Playtika

# ARCHITECTURE @ SCALE

## From prototype to the complex system

Kanstantsin Slisenka | Software Architect

# KANSTANTSIN SLISENKA

## SOFTWARE ARCHITECT

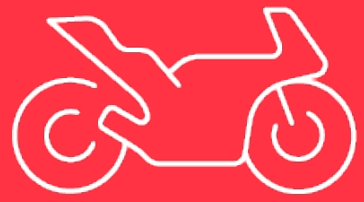
Playtika




**50M+ DOWNLOADS***

*according to the Google Play

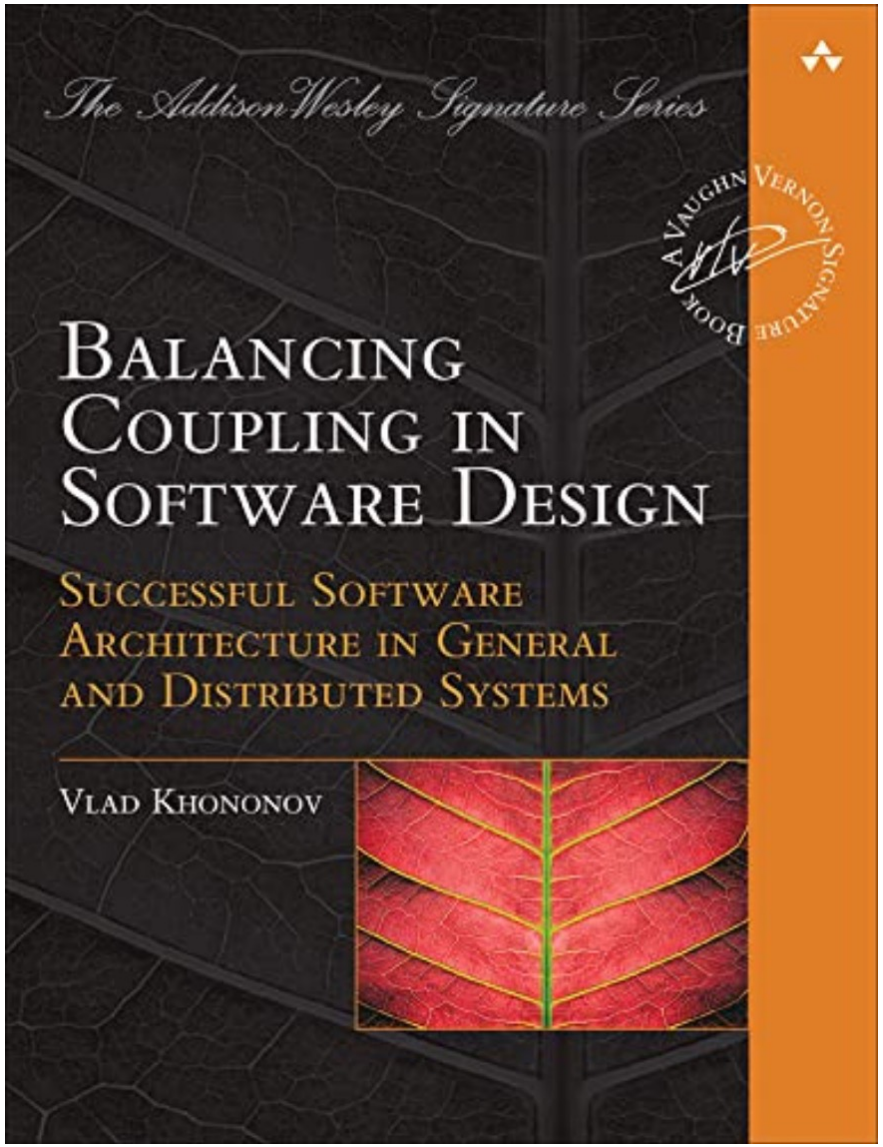
Playtika



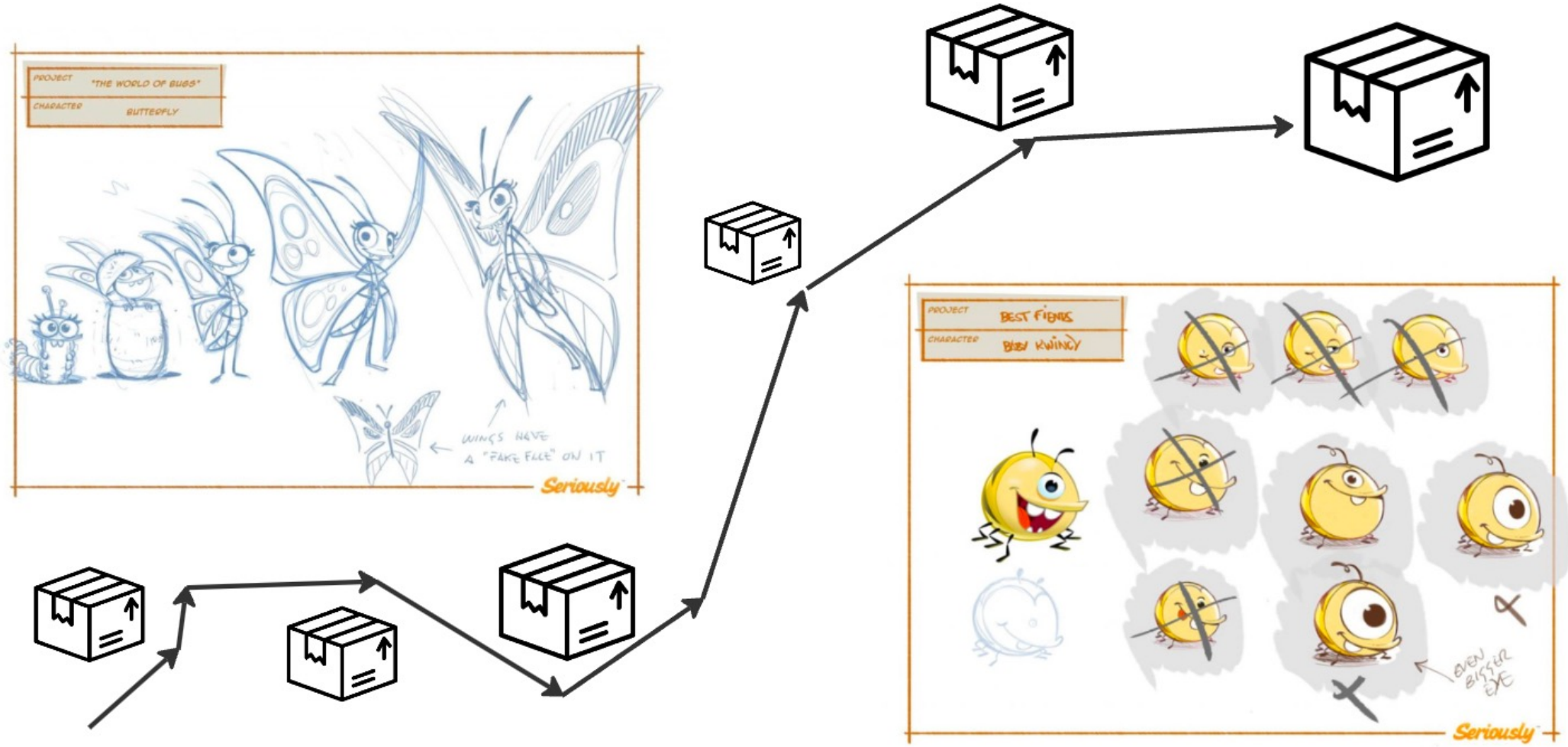
1.0 1.5 2.0

Playtika

The value of a software system is not merely a reflection of its current functionality but also its **ability to evolve, to grow, and to address future requirements**



# SOFTWARE OFTEN STARTS SMALL



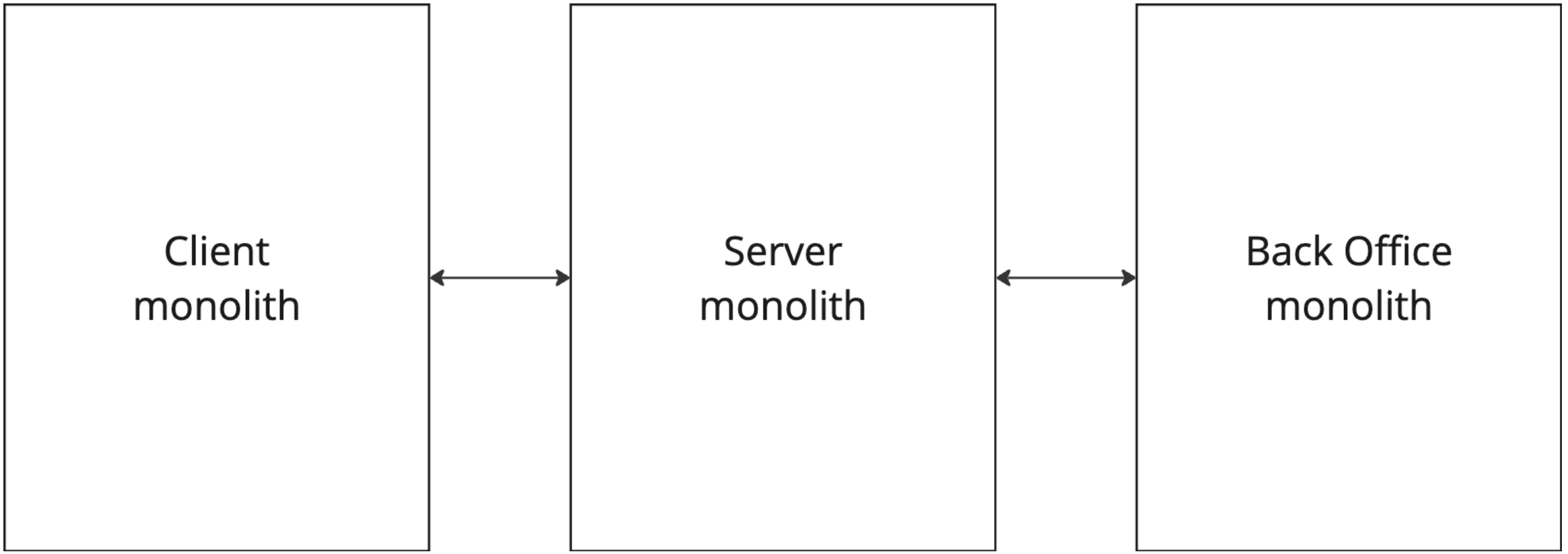
# ARCHITECTURE 1.0

# GOAL

## QUICK PROTOTYPING

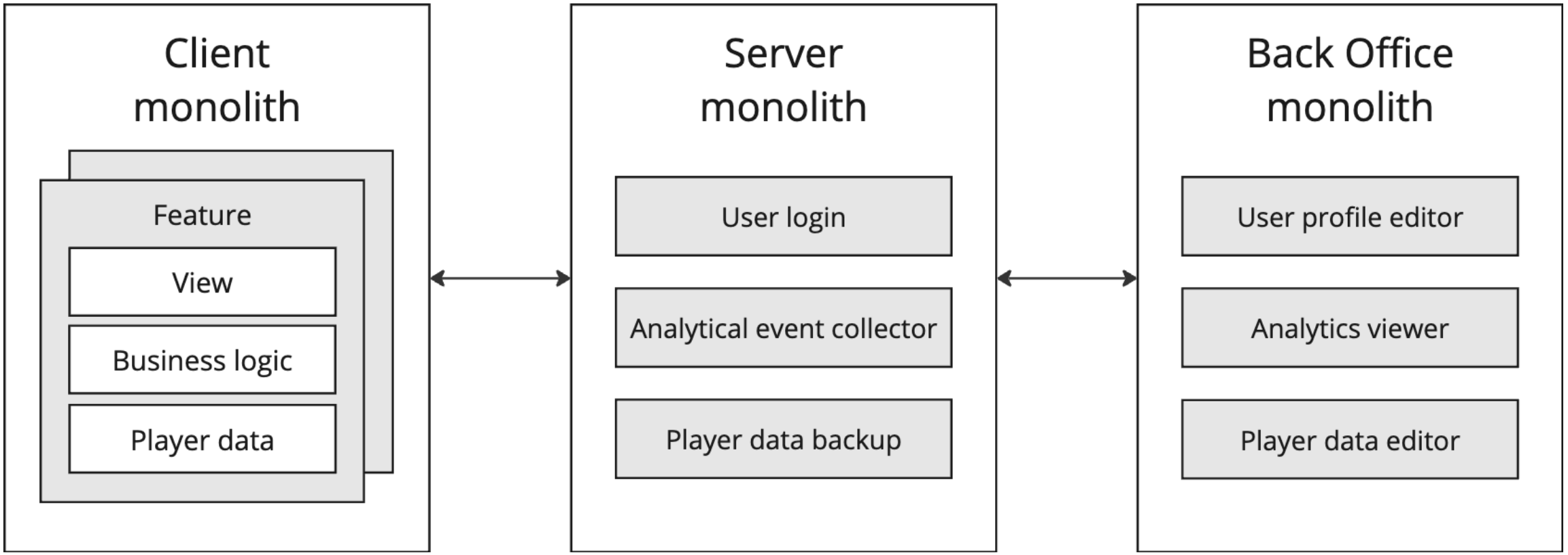
Playtika

# ARCHITECTURE 1.0

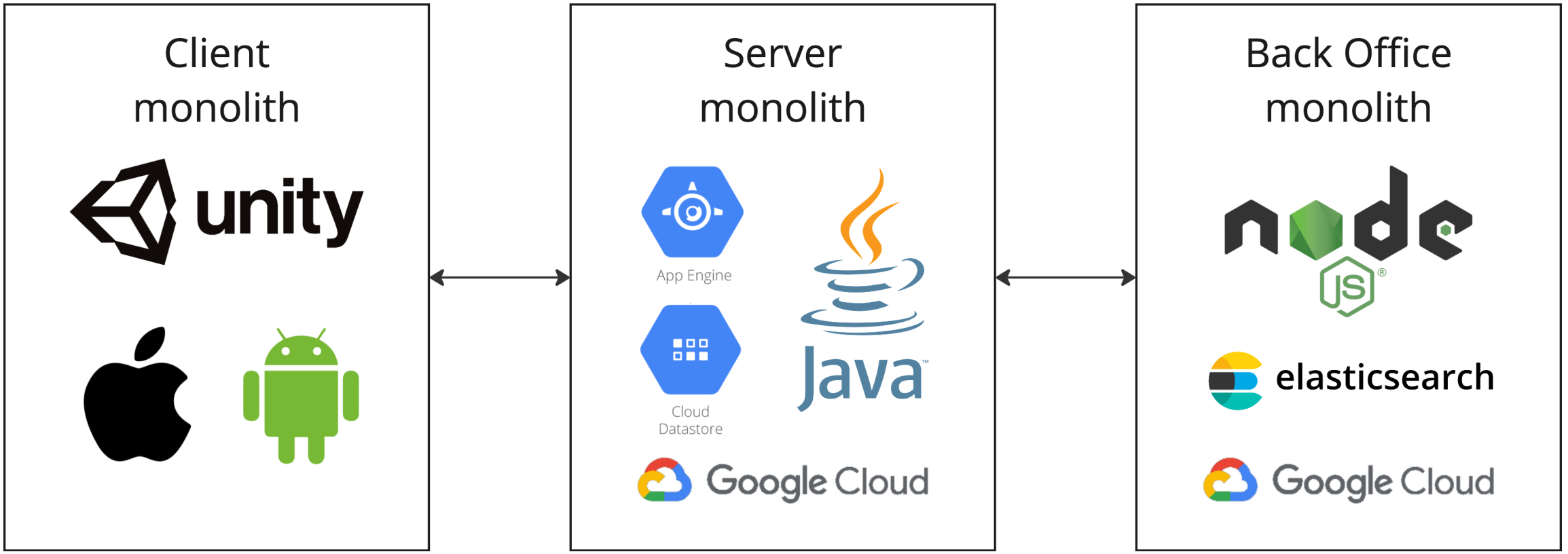


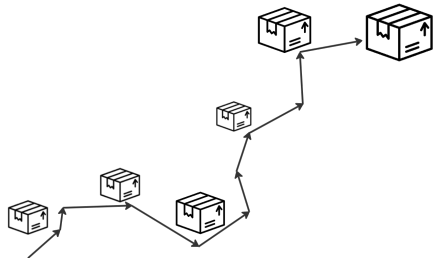

Playtika

# ARCHITECTURE 1.0

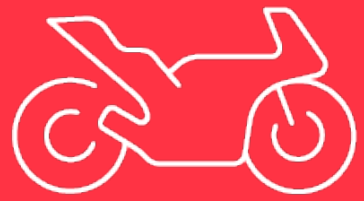


# ARCHITECTURE 1.0





QUICK PROTOTYPING
LESS RESTRICTIONS



**1.0** **1.5** **2.0**

MONOLITH

(NO BOUNDARIES)

