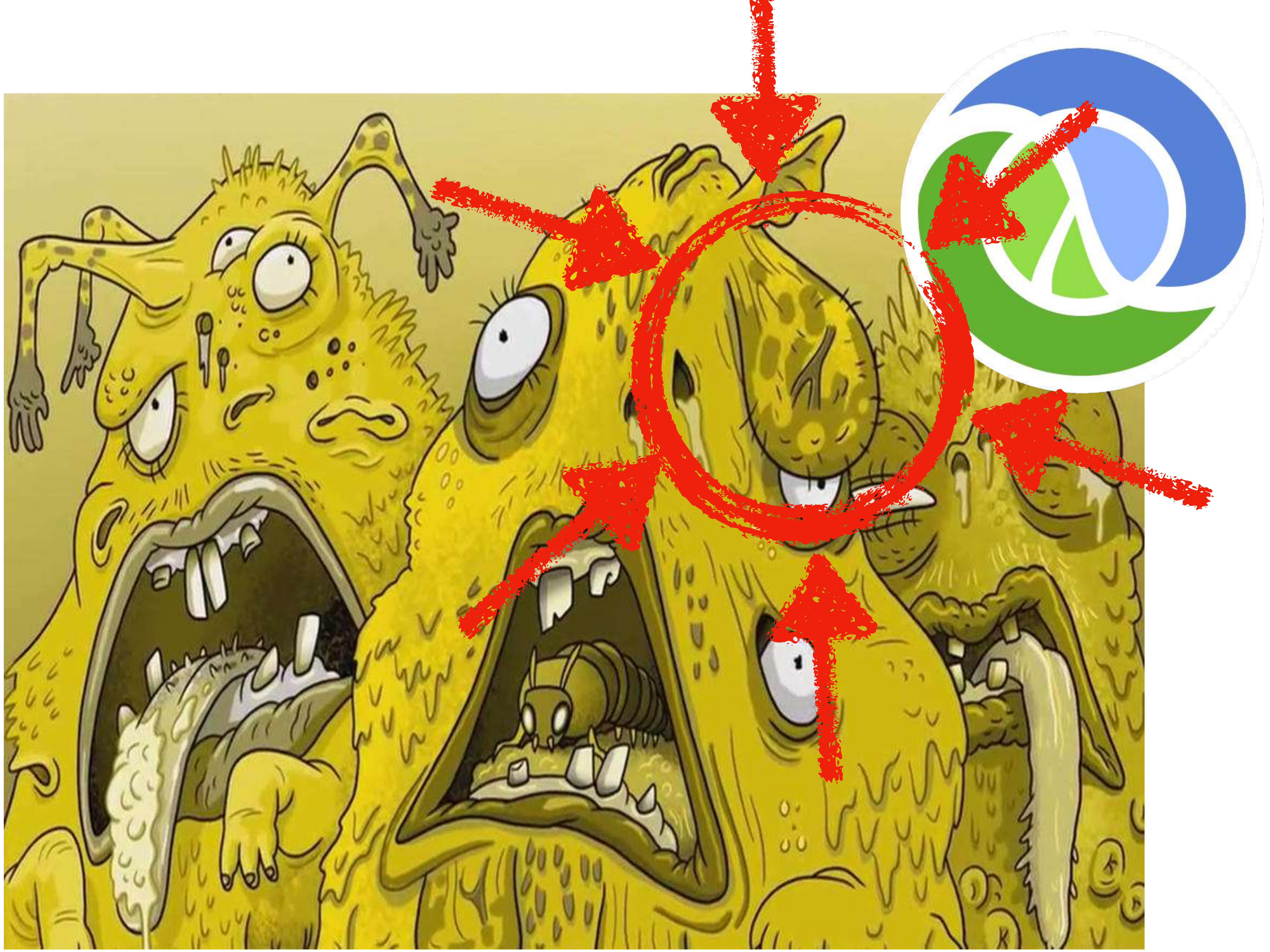
Java, with a Clojure mindset

@DanLebrero
www.akvo.org



Clojure

- Functional
- Hosted
- Dynamic
- Strongly typed
- Lisp

100% BONUS
+200 FreeSpins
ON 1ST DEPOSIT

New players only, min deposit £10, 1st deposit: 100% bonus up to £100 & 20 Free Spins daily for 10 days in a row, first deposit + bonus funds must be wagered 35x (game weighting applies). Max. bet while playing with bonus £6. Neteller and Skrill deposits excluded from offer. Game weighting and T&Cs apply.

Give 10$ free cash
if client makes 1000 bets
in less than 1 week

Functional (vs OO)

Pure Functions

makeagif.com

“Programmers are constantly in maintenance mode.”

— **The Pragmatic Programmer**

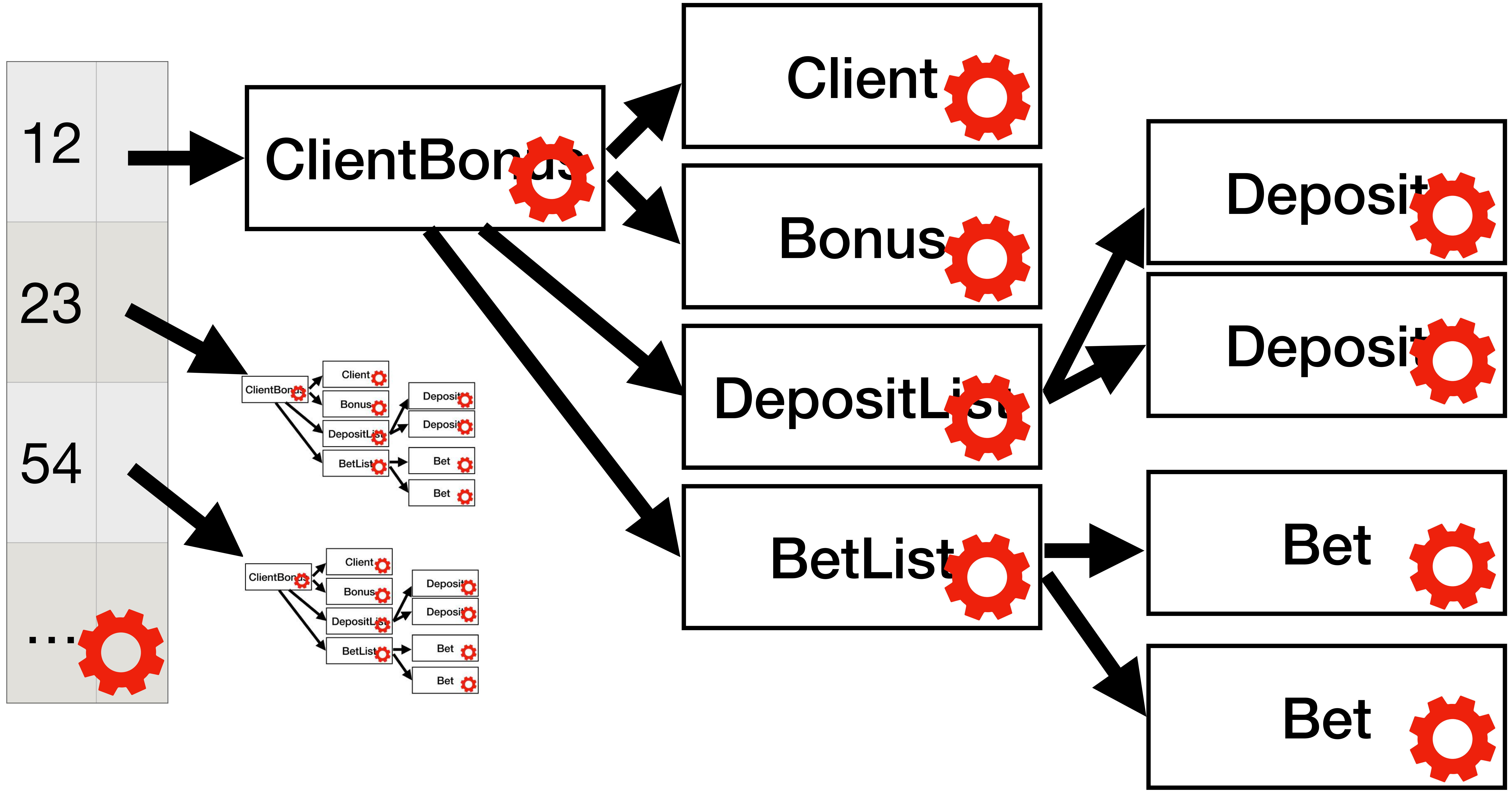
Side Effects

COOLJUNK PRODUCTIONS

Side effects are the enemy

$$\begin{tikzcd}
 & & \text{Side Effects} \arrow[dl] \arrow[drr] & & & \\
 & \text{State} & & & \text{IO} \arrow[dl] \arrow[dr] & \\
 & & & \text{Co-effects} & & \text{Effects}
\end{tikzcd}$$

