Gaming and Betting

Distributed computing Enterprise challenges

Sławek Zajac, Technical Director at Grand Parade (a William Hill company)

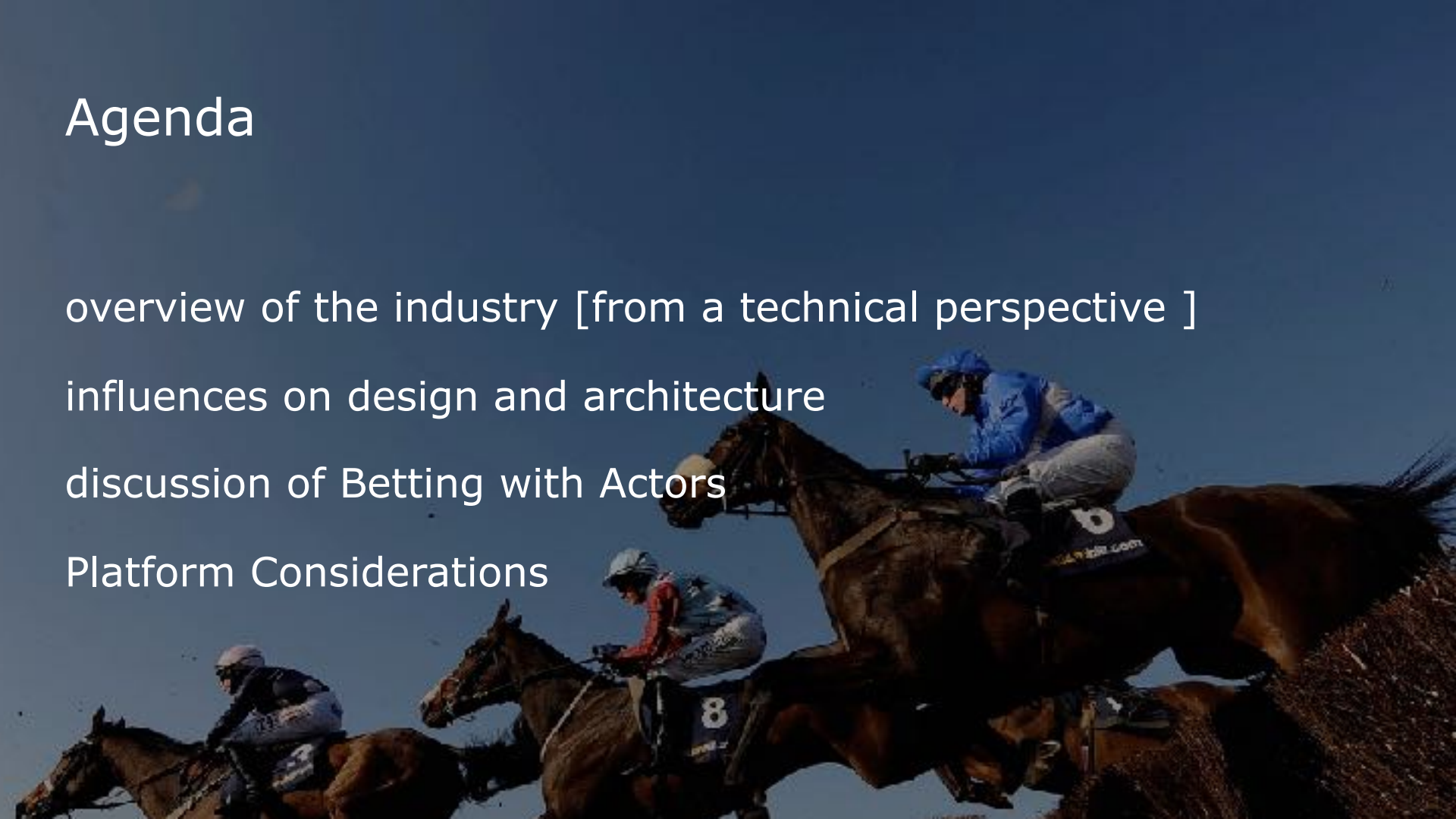
Agenda

overview of the industry [from a technical perspective]