Playtika

# SOFTWARE EXPANDS WHEN THE VALUE IS PROVEN

# MORE TEAMS, STAKEHOLDERS, FEATURES



Playtika

## ARCHITECTURE 1.5

# GOAL

## UNLOCKING PARALLEL DEVELOPMENT AND RELEASES

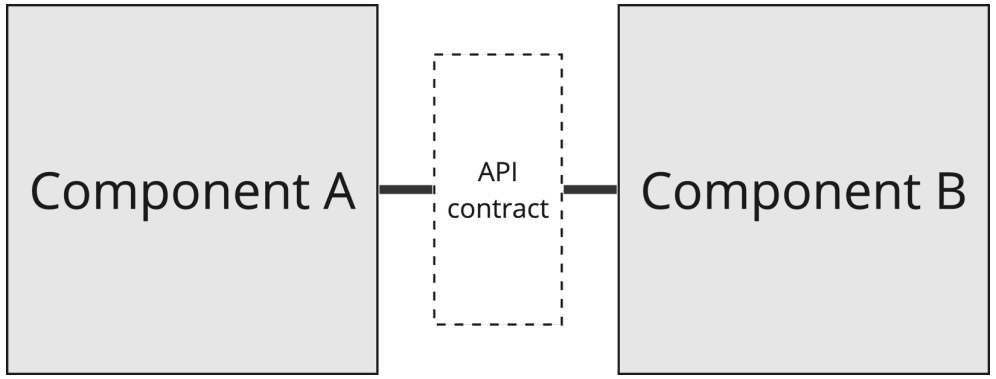
Playtika

# COUPLING

## SHARED KNOWLEDGE

## SHARED LIFE CYCLE

# CONTRACT COUPLING

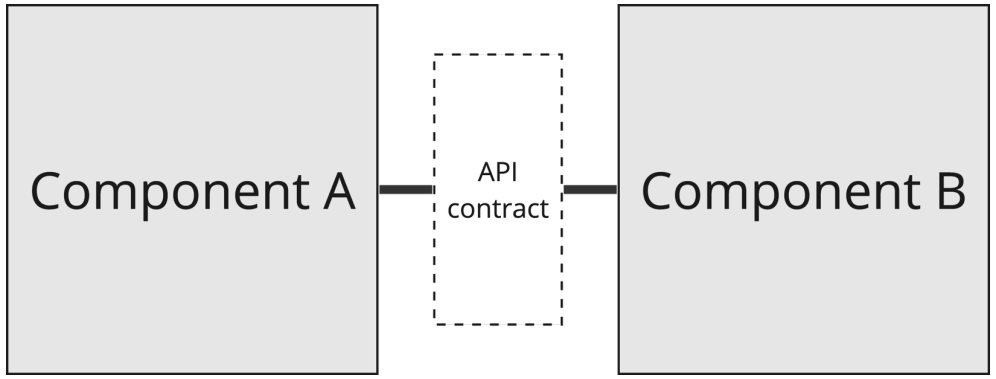



Changes can be isolated

Minimum shared knowledge

Playtika

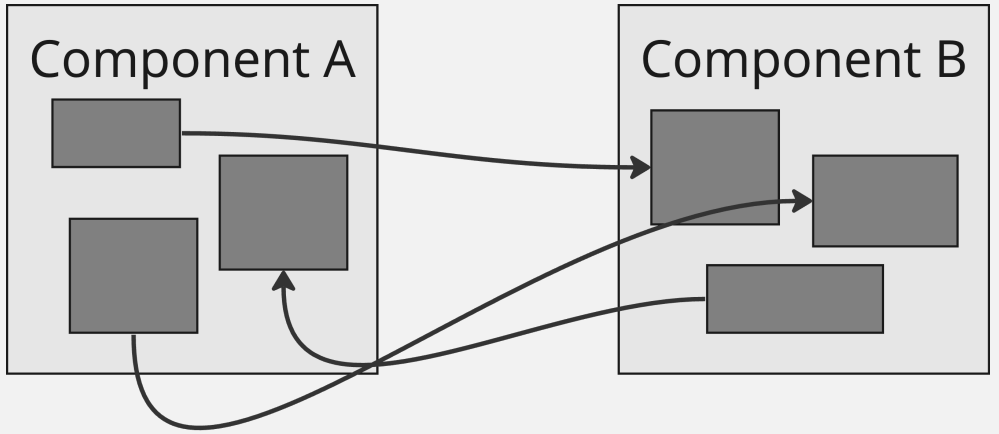
# CONTRACT COUPLING




Changes can be isolated

Minimum shared knowledge

# FUNCTIONAL, MODEL, INTRUSIVE COUPLING

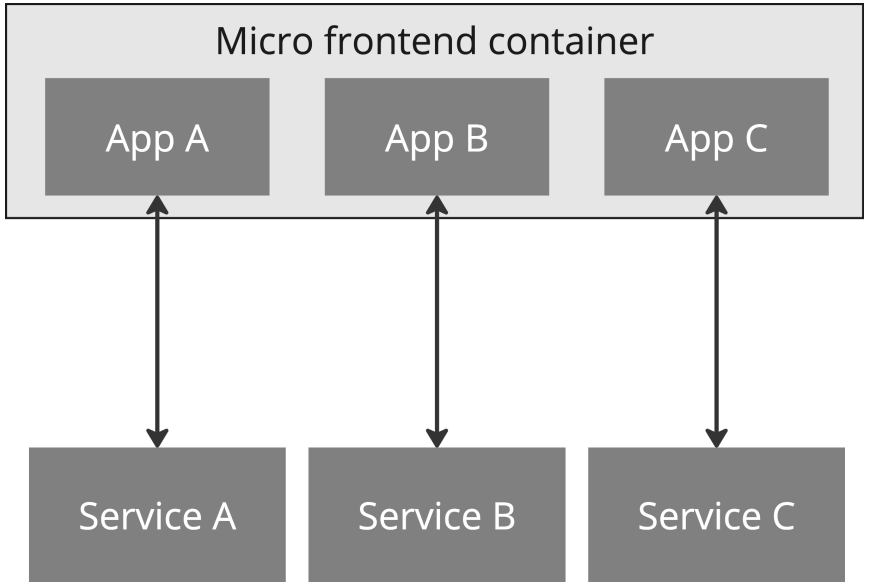



Changes in A trigger changes in B

A lot of shared knowledge

Playtika

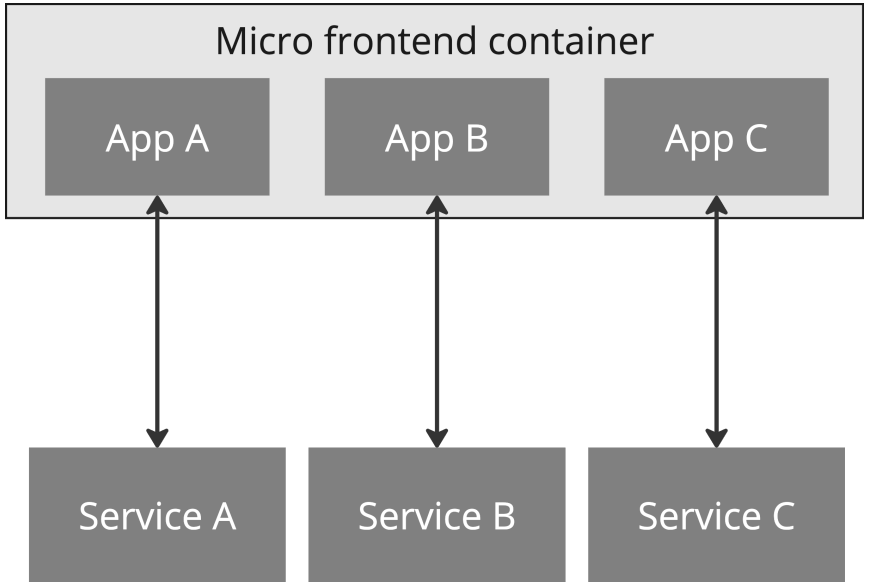
# MICRO SERVICES




Independent releases

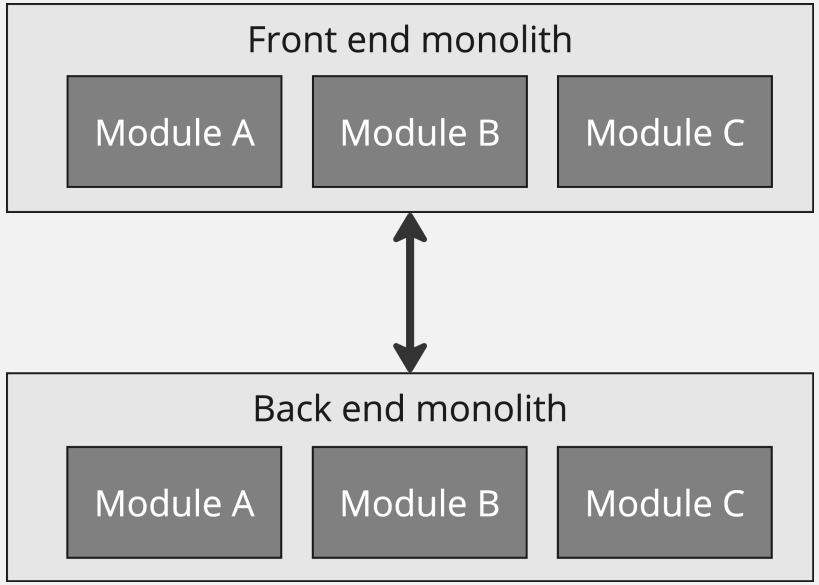
Playtika

# MICRO SERVICES




Independent releases

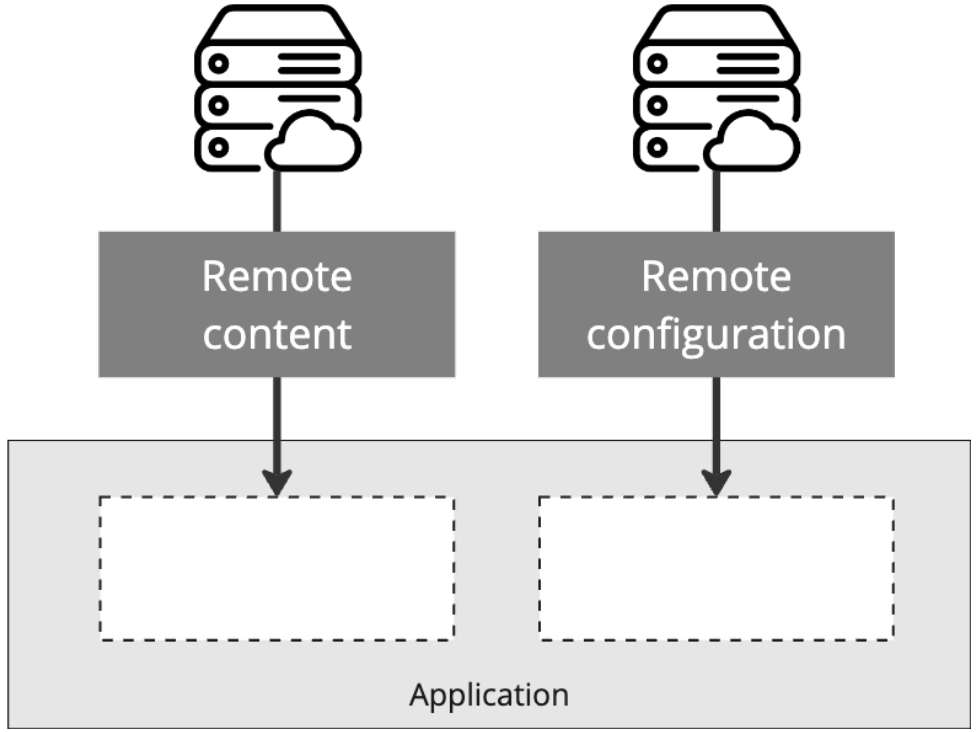
# MONOLITHS




Released only together

Playtika

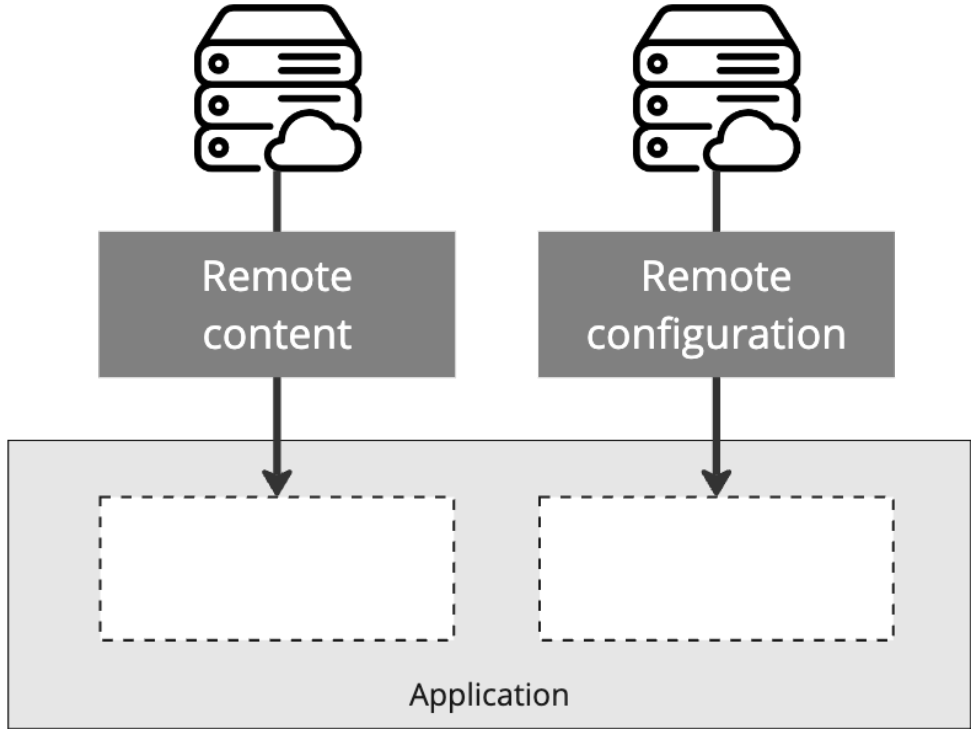
# REMOTE CONTENT




Remote content and application are released separately

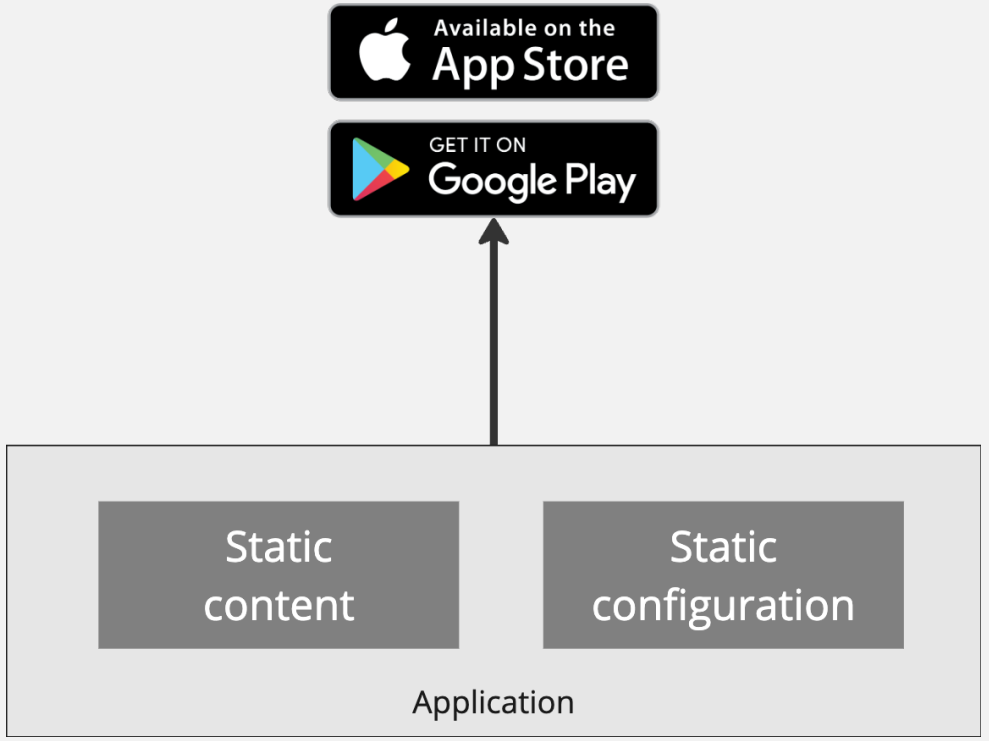
Playtika

# REMOTE CONTENT




Remote content and application are released separately

# STATIC CONTENT




Static content and application are released together

Playtika

# BUSINESS DRIVERS

MORE CONTENT

PLAYER SEGMENTATION

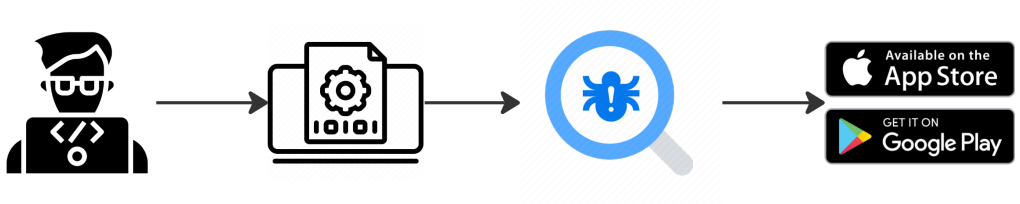
SOCIAL MECHANICS