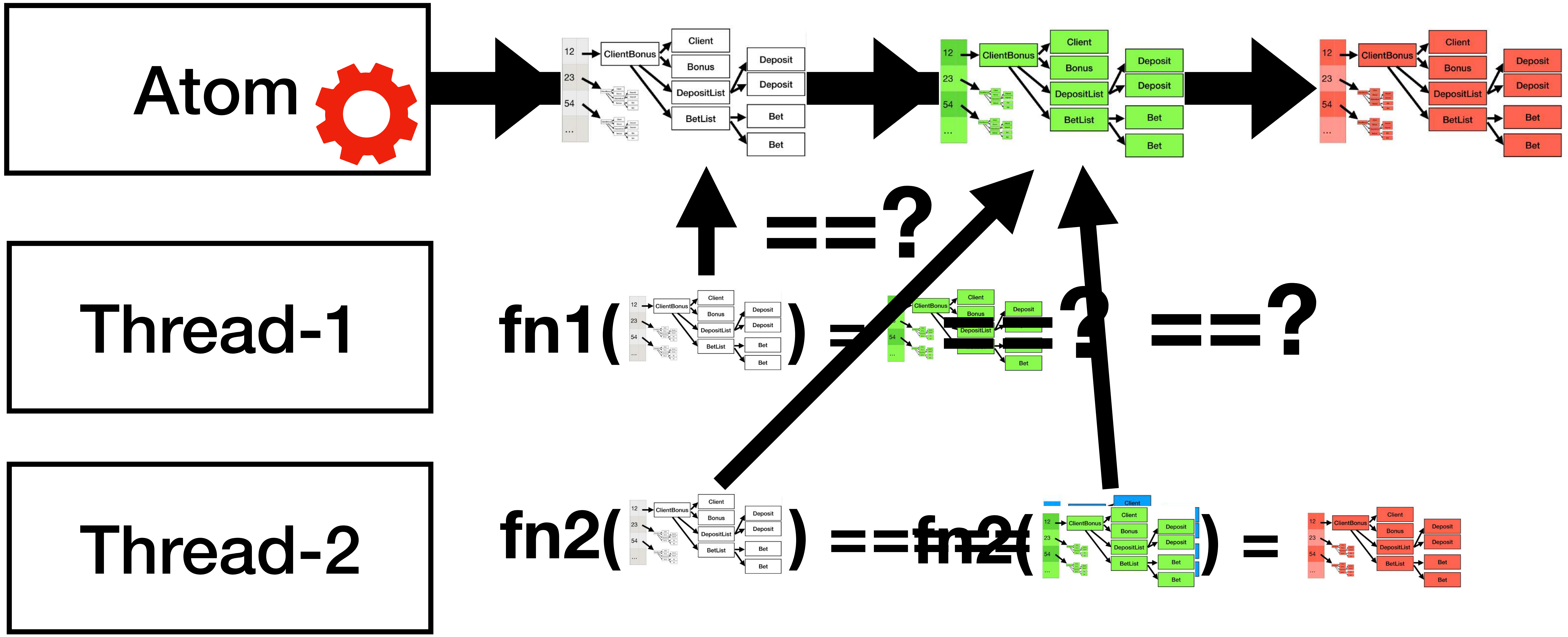
State

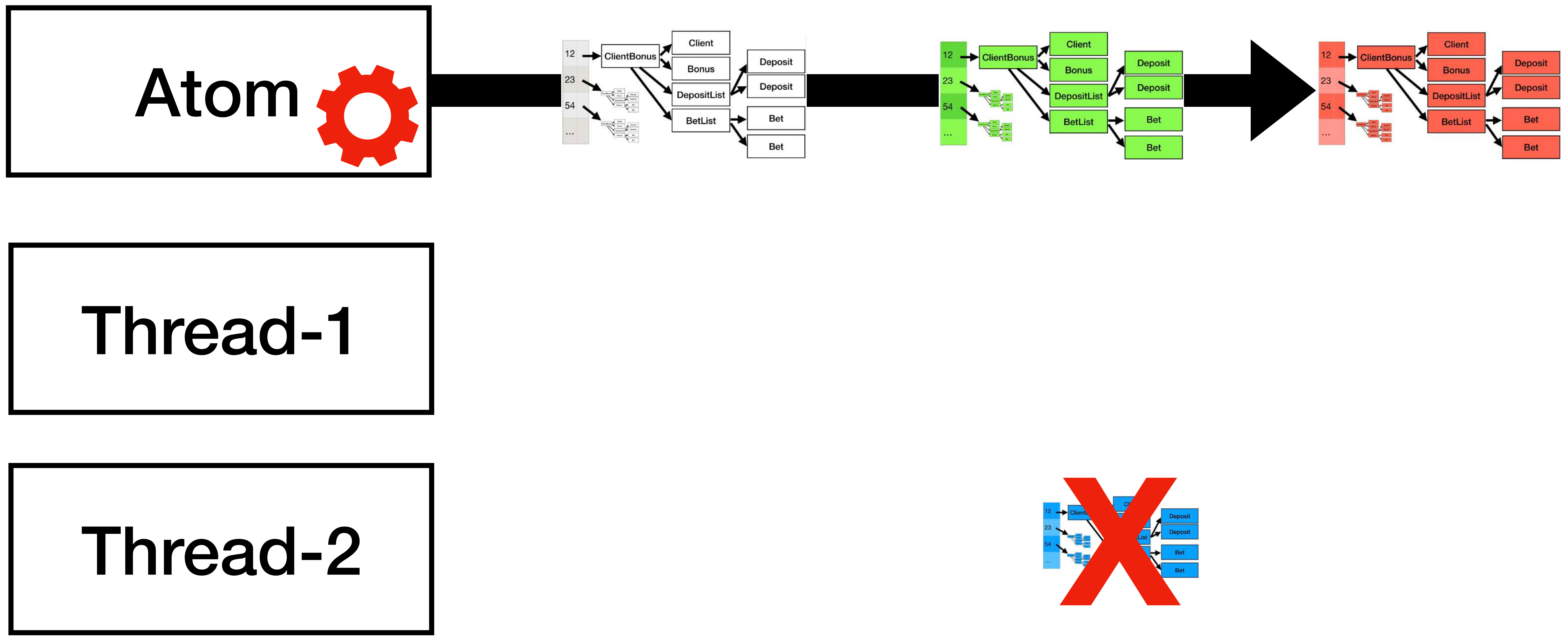

12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet

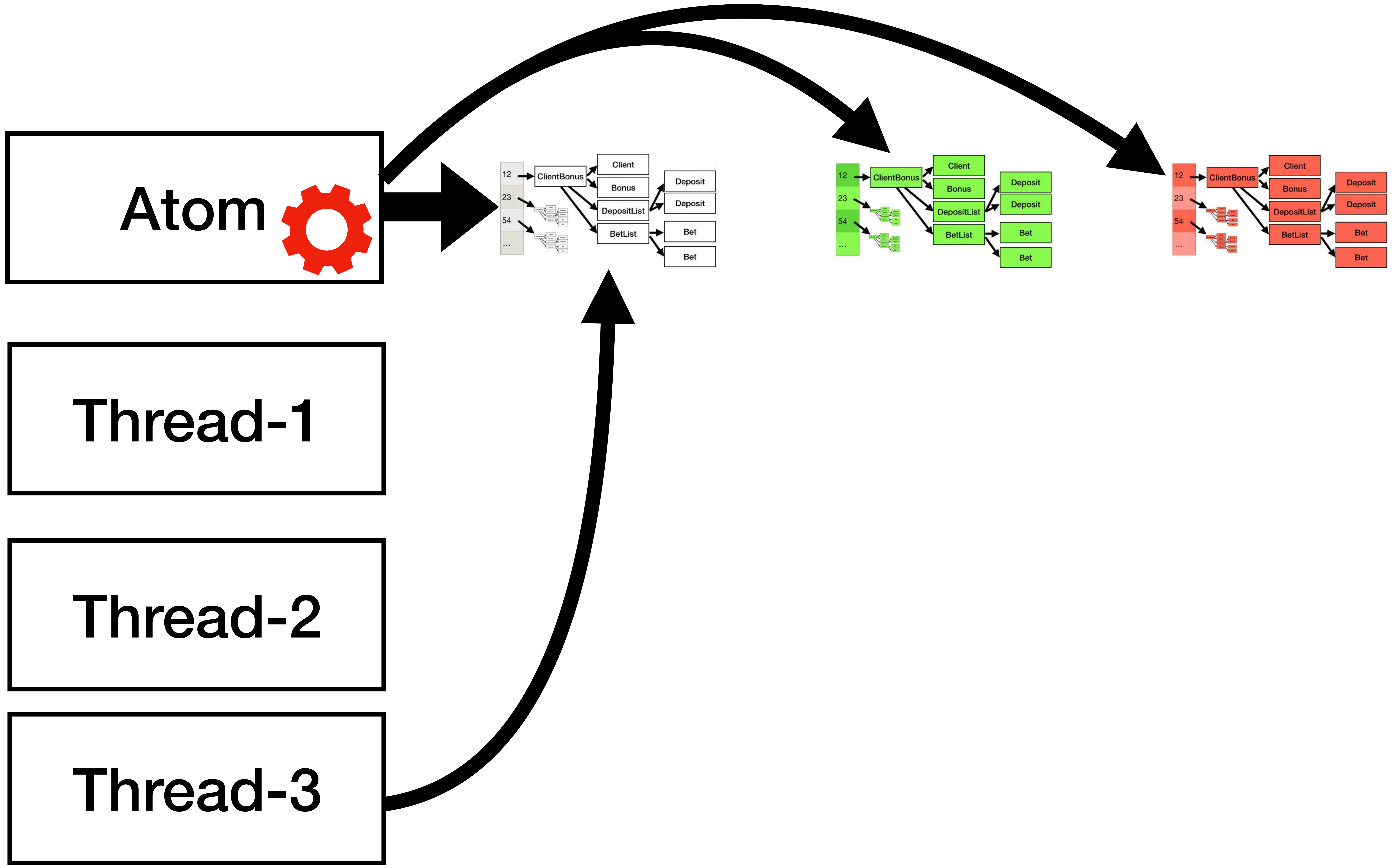
Atom

=~

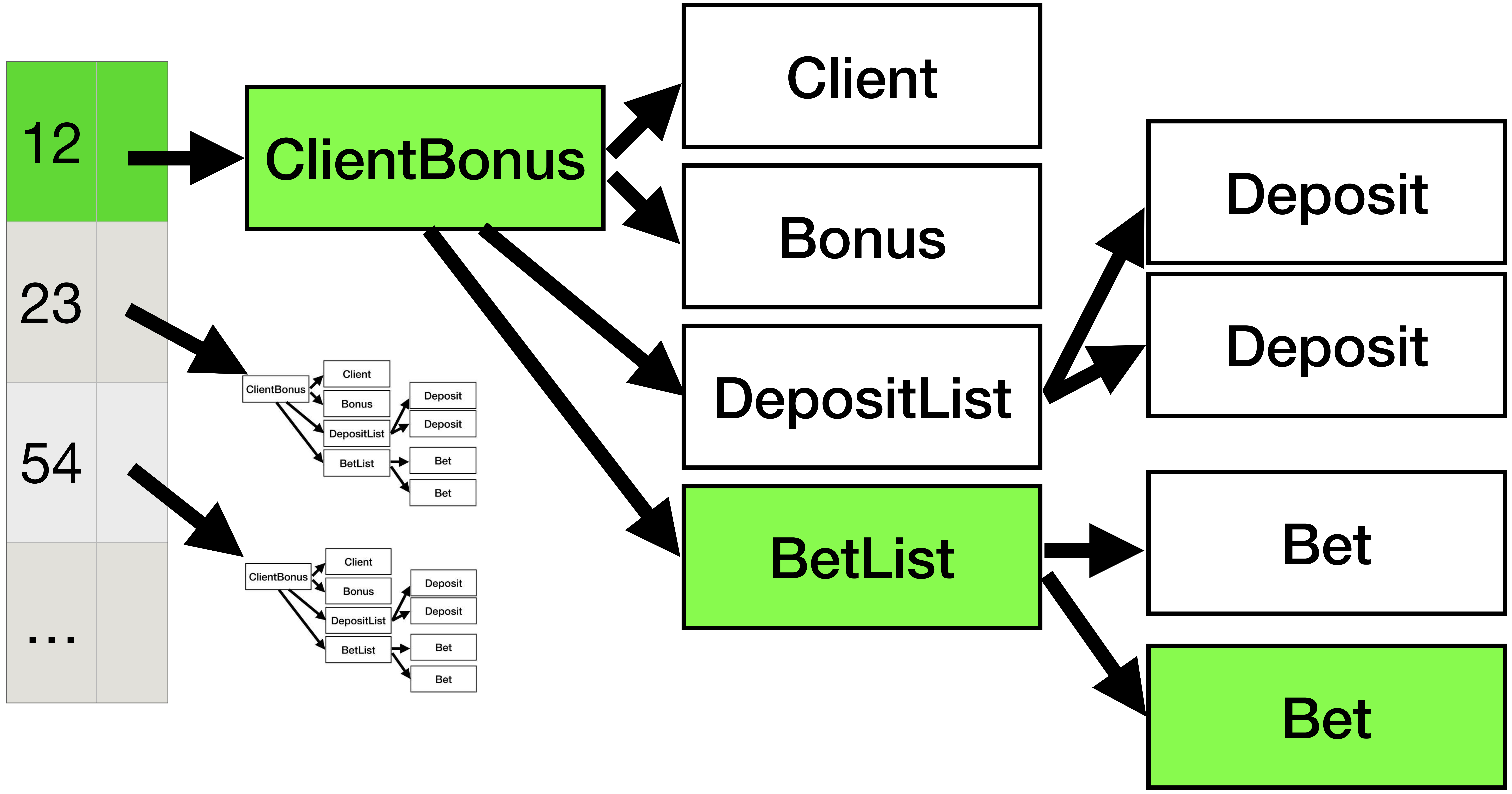
j.u.c.a.AtomicReference


Atom
Thread-1
Thread-2
12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet
==?
fn1(
) =
==?
==?
fn2(
) ==fn2(
) =