influences on design and architecture

discussion of Betting with Actors

Platform Considerations



Betting Today

Poker

Blackjack

Casino/Slots

Sports

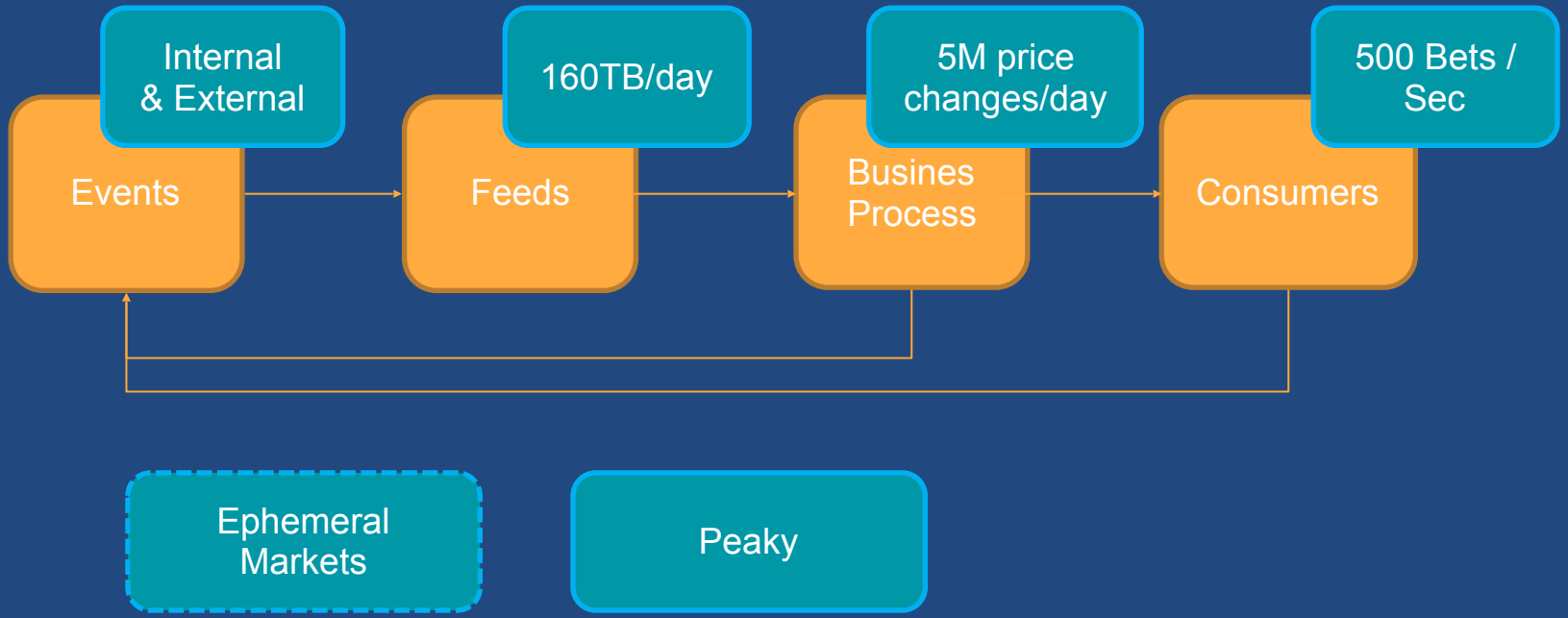
Example: Tennis