INFRASTRUCTURE OPTIMIZATION

# CONTENT IN THE GAME



# STATIC CONTENT RELEASE

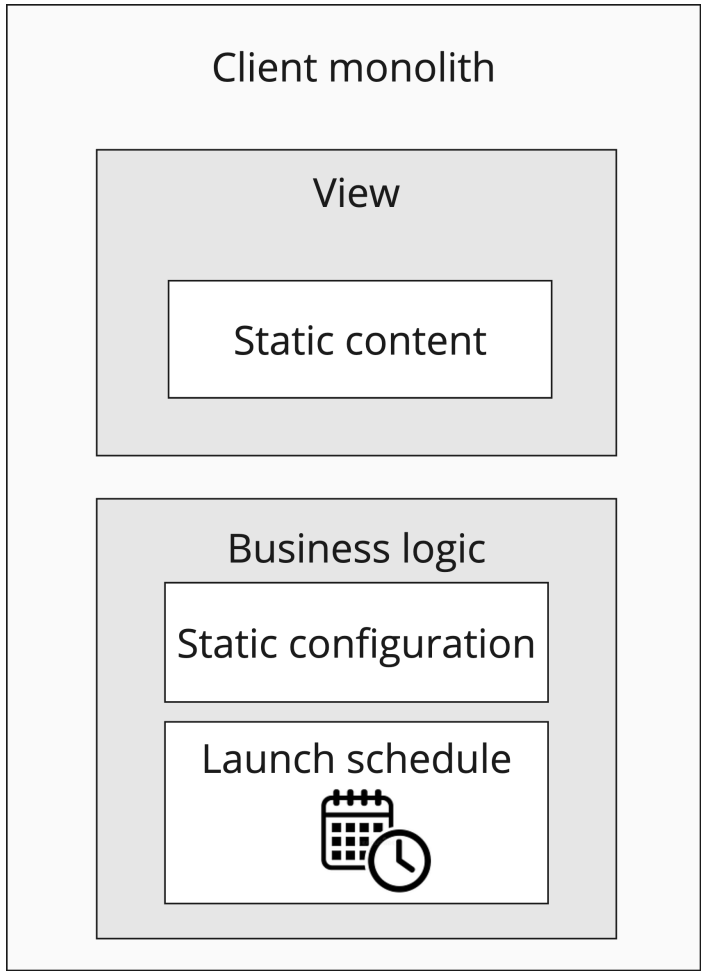



code change → build → qa → release

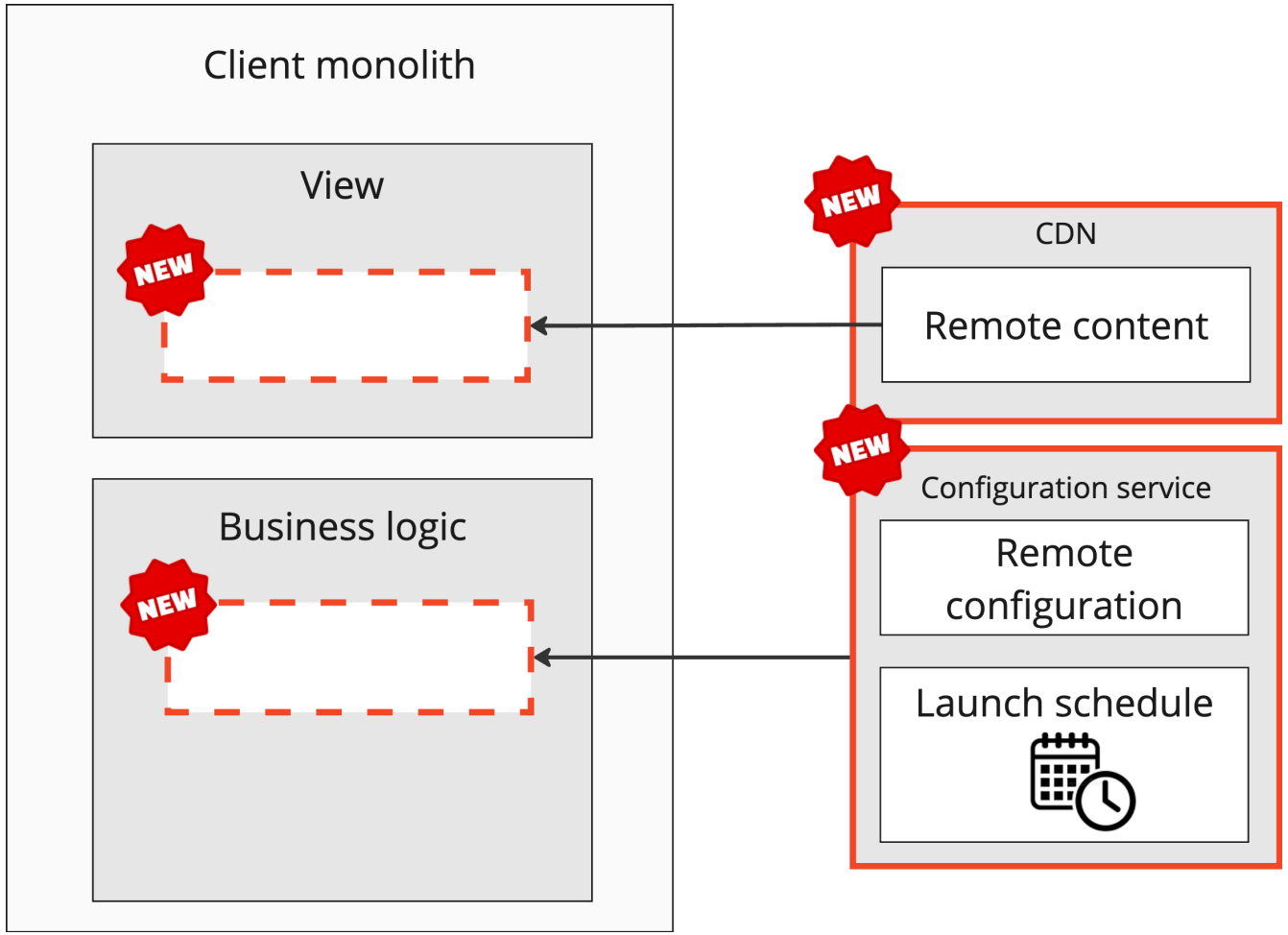
# REMOTE CONTENT RELEASE

content change → upload → download

# CONTENT MANAGEMENT SYSTEM

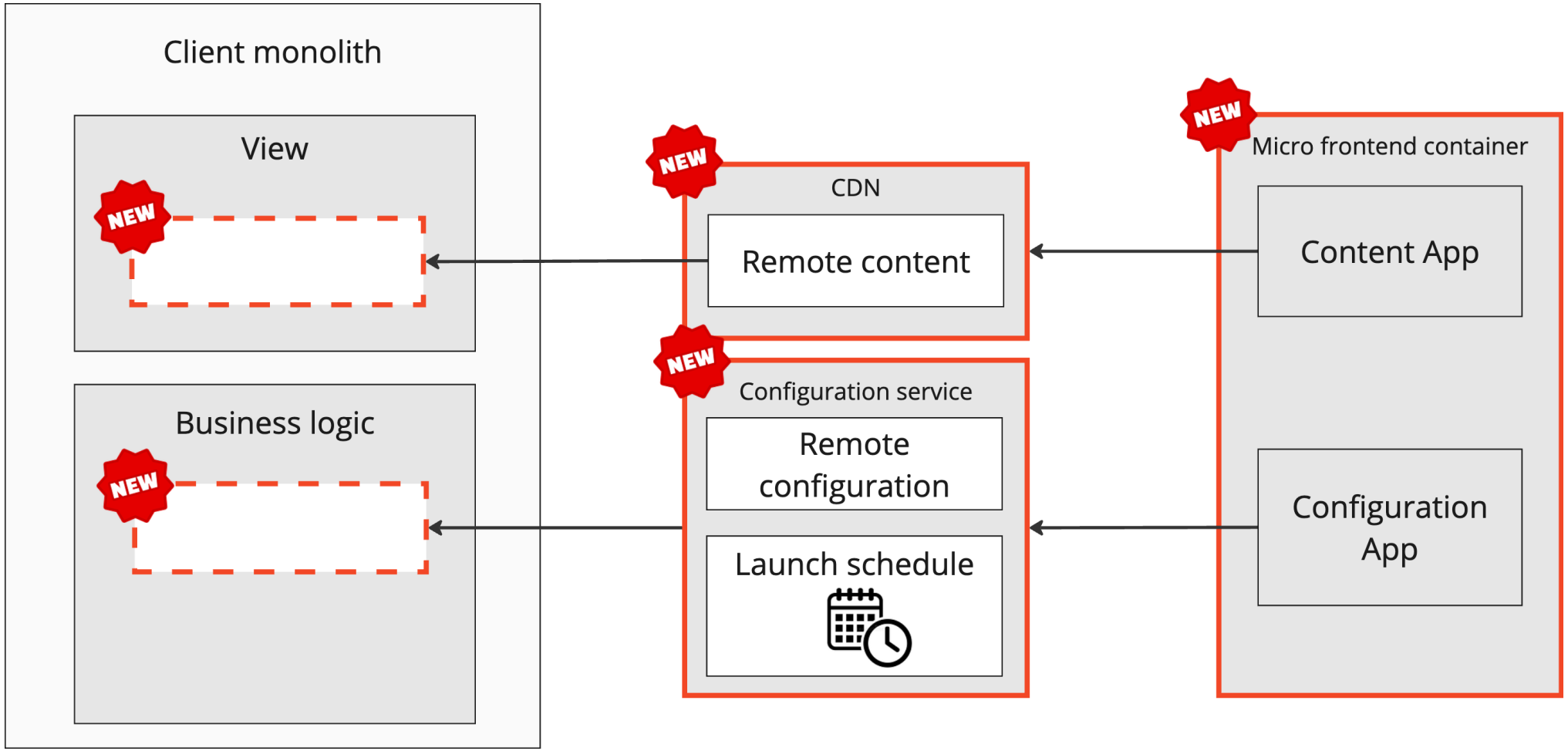


# CONTENT MANAGEMENT SYSTEM




Playtika

# CONTENT MANAGEMENT SYSTEM




Playtika

# BUSINESS DRIVERS

MORE CONTENT

PLAYER SEGMENTATION

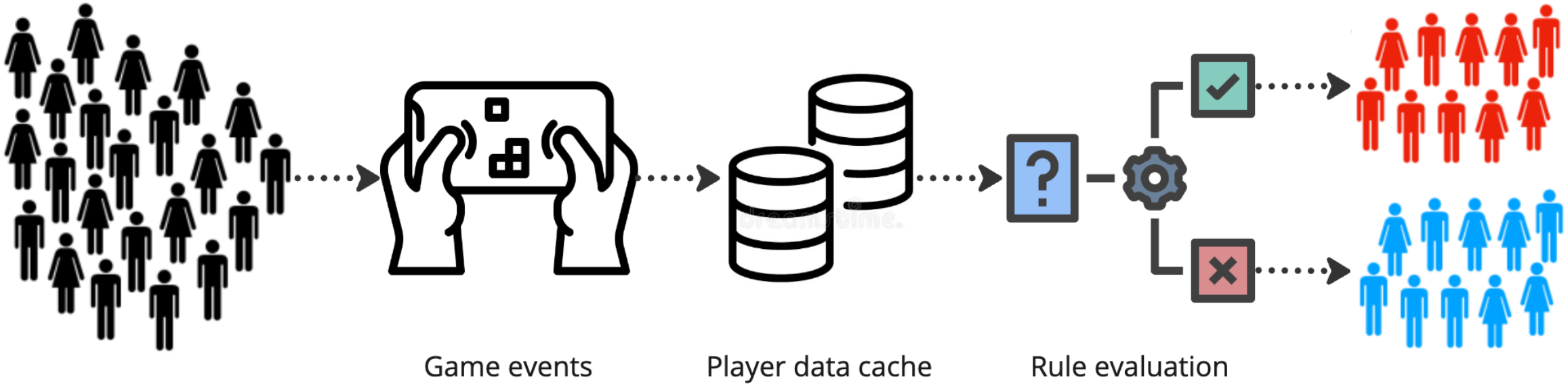
SOCIAL MECHANICS

INFRASTRUCTURE OPTIMIZATION

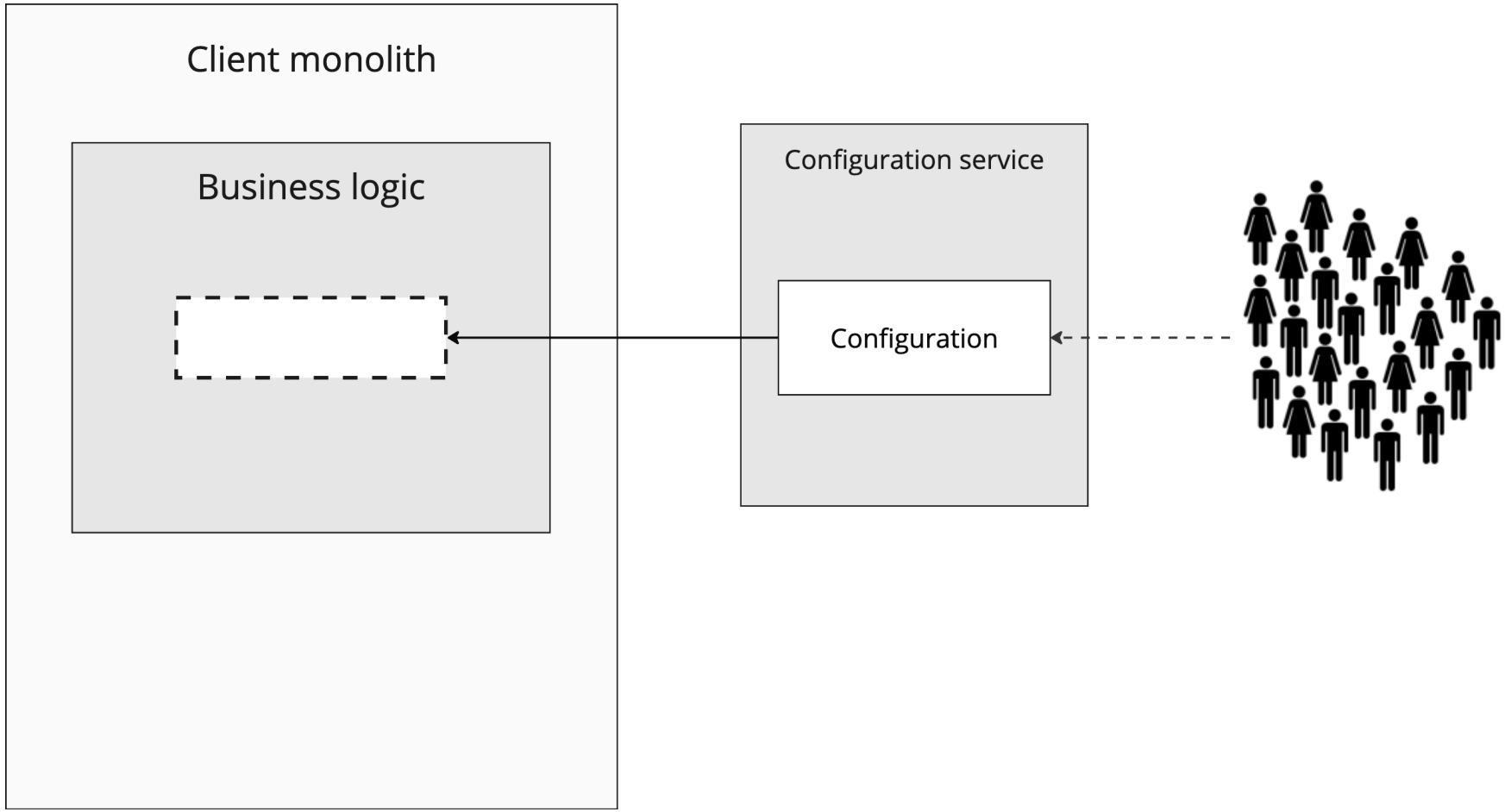
Playtika

# PLAYER SEGMENTATION EXAMPLE

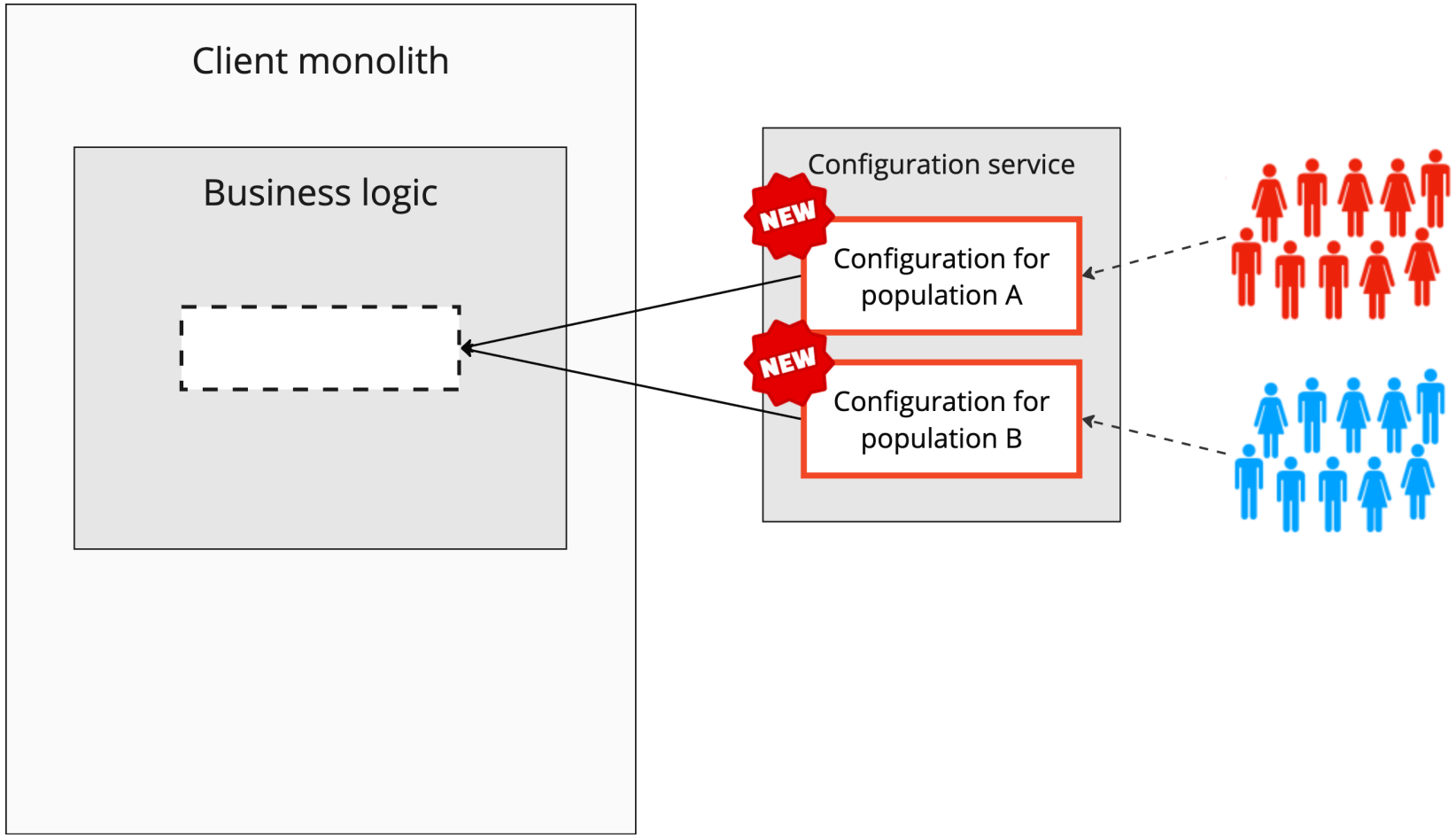
# PLAYER SEGMENTATION



# SEGMENTATION SYSTEM

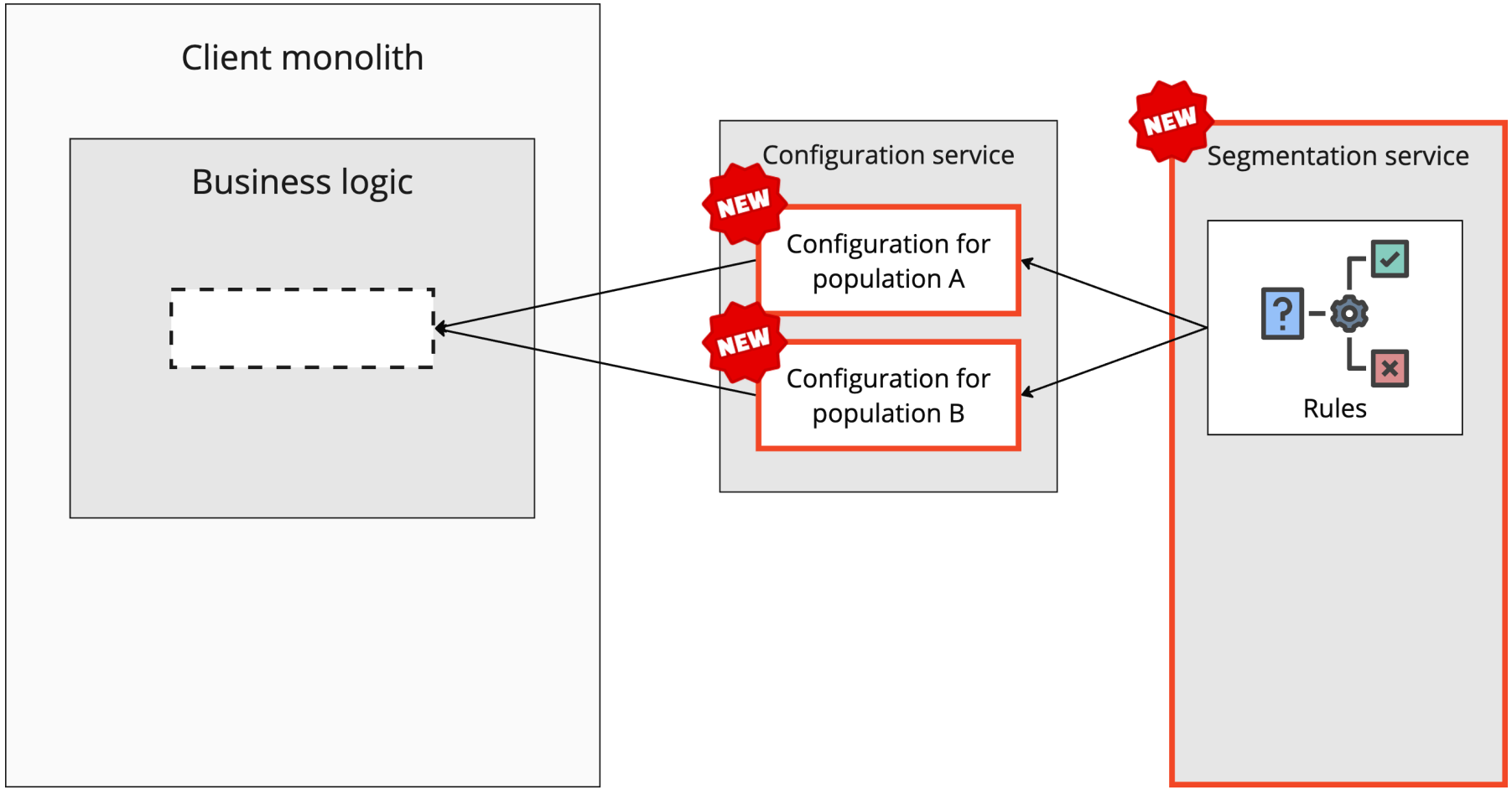


# SEGMENTATION SYSTEM

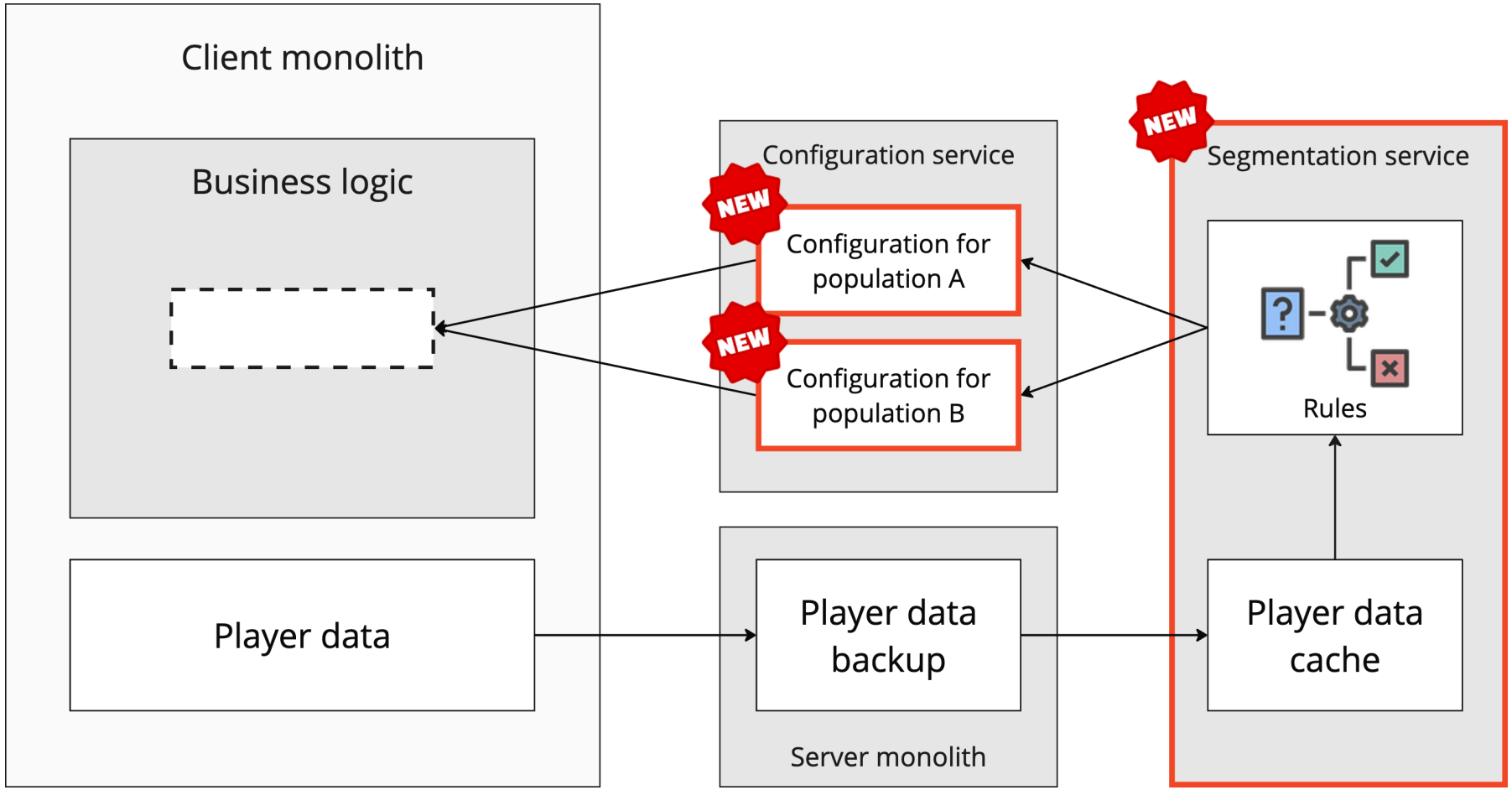



Playtika

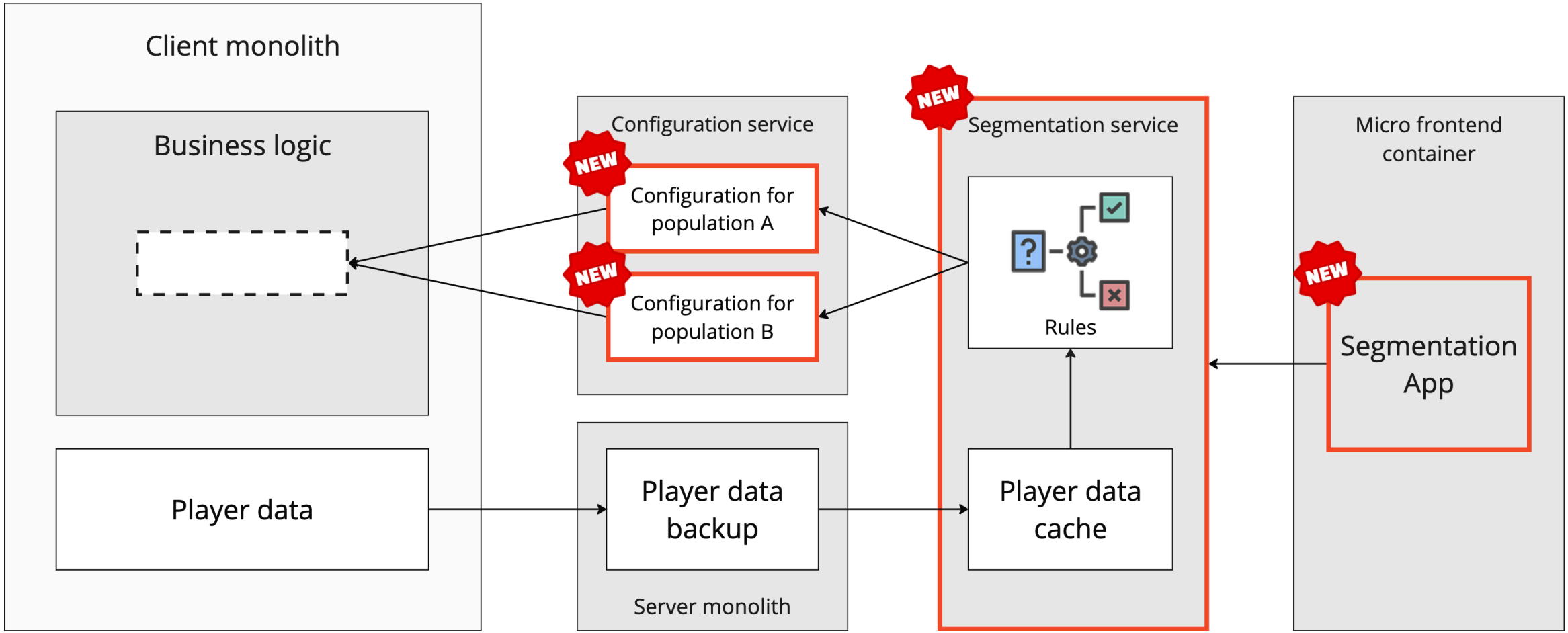
# SEGMENTATION SYSTEM



# SEGMENTATION SYSTEM



# SEGMENTATION SYSTEM



# BUSINESS DRIVERS

MORE CONTENT

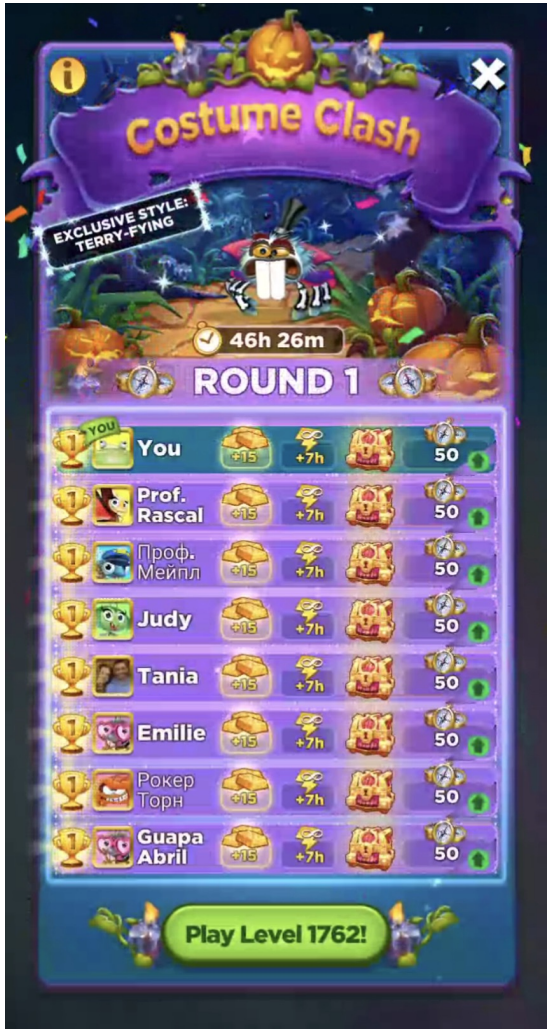
PLAYER SEGMENTATION

SOCIAL MECHANICS

INFRASTRUCTURE OPTIMIZATION

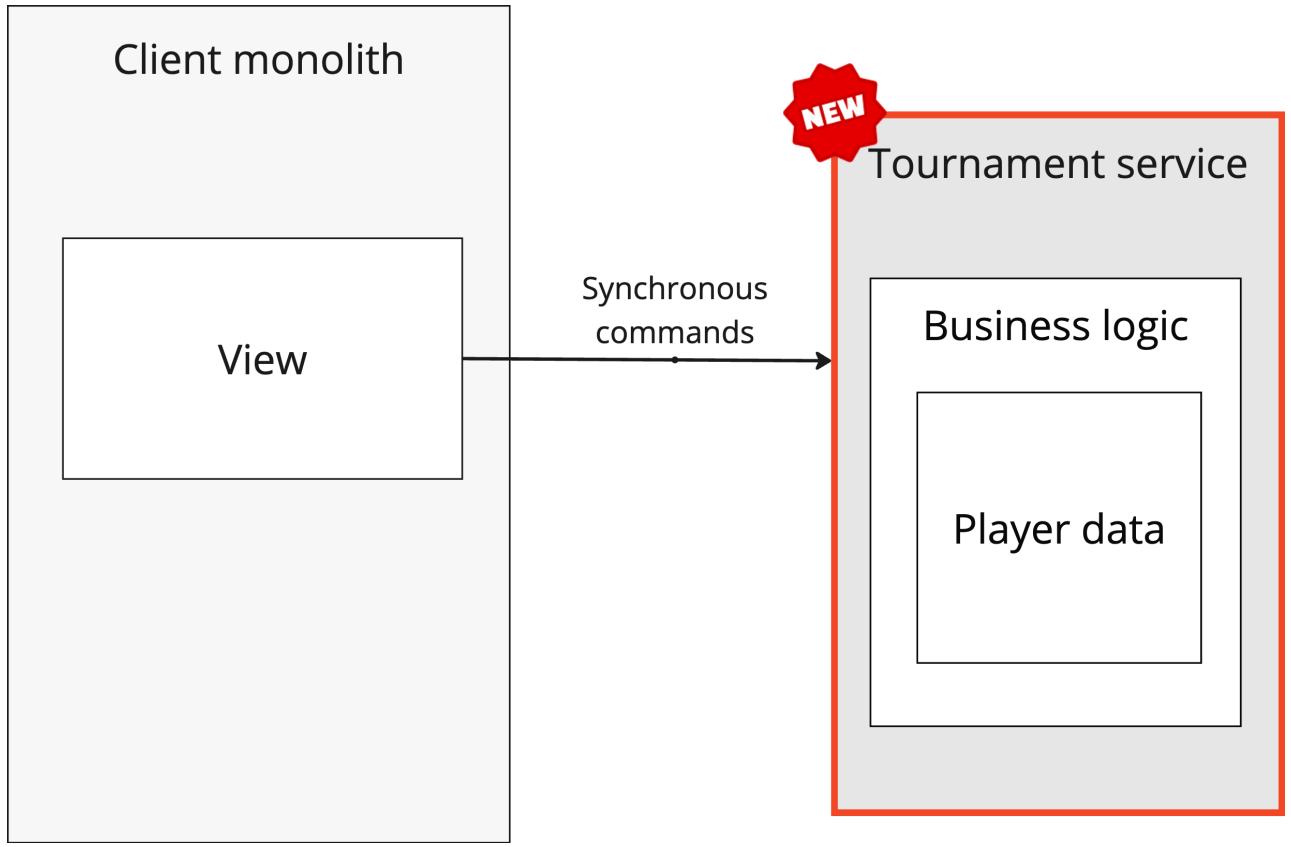
Playtika

# TOURNAMENTS

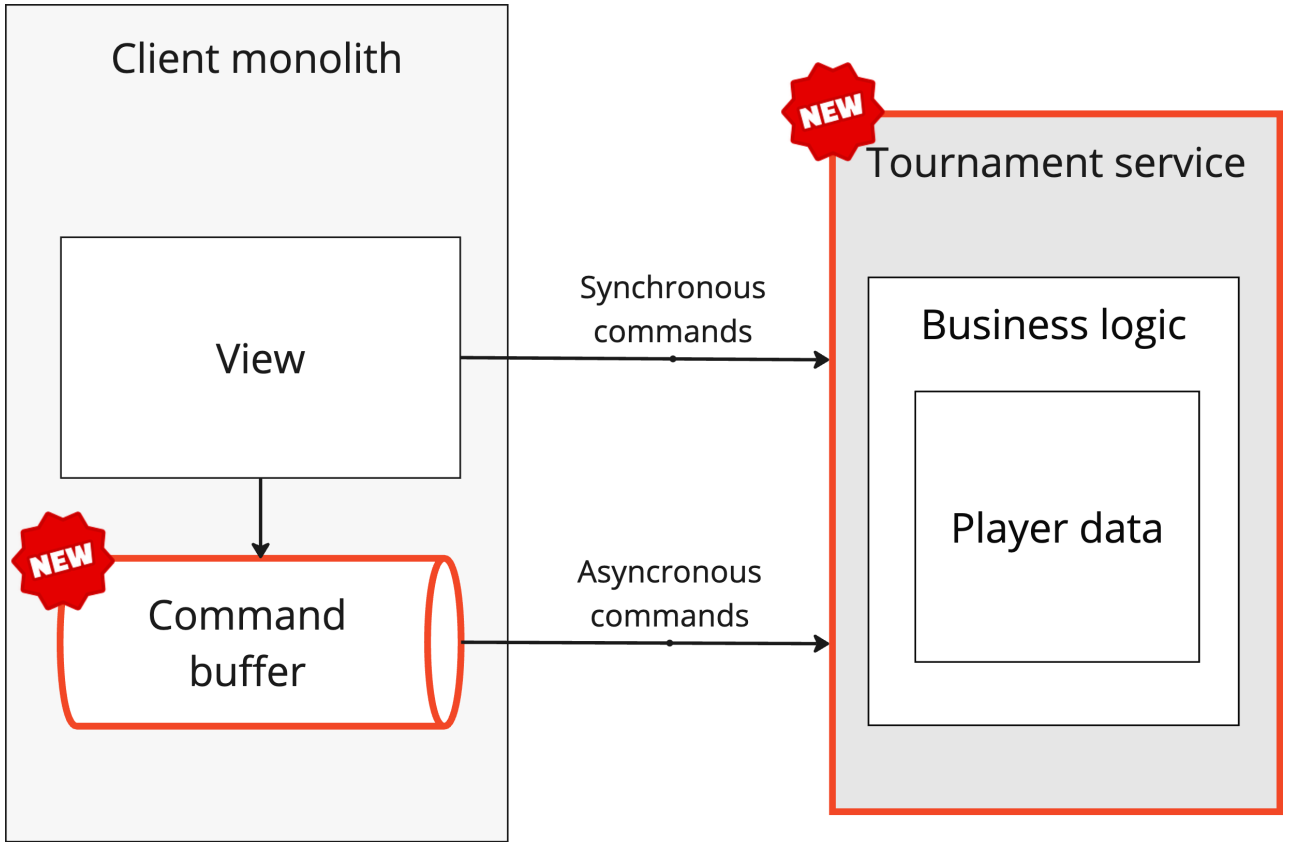


# TOURNAMENT LEAGUES WITH SEGMENTATION


# TOURNAMENTS SYSTEM

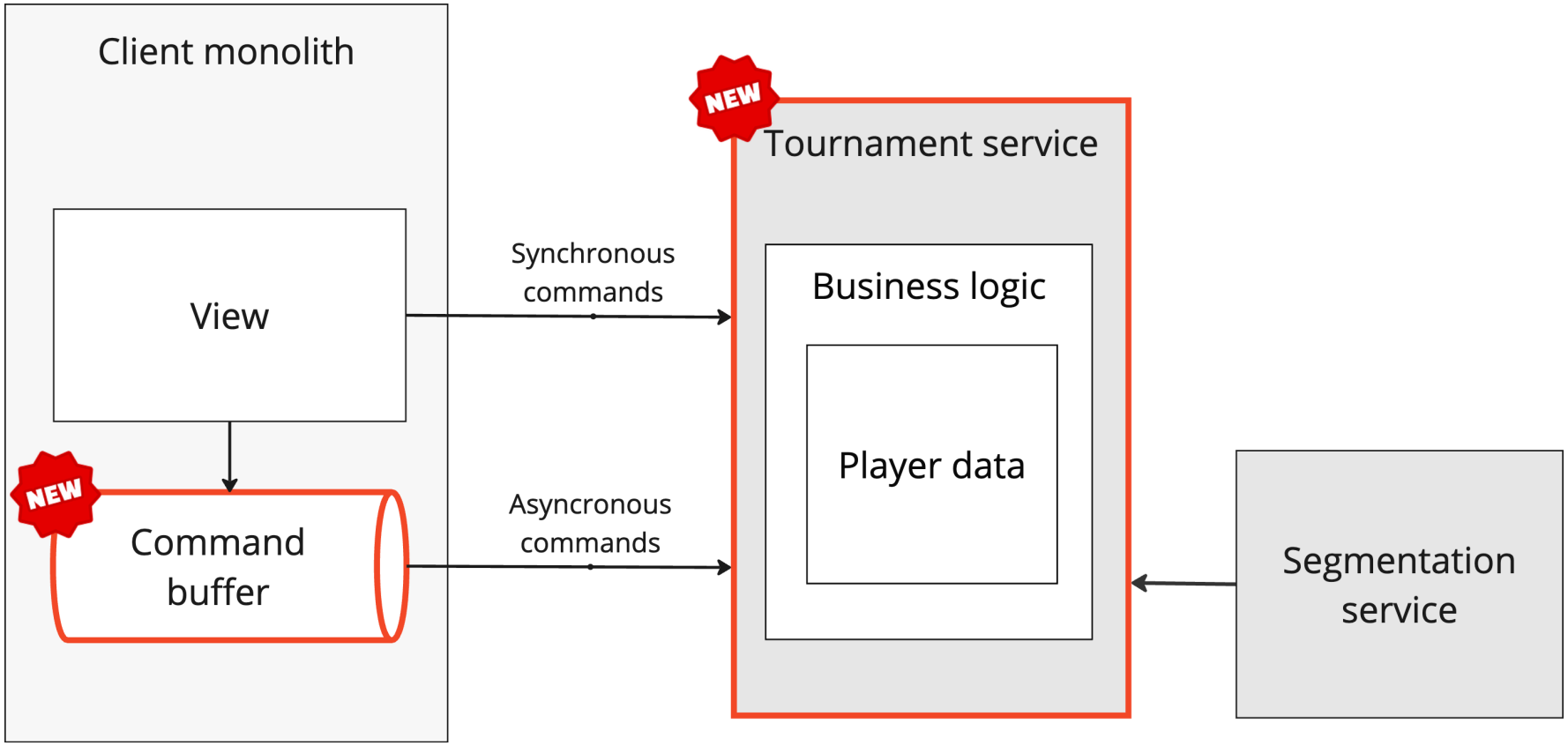


# TOURNAMENTS SYSTEM

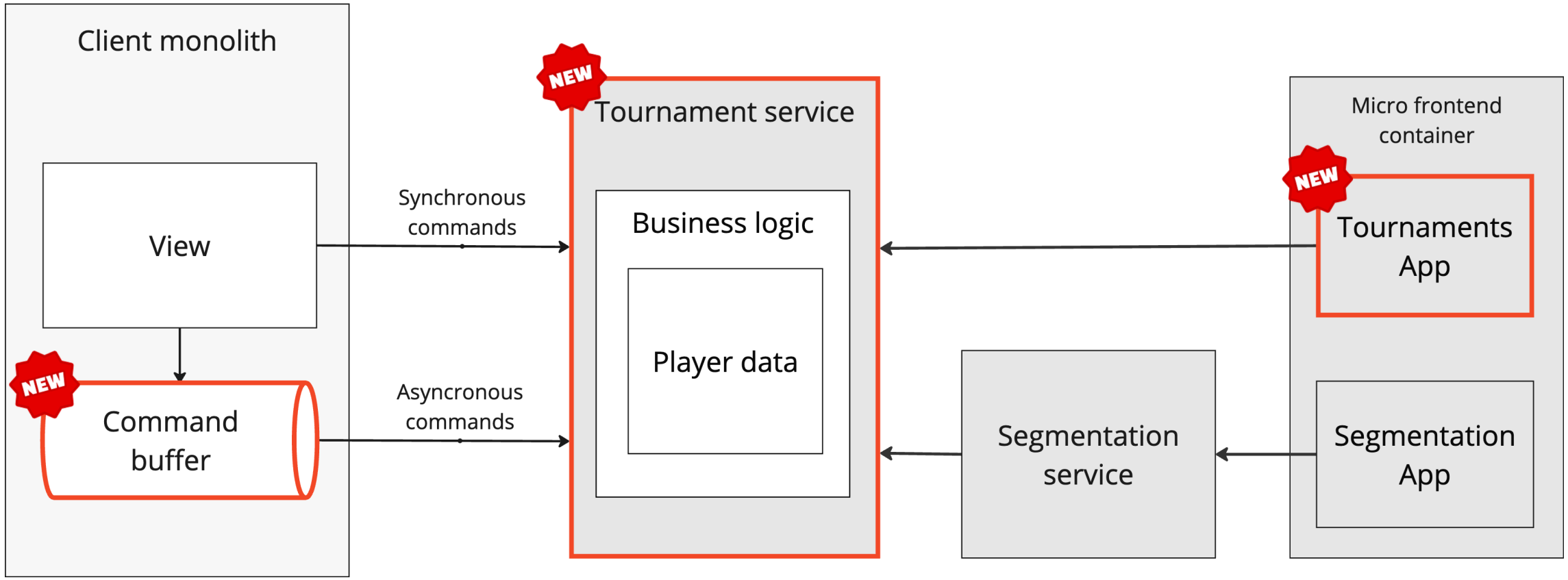



