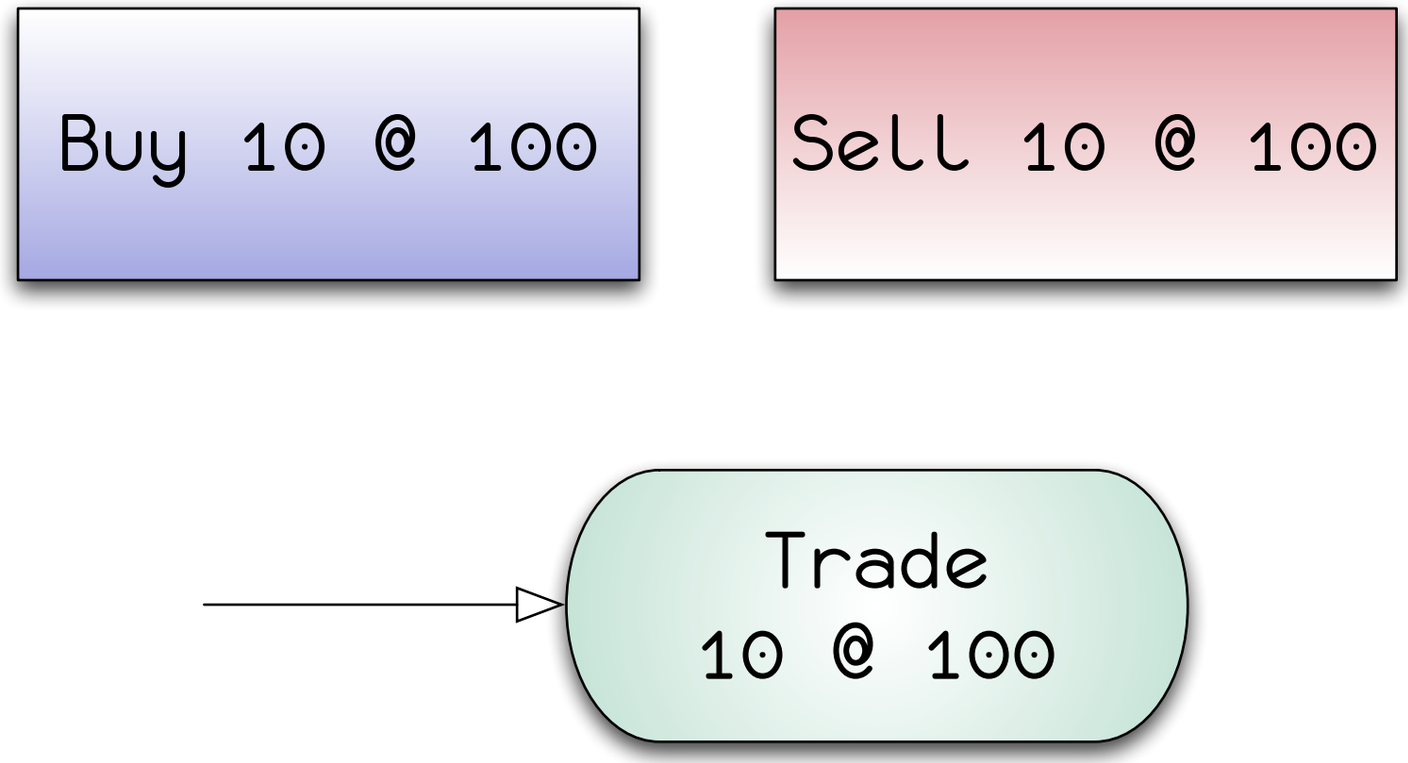
The Erlang Stock Exchange

- Kenny Stone
- Trading Systems
- C/C++, Ruby, Erlang

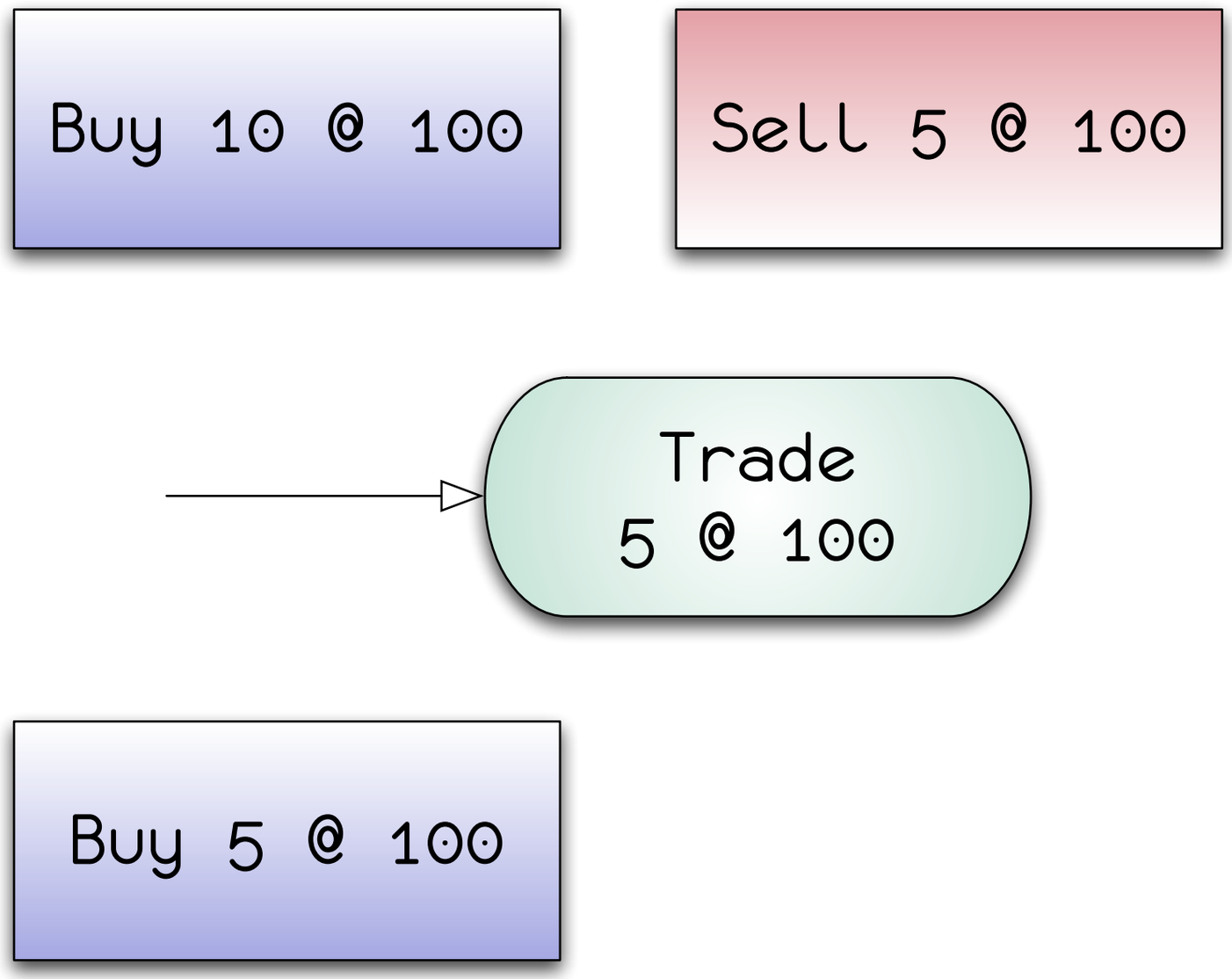
What is a stock exchange?

