Why you may prefer SDK-based server tech to GBaaS

Great Technology For Great Games

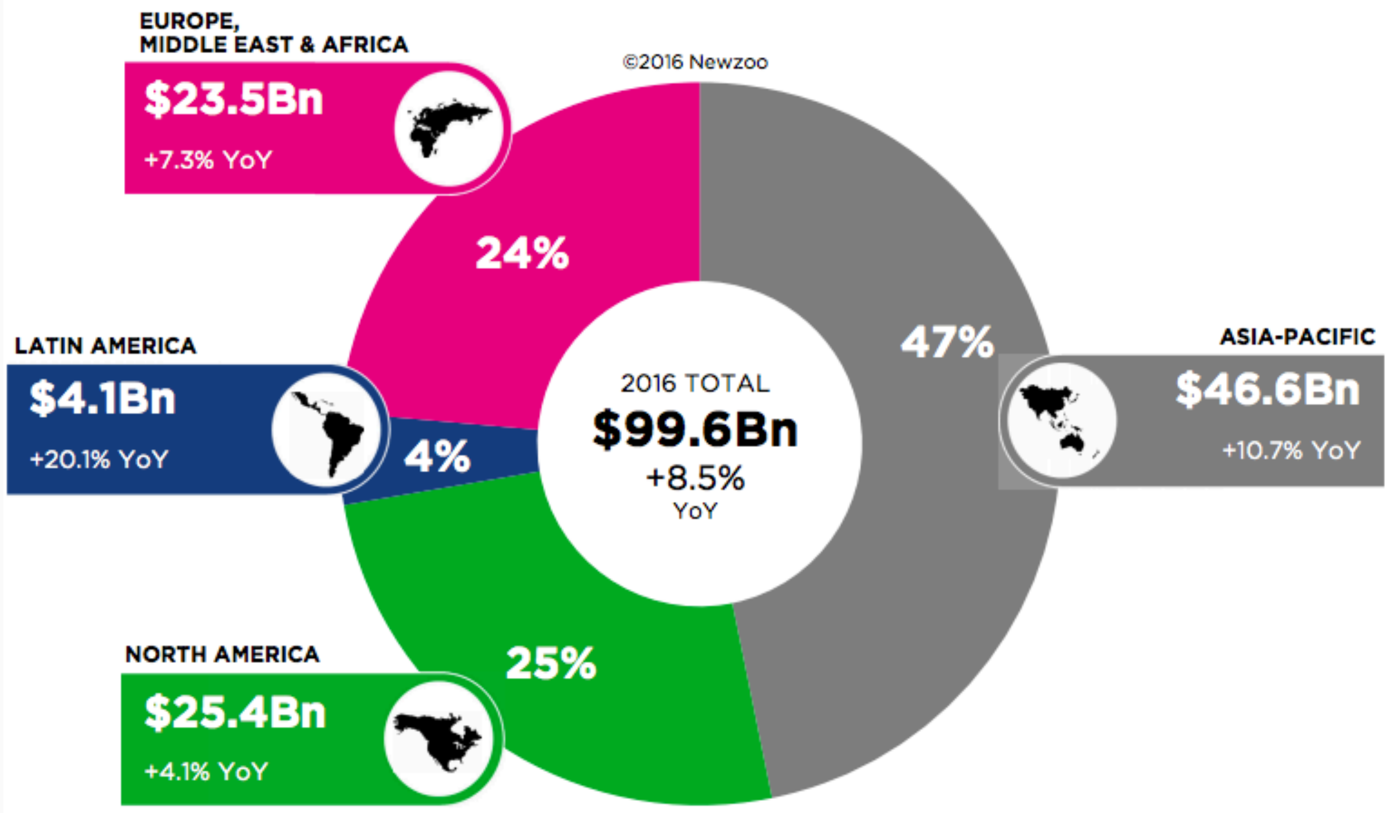
DK Moon

dkmoon@ifunfactory.com

Market Trend Glance

2016 GLOBAL GAMES MARKET

PER REGION WITH YEAR-ON-YEAR GROWTH RATES



Source: ©Newzoo | Global Games Market Report Premium
newzoo.com/globalreportpremium/