Playtika

# TOURNAMENTS SYSTEM



# TOURNAMENTS SYSTEM




Playtika

# BUSINESS DRIVERS

MORE CONTENT

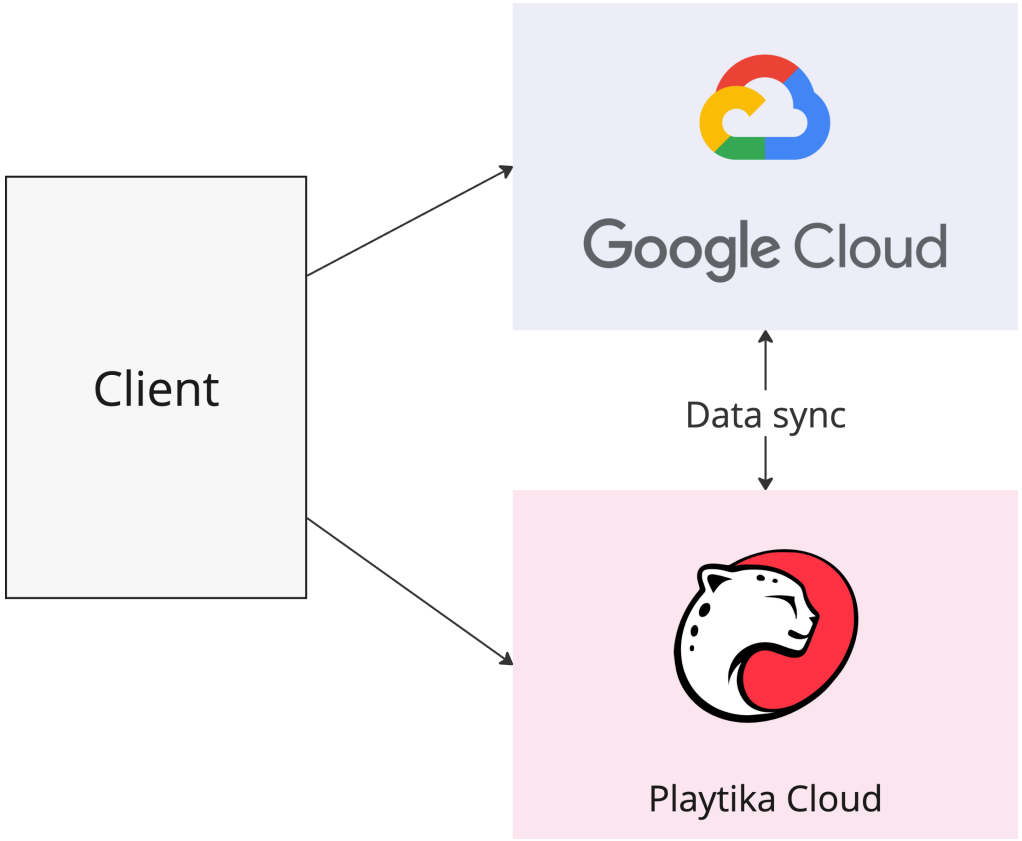
PLAYER SEGMENTATION

SOCIAL MECHANICS

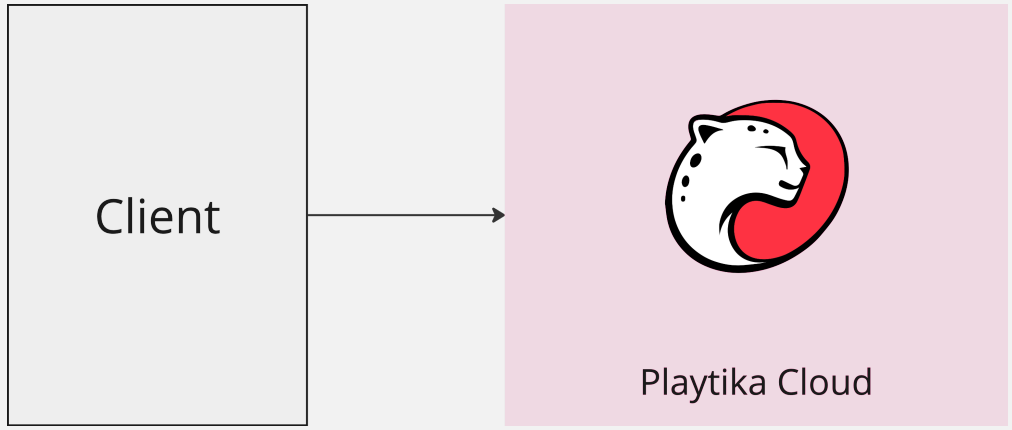
INFRASTRUCTURE OPTIMIZATION

Playtika

# CURRENT

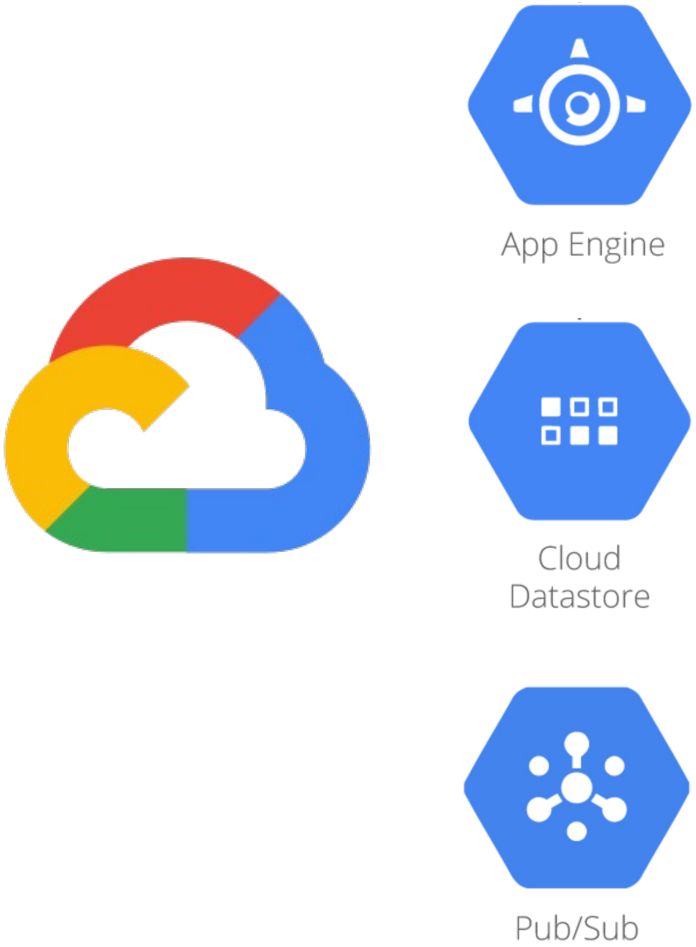



# TARGET




Playtika

# GOOGLE CLOUD




# PLAYTIKA

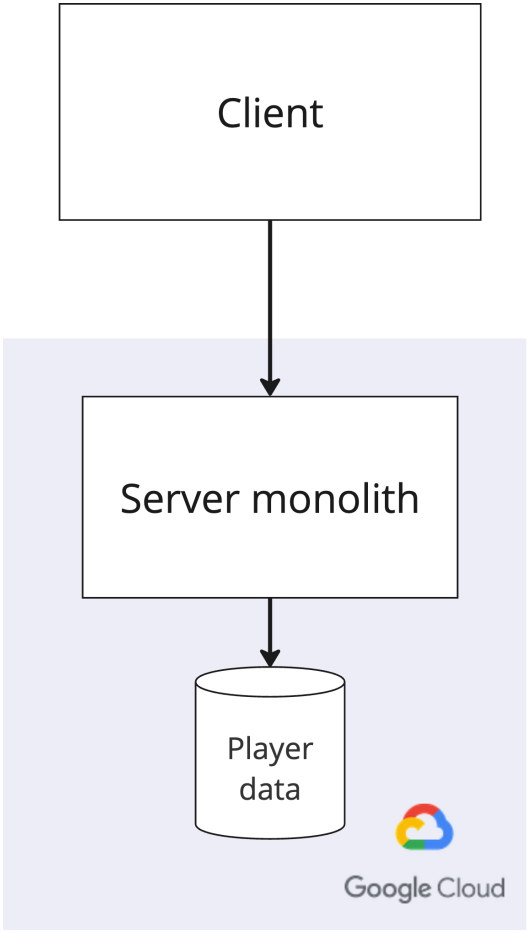
Spring Cloud

kubernetes

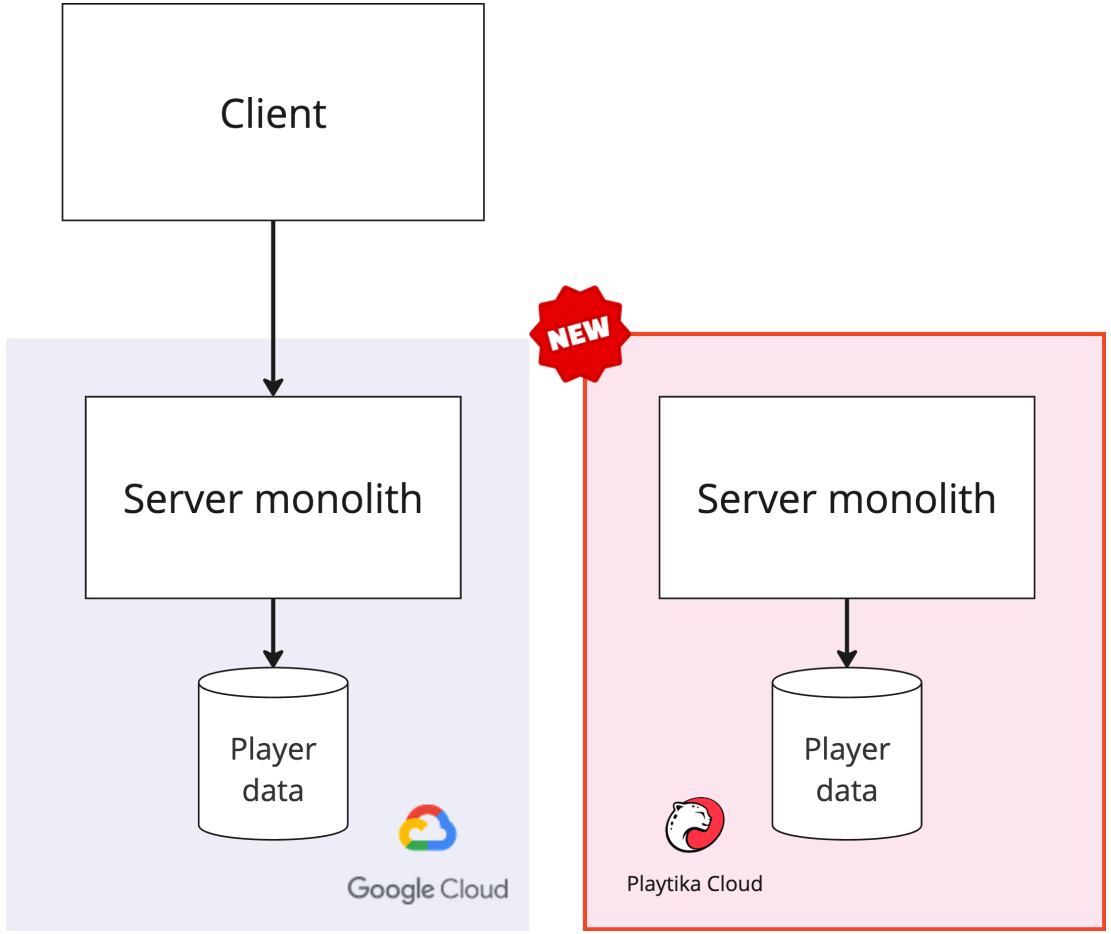
AEROSPIKE

kafka

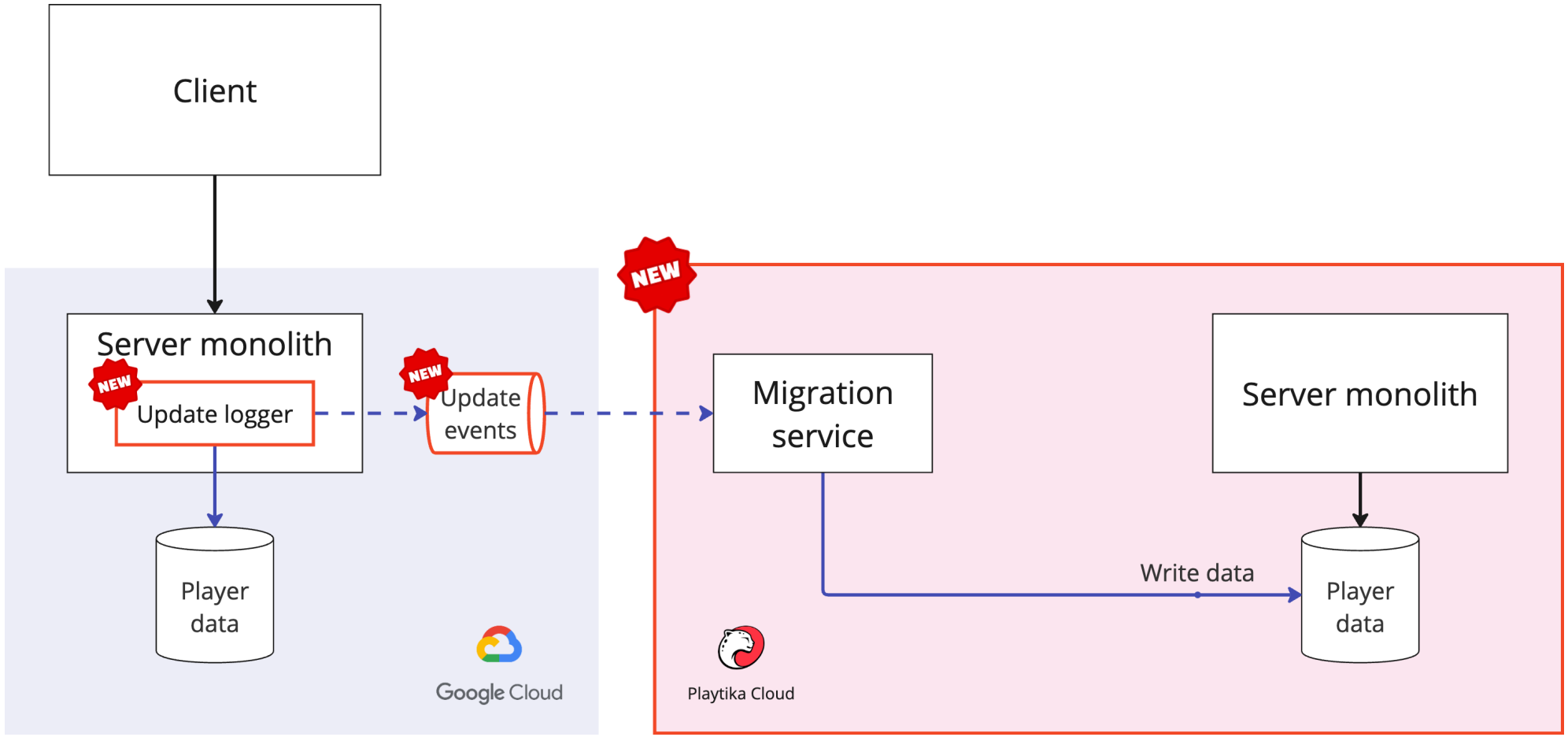
# DATA MIGRATION ARCHITECTURE



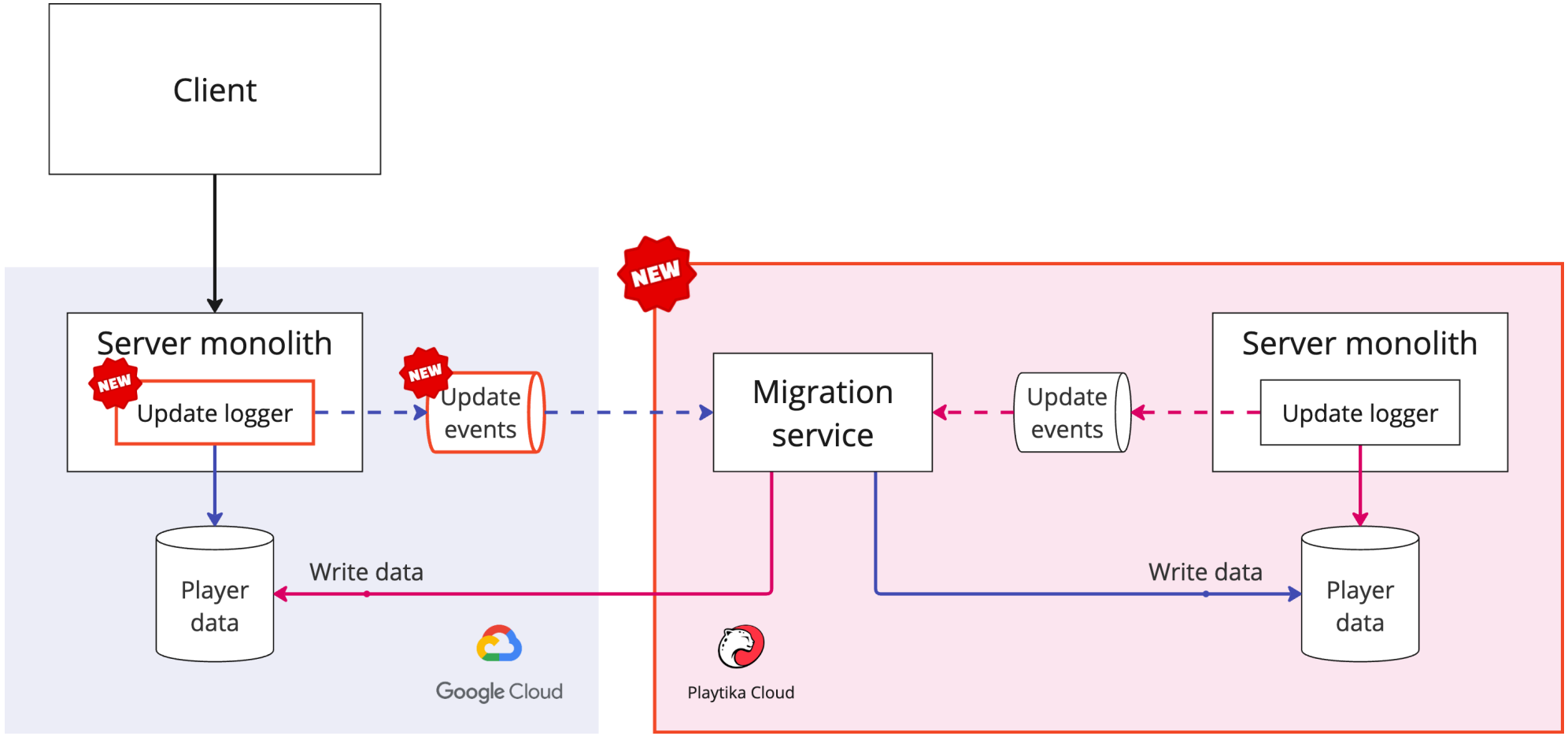

# DATA MIGRATION ARCHITECTURE



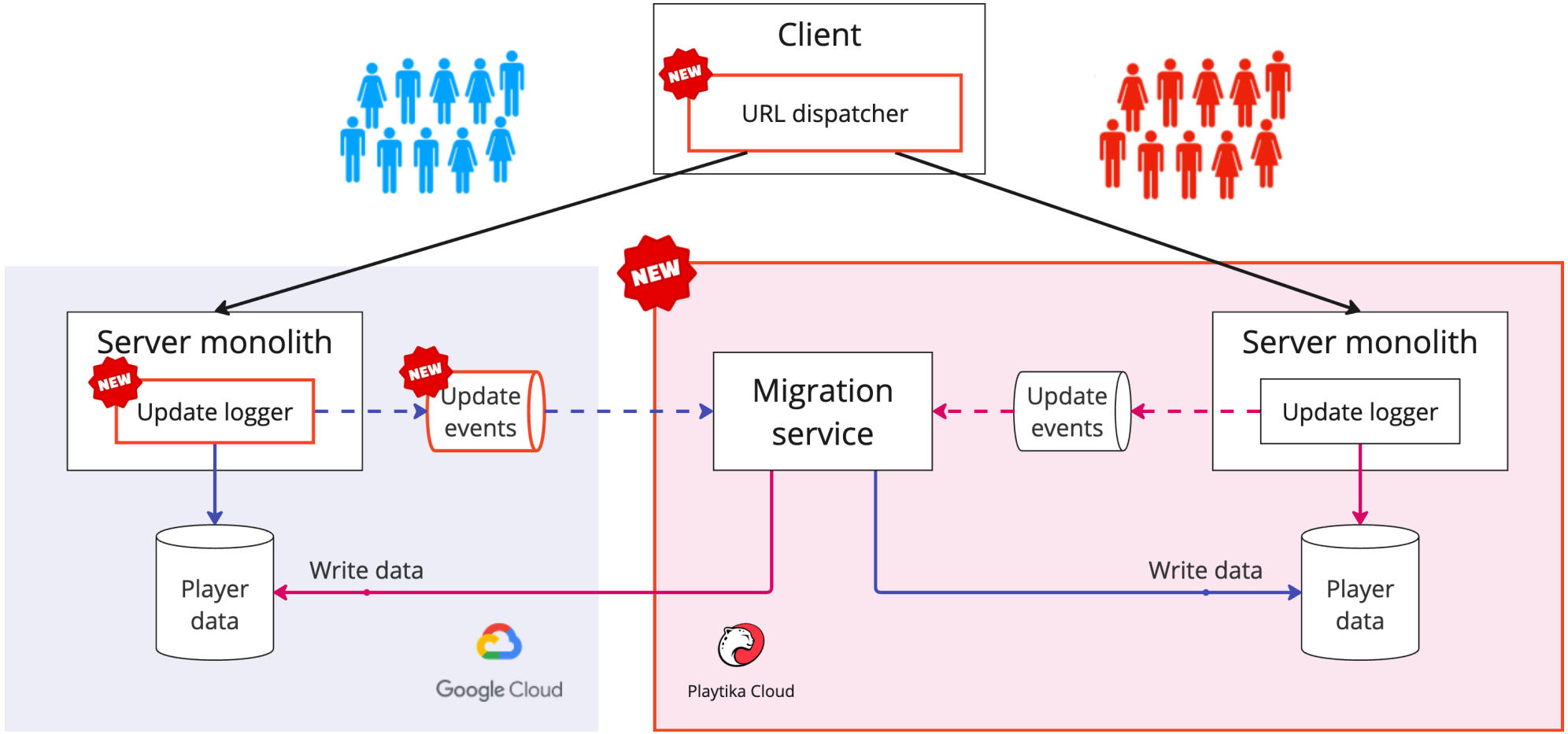
# DATA MIGRATION ARCHITECTURE



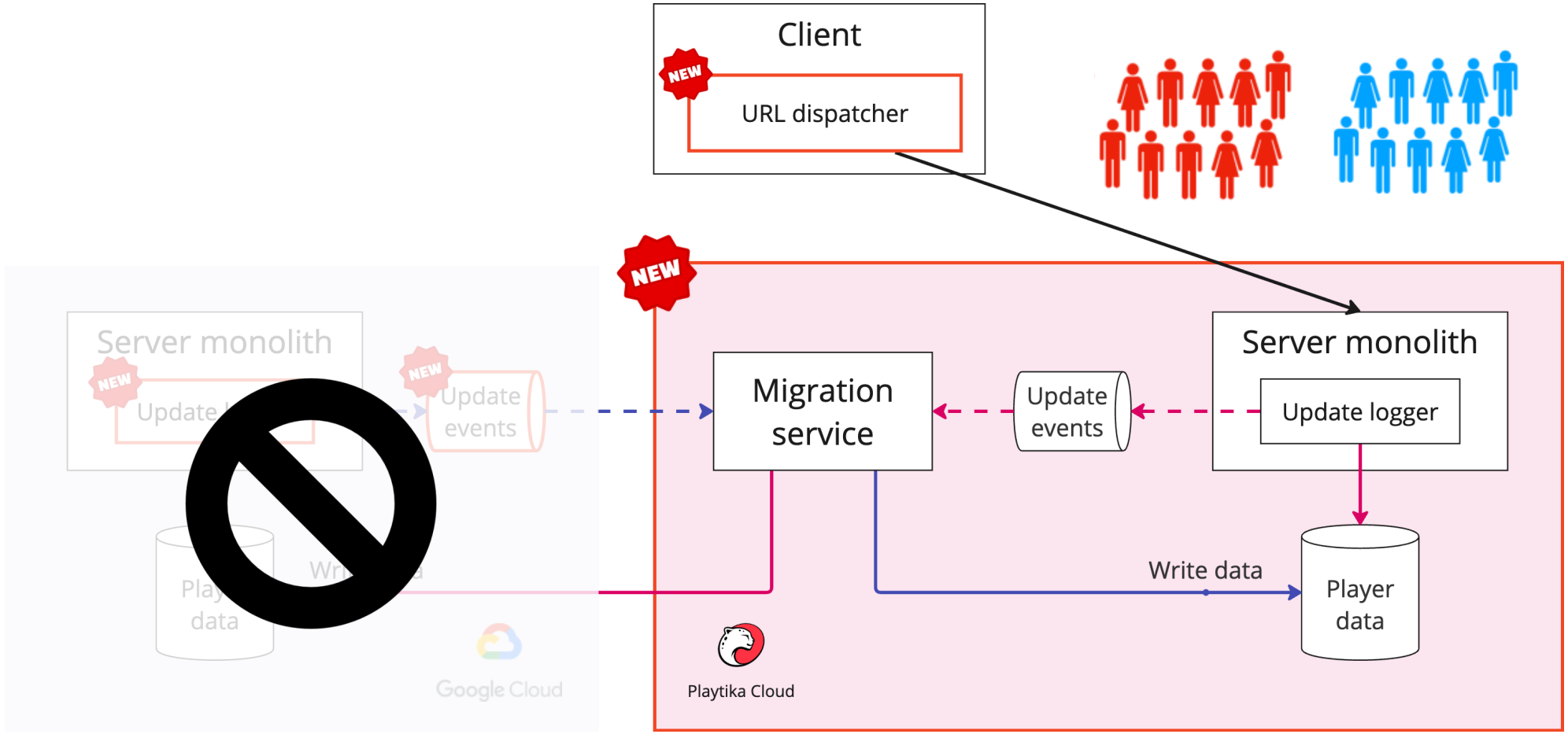
# DATA MIGRATION ARCHITECTURE



# DATA MIGRATION ARCHITECTURE



# DATA MIGRATION ARCHITECTURE



# MIGRATION APPROACH

1. Active data sync
2. Passive data sync
3. Discrapency analysis
4. Technical test with rollback
5. Gradual migration

Playtika

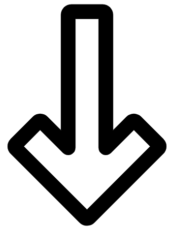
# MIGRATION STATISTICS

0

Downtime

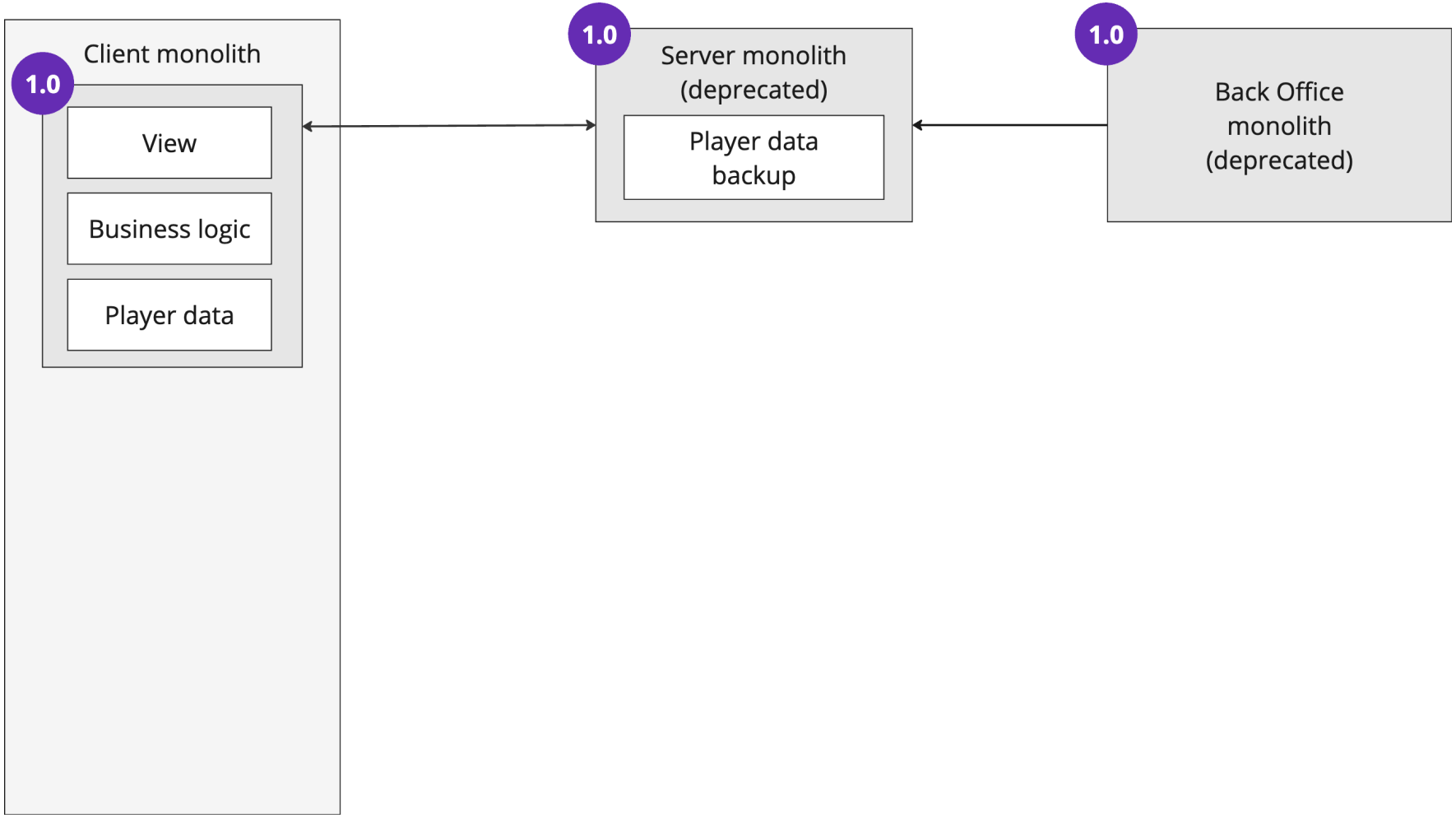
50 TB

Player data migrated

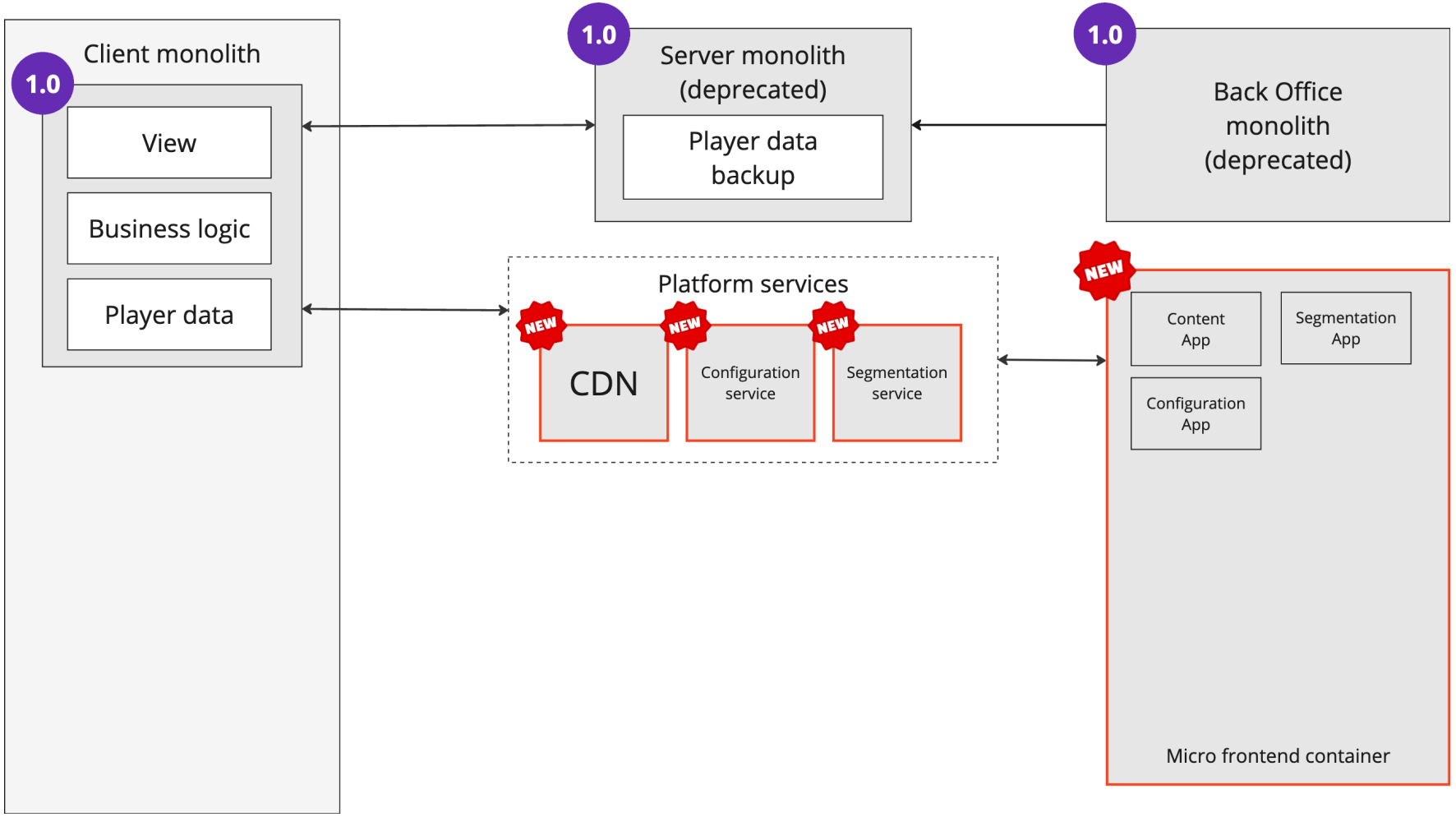


Infrastructure and maintenance costs

# ARCHITECTURE 1.5

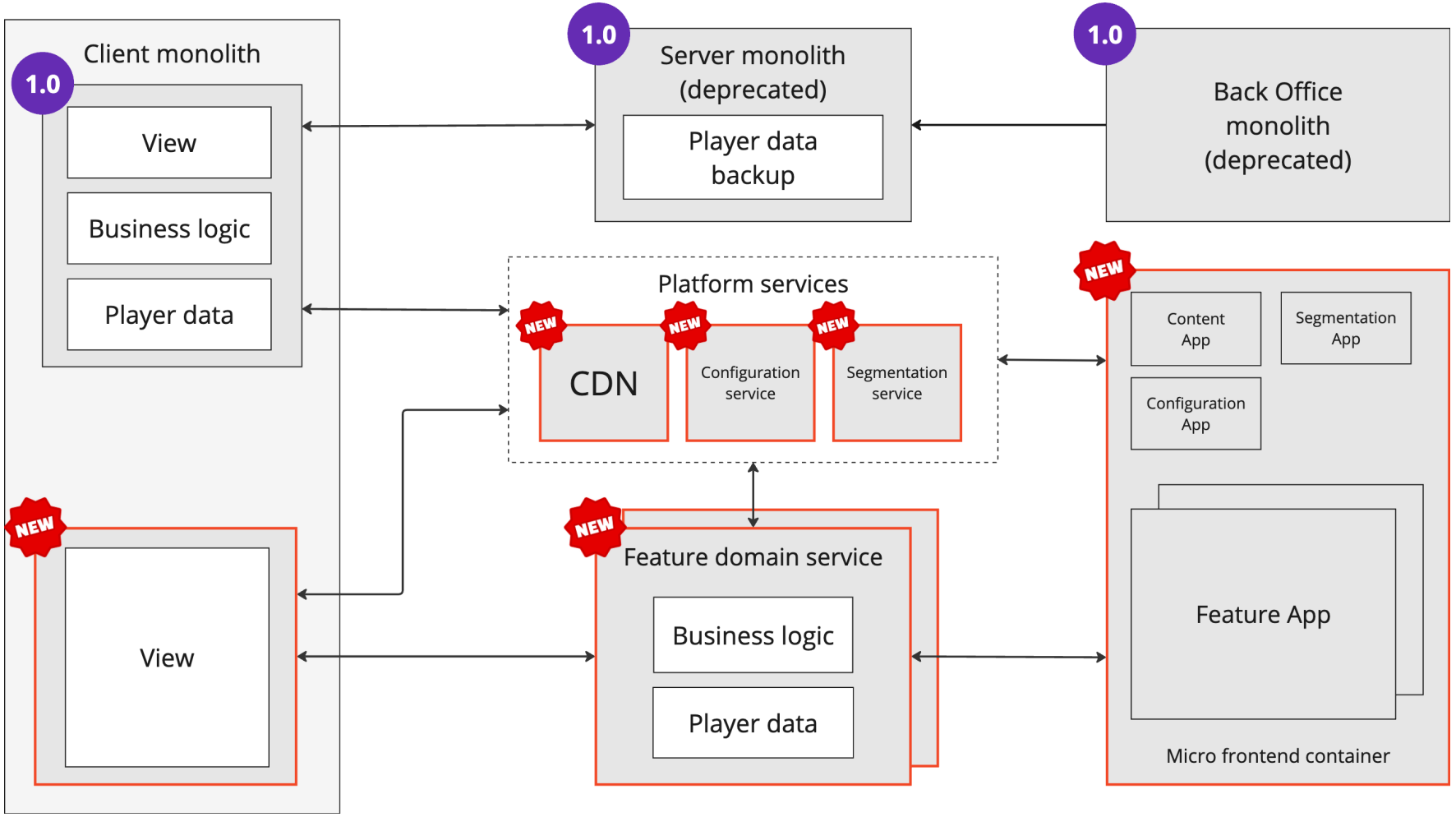


# ARCHITECTURE 1.5

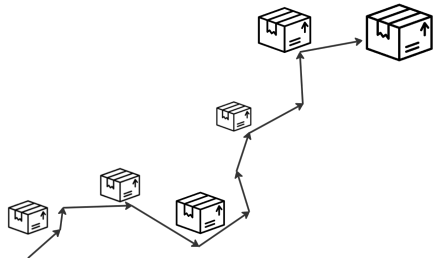



Playtika