- Electronically match buyers and sellers
- “Matching engine”
- Stock exchange = matching engine for stocks

Matching Engine



Matching Engine



Keeping the Book

Buy 10 @ 110

Buy 20 @ 105

Buy 10 @ 100

Keeping the Book

Buy 25 @ 110

Buy 10 @ 110

Sell 10 @ 110

105

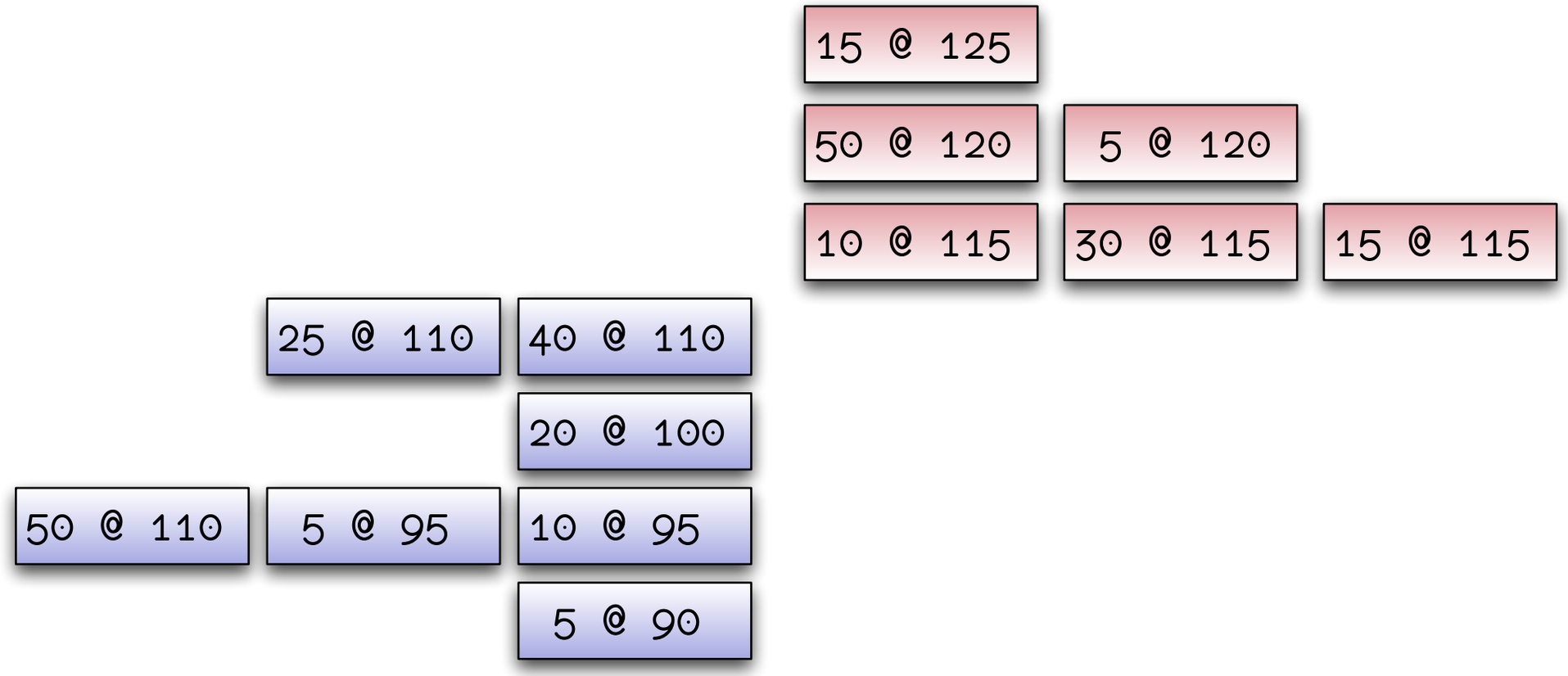
Buy 30 @ 105

Buy 20 @ 105

Buy 10 @ 100

Trade
10 @ 110

Keeping the Book

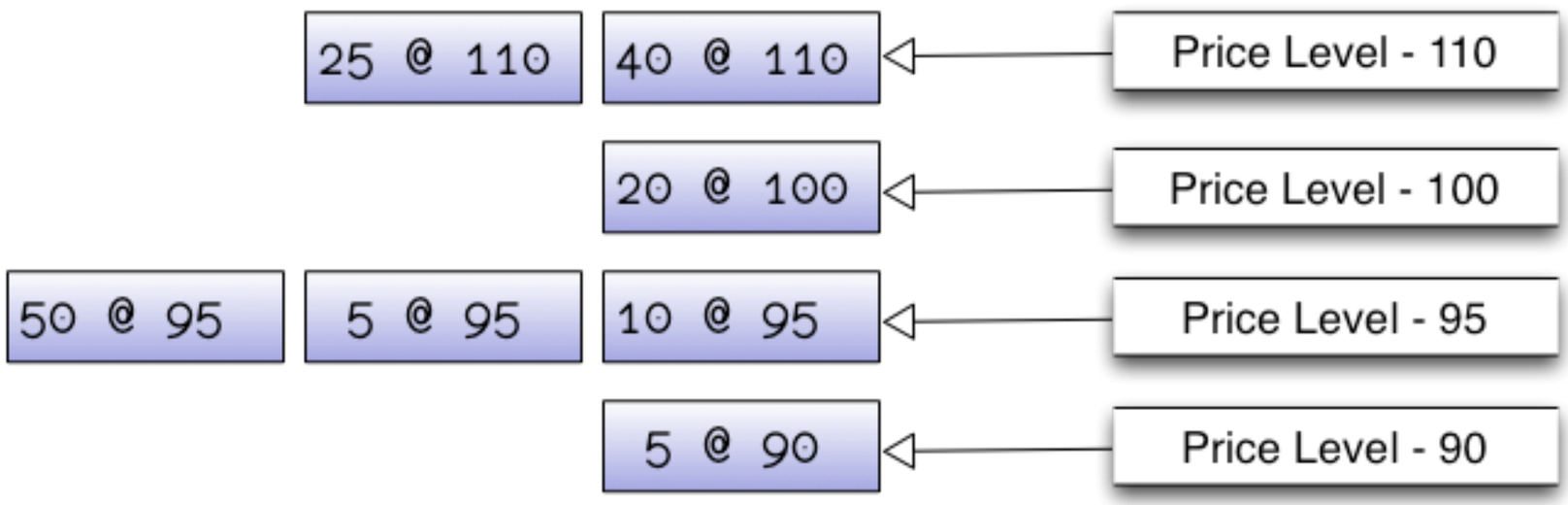


Matching Engine

- Simple Price/Time Priority Algorithm
- Keep the book
- N actors
- Why Erlang?

The Concurrent Engine

- Price Levels
- Insert and Match Operations can happen concurrently per price level
- Linked List of Price Levels