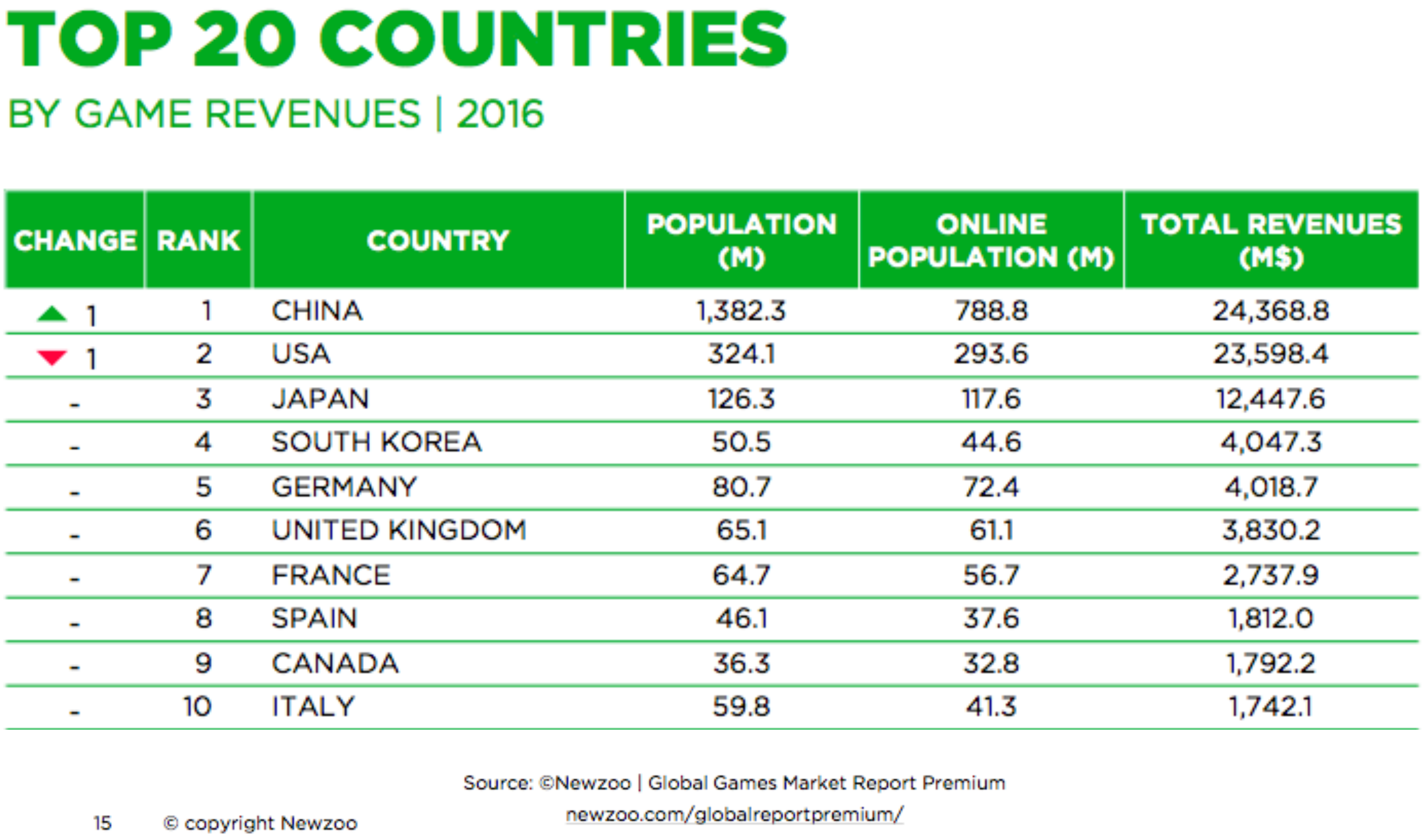
In 2016,

58%

of growth of the global games market comes from the Asia-Pacific region

newzoo GAMES

Market Trend Glance



TOP 20 COUNTRIES

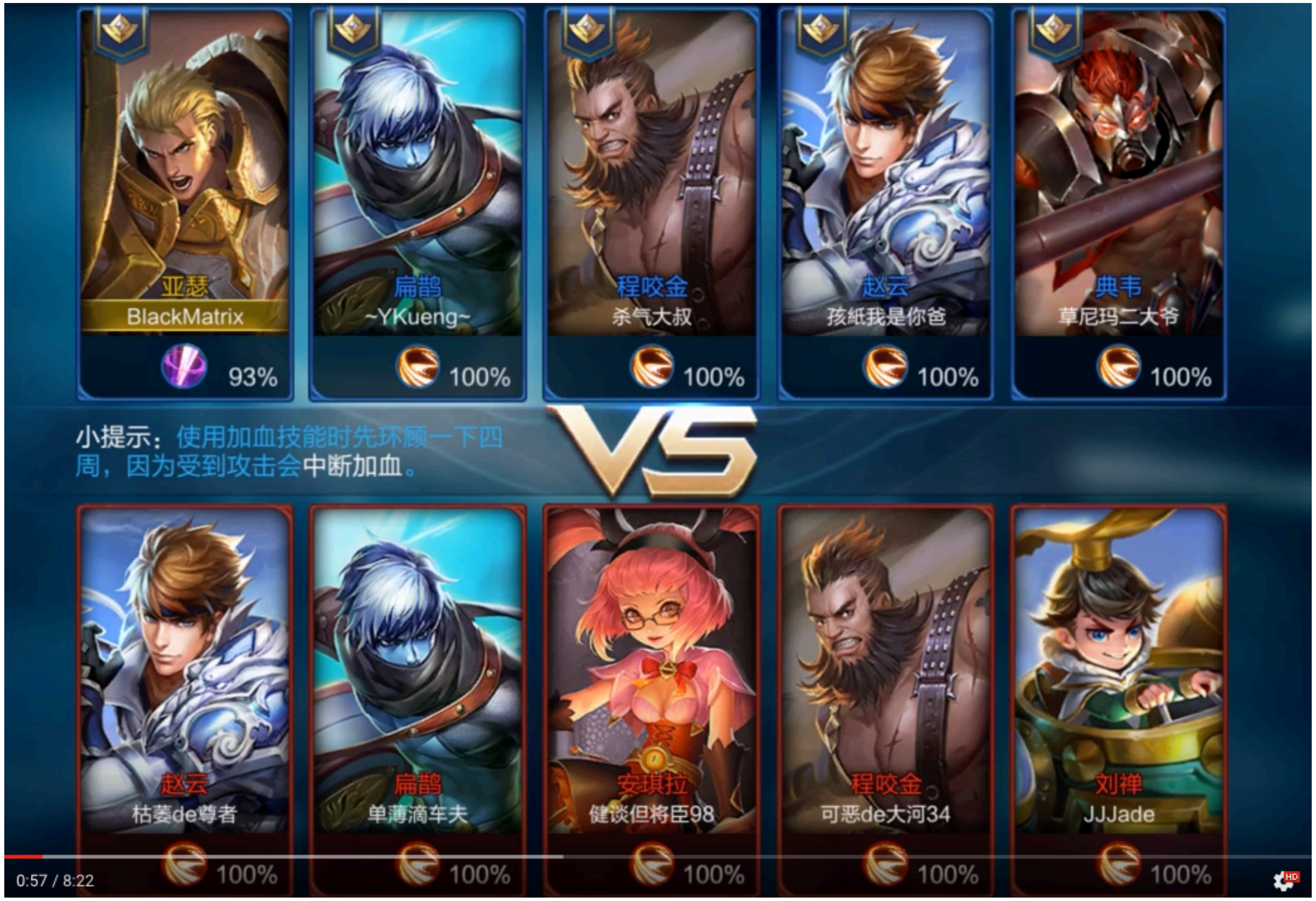
BY GAME REVENUES | 2016

CHANGE	RANK	COUNTRY	POPULATION (M)	ONLINE POPULATION (M)	TOTAL REVENUES (M$)
▲ 1	1	CHINA	1,382.3	788.8	24,368.8