Atom
Thread-1
Thread-2
12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet
12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet
12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet
12
23
54
...
Deposit
Deposit
Bet
Bet



Atom
Thread-1
Thread-2
Thread-3
12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet

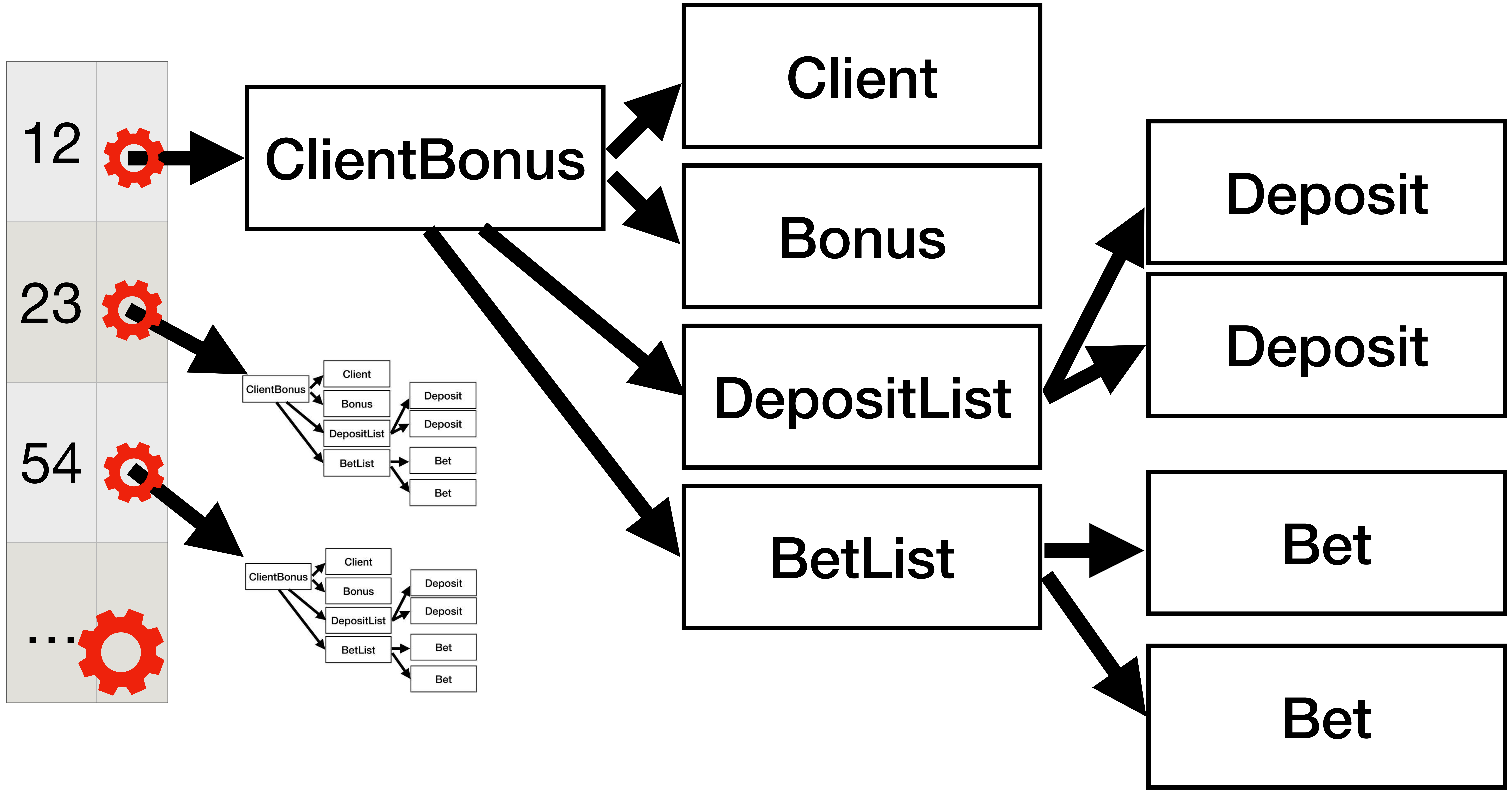

12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet

“Is this thread safe?”

— **Every Java developer, every day.**

1. State is consistent
2. Function must be pure
3. Only safe within the pure function

```
public class ClientBonus {

    private final Client client;
    private final Bonus bonus;
    private final DepositList deposits;

…
```

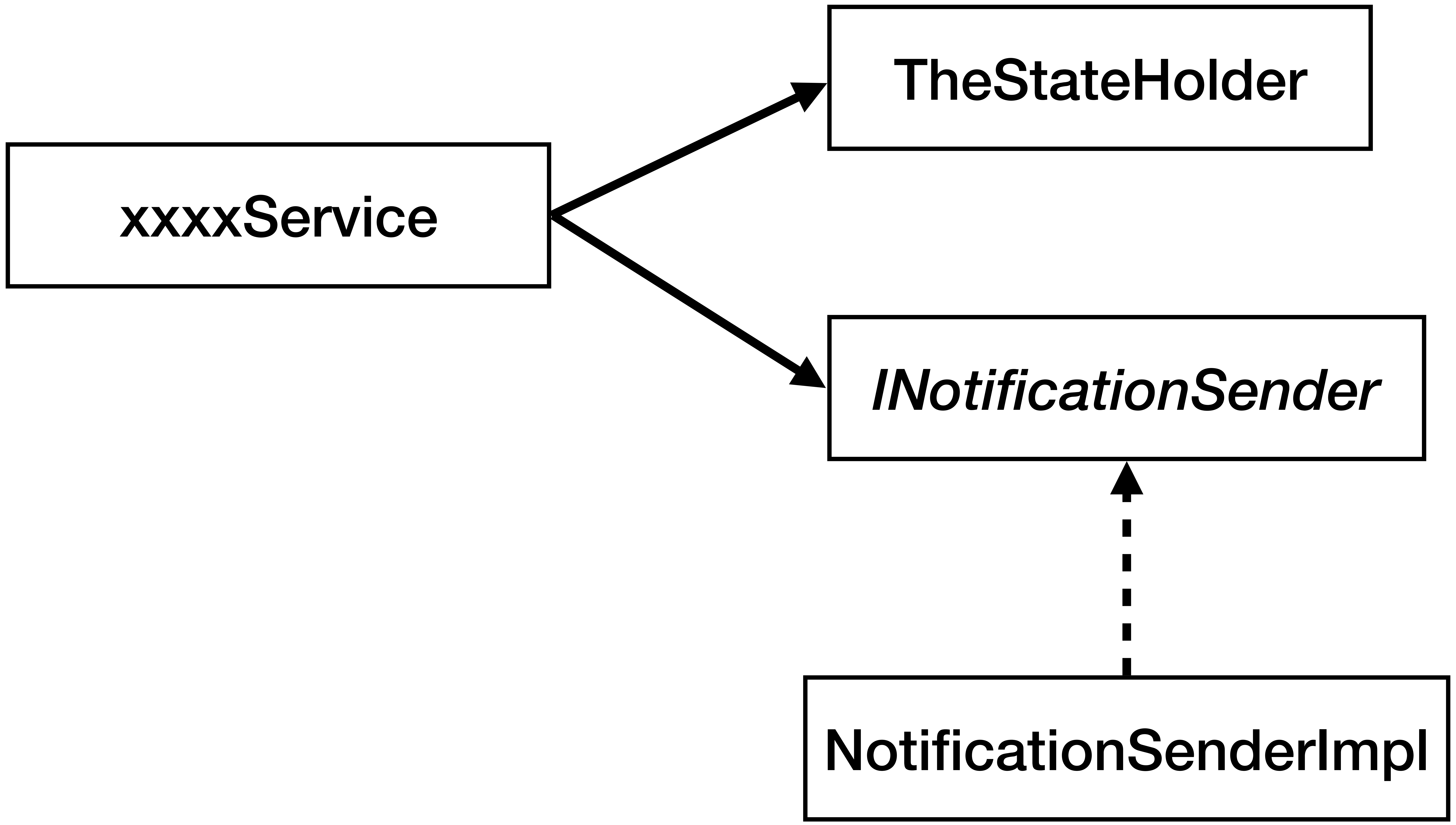

12
23
54
...
ClientBonus
Client
Bonus
DepositList
BetList
Deposit
Deposit
Bet
Bet

```
public interface ConcurrentMap<…> extends Map<…> {

V compute(K key, BiFunction<…> remappingFunction)
V computeIfAbsent(K key, Function<…> mappingFunction)
V computeIfPresent(K key, BiFunction<…> remappingFunction)
…
}
```

```java
public class TheStateHolder {

    private final Map<Long, ClientBonus> state = new ConcurrentHashMap<>();

    public ClientBonus nextState(Long client, Bet bet) {
        return state.computeIfPresent(
                client,
                (k, currentState) -> currentState.nextState(bet));
    }
```

```
public class ClientBonus {

    private final Client client;
    private final Bonus bonus;
    private final DepositList deposits;

    public ClientBonus nextState(Bet bet) {
        ...
    }

...
```