The Concurrent Engine

NEW ORDER: 10 @ 95

50 @ 110

25 @ 110

40 @ 110

20 @ 100

10 @ 95

5 @ 95

5 @ 90

The Concurrent Engine

NEW ORDER: 65 @ 110

50 @ 110

25 @ 110

40 @ 110

TRADE - 65 @ 110

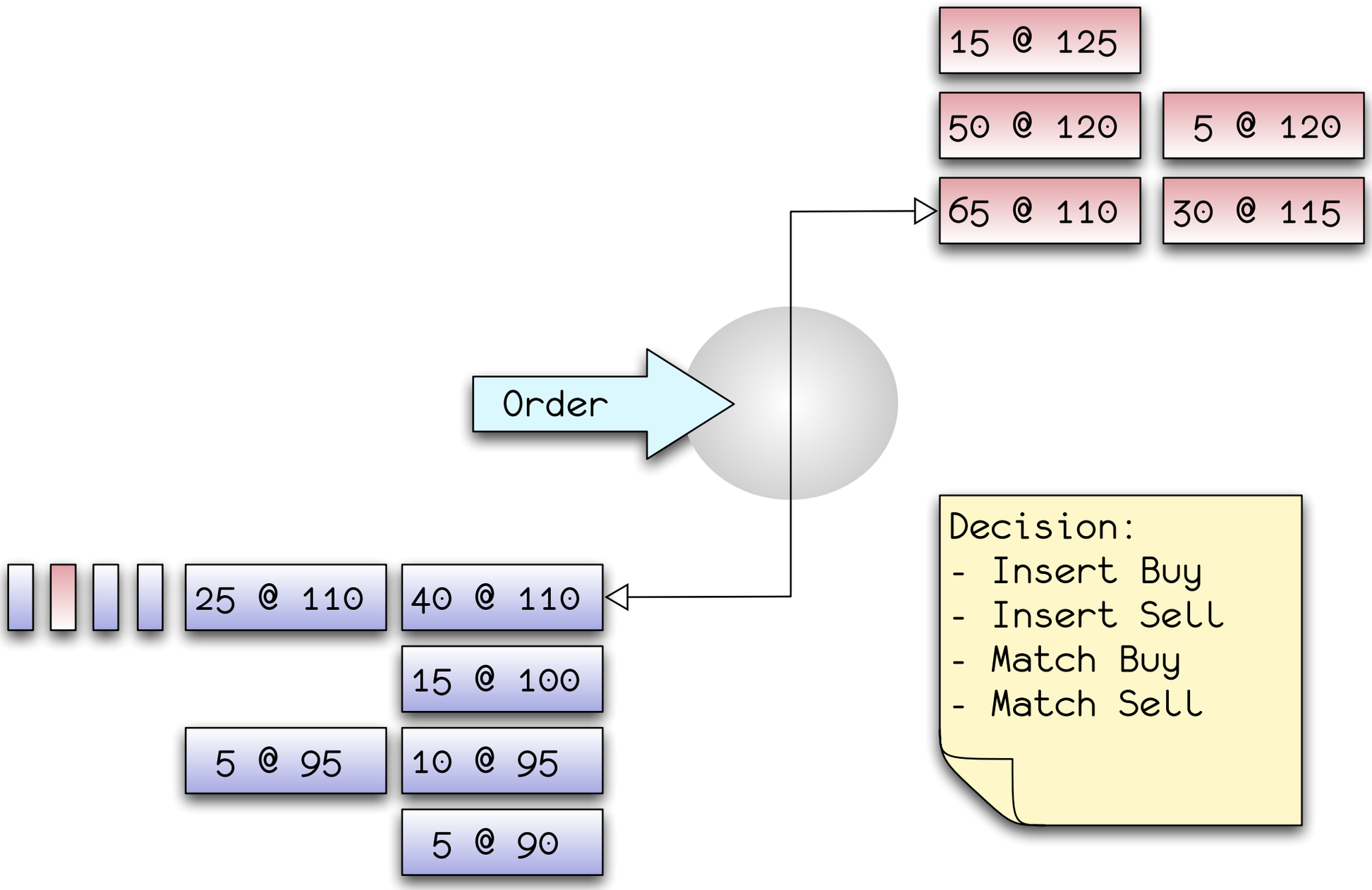
20 @ 100

10 @ 95