▼ 1	2	USA	324.1	293.6	23,598.4
-	3	JAPAN	126.3	117.6	12,447.6
-	4	SOUTH KOREA	50.5	44.6	4,047.3
-	5	GERMANY	80.7	72.4	4,018.7
-	6	UNITED KINGDOM	65.1	61.1	3,830.2
-	7	FRANCE	64.7	56.7	2,737.9
-	8	SPAIN	46.1	37.6	1,812.0
-	9	CANADA	36.3	32.8	1,792.2
-	10	ITALY	59.8	41.3	1,742.1

Source: ©Newzoo | Global Games Market Report Premium
newzoo.com/globalreportpremium/

15

Chinese Top Game
King of Glory by Tencent



https://www.youtube.com/watch?v=PzKQuURLPxg

Korean Top Game
Lineage II Revolution by Netmarble

http://www.mmorpg.news/2016/10/lineage-ii-revolution-gameplay.html

Lessons from Market Trends

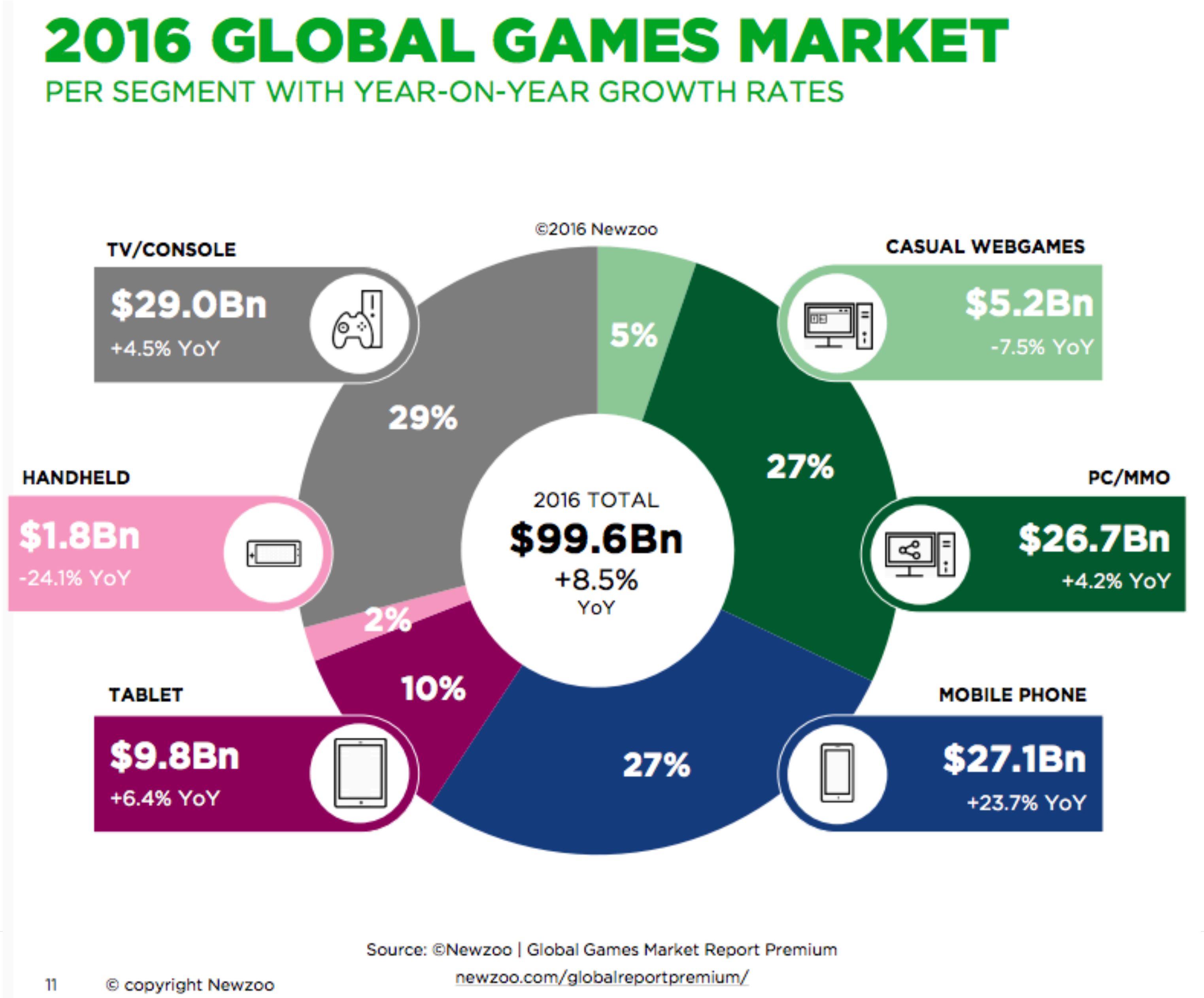
At least China, Japan, and Korea..