Federer vs Nadal, 5th set Australian Open 2017

Nadal leads 3:2 in the fifth

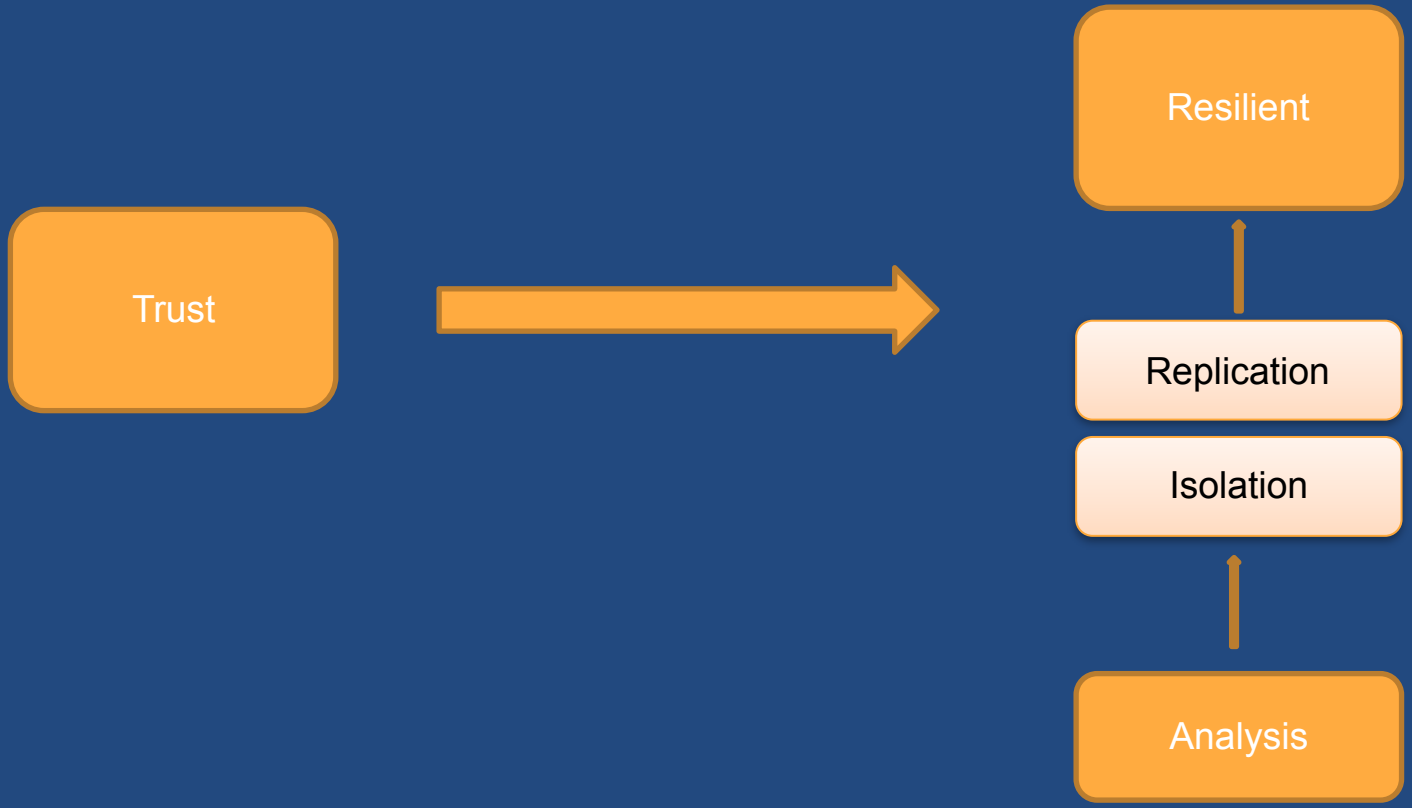
Federer breaks to 3:3

View of the data

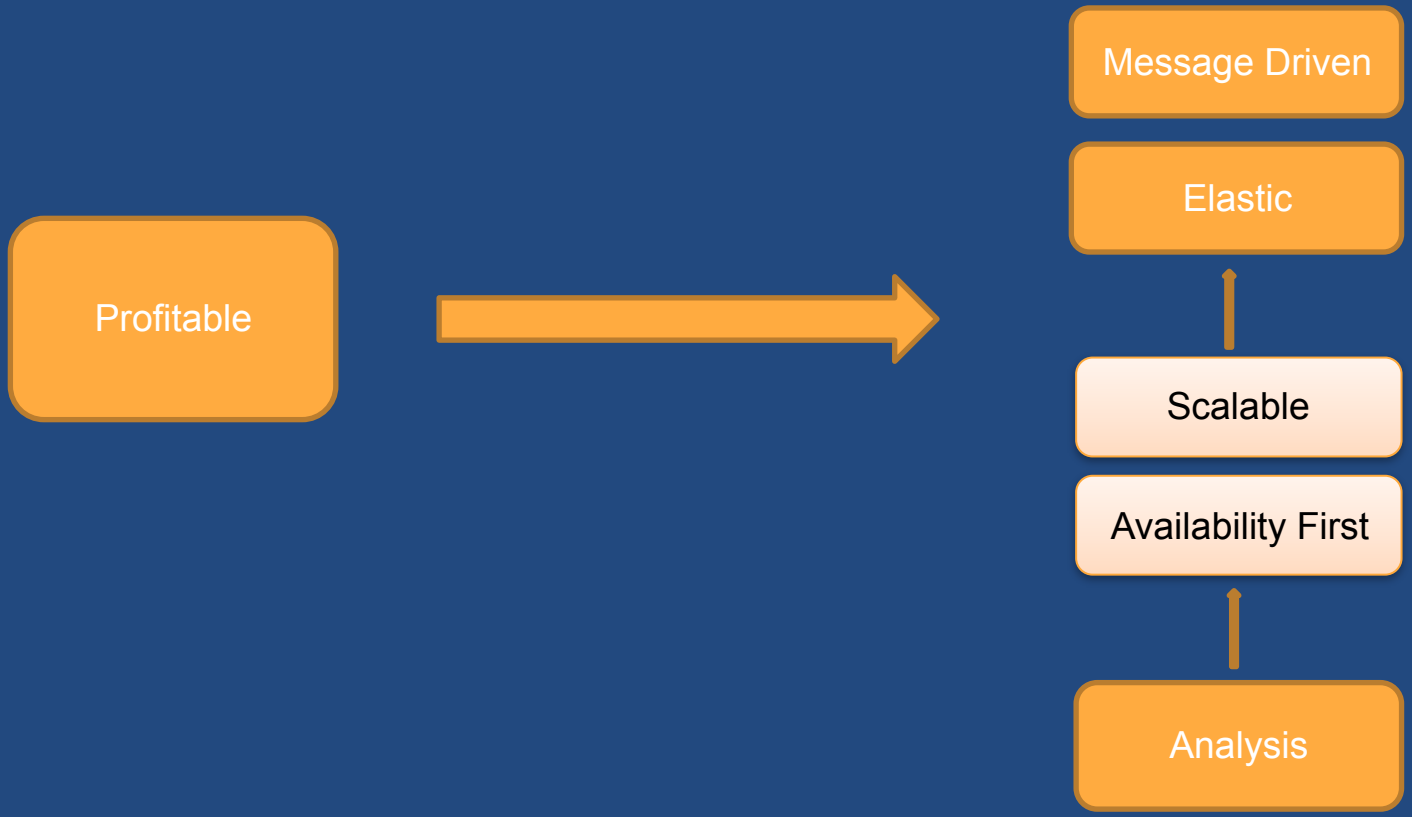