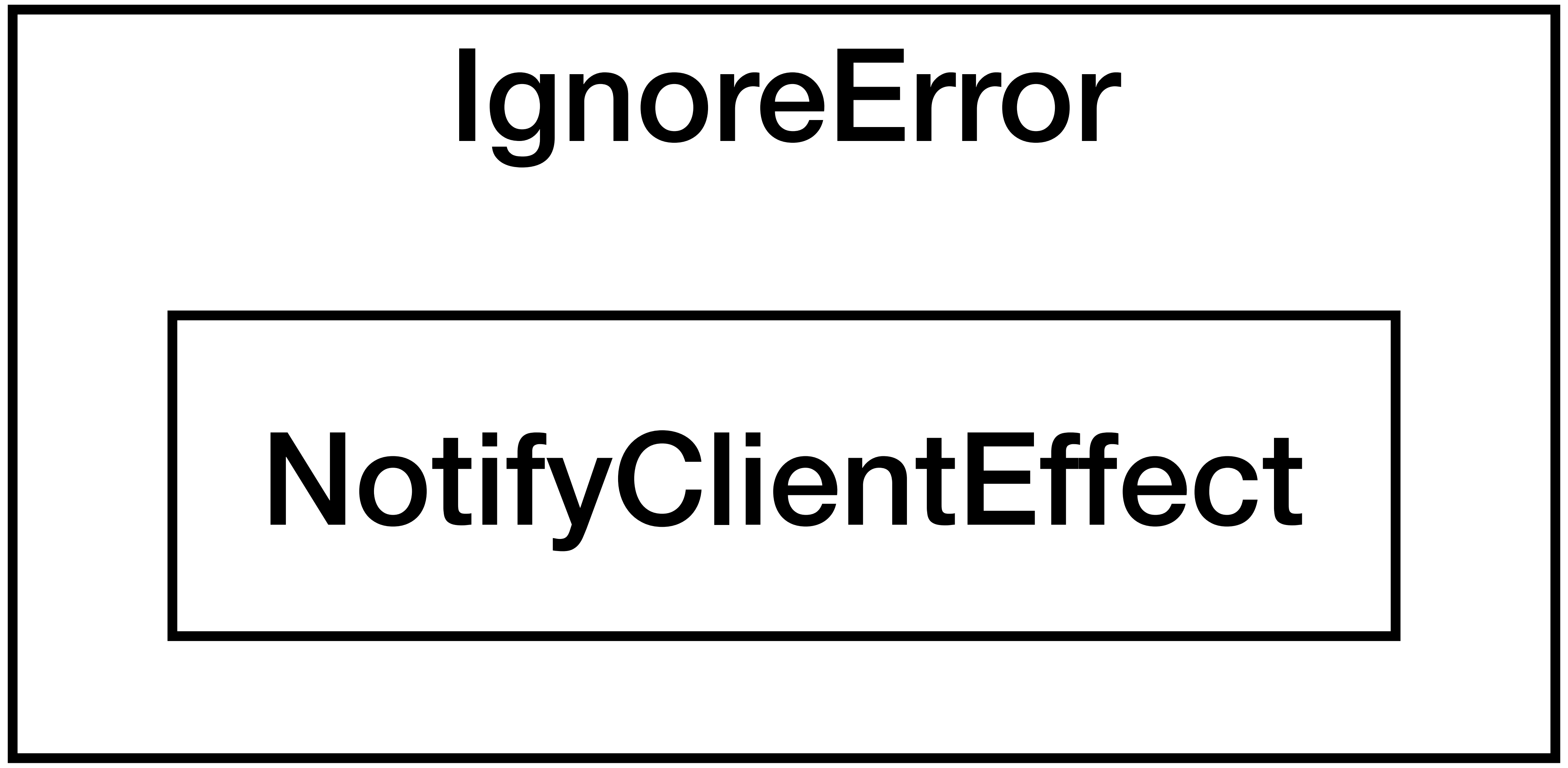
Effects

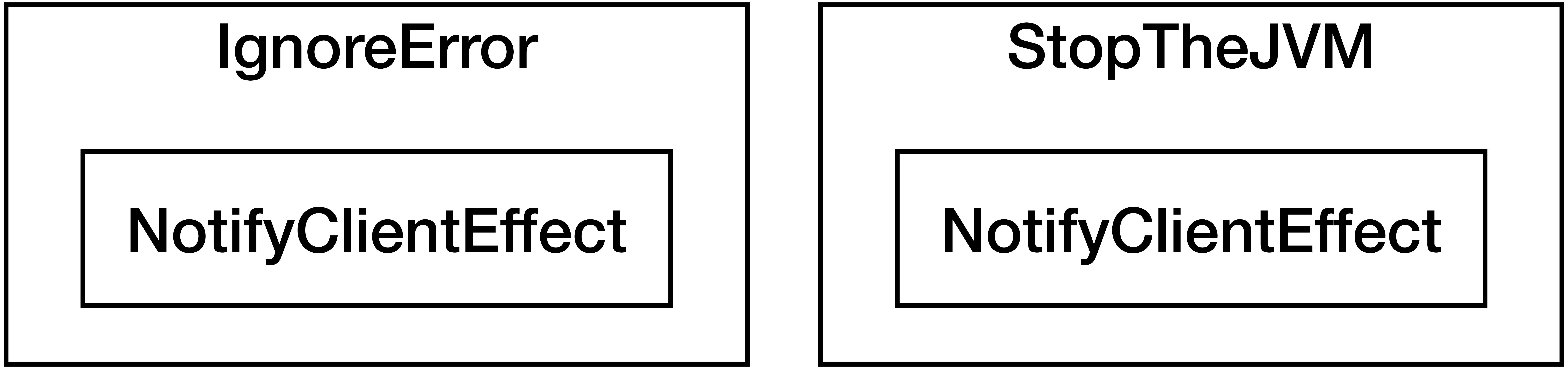

TheStateHolder
xxxxService
INotificationSender
NotificationSenderImpl

```
public class ClientBonus {

    private final Client client;
    private final Bonus bonus;
    private final DepositList deposits;

    public ClientBonus nextState(Bet bet) {
        ...
    }

...
```

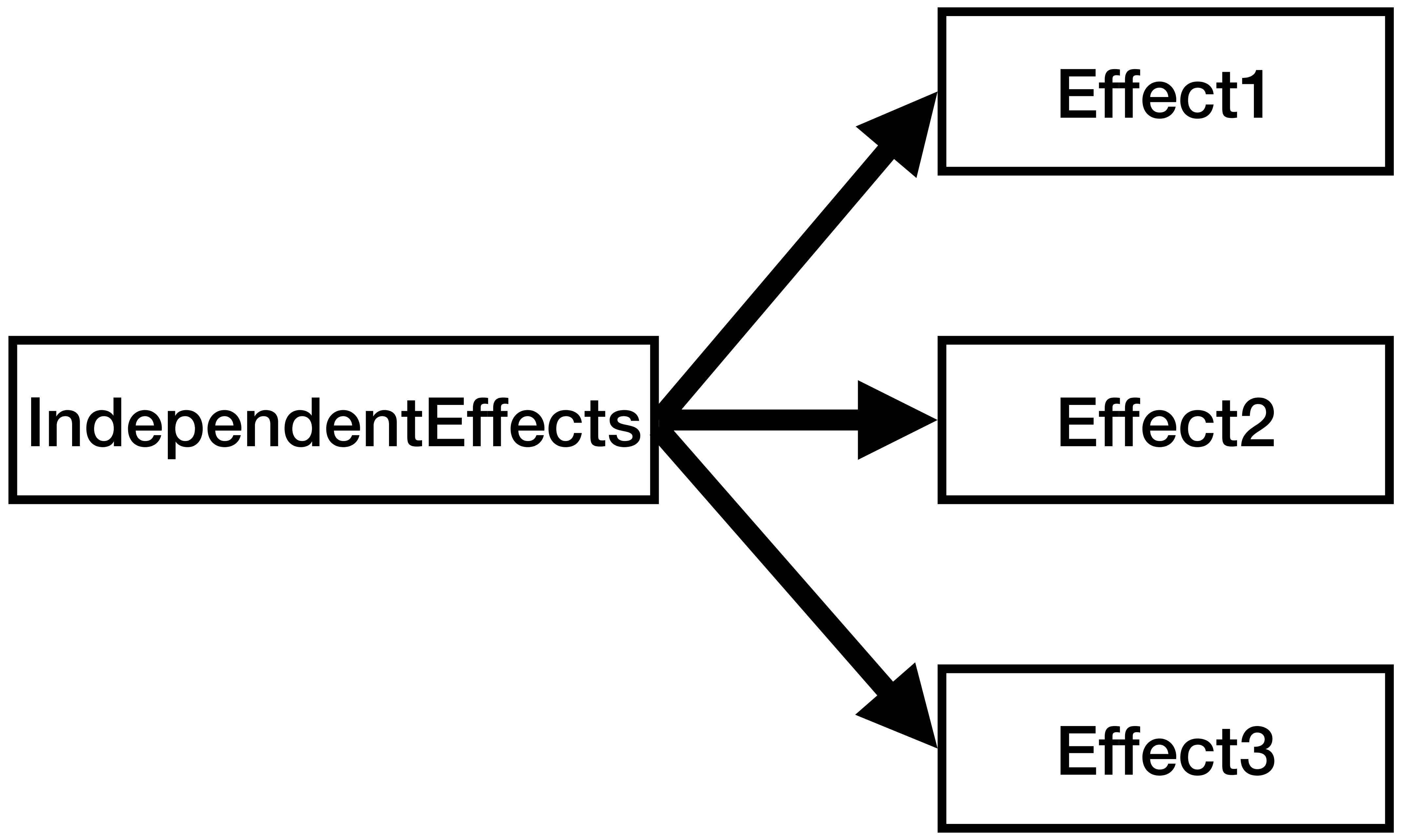
```
public class ClientBonus {

    private final Client client;
    private final Bonus bonus;
    private final DepositList deposits;

    public Pair<ClientBonus,Effects> next(Bet bet) {
        ...
    }

...
```

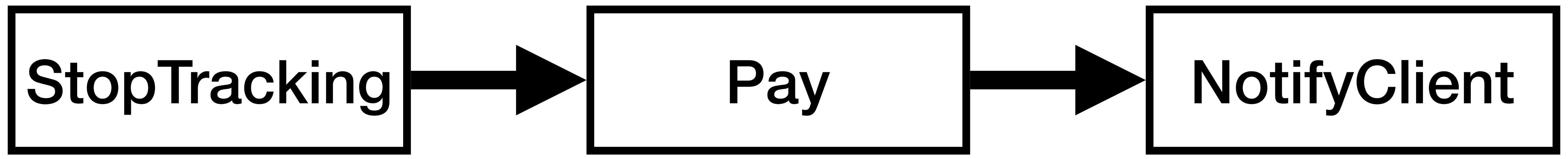
NotifyClientEffect


IgnoreError
NotifyClientEffect


IgnoreError
NotifyClientEffect
StopTheJVM
NotifyClientEffect

$$\begin{tikzcd}\boxed{\text{SequentialEffects}} \arrow[r] & \boxed{\text{Effect1}} \arrow[r] & \boxed{\text{Effect2}} \arrow[r] & \boxed{\text{Effect3}}\end{tikzcd}$$


IndependentEffects
Effect1
Effect2
Effect3


StopTracking
Pay
NotifyClient

```
public interface Effect {
    void run(AllDependencies dependencies);
}
```

```
Pair<ClientBonus, Effects> pair = theStateHolder.nextState(bet);
pair.effects.run(dependencies);
```

```
Pair<ClientBonus, Effects> pair = theStateHolder.nextState(bet);
pair.effects.run(dependencies);
```

“Is this thread safe?”

Thread-1 → Pair<> pair = theStateHolder.nextState(bet);

Thread-2 → Pair<> pair = theStateHolder.nextState(bet);

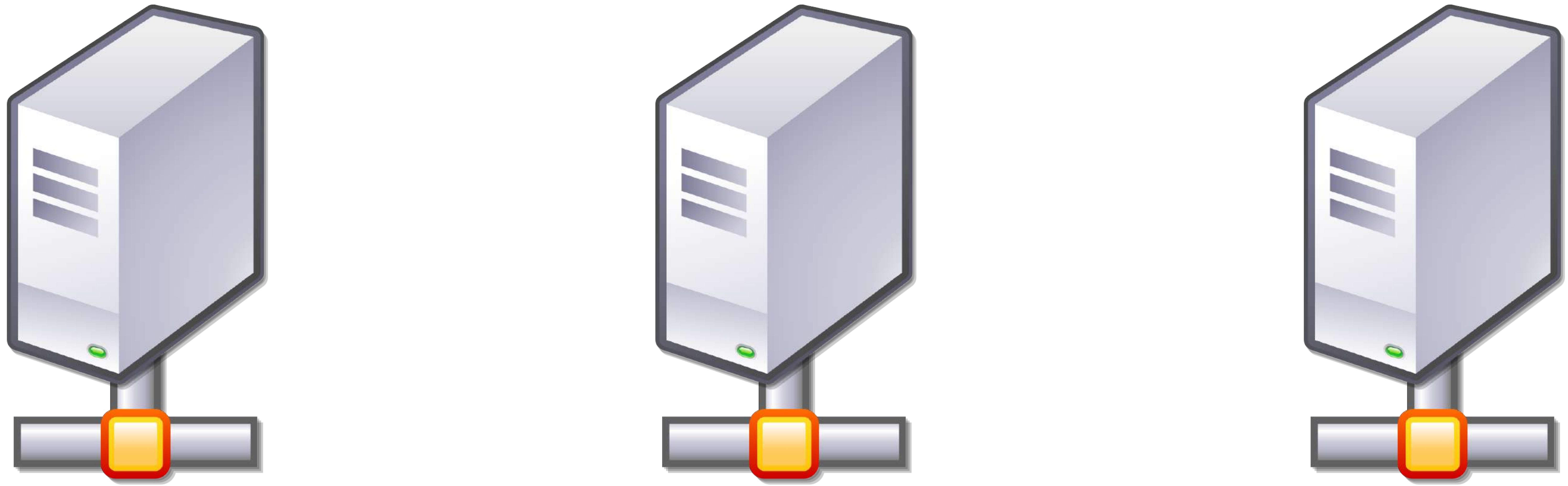
Thread-1 → Pair<> pair = theStateHolder.nextState(bet);