Real-time is now common.

Even MMO is becoming popular.

And the trends will spread to the western market, *eventually*.

Market Trend Glance



About This Talk

What covered:

Commonalities of GBaaS implementations & Differences in designs

GBaaS Pros & Cons

SDK-based Pros & Cons

What not covered:

Stability of each GBaaS implementation

Game Backend-as-a-Service (GBaaS)

Backend-as-a-Service (BaaS) for *gaming*

Game Backend-as-a-Service (GBaaS)

Backend-as-a-Service (BaaS) for *gaming*

That is…

> “Providing overall infrastructure (i.e. backend) for game services as a turn key solution, in the form of cloud service”

Game Backend-as-a-Service (GBaaS)

Backend-as-a-Service (BaaS) for *gaming*

That is…

> "Providing overall infrastructure (i.e. backend) for game services as a turn key solution, in the form of cloud service"

This includes…

1) **Physical components** like Server, DB, and Network, etc.
2) **Logical components** (programming primitives) for in-game system.
3) **Operational components** like game/player mgmt.

GBaaS Providers in the Market

✓ Photon by Exit Games (Hamburg, German, founded in 2003)

✓ GameSparks (Dublin, Ireland, founded in 2013)

✓ PlayFab (Seattle, USA, founded in 2014)

Photon Summary

- ✓ Client-Server model + TCP/UDP/HTTP/WS.
- ✓ Core implementation in C++.
- ✓ Replication of a backend instance across regions.
  The client chooses a regional gateway server to connect to.

Photon Summary

- ✓ Client-Server model + TCP/UDP/HTTP/WS.
- ✓ Core implementation in C++.
- ✓ Replication of a backend instance across regions.
  The client chooses a regional gateway server to connect to.

- ✓ Overall flow is designed around room (game session) :
  Room creation → Play → Room termination.
- ✓ So is API. Mostly about room management.

Photon Summary

- ✓ Client-Server model + TCP/UDP/HTTP/WS.
- ✓ Core implementation in C++.
- ✓ Replication of a backend instance across regions.
  The client chooses a regional gateway server to connect to.

- ✓ Overall flow is designed around room (game session) :
  Room creation → Play → Room termination.
- ✓ So is API. Mostly about room management.

- ✓ Limited dashboard features
- ✓ The client explicitly declares a version to match.

GameSparks Summary

- ✓ Node.js workers pool + MongoDB
- ✓ WSS for Async API + TCP/UDP for real-time API
- ✓ Replication of a backend instance across regions.
  The client connects to a gateway server for a primary region.