Choose Responsive to provide entertainment

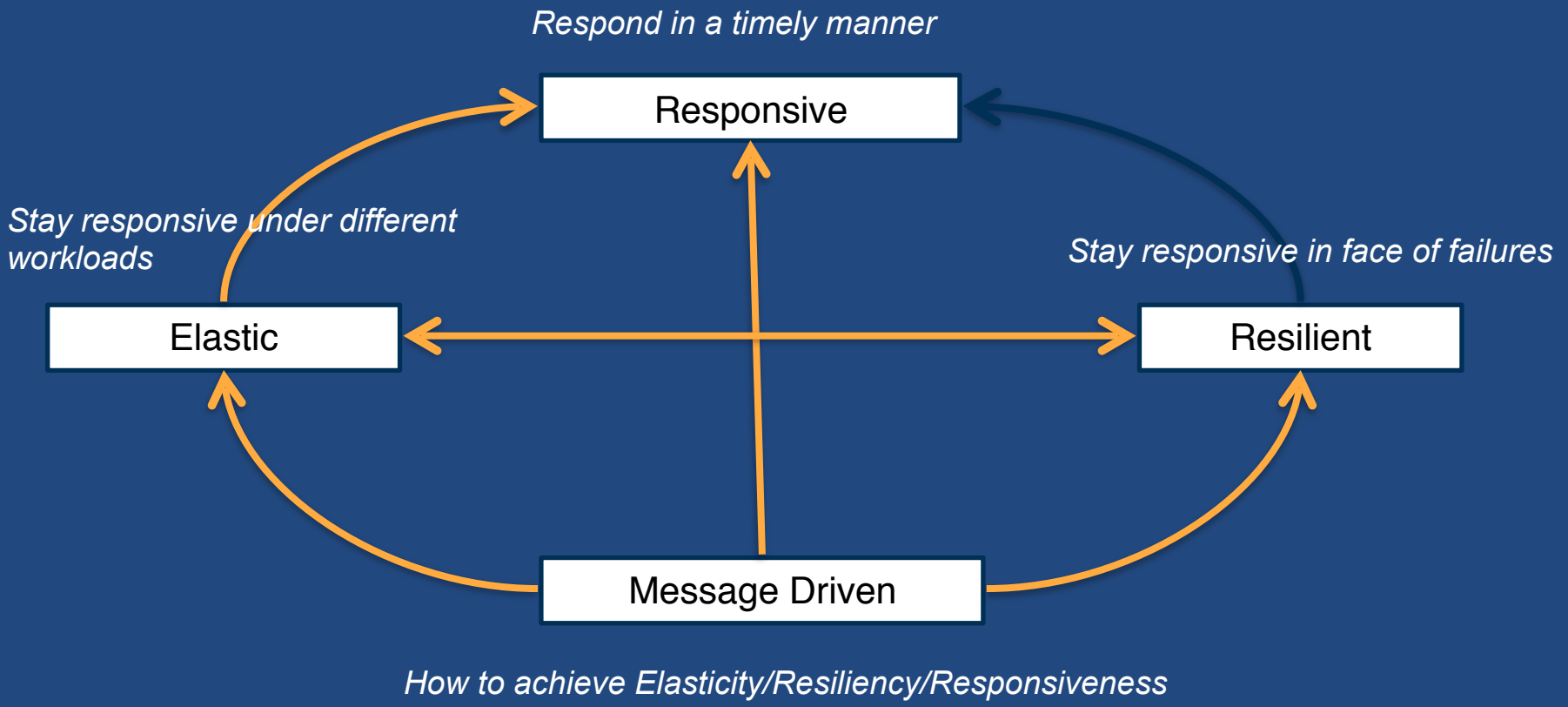
Resilience to build trust



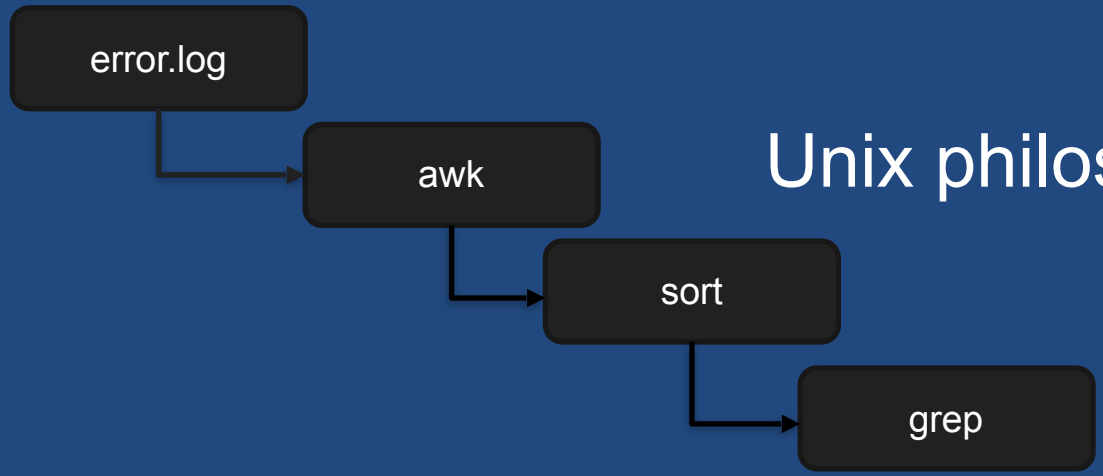
Elasticity and Message Driven to drive profitability



