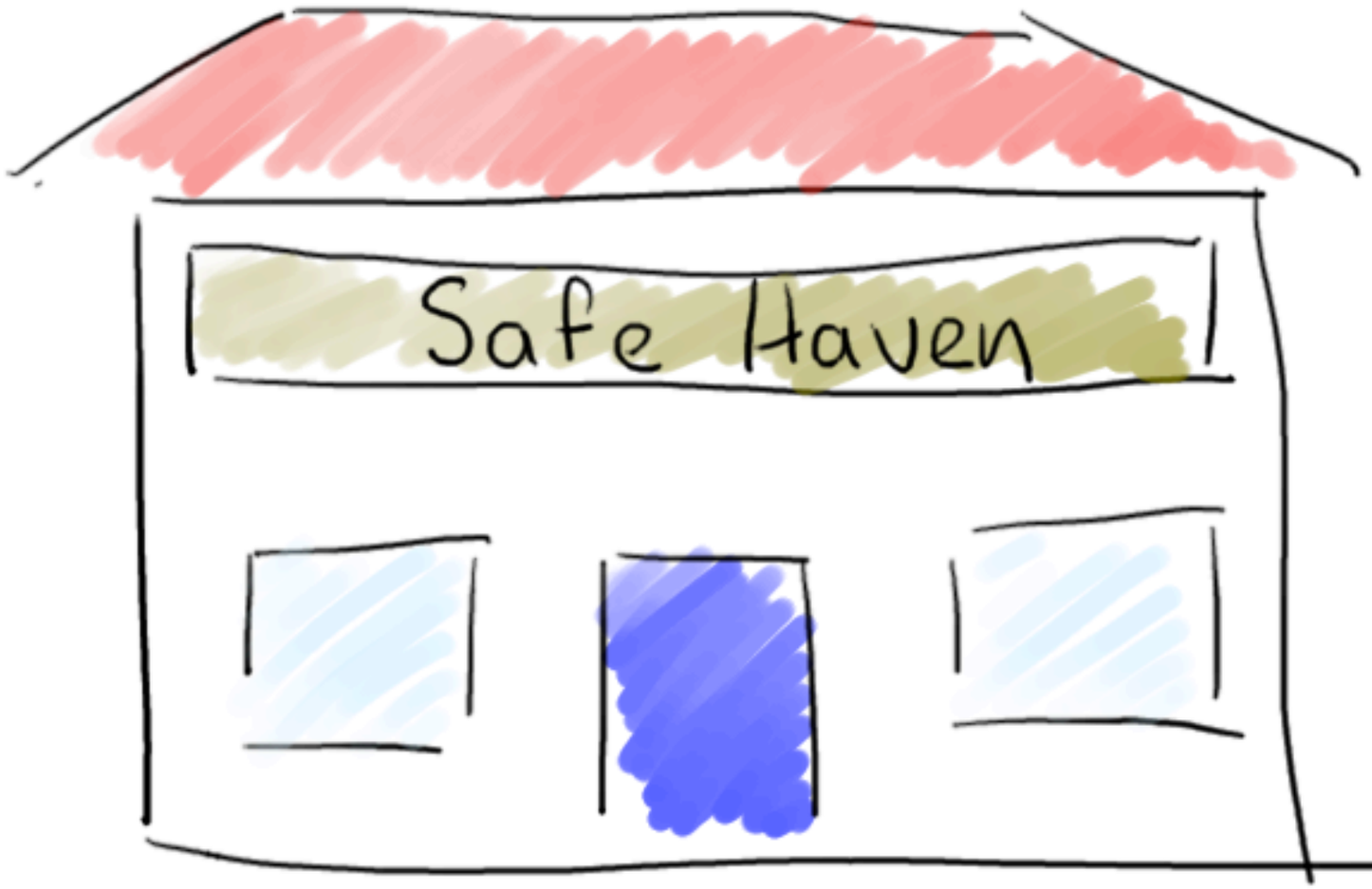
# Design Goals

- Consistency
- Cleaner design
- **Intuitive API...**
- Sane Exception handling
- Test friendly
- Backwards compatible

# Not Dead Yet!




# Tutorial/hack session




# This talk

# Design is a Process, not a Document

# Q & A

10gen | the MongoDB company

mongoDB

1. Are you using the Java driver?

# 2. What do you like about it?

# 3. What are your pain points?

# Design is a Process, not a Document

# Your Questions

10gen | the MongoDB company

mongoDB