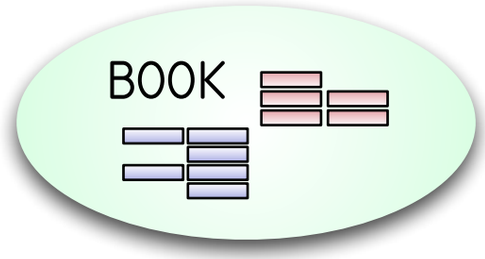
5 @ 95

5 @ 90

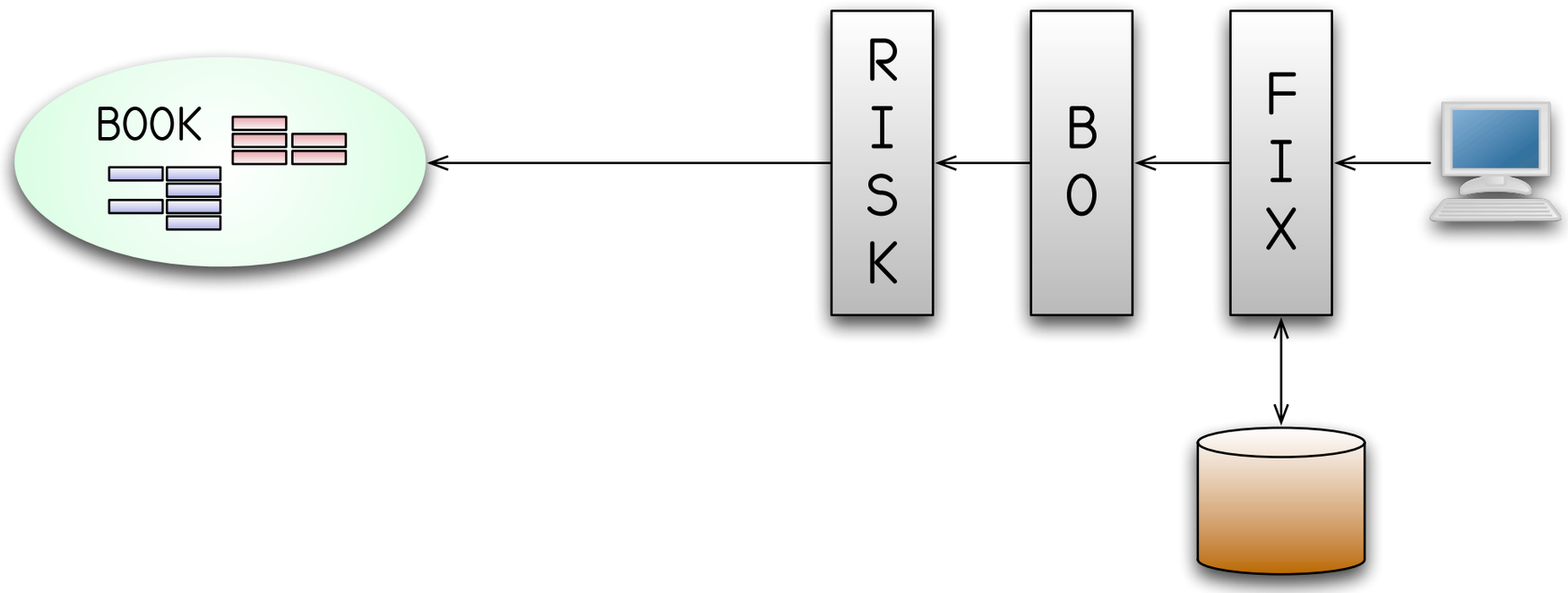
The Concurrent Engine



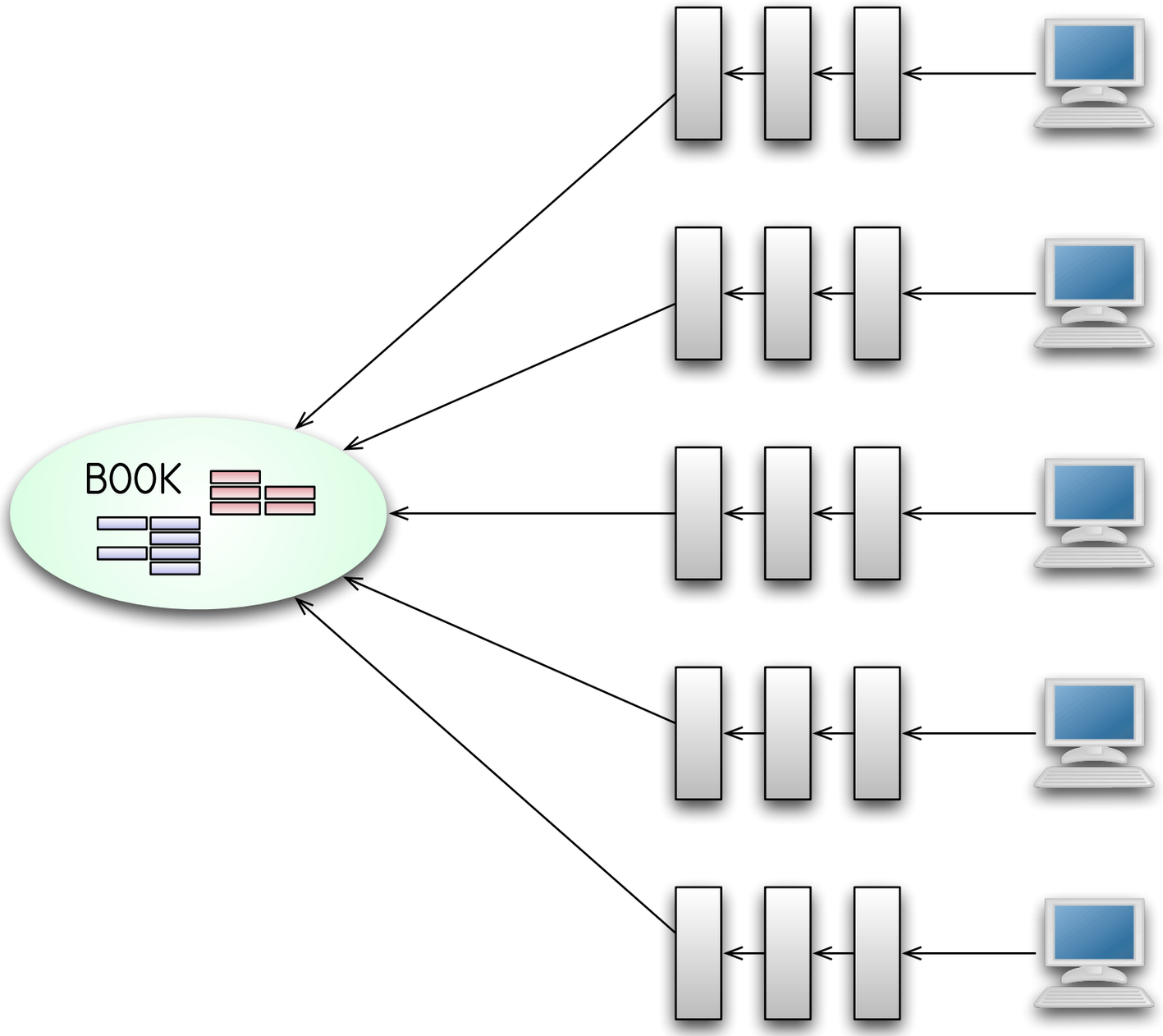
The Concurrent Engine



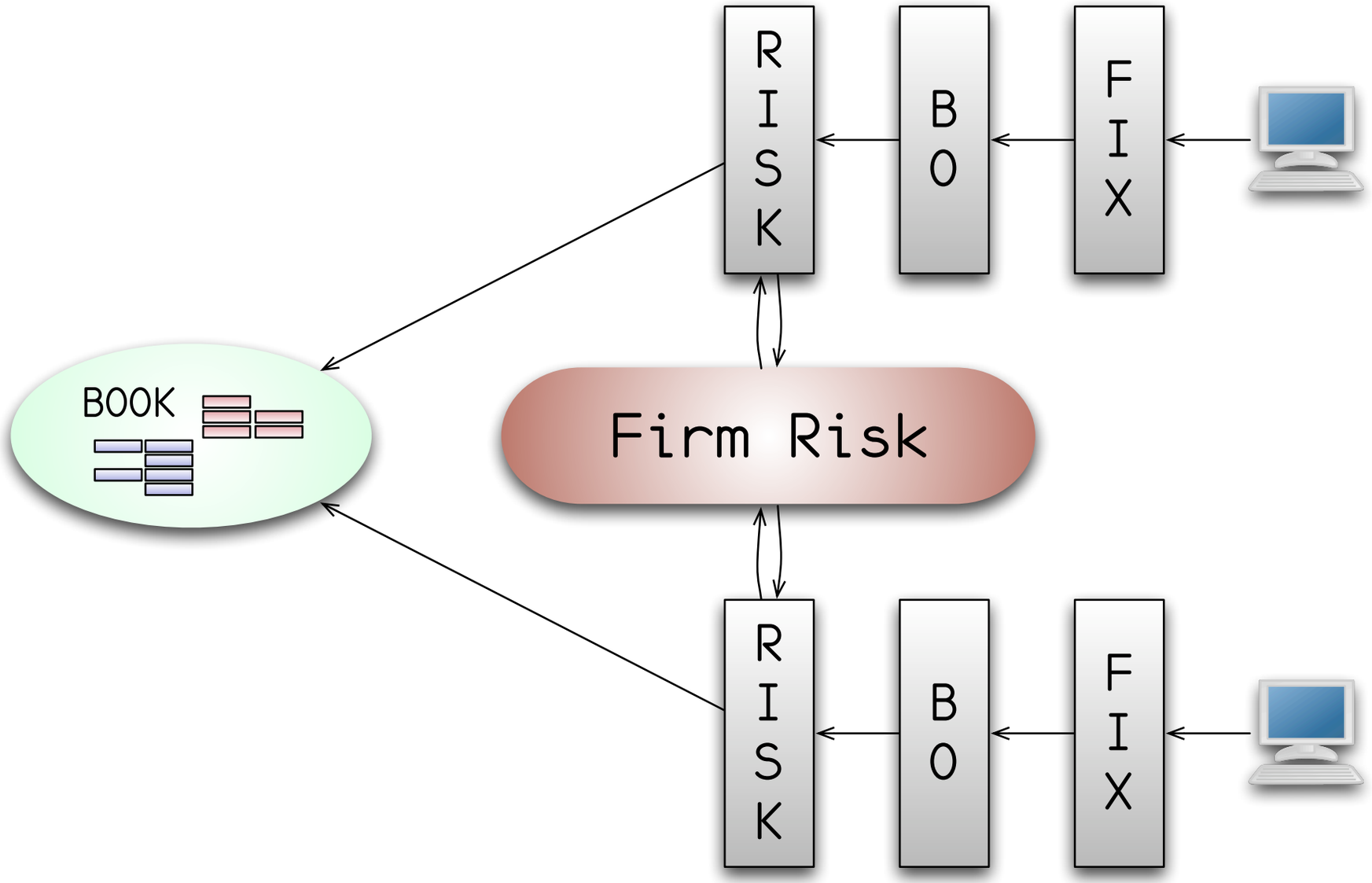
The Concurrent Engine



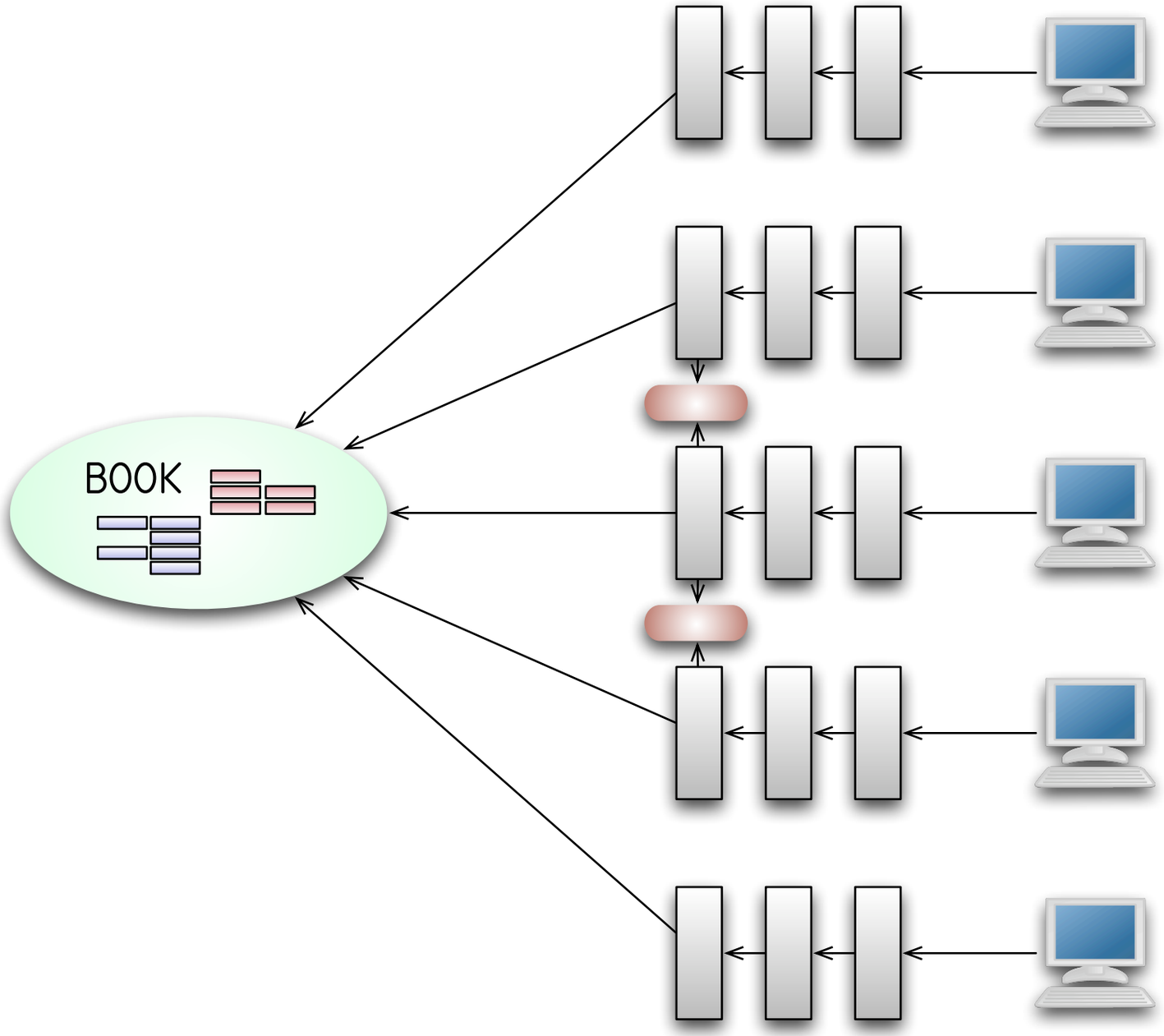