Guiding Principles



Provide the means of composing different feeds

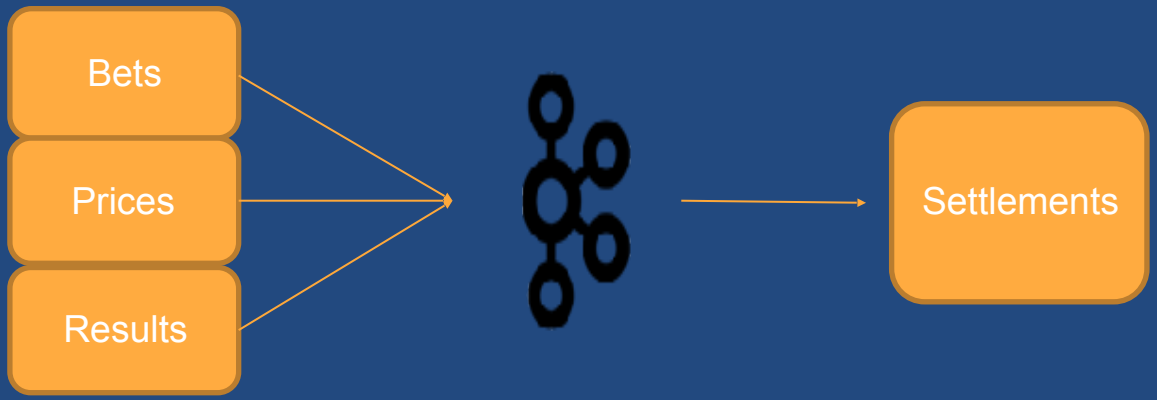


Unix philosophy

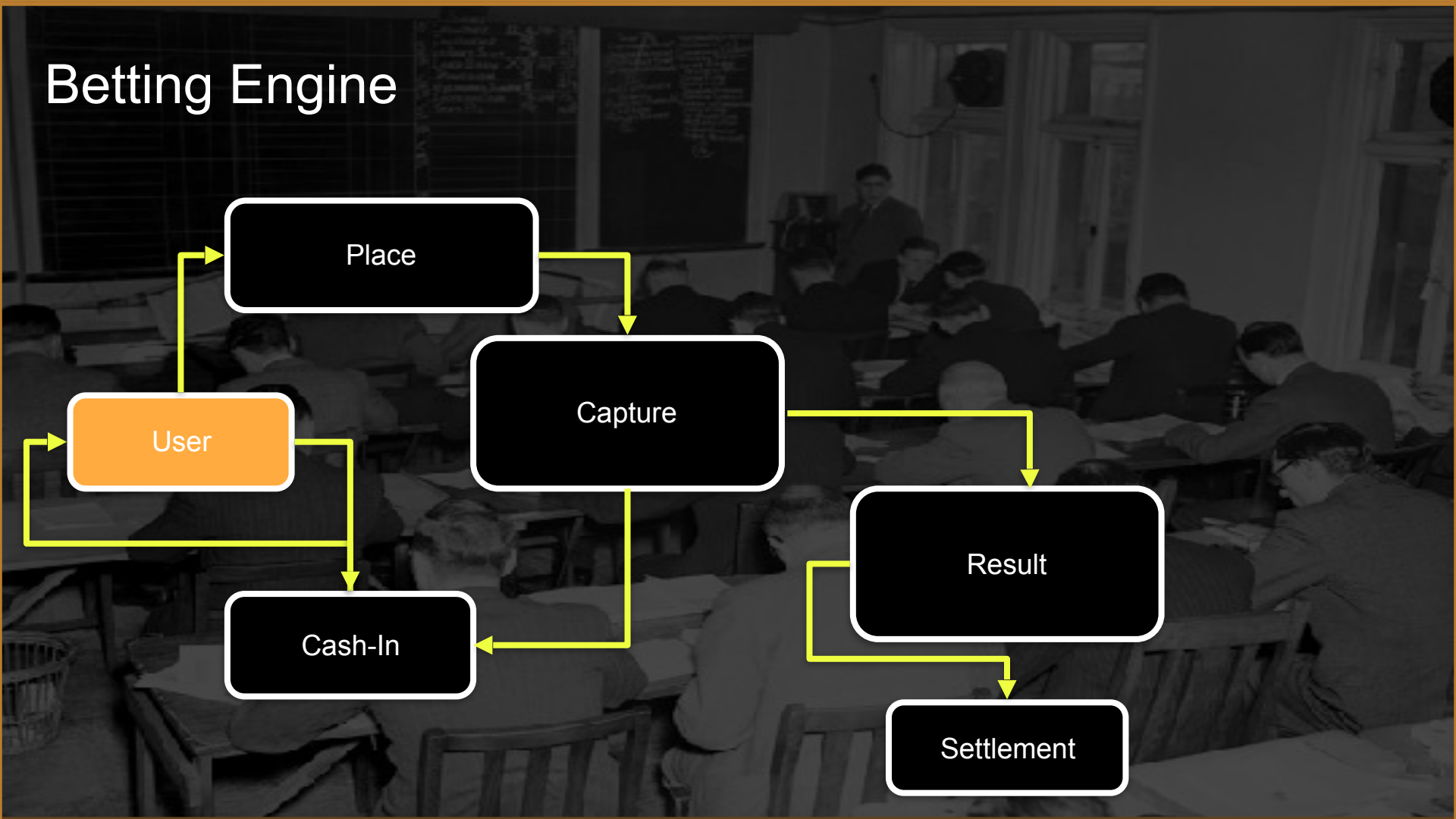
Compose in a distributed system