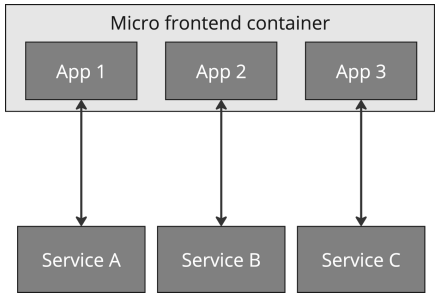
# ARCHITECTURE 1.5



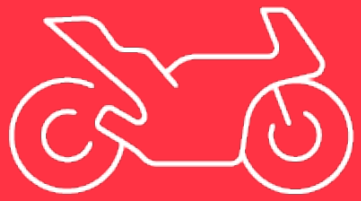

Playtika



QUICK PROTOTYPING
LESS RESTRICTIONS




UNLOCKING PARALLEL
DEVELOPMENT AND RELEASES



**1.0** **1.5** **2.0**

MONOLITH

(NO BOUNDARIES)

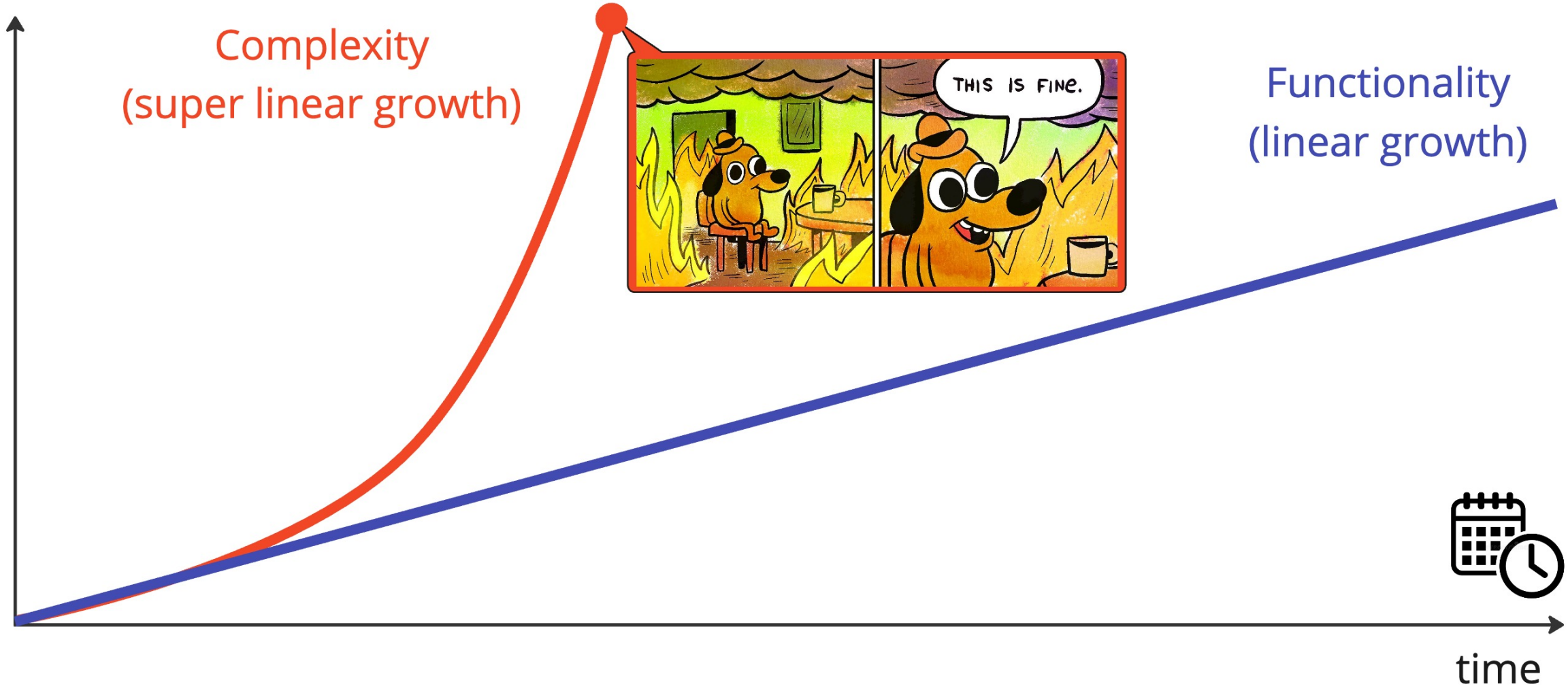
APPLICATION AND CONTENT
RELEASE SEPARATION

MICRO
SERVICES

MICRO
FRONTENDS

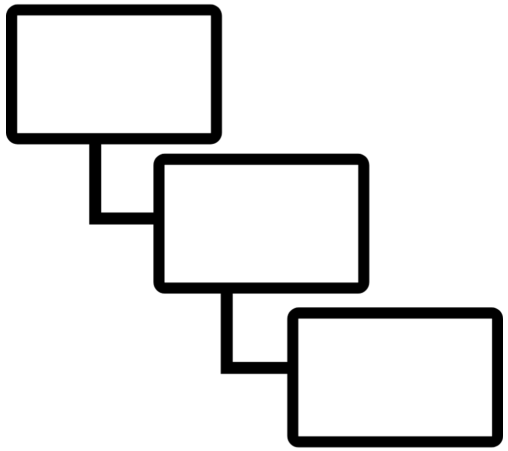
Playtika

# DEVELOPMENT COMPLEXITY

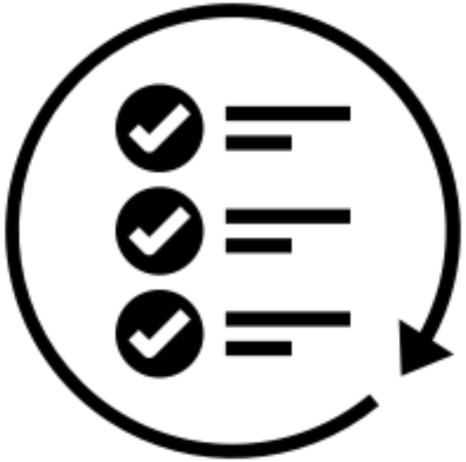



Playtika

# SOURCES OF COMPLEXITY



Cascading changes due to strong coupling



Developing similar solutions from scratch

High cognitive load

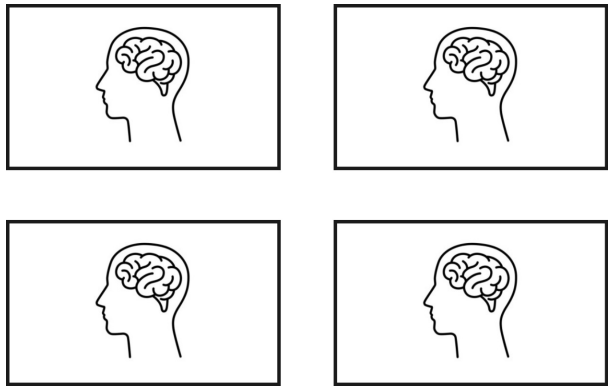
# ARCHITECTURE 2.0

# GOAL

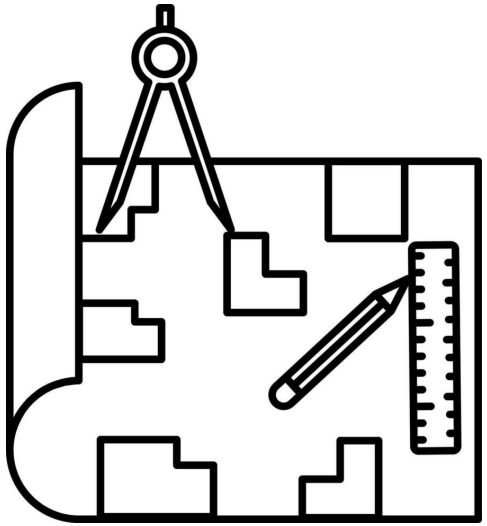
## REDUCE DEVELOPMENT COMPLEXITY AND SCALE THE ORGANIZATION

Playtika

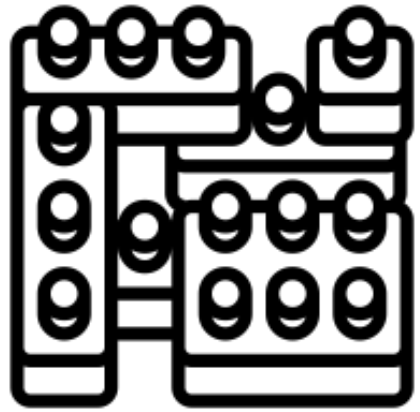
# ARCHITECTURE 2.0 DEFINITION



Revisit the system context boundaries

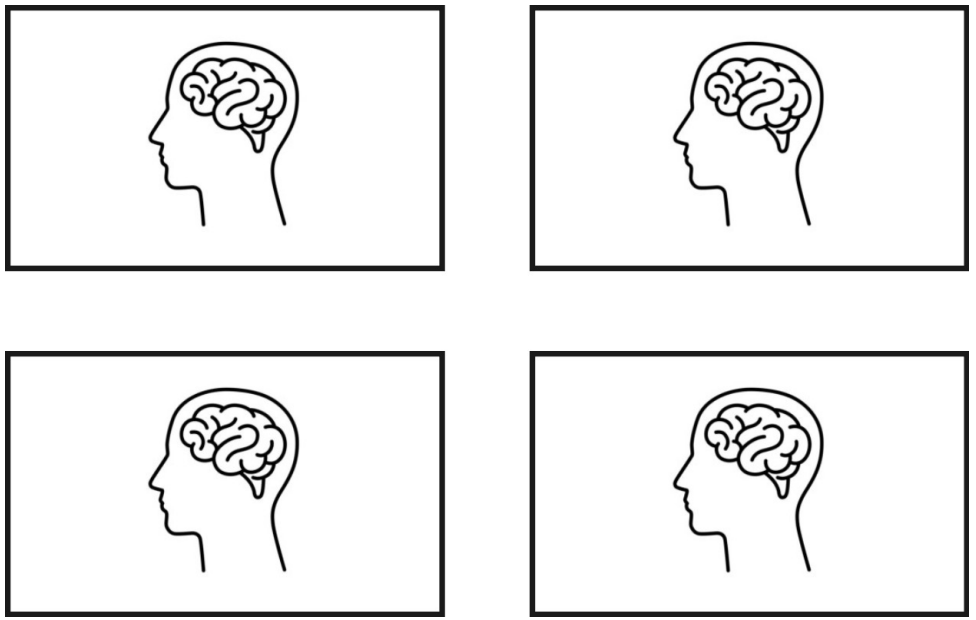


Define feature design standards



Build generic reusable solutions

# CONTEXT BOUNDARIES



Playtika

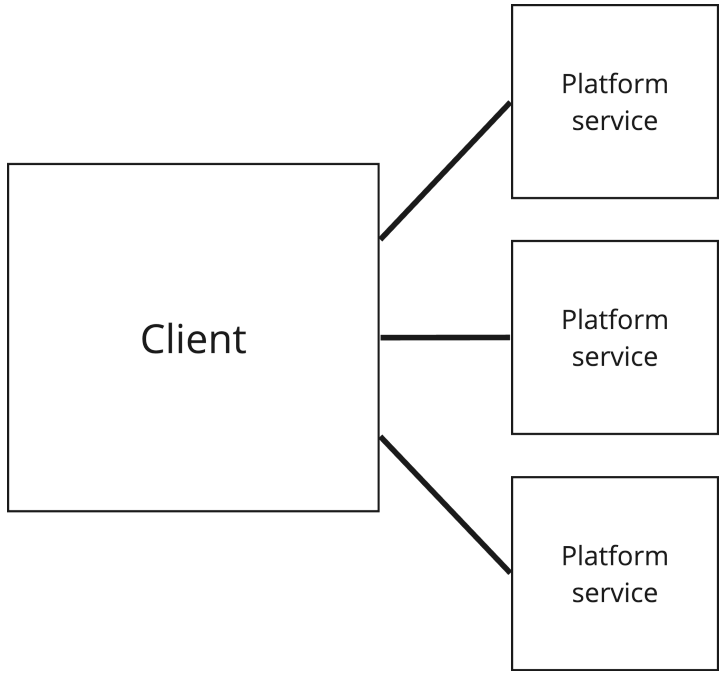
**VisArch**
@ruthmalan

Follow

Architectural design is system design. System design is contextual design — it is inherently about boundaries (what's in, what's out, what spans, what moves between), and about tradeoffs. It reshapes what is outside, just as it shapes what is inside.