The Concurrent Engine



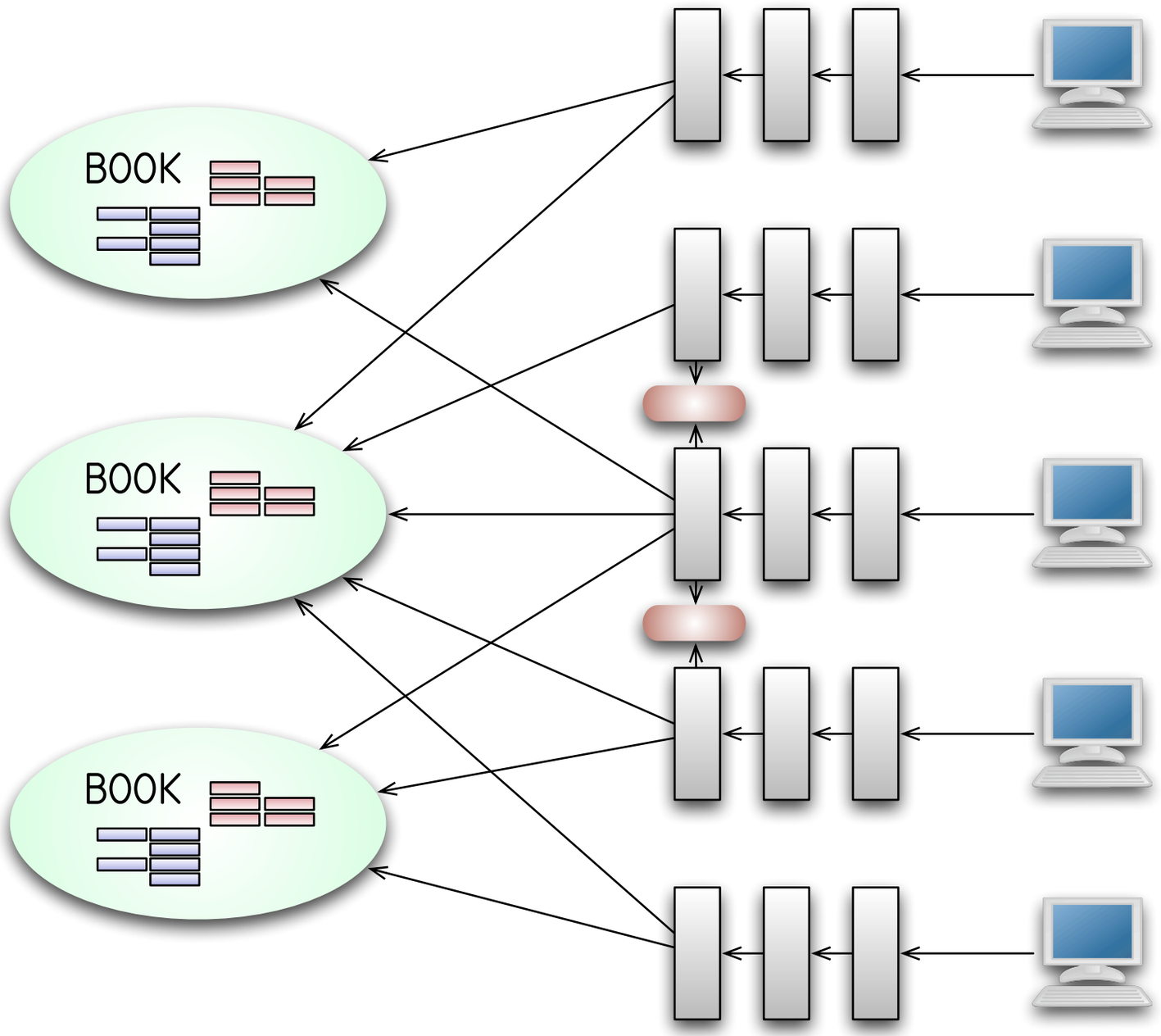
The Concurrent Engine



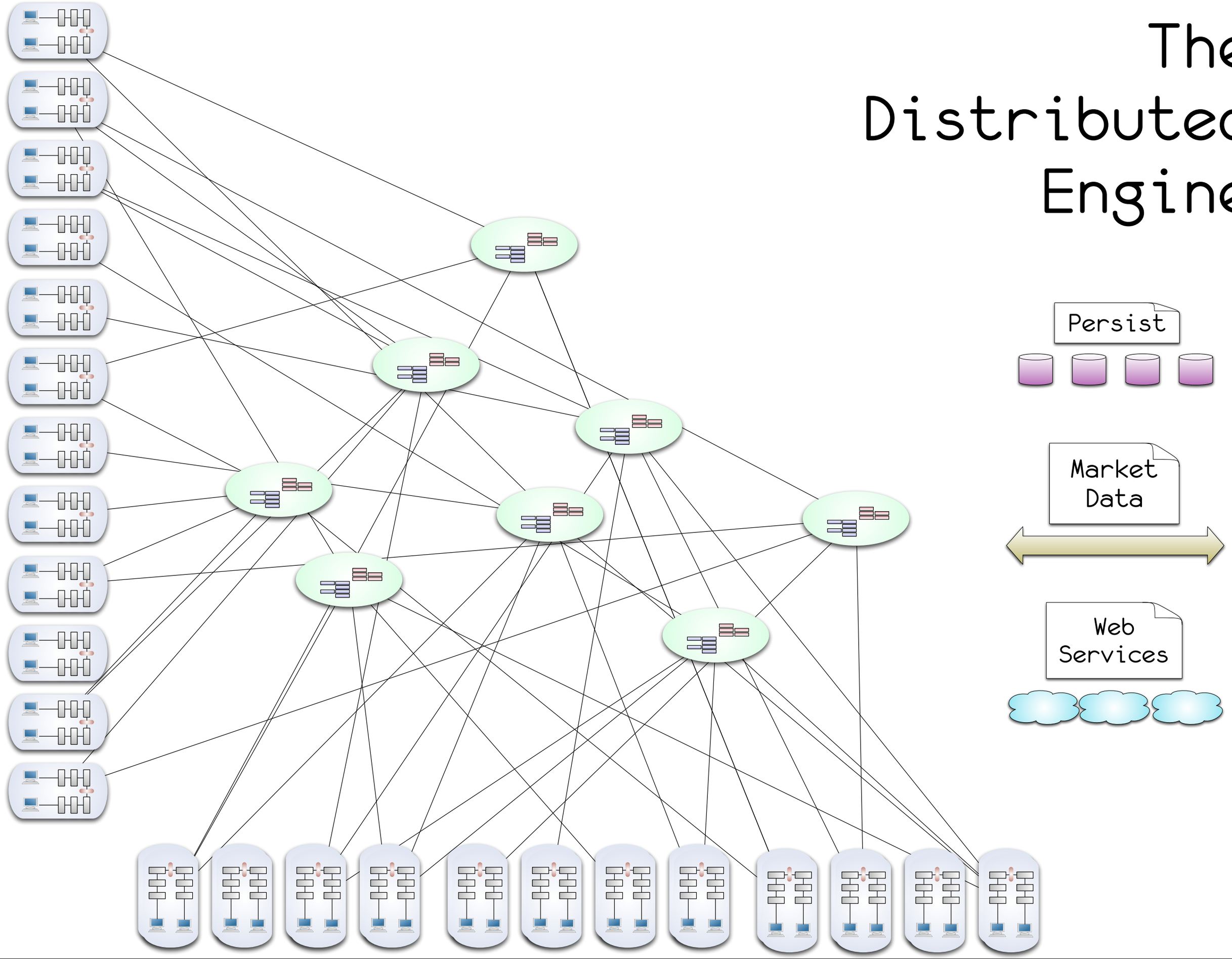
The Distributed Engine



The Distributed Engine



The Distributed Engine

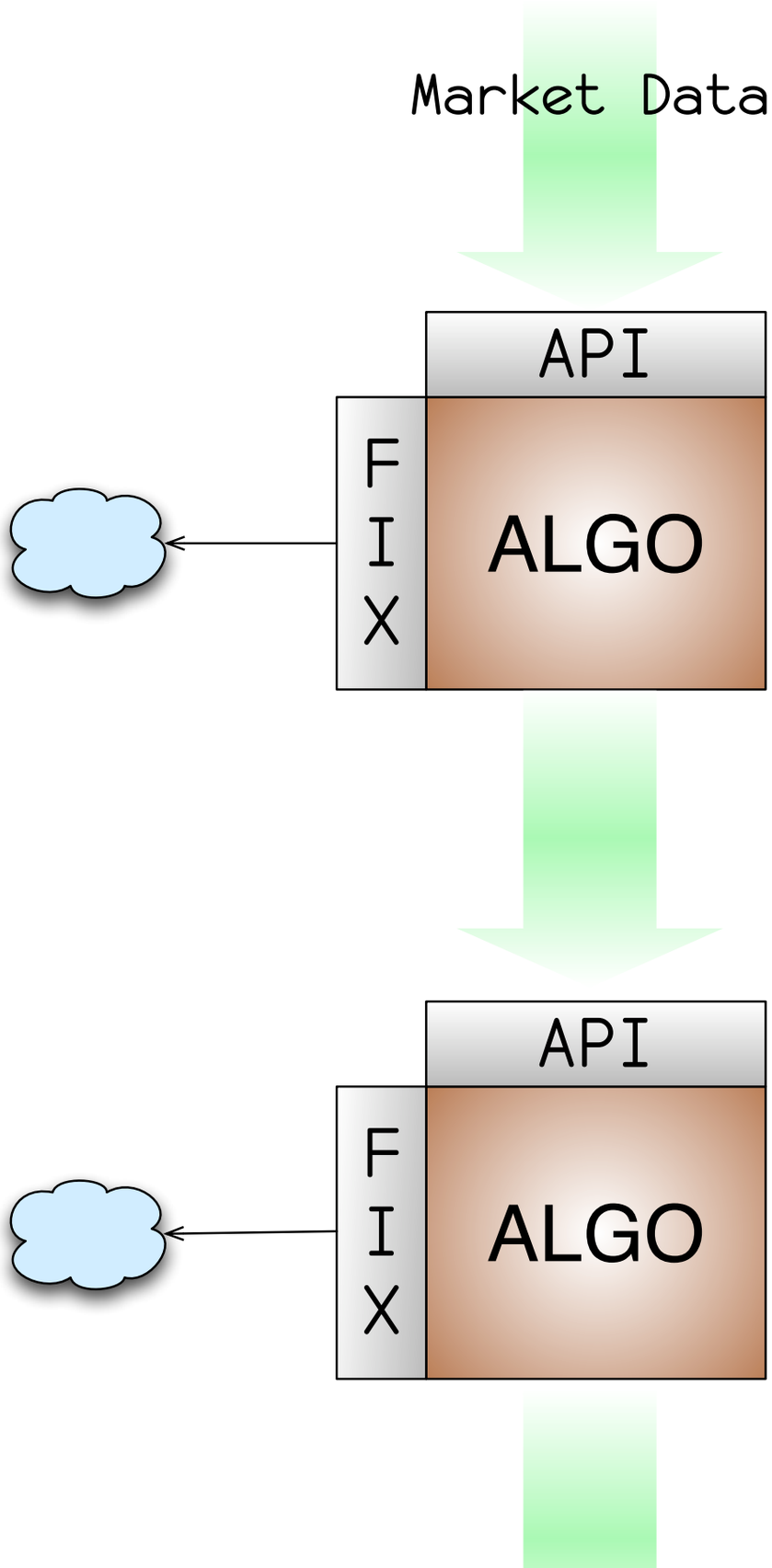