Provide fault tolerance

Composing Feeds



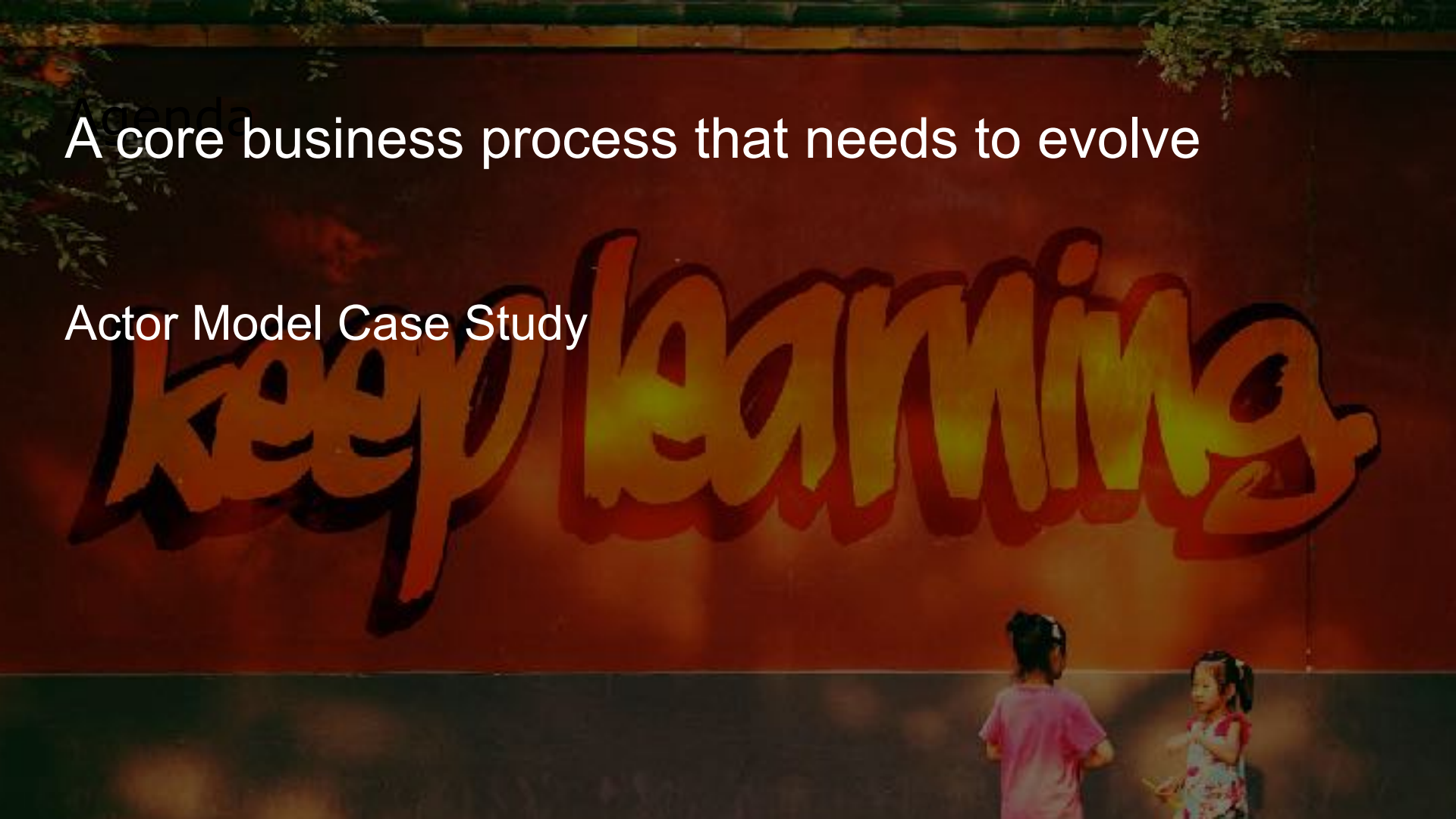
Betting Engine



Agenda

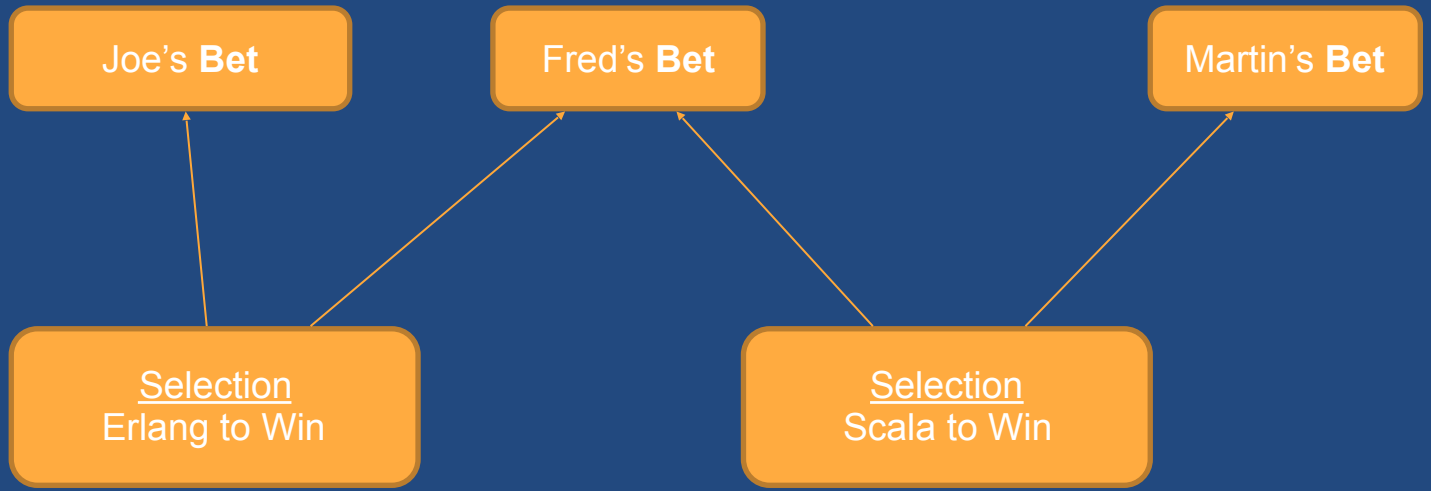
A core business process that needs to evolve