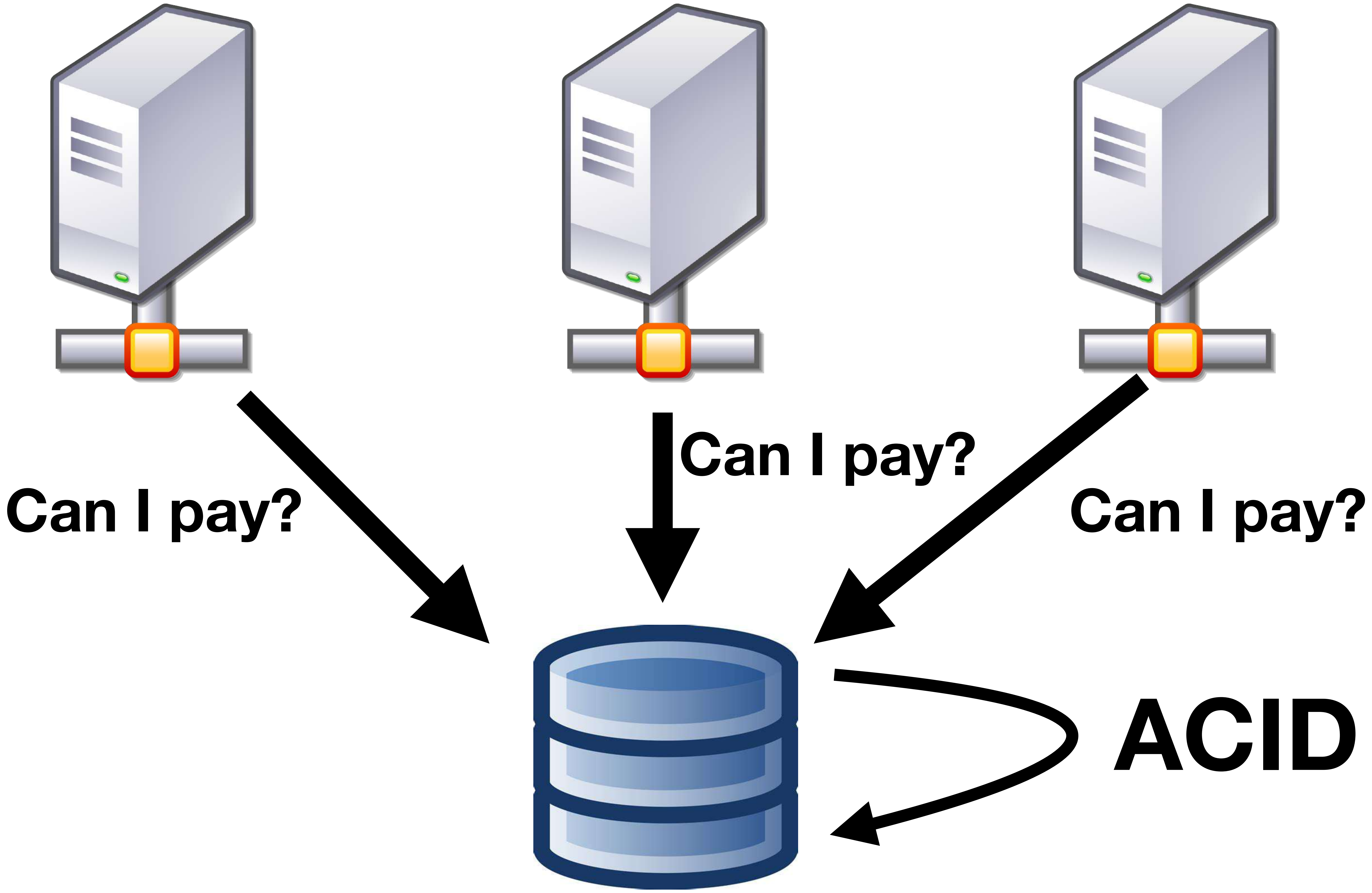
Thread-2 → Pair<> pair = theStateHolder.nextState(bet);

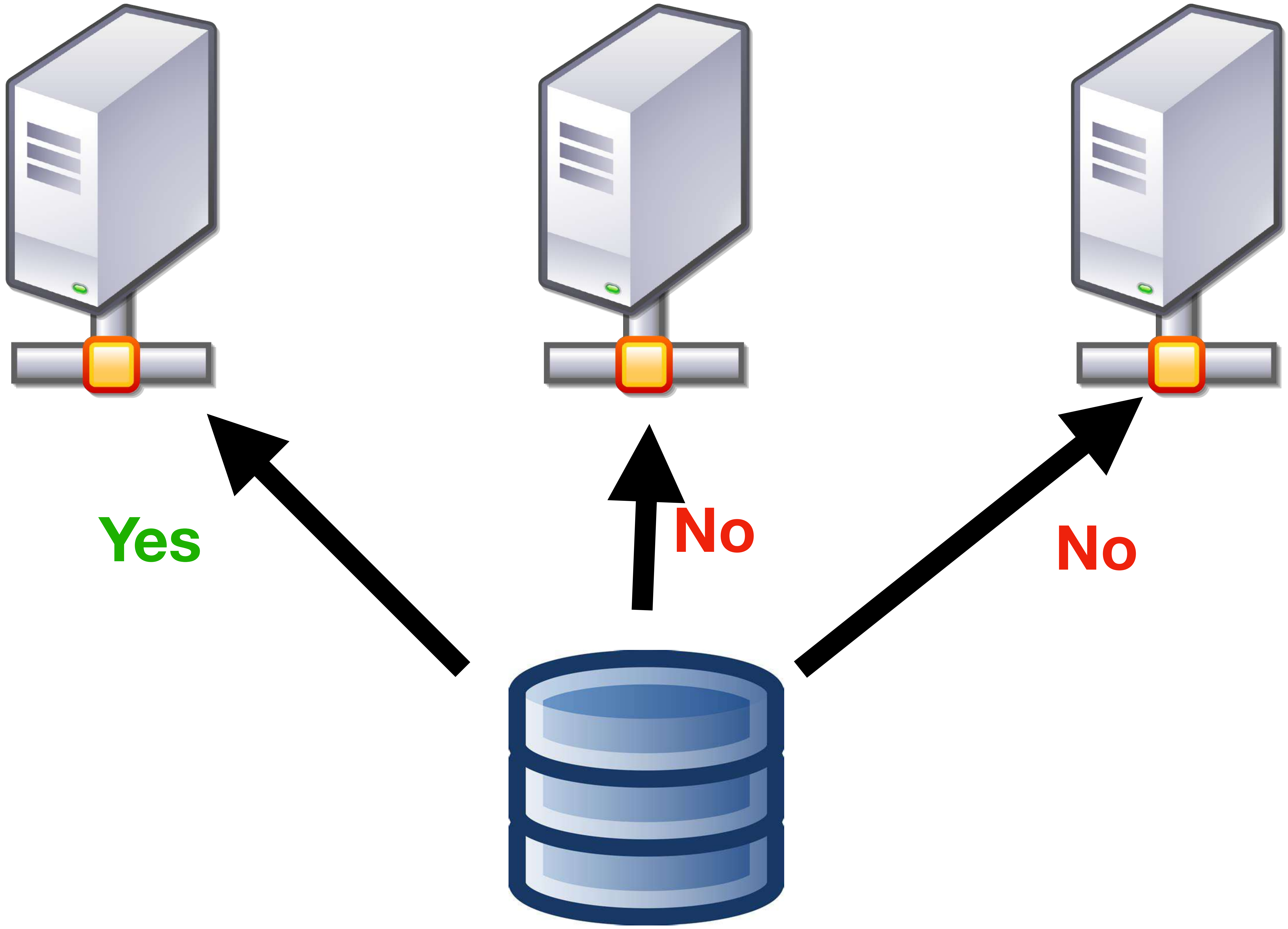
Thread-2 → pair.**effects**.run(dependencies);

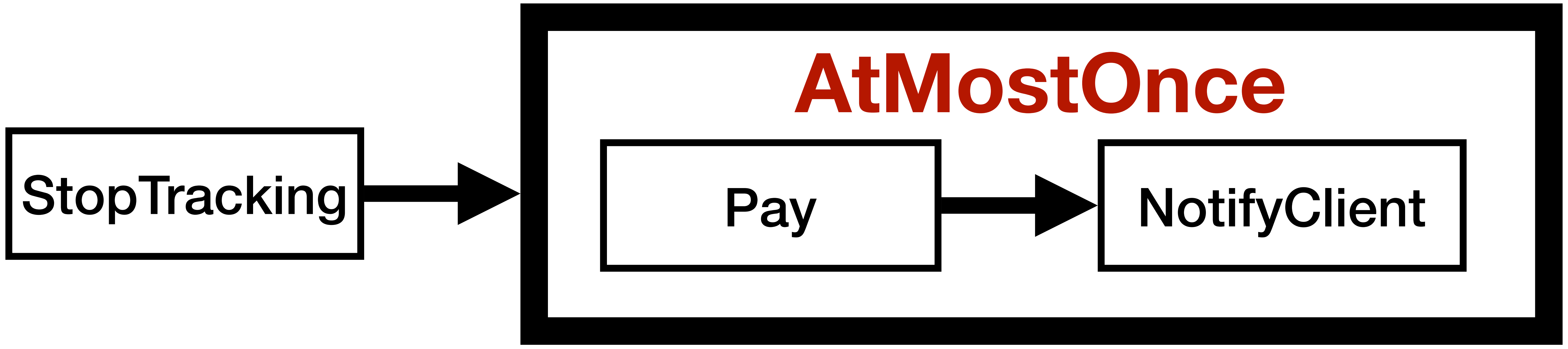
Thread-1 → pair.**effects**.run(dependencies);

Agents (=~~~ Actors)




Can I pay?
Can I pay?
Can I pay?
ACID


Yes
No
No


StopTracking
AtMostOnce
Pay
NotifyClient

Co-Effects

kafka

Event Sourcing

🫀

Functional Programming

Event Sourcing

❤

Functional Programming

```
public class KafkaConsumer {

    private AllDependencies allDependencies;
    private TheStateHolder theStateHolder;

    public void run() {
        while (!stop) {
            Bet bet = readNext();
            Effects effects = theStateHolder.event(bet);
            effects.run(allDependencies);
        }
    }

…
}
```