4:50 PM · Jan 5, 2019

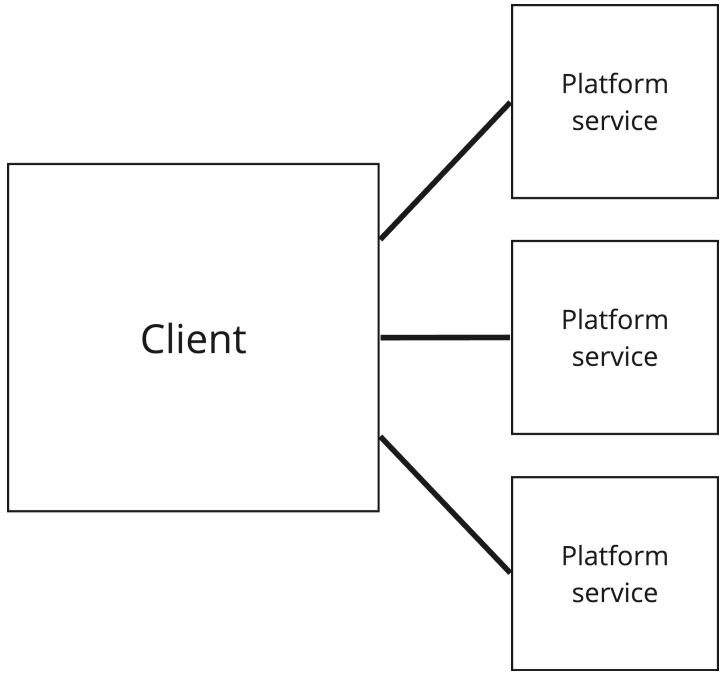
# 1.5




Client and platform services share knowledge

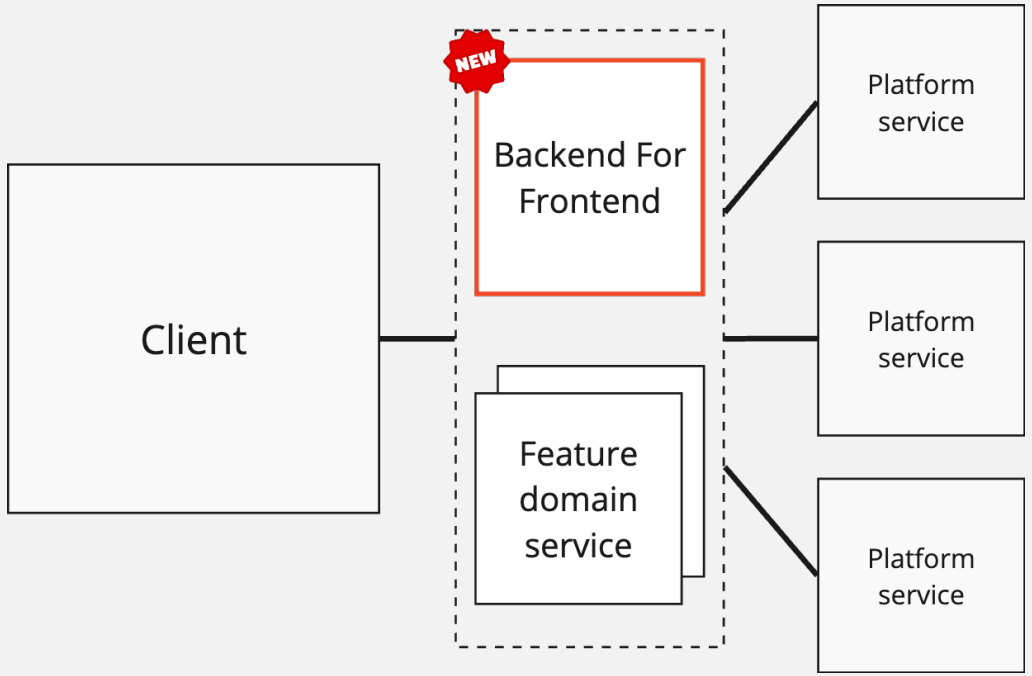
Playtika

# 1.5




Client and platform services share knowledge

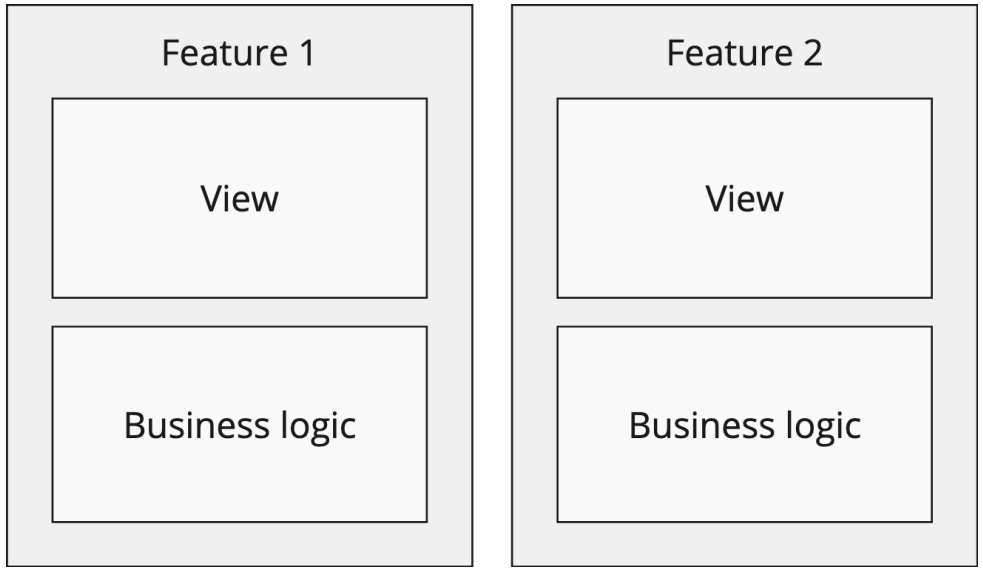
# 2.0




Knowledge and changes are isolated

Playtika

# 1.5

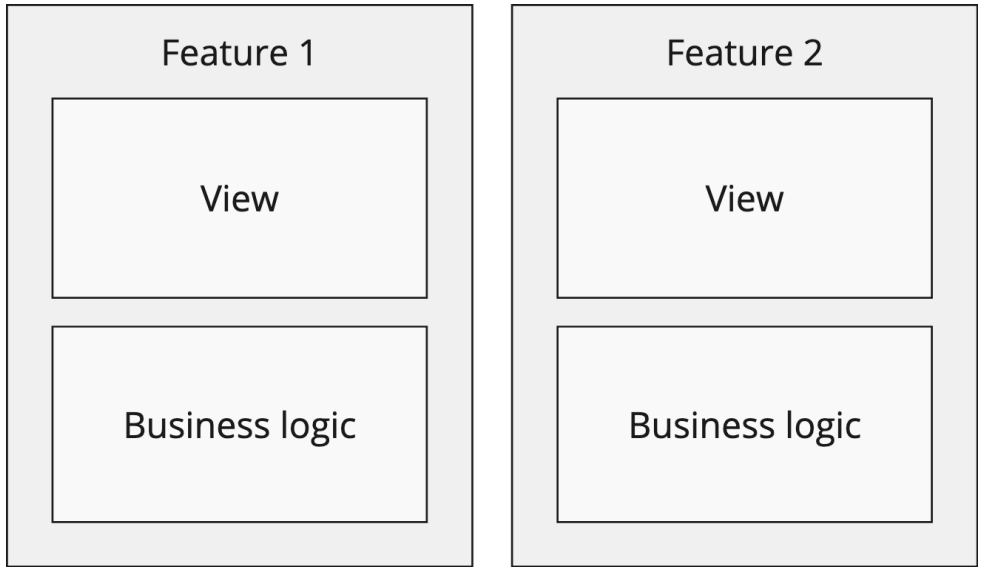



Repeated design and development efforts for every feature

Cross-feature changes requires patching the whole game

Playtika

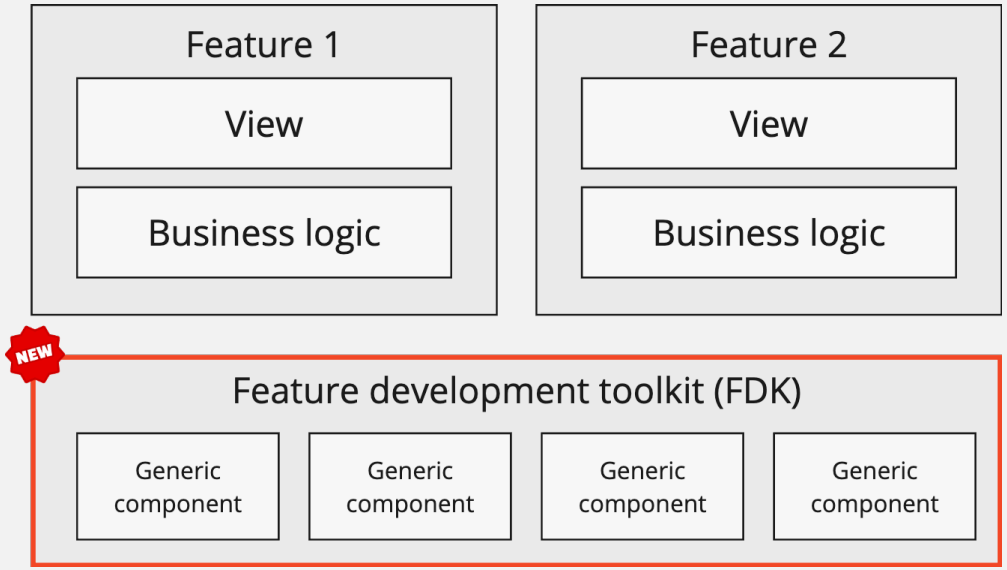
# 1.5




Repeated design and development efforts for every feature

Cross-feature changes requires patching the whole game

# 2.0

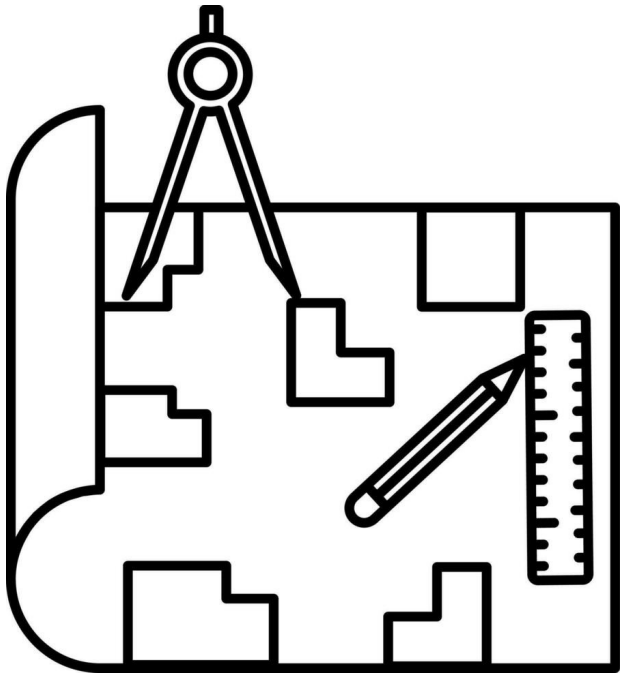



Focus on features, not on tech decisions

Cross-feature changes require change in FDK only

Playtika

# FEATURE DESIGN STANDARDS



Playtika

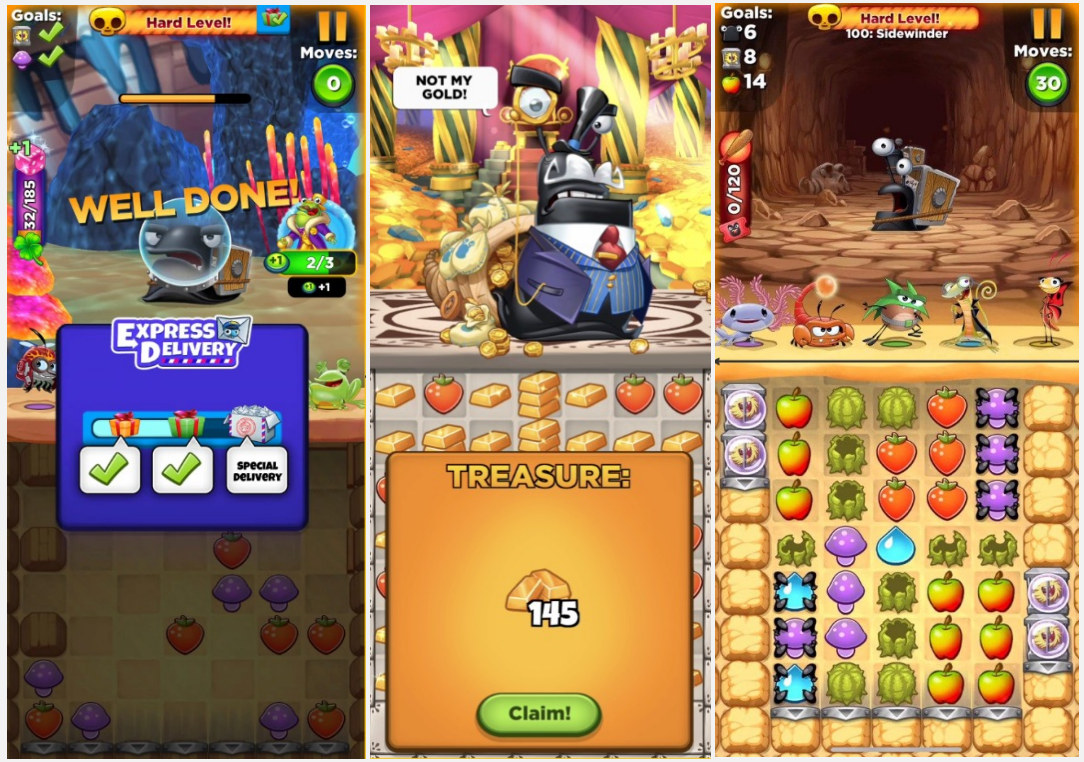
# ONLINE-FIRST

Meta or multiplayer games with no active user interaction



# OFFLINE-FIRST

In level mechanics with active user interaction




Playtika

# ONLINE-FIRST

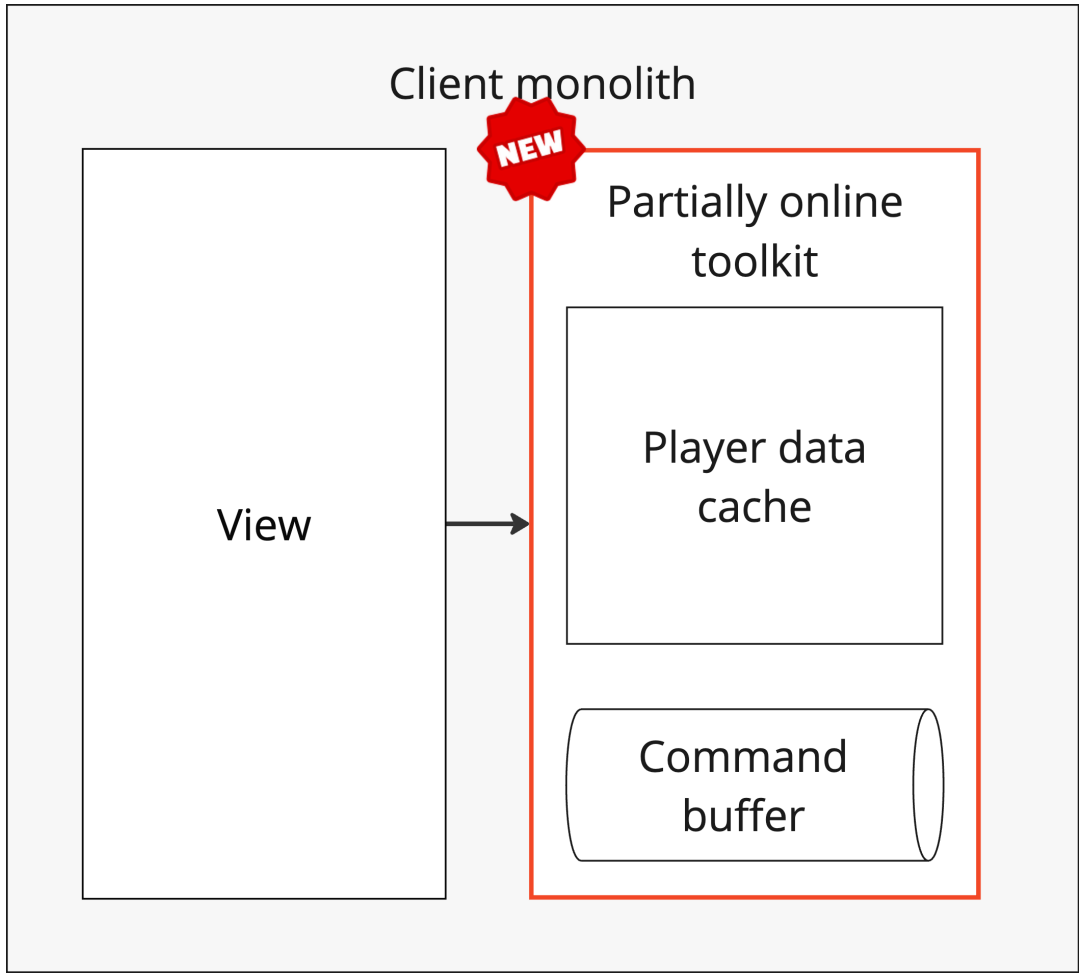
Meta or multiplayer games with no active user interaction



Require the Internet connection to play, limited functionality works offline

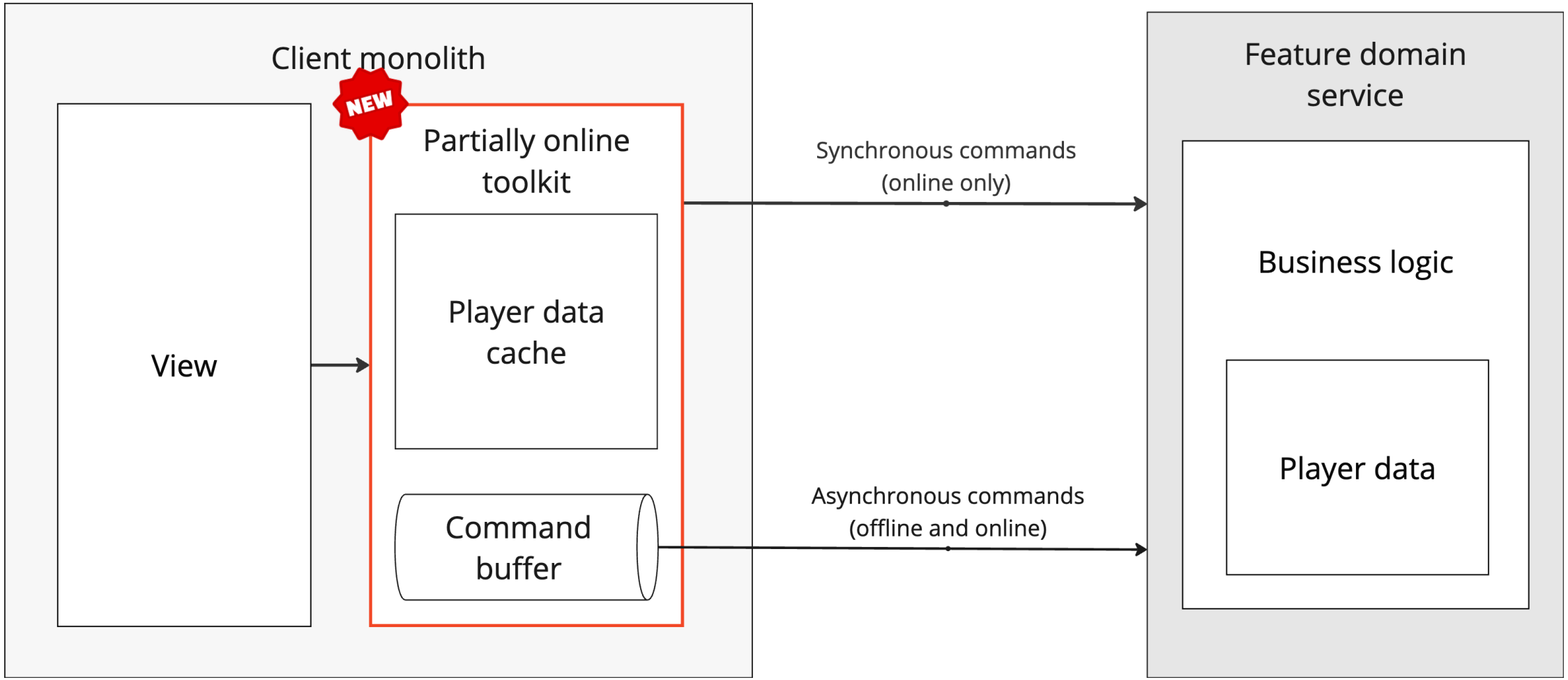
<u>Server is source of truth for the player state</u>

# ONLINE-FIRST ARCHITECTURE




Playtika

# ONLINE-FIRST ARCHITECTURE

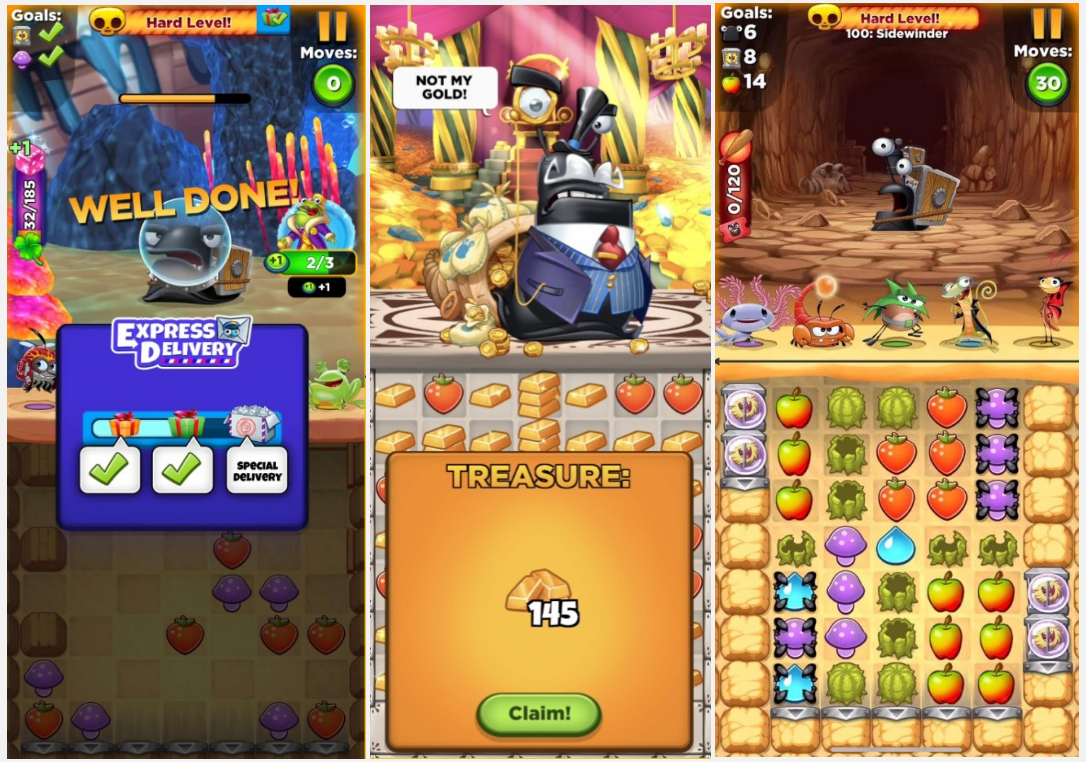


# OFFLINE-FIRST

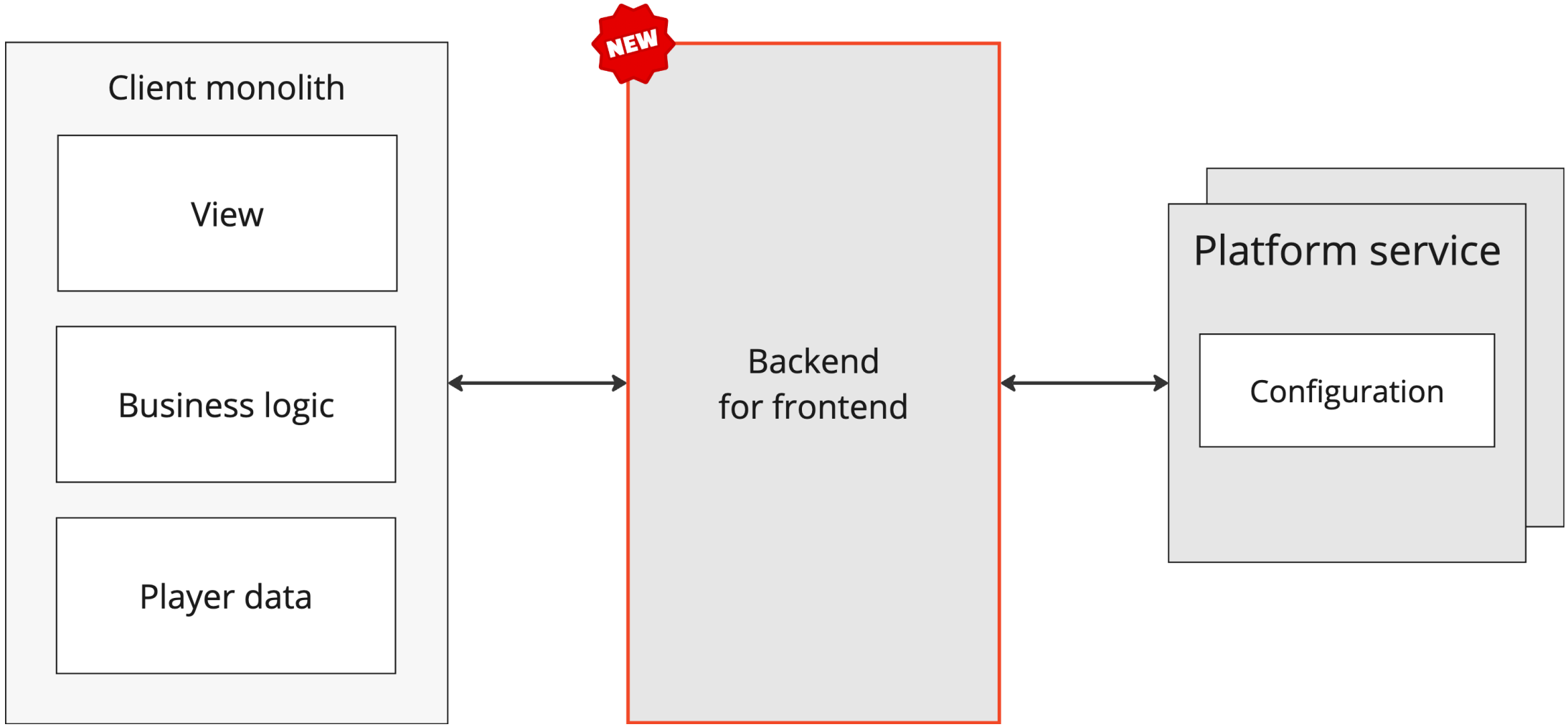
Require Internet connection to download the content and configuration