Actor Model Case Study

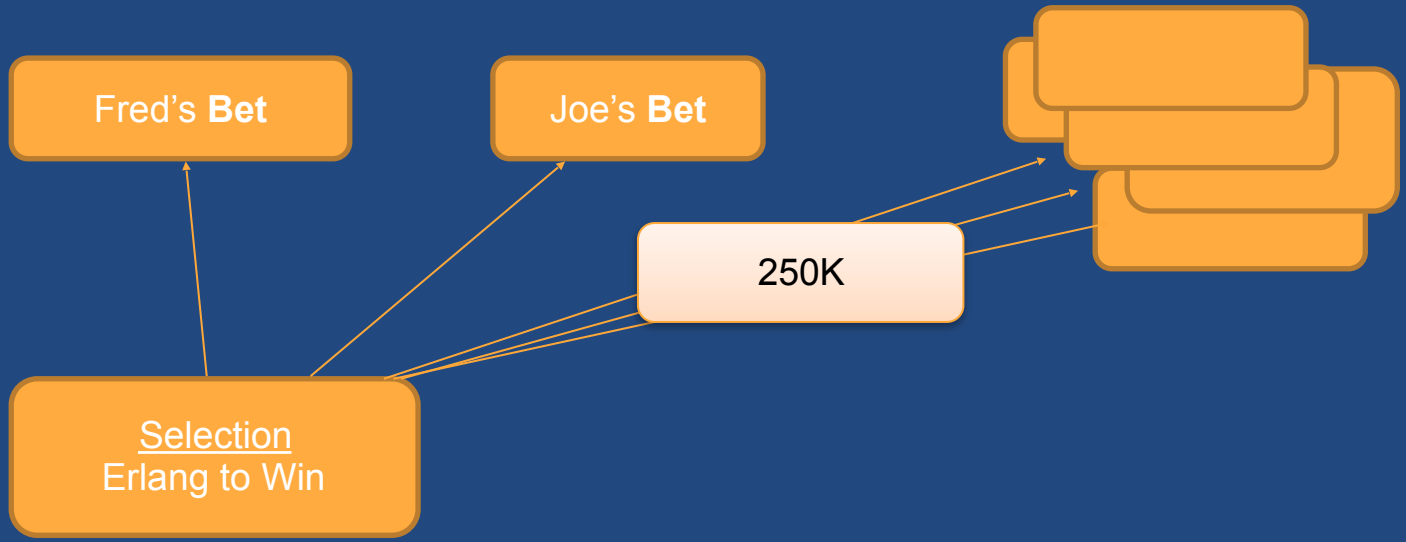


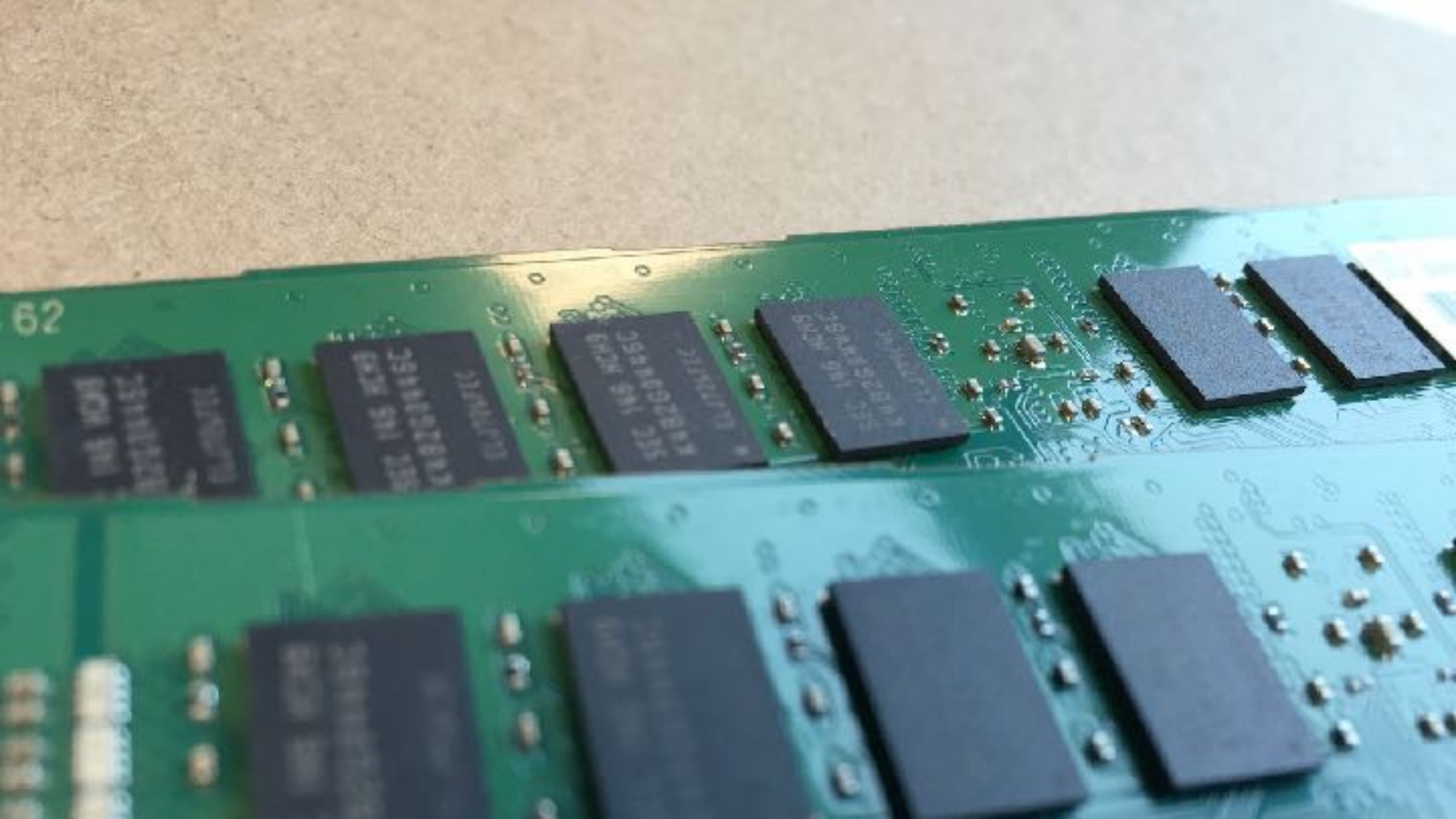
Actors

Actors


Joe's **Bet**
Fred's **Bet**
Martin's **Bet**
Selection
Erlang to Win
Selection
Scala to Win

Actors





Outcome

Concurrency was good 10M processes in a VM (vertical scalability)

Memory 65GB or RAM (usual Grand National traffic)

250K messages sent from one selection [slow] (x 1K) = 250M

supervision hard (for 250K children), looked at sharding

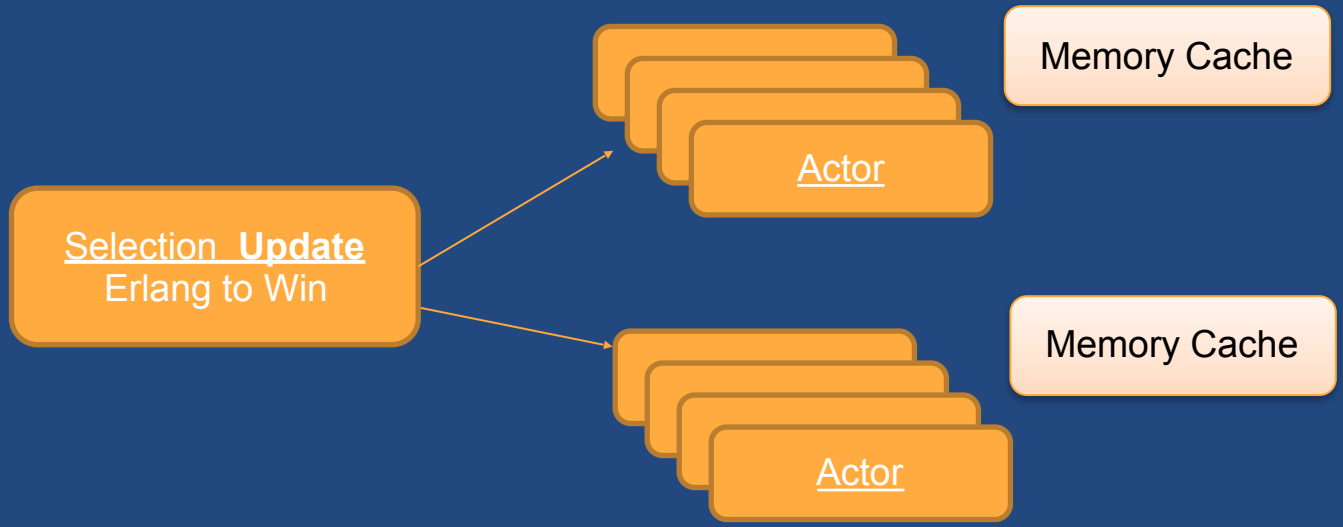
Key realization

Modelling Bet tree structure with Actors for changing state is problematic.