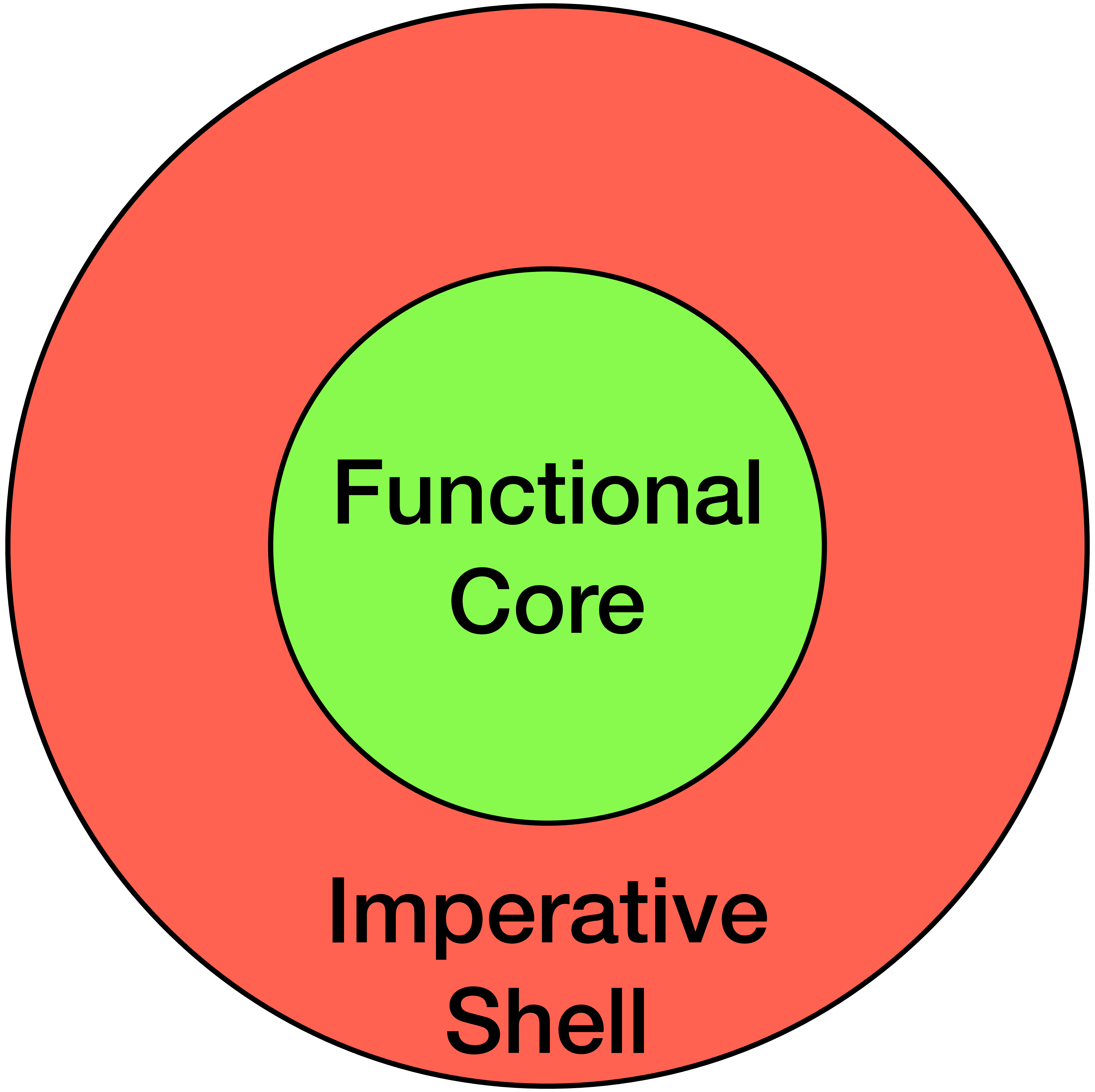
```
public class KafkaConsumer {

    private AllDependencies allDependencies;
    private TheStateHolder theStateHolder;

    public void run() {
        while (!stop) {
            Bet bet = readNext();
            Effects effects = theStateHolder.event(bet);
            effects.run(allDependencies);
        }
    }

…
}
```

```
public class KafkaConsumer {

    private AllDependencies allDependencies;
    private TheStateHolder theStateHolder;

    public void run() {
        while (!stop) {
            Bet bet = readNext();
            Effects effects = theStateHolder.event(bet);
            effects.run(allDependencies);
        }
    }

…
}
```

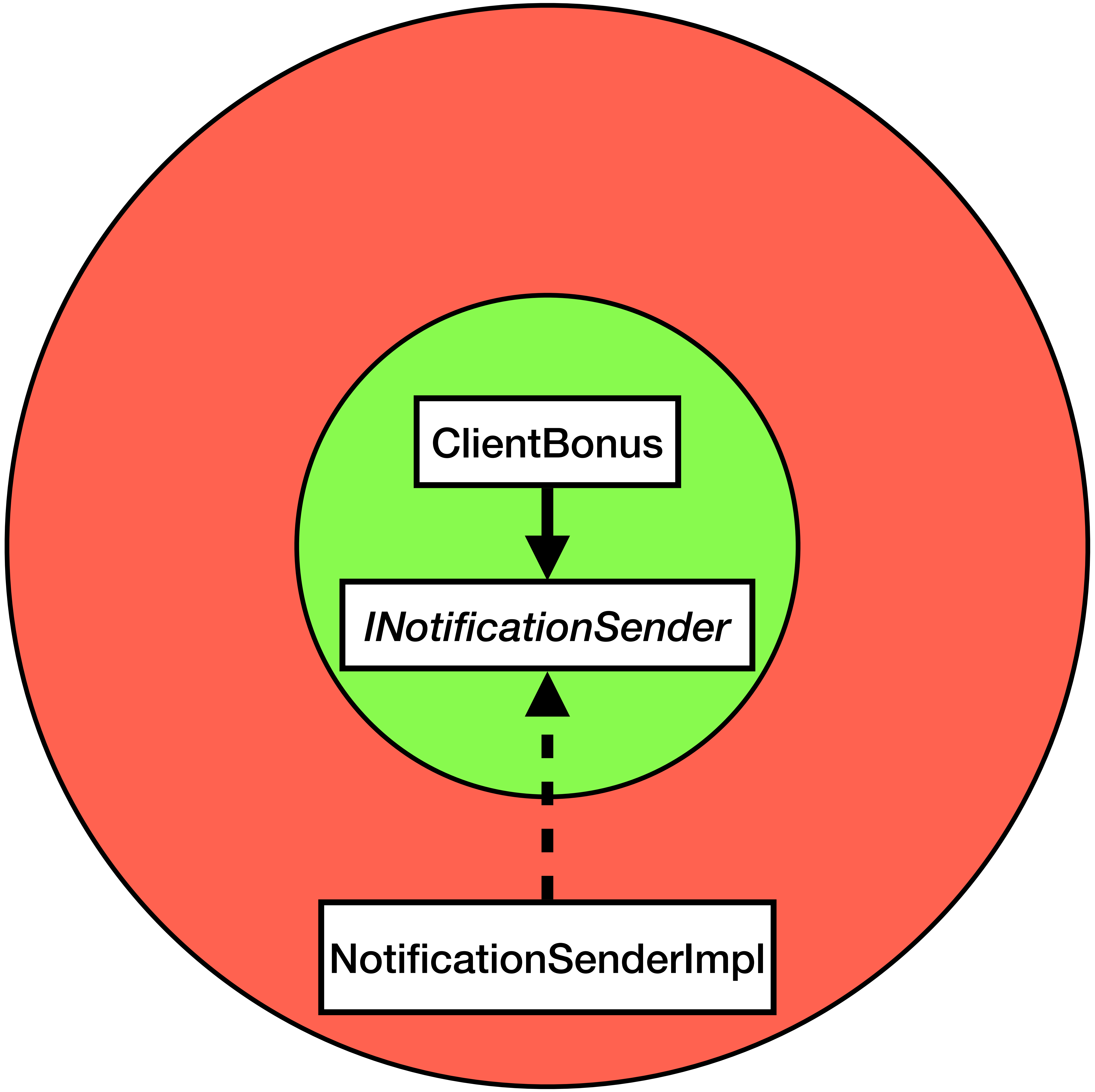
```
public class KafkaConsumer {

  private AllDependencies allDependencies;
  private TheStateHolder theStateHolder;

  public void run() {
    while (!stop) {
      Bet bet = readNext();
      Effects effects = theStateHolder.event(bet);
      effects.run(allDependencies);
    }
  }

…
}
```

```
public class KafkaConsumer {

    private AllDependencies allDependencies;
    private TheStateHolder theStateHolder;

    public void run() {
        while (!stop) {
            Bet bet = readNext();
            Effects effects = theStateHolder.event(bet);
            effects.run(allDependencies);
        }
    }

…
}
```

- No getters
- No locks or synch blocks
- No try/catch
- No logging
- No mocks
- No useless interfaces



https://www.destroyallsoftware.com/screencasts/catalog/functional-core-imperative-shell


ClientBonus
INotificationSender
NotificationSenderImpl

Dynamic (vs Static) typing

```
clientBonus = Map.of(
        "client", Map.of("id", "123233"),
        "deposits",
        List.of(
                Map.of("amount", 3,
                            "type", "CASH"),
                Map.of("amount", 234,
                            "type", "CARD")));
```

```java
public class Bet {
    private String id;
    private int amount;
    private long timestamp;
}
```

“It is better to have 100 functions operate on one data structure than 10 functions on 10 data structures.”

— **Alan Perlis**

```
{:type :bet
 :id "client1"
 :amount 23
 :timestamp 123312321323}
```

```
{:type :bet
 :id "client1"
 :amount 23
 :timestamp 123312321323}
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

```
{:request-method :get
 :uri            “/foobaz"
 :query-params   {"somekey" “somevalue"}
 :headers        {"accept-encoding" "gzip, deflate"
                  "connection"      "close"}
 :body           nil
 :scheme         :http
 :content-length 0
 :server-port    8080
 :server-name    “localhost"}
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

```clojure
{:status  200
 :headers {"Content-Type" "text/html"}
 :body    "Hello World"}}
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

```
{:select [:id :client :amount]
 :from   [:transactions]
 :where  [:= :client "a"]}
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

```
[{:id 1 :client 32 :amount 3}
 {:id 2 :client 87 :amount 7}
 {:id 3 :client 32 :amount 4}
 {:id 4 :client 40 :amount 6}]
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

```
[:html
    [:body
        [:p "Count: 4"]
        [:p "Total: 20"]]]
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

```
{:web-server          {:listen 8080}
 :db-config           {:host     "xxxx"
                       :user     "xxxx"
                       :password "xxxx"}
 :http-defaults       {:connection-timeout 10000
                       :request-timeout    10000
                       :max-connections    2000}
 :user-service        {:url “http://user-service”
                       :connection-timeout 1000}}
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