<u>Client is source of truth for the player state</u>

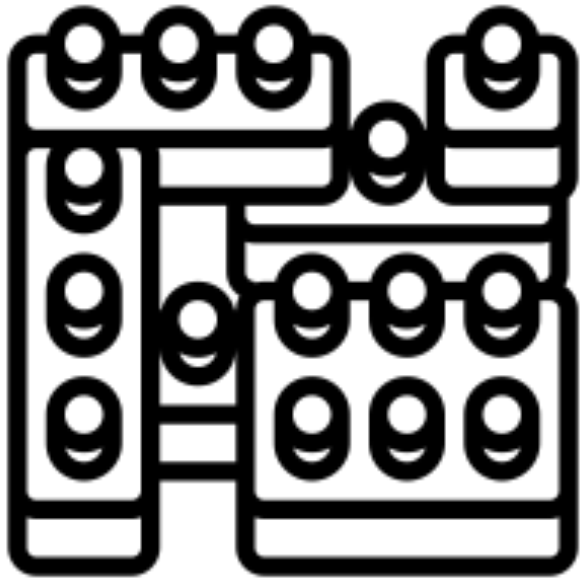
In level mechanics with active user interaction



# OFFLINE-FIRST ARCHITECTURE



# FEATURE DEVELOPMENT TOOLKIT (FDK)



Playtika

# REWARDING FLOW





Earn resources

Spend resources

# ARCHITECTURE

Each feature has specific implementation of the

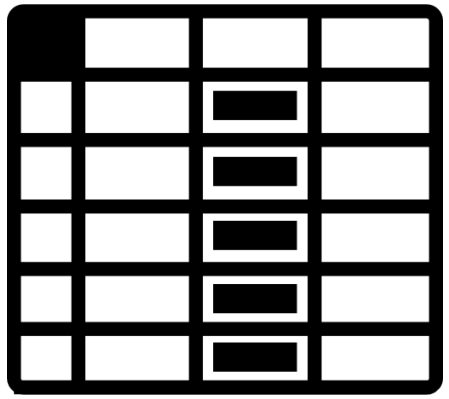
“rewarding flow”


# CONSEQUENCES

- Adding new game resource type requires patching all features
- Each feature requires own ETL pipelines to process analytical events
- Extra efforts for supporting multiple implementations

Playtika

# REWARDING TOOLKIT (FDK)



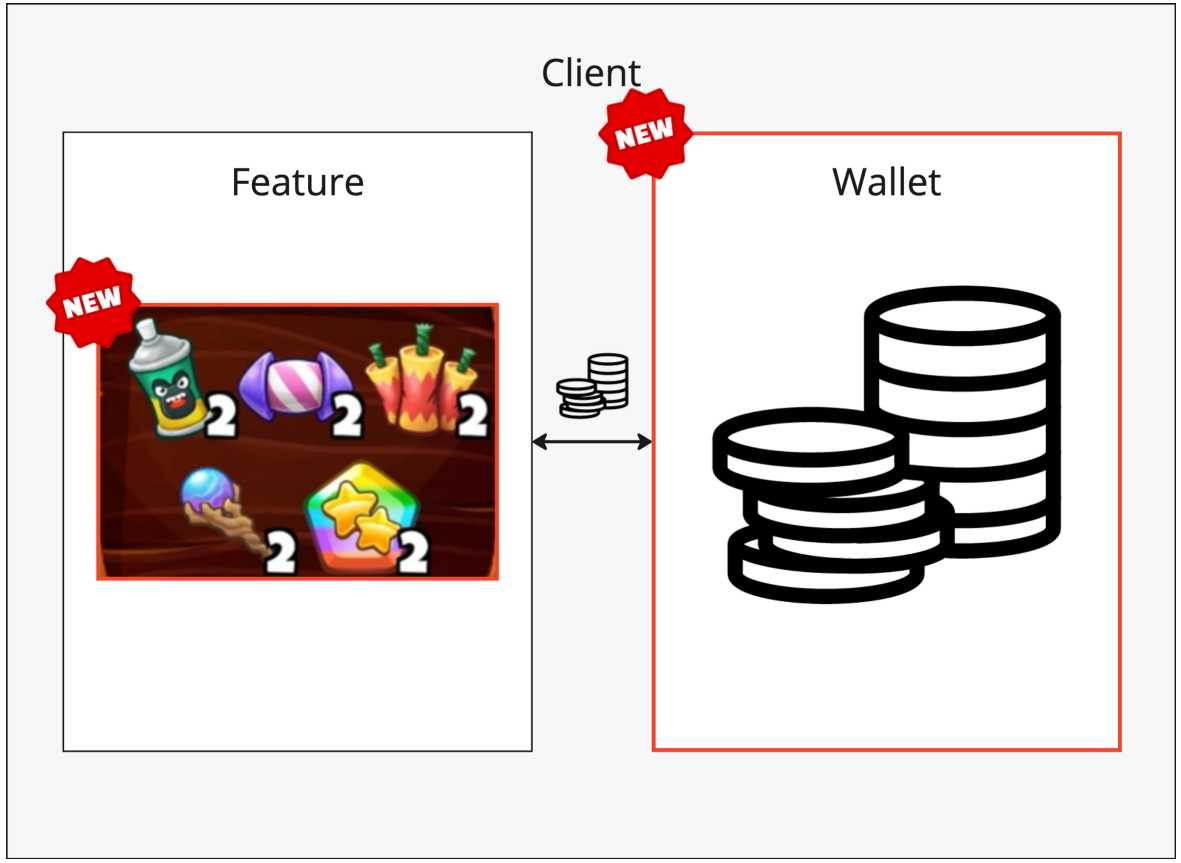

Configuration system

UI components

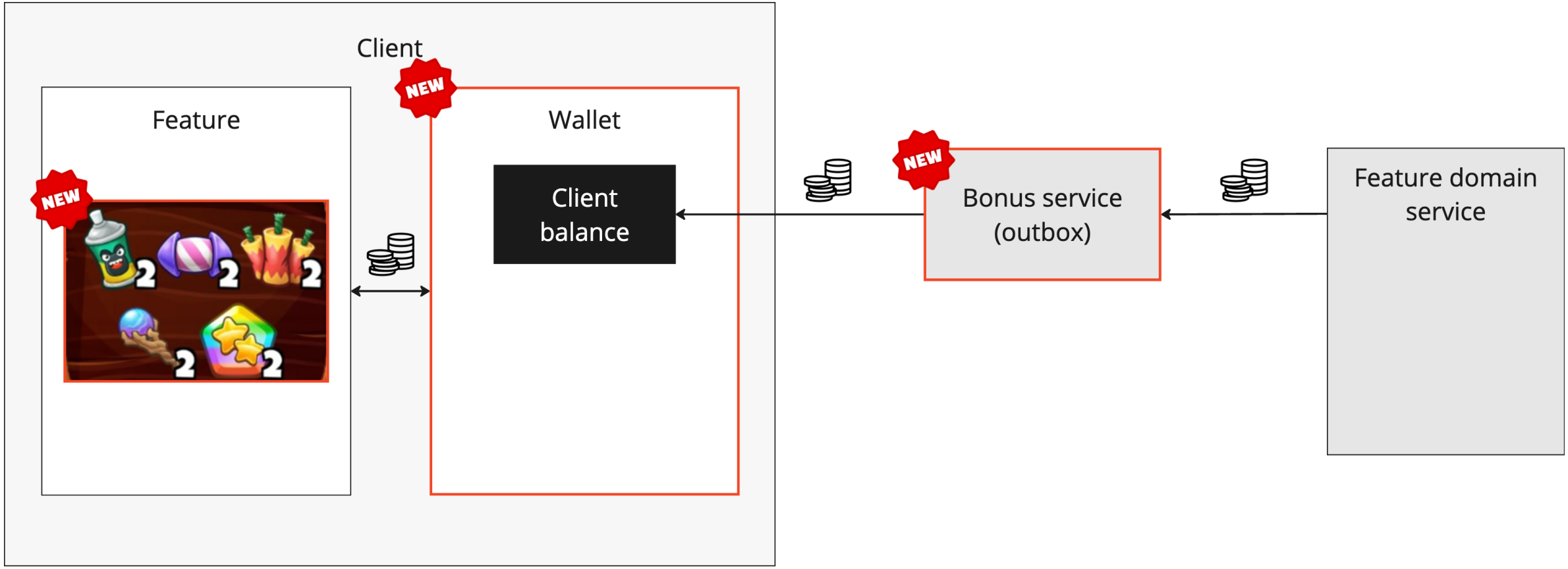
Transaction flow

Analytics

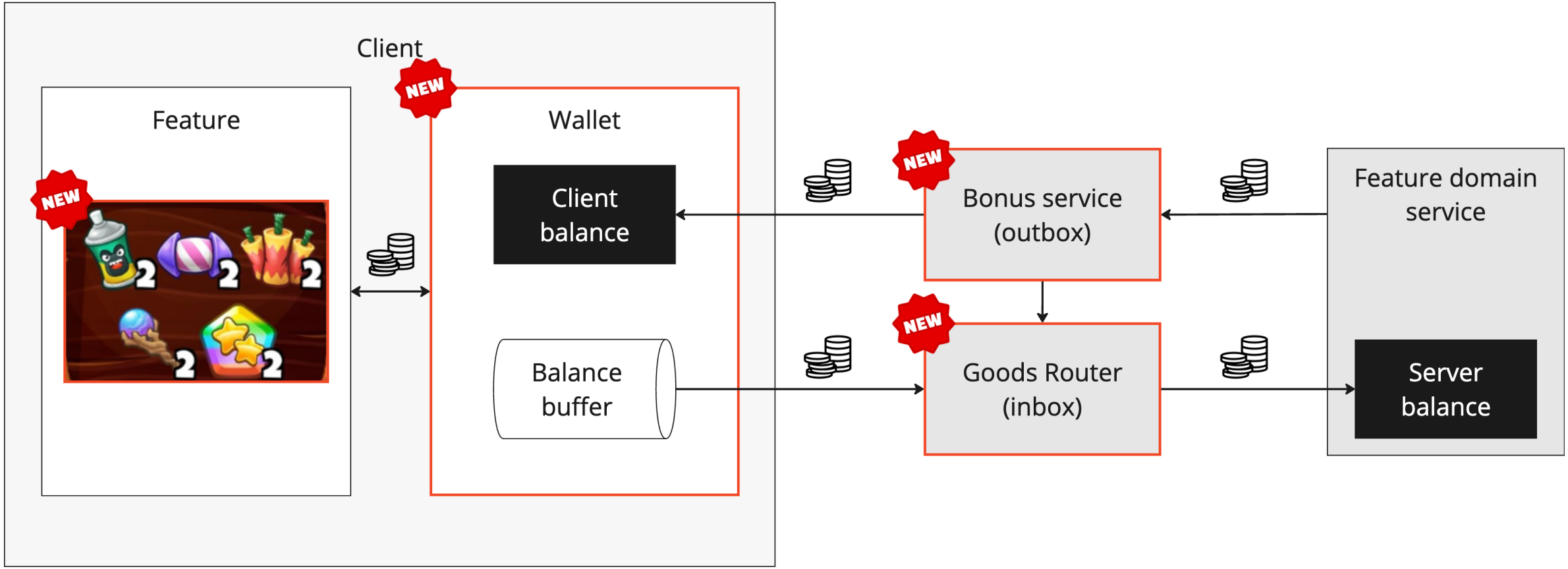
# REWARDING FLOW ARCHITECTURE



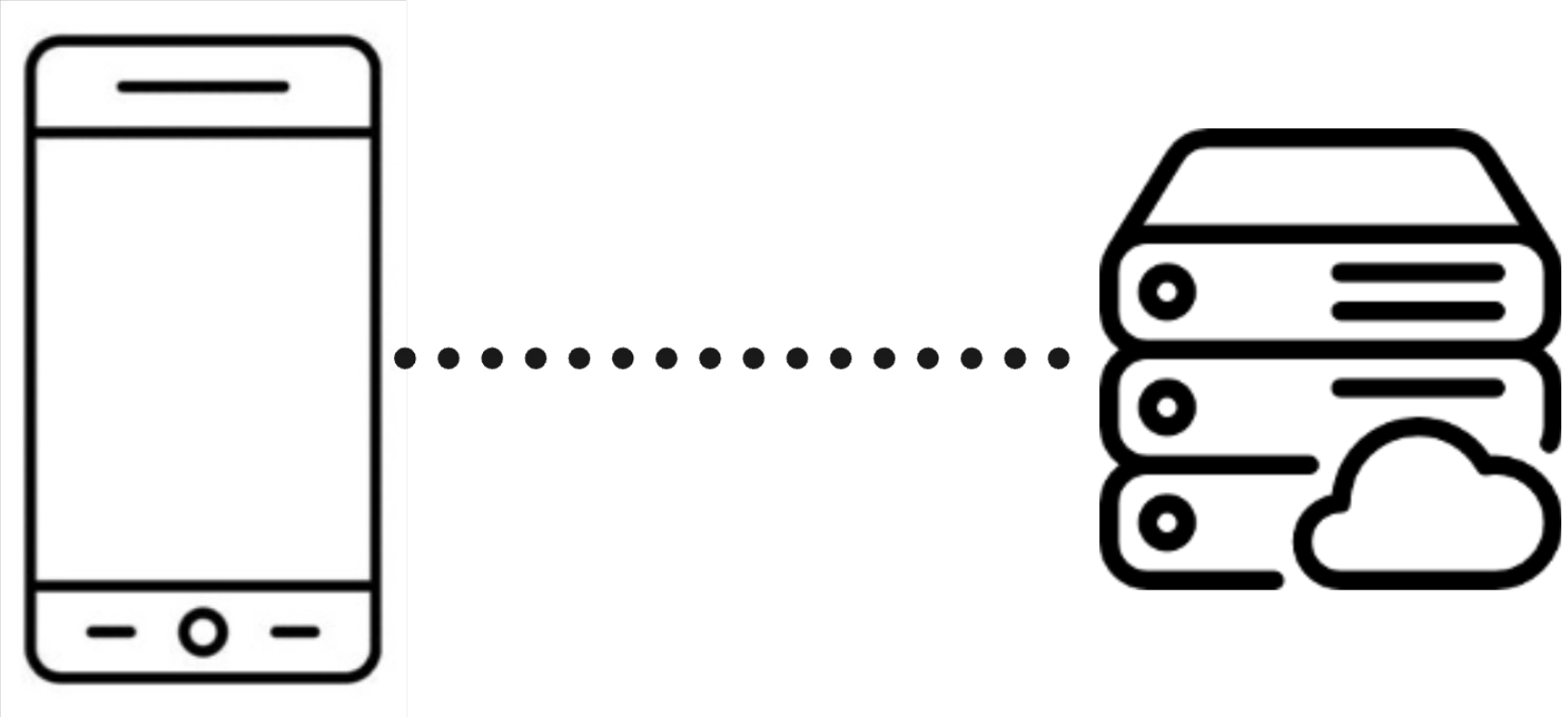
# REWARDING FLOW ARCHITECTURE



# REWARDING FLOW ARCHITECTURE



# PLAYER STATE SYNCHRONIZATION



Playtika

# ARCHITECTURE

Each feature has specific implementation  of the

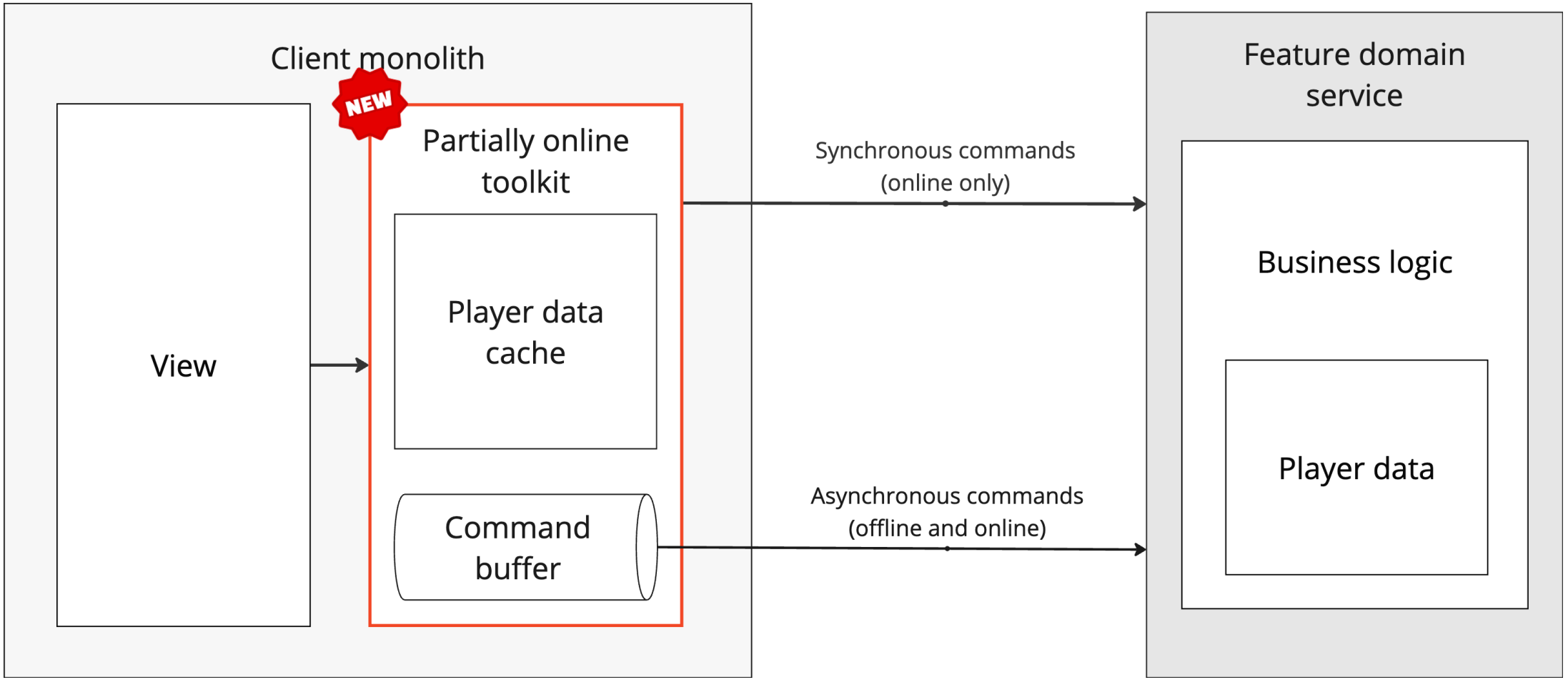
“player state synchronization”


# CONSEQUENCES

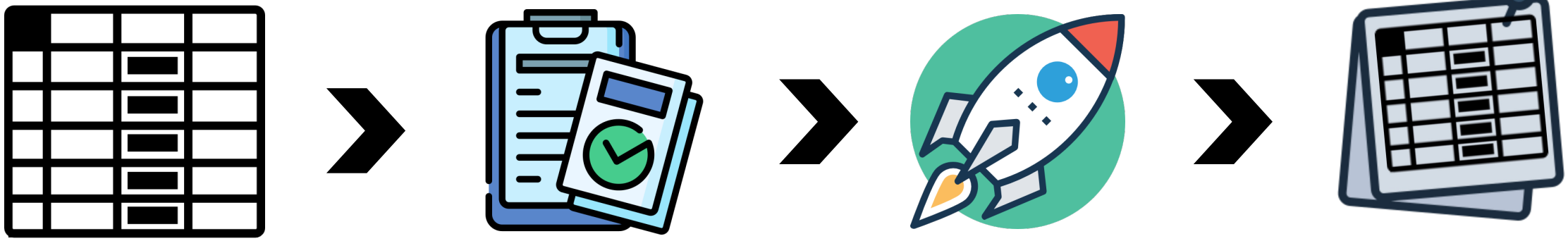
- Game freezes
- Data discrepancies
- Extra efforts for supporting multiple implementations

Playtika

# PARTIALLY ONLINE TOOLKIT (FDK)



# GAME OPERATIONS FLOW



Configuration

Pre-launch verification

Launch

Stickiness

# ARCHITECTURE

Each feature has specific implementation of the

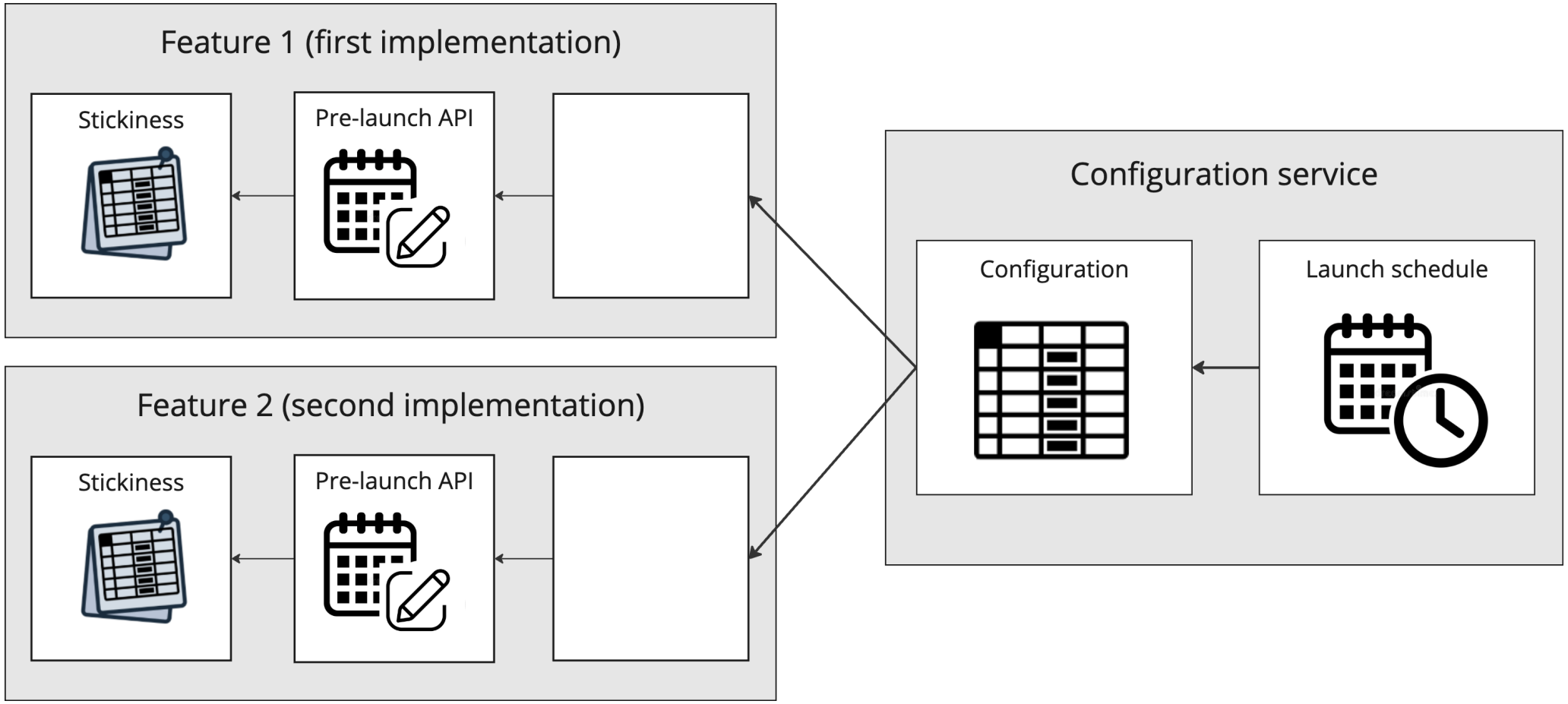
“game operations flow”


# CONSEQUENCES

- Some features have no stickiness or pre-launch mode
- Extra efforts for supporting multiple implementations