By using Erlang,
I can build one and
scale to many

The Trading Industry

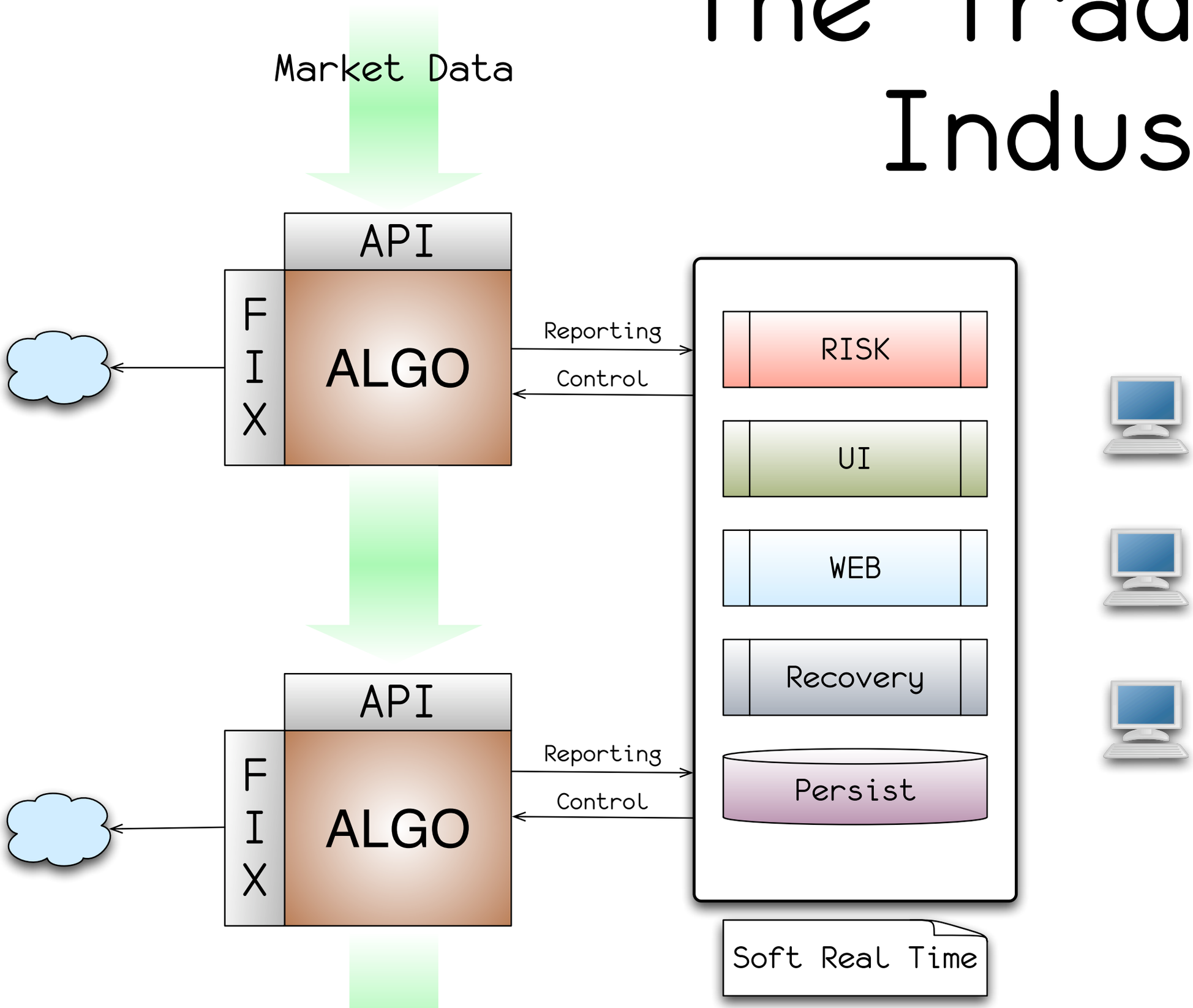
- High frequency, low latency
- µSeconds matter
- The critical path cannot be compromised

The Trading Industry



- Market Data Input, Orders Output
- Critical path can be very latency sensitive
- Need systems around Algo engine

The Trading Industry



The Trading Industry

- Challenges of Erlang
  - It's very different.
    - Customers aren't going to ask for it.
    - Developers don't know anything about it.
- Software is about choosing the right tools.
- A business is about execution.

The Trading Industry

- What we're building:
  - Highly event driven systems
  - Distributed systems
  - Systems that scale
  - Systems that are fault tolerant

Thanks

Kenny Stone
kstone@connamara.com