```
{:id          :string
 :name        :string
 :deposits    [{:id          :string
                :amount      :int
                :timestamp   :long}]}
```

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

- Business logic
- Infrastructure code
- Configuration
- Reflection/Metadata

apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

Dynamic (vs Static) development

REPL

REPL Driven Development

Dan Lebrero

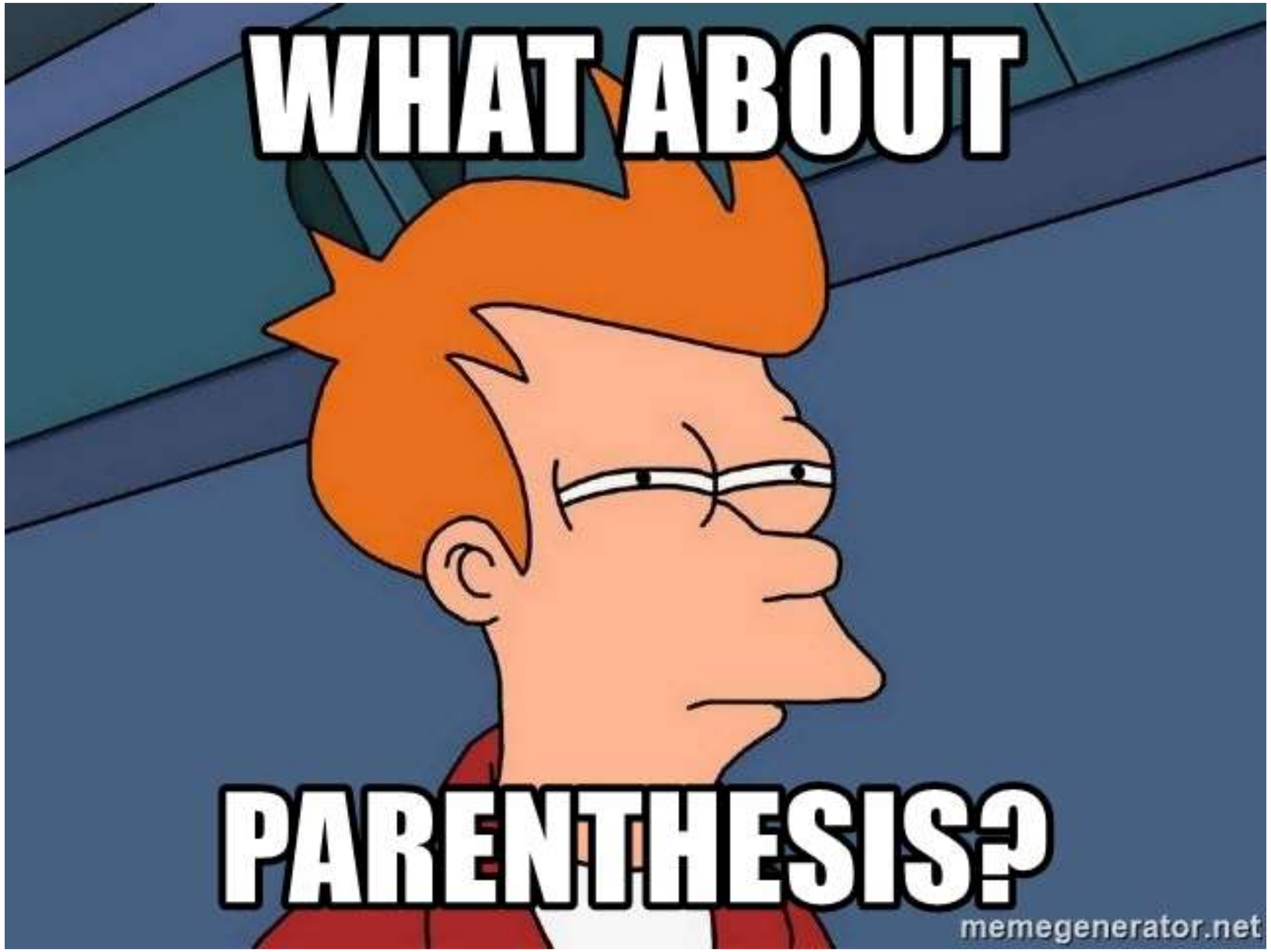
geecon

Prague, 18-19 October 2018

https://danlebrero.com/repl

Lisp (vs Fortan)

NO MORE ;
NO MORE ,
memegenerator.net


WHAT ABOUT
PARENTHESIS?
memegenerator.net

```
.filter(removeCsvHeaders(firstHead
.map(splitCsvString())
.map(convertCsvToMap(csvHea
.map(convertToJson(eventCrea
```

16

```
(filter not-header
(map parse-cs
(map (partial
(map ->event
```

10

-40%!!!

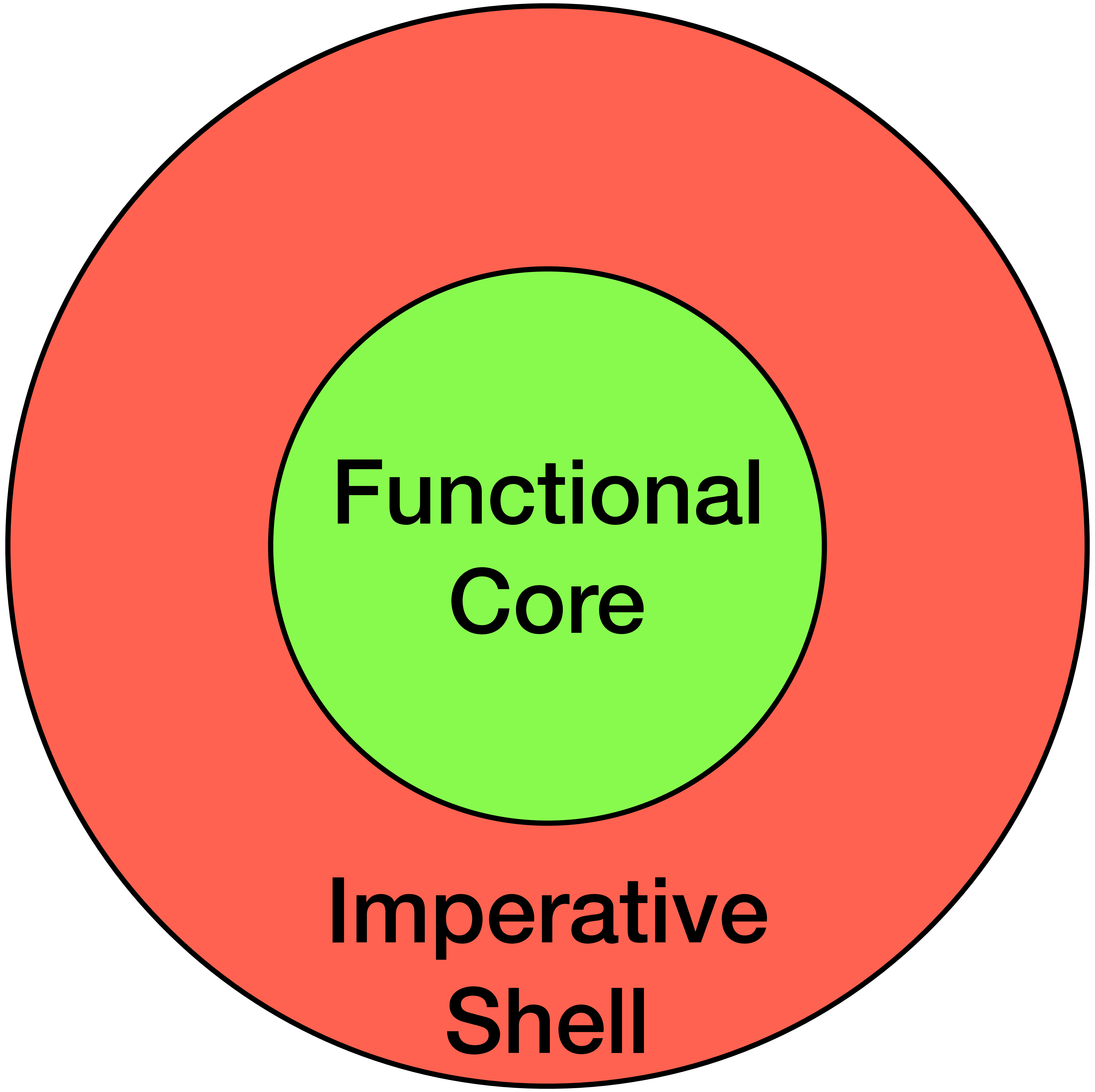
JAVA APPS...
MORE PARENS THAN CLOJURE?
AND TEN TIMES MORE CODE
imgflip.com

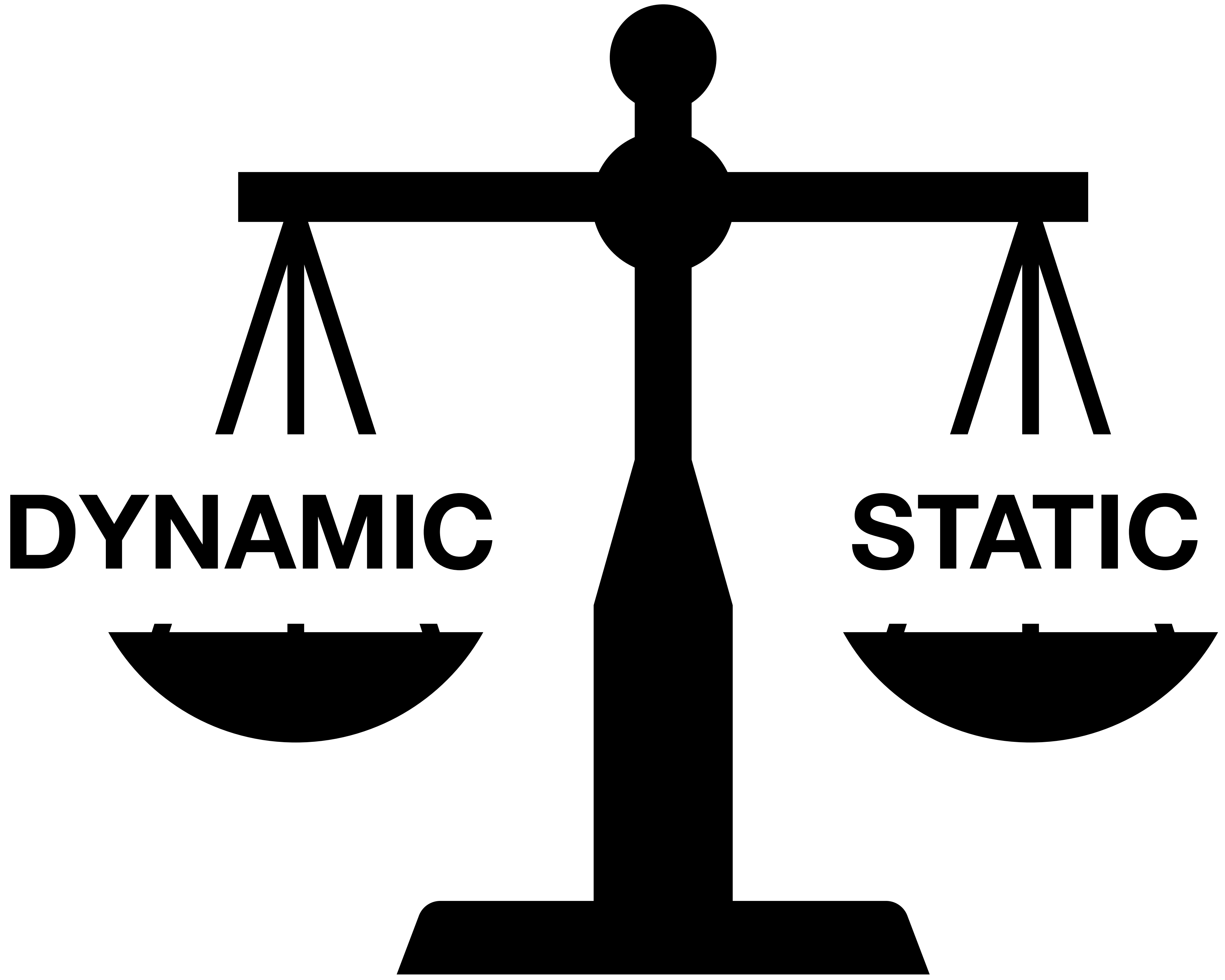
```
List.of(
    new Symbol("defn"),
    new Symbol("plus-one"),
    List.of(
        new Symbol("a"),
        new Symbol("b")),
    Map.of(
        new Keyword("time"), List.of(new Symbol("System/currentTimeMillis")),
        new Keyword("result"), List.of(
            new Symbol("+"),
            new Symbol("a"),
            new Symbol("b"),
            new Long(1))));
```