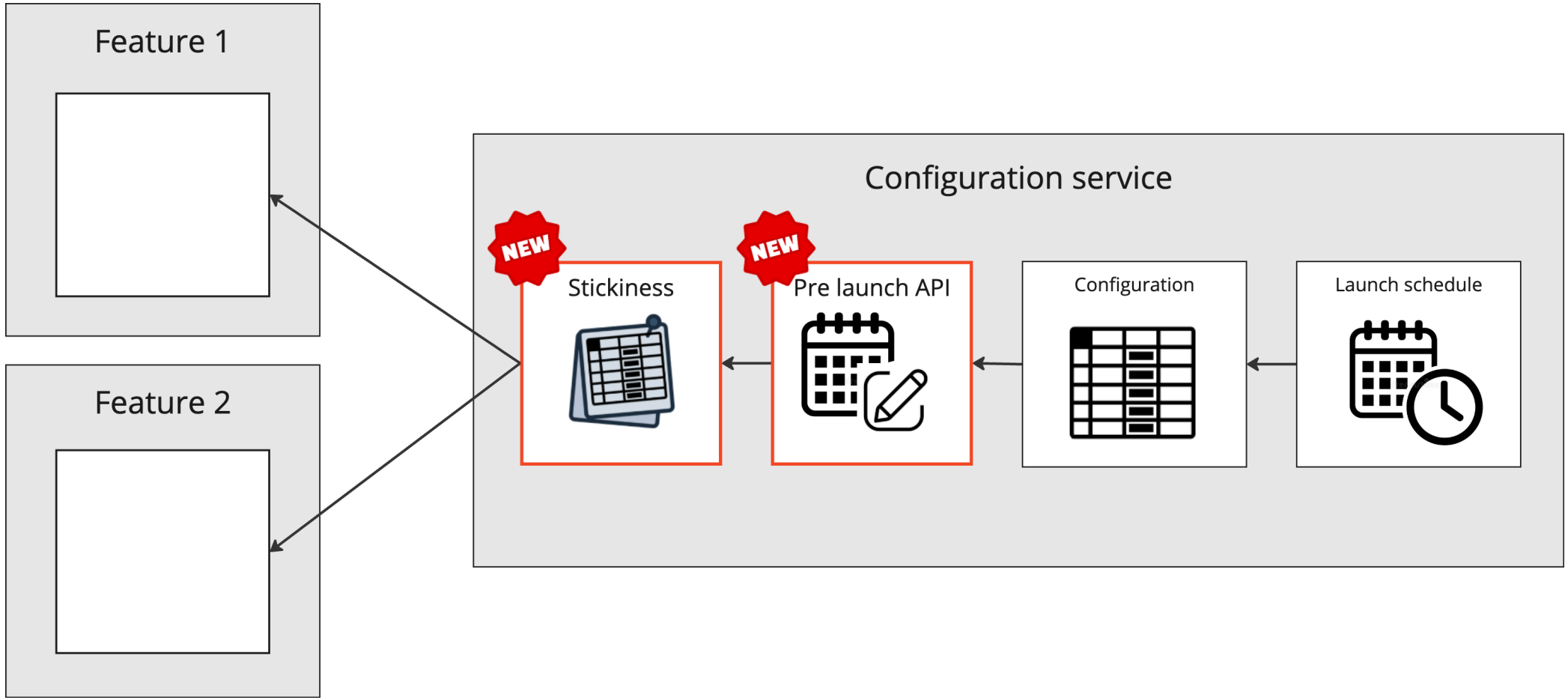
Playtika

# CONFIGURATION SYSTEM 1.5



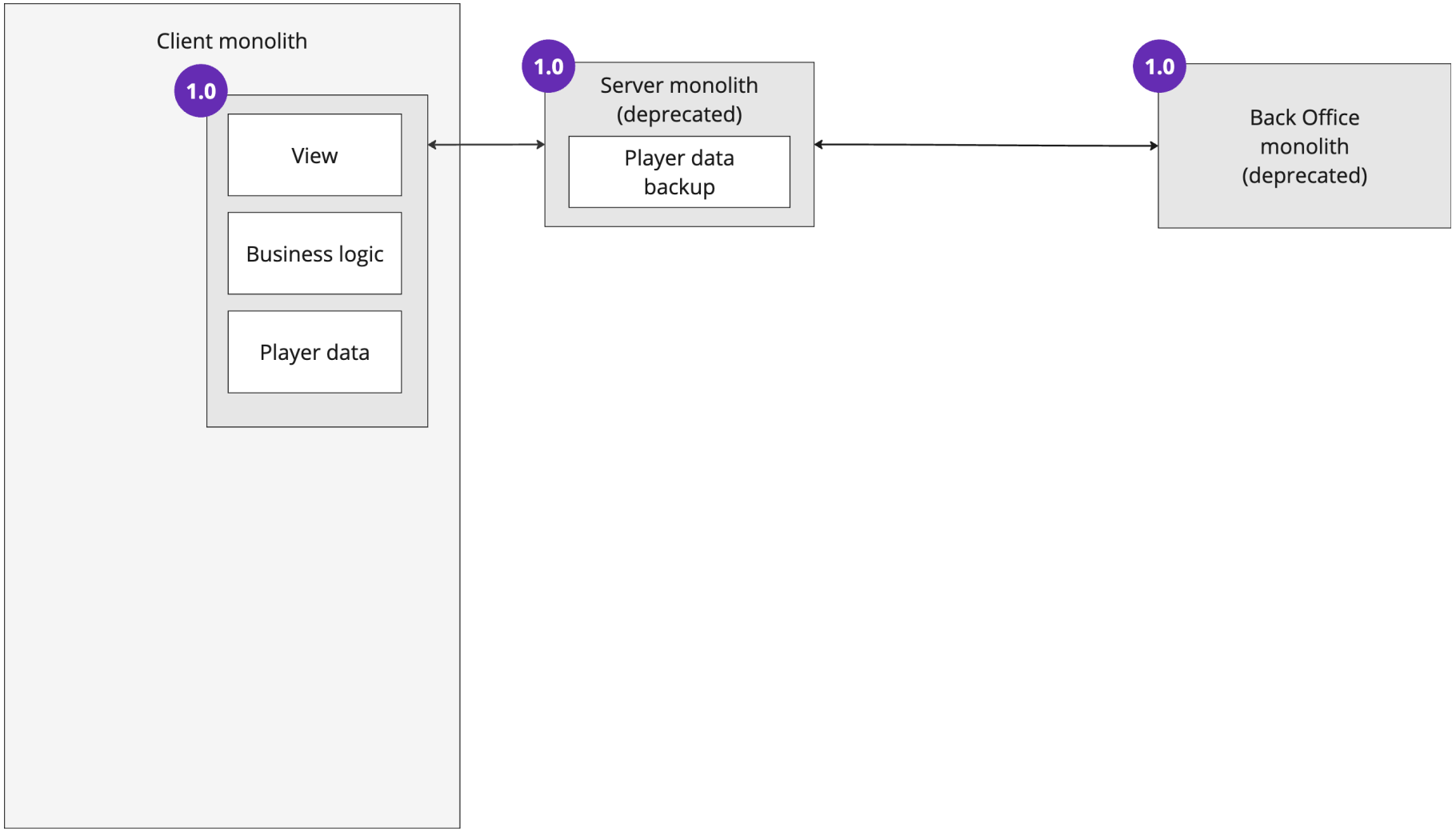

Playtika

# CONFIGURATION SYSTEM 2.0

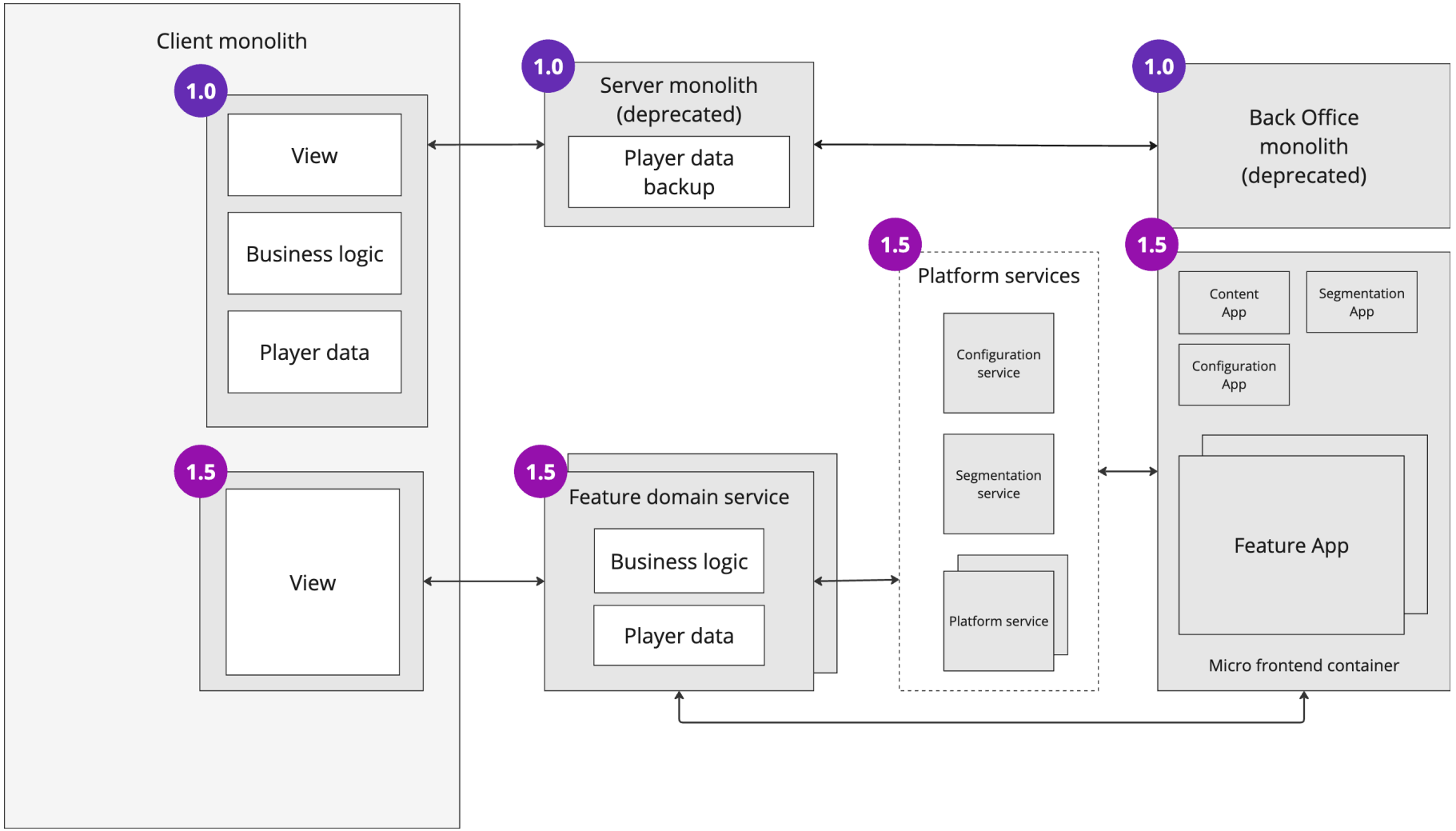



Playtika

# ARCHITECTURE 2.0

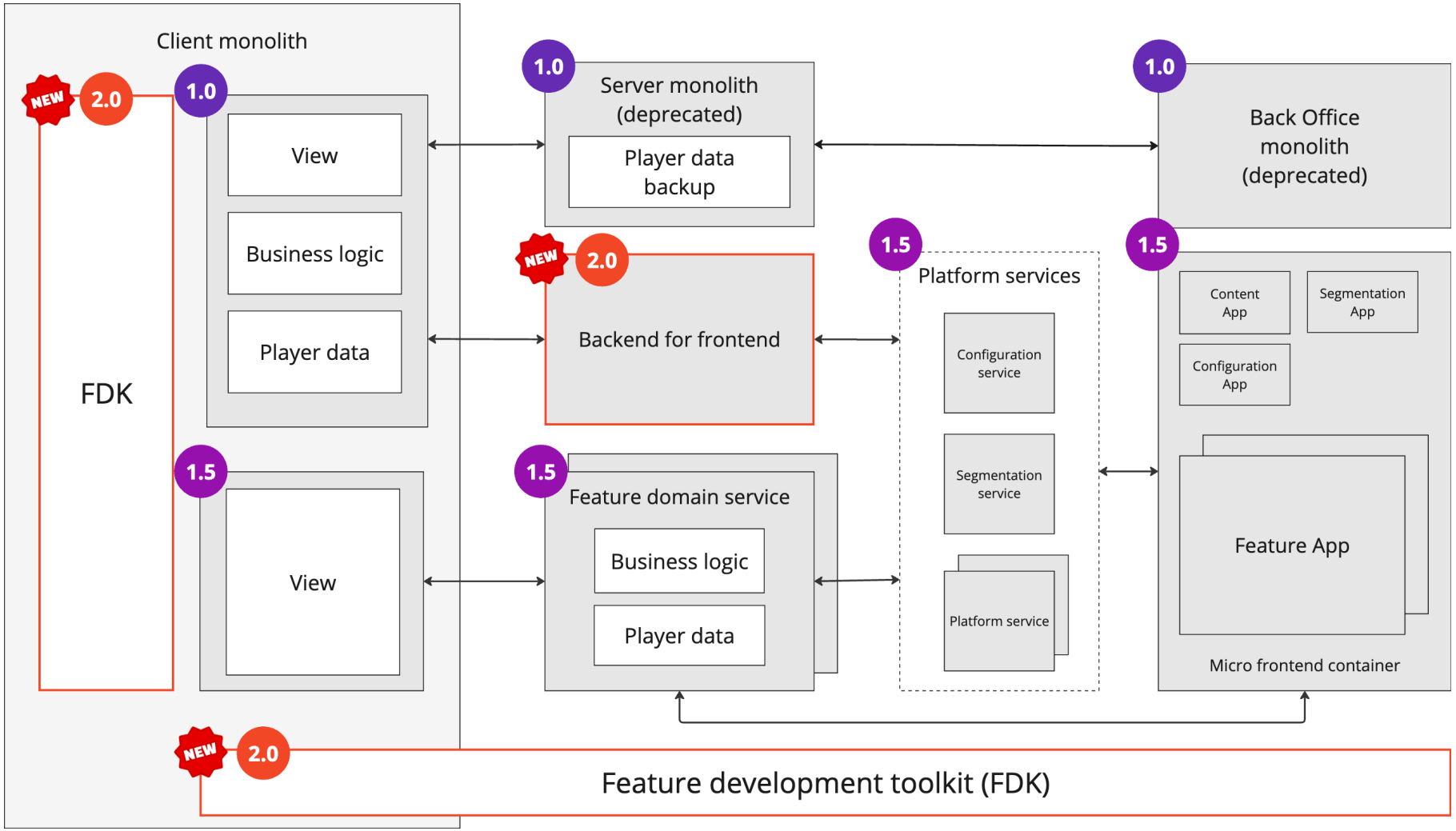


# ARCHITECTURE 2.0




Playtika

# ARCHITECTURE 2.0




Playtika

# DELIVERY APPROACH

## BUILDING A SOCIO-TECHNICAL SYSTEM

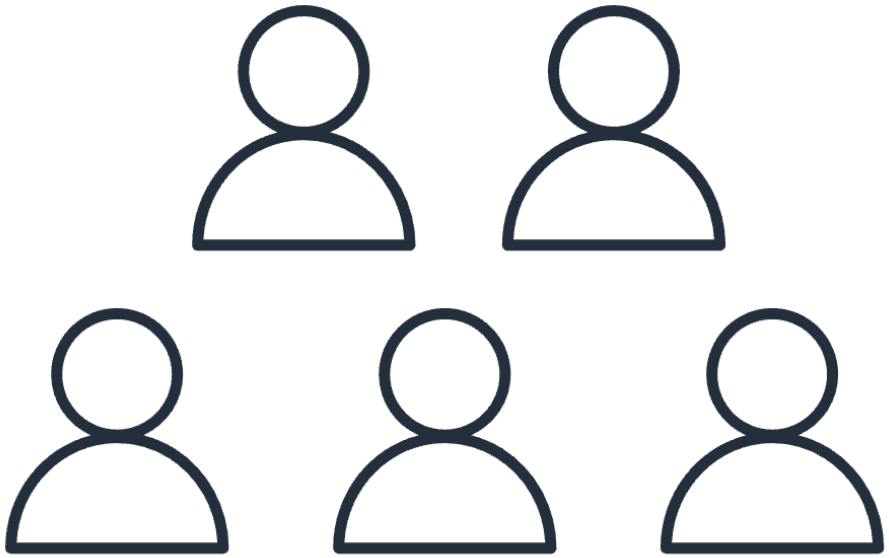
Playtika

# CONWAY'S LAW

"An organization's system design will be a copy of the organization's communication structure”

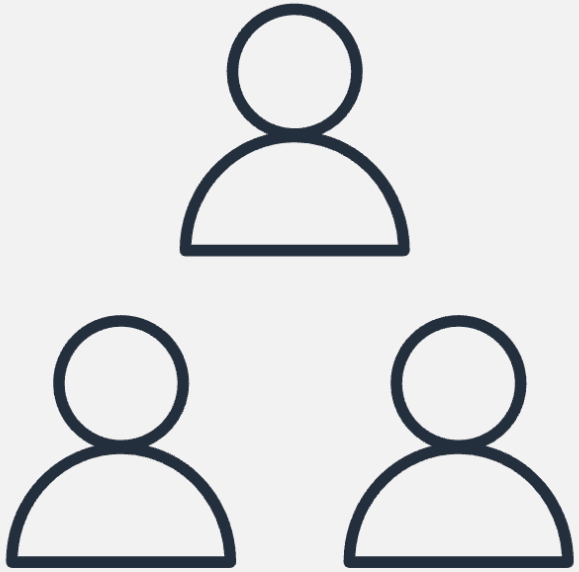
Melvin Conway
“How do committees invent?” (1968)

# COBOL



5 phases of compilation*

# ALGOL

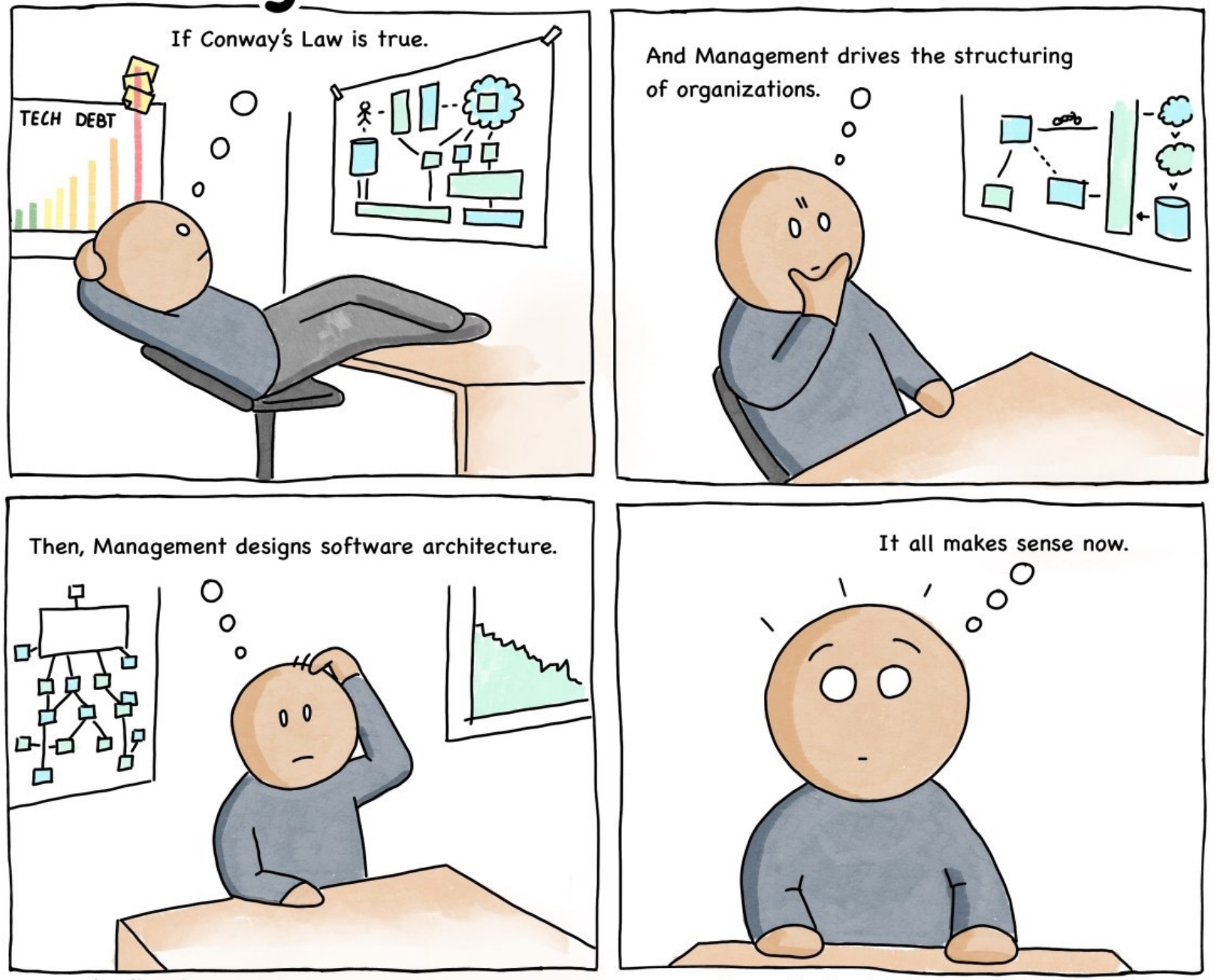


3 phases of compilation*

*source: Melvin Conway “How do committees invent?” (1968)

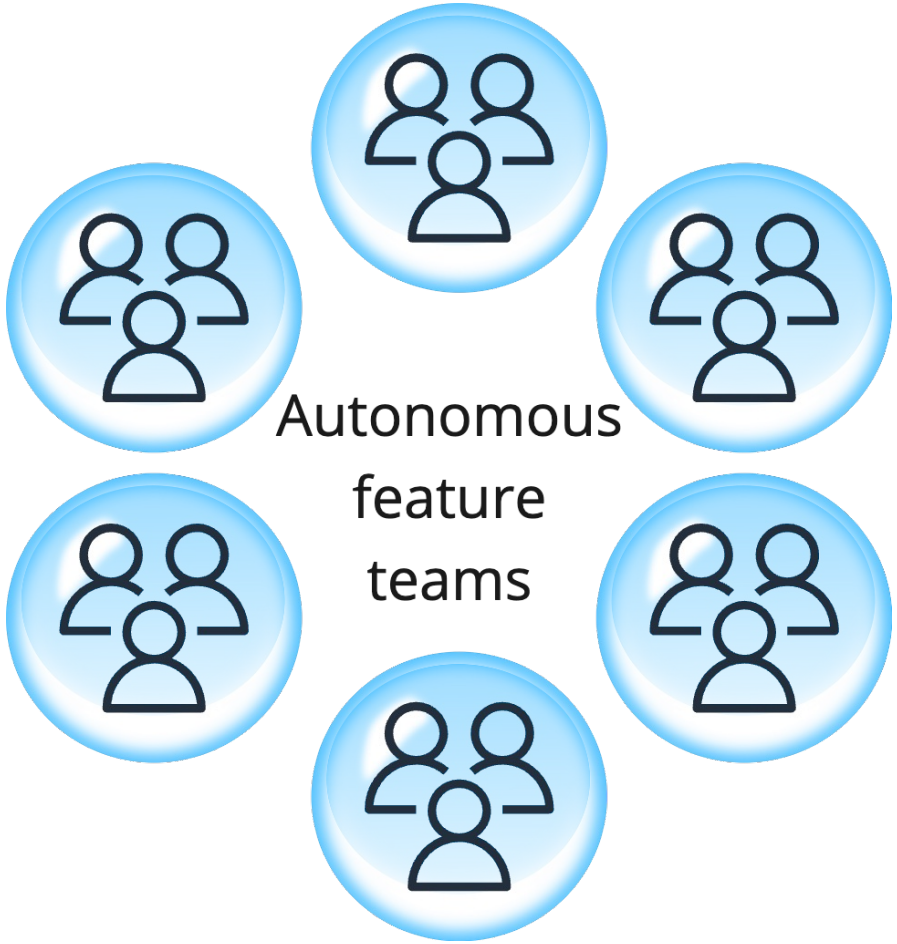
Playtika

# Comic Agilé




Playtika

# TEAM TOPOLOGY




# CONSIQUENCES

Each feature has a custom implementation of <...>

Extra efforts for supporting multiple implementations of <...>


Playtika

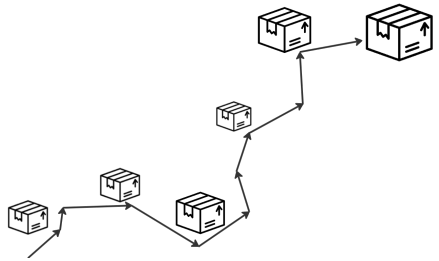
# INVERSE CONWAY MANEUVER

Architecture transformation team

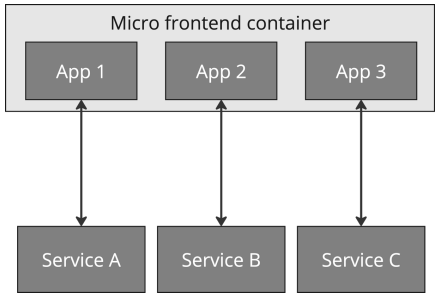
Autonomous feature teams

Evolving your team and organizational structure to promote your desired architecture

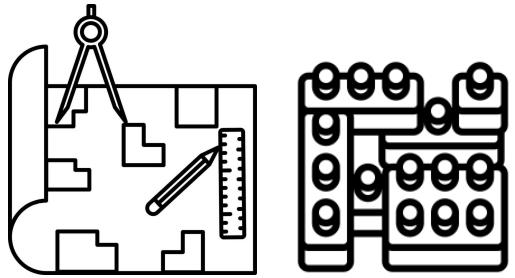
Playtika



QUICK PROTOTYPING
LESS RESTRICTIONS




UNLOCKING PARALLEL
DEVELOPMENT AND RELEASES



REDUCE COMPLEXITY
AND SCALE THE ORGANIZATION

**1.0**

**1.5**

**2.0**

MONOLITH

(NO BOUNDARIES)

APPLICATION AND CONTENT
RELEASE SEPARATION

MICRO
SERVICES

MICRO
FRONTENDS

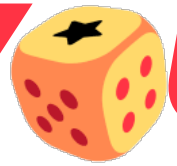
FEATURE
DESIGN
STANDARDS

FEATURE
DEVELOPMENT
TOOLKIT

INVERSE
CONWAY
MANEUVER

Playtika

# THANK YOU!



Playtika