Millions [up to 250M] of messages flowing in bursts creates problems

Next step [Actors as units of computation]

Next Iteration



Results

Usage of RAM decreased significantly [2GB]

Achieved good level of concurrency (with sharding)

Likely the way forward but need to think about how to achieve concurrency effectively [tricky edge cases to solve]

Personalization

Omnia Platform (Lambda Architecture)

Chronos (Data Source)

NeoCortex (Speed Layer)

Fates (Batch Layer)

Hermes (Serving Layer)

https://www.lightbend.com/resources/case-studies-and-stories/uks-online-gaming-leader-bets-big-on-reactive-to-drive-real-time-personalization

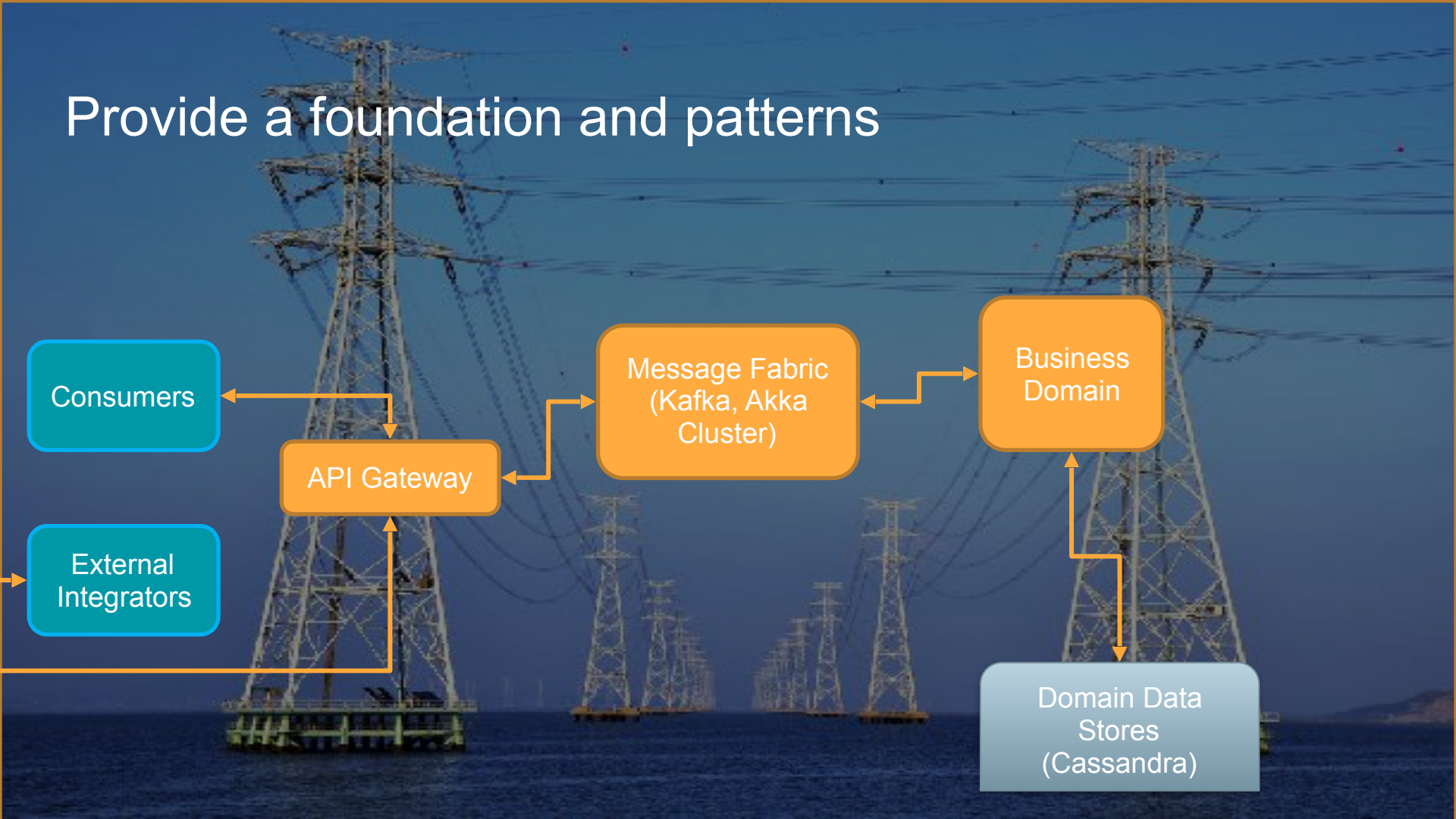
Enterprise Challenge (not a solution challenge)

Provide common platform for services [betting, personalization]

Enable service communication and composability

Maintain consistent design principles

Provide a foundation and patterns



Key considerations going forward

Conway’s law (and the reverse)

CQS, CQRS, Event Sourcing, DDD, Messaging contracts (need to be upheld by all)

Scala can be a challenge so continuous learning is necessary

Front-End cannot be an afterthought

Summary & Q&A