```
List.of(
    new Symbol("defn"),
    new Symbol("plus-one"),
    List.of(
        new Symbol("a"),
        new Symbol("b")),
    Map.of(
        new Keyword("time"), List.of(new Symbol("System/currentTimeMillis")),
        new Keyword("result"), List.of(
            new Symbol("+"),
            new Symbol("a"),
            new Symbol("b"),
            new Long(1))));
```

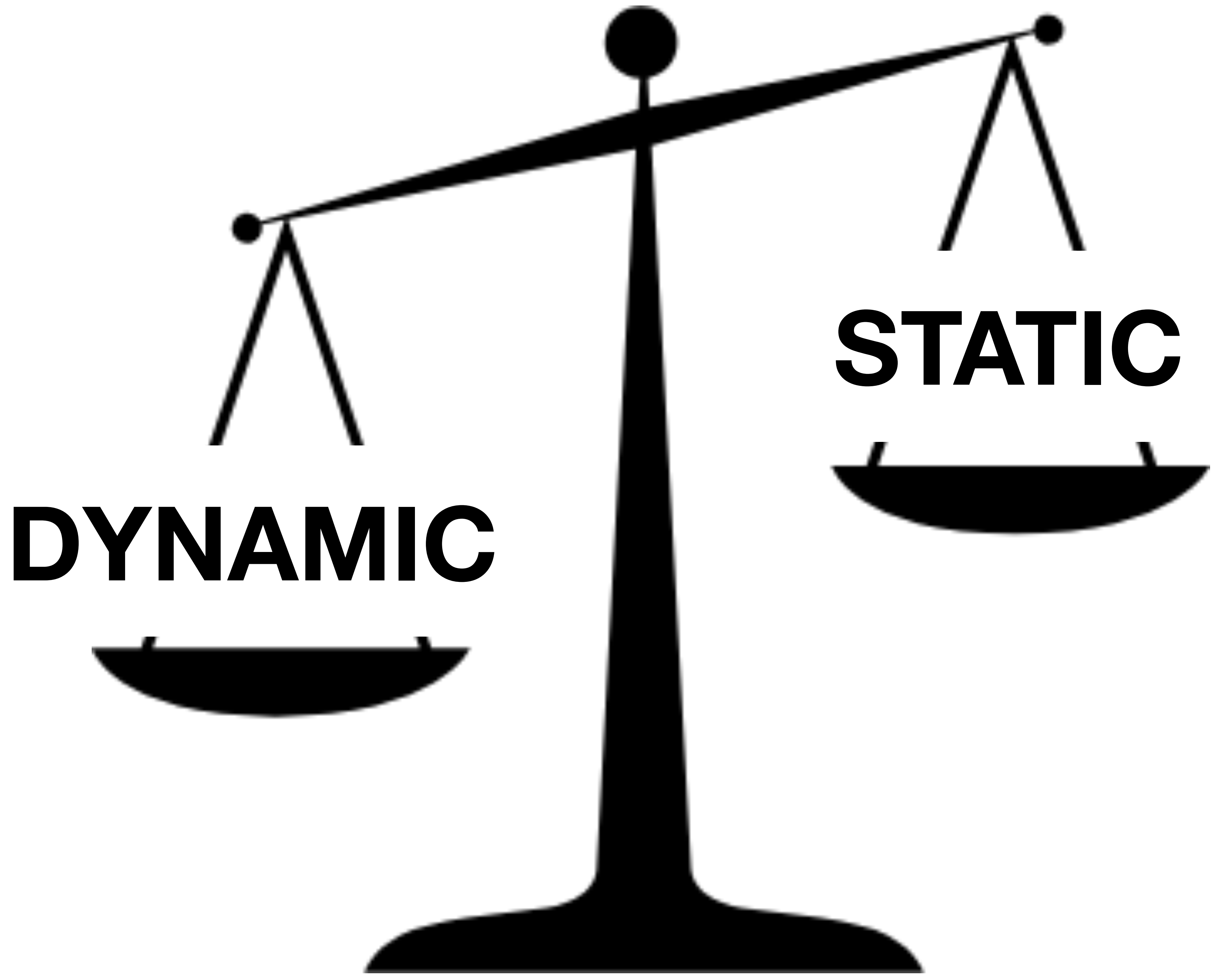
apply butlast concat cons count cycle diff distinct distinct? drop drop-last drop-while empty empty? every? ffirst filter first flatten fnext for frequencies group-by interleave interpose into into-array keep keep-indexed last lazy-cat map map-indexed mapcat next nfirst nnext not-any? not-empty not-every? nth nthnext partition partition-all partition-by pmap postwalk prewalk rand-nth reduce reductions remove replace rest reverse second seq? seque set shuffle some sort sort-by split-at split-with take take-nth take-while to-array-2d vec walk when-first assoc pop subvec replace conj rseq update-in update get get-in contains? find keys vals assoc assoc-in dissoc merge merge-with select-keys update-in update rename-keys map-invert reduce-kv dissoc-in disj join select project union difference intersection index

Summary


Functional Core
Imperative Shell



TYPES

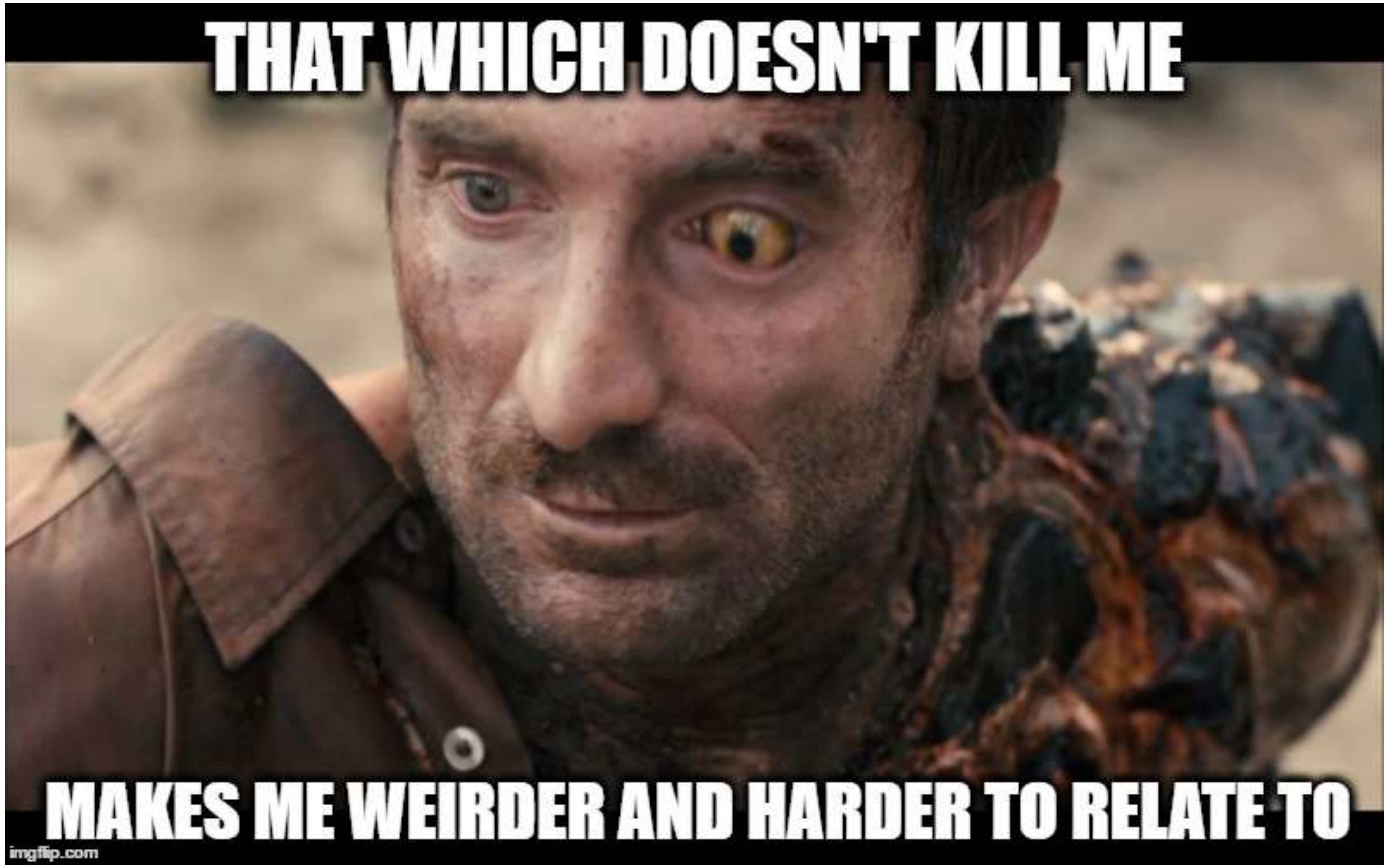


DEVELOPMENT

“A language that doesn’t affect the way you think about programming, is not worth knowing.”

— **Alan Perlis**

Microsoft
Language
Pascal
ABAP
SYMPL
C/AL
Cesil
Cg
Scala
HAL/S
Rust
Bistro
GDL
JOSS
Genie
Inform
SenseTalk
Blue
SIMSCRIPT
Revolution
Sawzall
Chapel
Rapira
SIMPLE
CSKA
ALGOL
DATATRIEVE
PL/I
ELAN
C
Ajax
Agora
Clojure
Object
Autolt
Microcode
code
Lynx
SAM76
Redcode
Logo
Not
Trac
Adenine
SALSA
ATS
Fjölnir
Fuxi
Assembly
TACPOL
FP
shell
language
E
A+
Lisp
VB
MicroScript
Squirrel
Windows/Dos
Kaleidoscope
IMP
C++
Fancy
SuperTalk
P
K
JavaScript
FL
JavaFX
R
Forth
b
T
Newspeak
I.C-3
ABC
Visual
D
GameMonkey
Prolog
GJ
L
XOTcl
dc
Cayenne
Go
Delphi
DASL
Caml
Z
ECMAScript
JASS
bc
GM
Constraint
S
EusLisp
ObjectScript
Oxygene
PEARL
G
Programming
Java
BLISS
notation
Script
PHP
Ruby
EXEC
RPG
M
Cyclone
CorVision
Action
REXX
F
RPL
Little
NET
Q
Simula
Boomerang
COBOL
GraphTalk
A
ChucK
Common
Agena
Strand
Turing
ISWIM
PL/SQL
Python
ORCA/Modula-2
rex
S/SL
Arc
Moby
Uniface
J
Robot
SP/k
HaXe
Eiffel
Coral
WATFOR


THAT WHICH DOESN'T KILL ME
MAKES ME WEIRDER AND HARDER TO RELATE TO
imgflip.com

???

@DanLebrero
dlebrero@gmail.com
danlebrero.com

akvo.org

@Akvo
https://akvo.org/

[illegible]