GameSparks Summary

- ✓ Node.js workers pool + MongoDB
- ✓ WSS for Async API + TCP/UDP for real-time API
- ✓ Replication of a backend instance across regions.
  The client connects to a gateway server for a primary region.

- ✓ Overall flow is designed around room (game session) :
  Room creation → Play → Room termination.
- ✓ Game session is initiated also by *Match* and *Challenge*.
- ✓ API for matching, ranking, teaming, achievements, virtual goods, etc.

GameSparks Summary

GAMESPARKS

- ✓ Node.js workers pool + MongoDB
- ✓ WSS for Async API + TCP/UDP for real-time API
- ✓ Replication of a backend instance across regions.
  The client connects to a gateway server for a primary region.

- ✓ Overall flow is designed around room (game session) :
  Room creation → Play → Room termination.
- ✓ Game session is initiated also by *Match* and *Challenge*.
- ✓ API for matching, ranking, teaming, achievements, virtual goods, etc.

- ✓ Various configuration, JS editing, & REST testing thru dashboard.
- ✓ Version mgmt by configuration snapshotting.

Photon vs. GS Commonalities

✓ Designed around game session (aka room)

- Game is thought of as a set of game sessions.
- API is designed to support game session mgmt.
- “Multiplay” is limited to the players in the same session.

Photon vs. GS Commonalities

✓ Designed around game session (aka room)

- Game is thought of as a set of game sessions.
- API is designed to support game session mgmt.
- “Multiplay” is limited to the players in the same session.

✓ Game contents implementation by basic API + logic extension

- Basic API is game logic agnostic.
- They provide a way to extending the API for game logic.

Photon vs. GS Commonalities

✓ Designed around game session (aka room)

- Game is thought of as a set of game sessions.
- API is designed to support game session mgmt.
- “Multiplay” is limited to the players in the same session.

✓ Game contents implementation by basic API + logic extension

- Basic API is game logic agnostic.
- They provide a way to extending the API for game logic.

✓ Automatically replicated across regions

- Global service comes for free.
- But can be problematic if data updates faster than replication.

Photon vs. GS Differences

✓ Network Topology: Photon Client-Server vs. GS full mesh

Photon vs. GS Differences

✓ Network Topology: Photon Client-Server vs. GS full mesh

✓ How custom logic is implemented

- By webhook: Photon allows invocation of external API on event.
- By code injection: GS allows developer to register JS and invoke it.

Photon vs. GS Differences

✓ Network Topology: Photon Client-Server vs. GS full mesh

✓ How custom logic is implemented

- By webhook: Photon allows invocation of external API on event.
- By code injection: GS allows developer to register JS and invoke it.

✓ Variety of logical components

- Photon: Limited to “room”.
- GS: matching, teaming, ranking, achievements, virtual economy, …

Photon vs. GS Differences

✓ Network Topology: Photon Client-Server vs. GS full mesh

✓ How custom logic is implemented

- By webhook: Photon allows invocation of external API on event.
- By code injection: GS allows developer to register JS and invoke it.

✓ Variety of logical components

- Photon: Limited to “room”.
- GS: matching, teaming, ranking, achievements, virtual economy, …

✓ Operational components

- Photon: Limited to CCU / Room monitoring.
- GS: logical components configuration/monitoring thru dashboard.

GBaaS Strength

✓ Very effective to single player games

- Single player tends to implement game logic on the client side.
- So, high level API for player mgmt is sufficient.

GBaaS Strength

✓ Very effective to single player games

- Single player tends to implement game logic on the client side.
- So, high level API for player mgmt is sufficient.

✓ Effective to session-based multiplayer games

- API is designed around game session, and so it well defines game session mgmt.

GBaaS Weakness

✓ Not for non session-based games

- Lack of API supporting open world style games.
  This is not “lack of implementation”, rather “design issue”.

GBaaS Weakness

✓ Not for non session-based games

- Lack of API supporting open world style games.
  This is not "lack of implementation", rather "design issue".

✓ Hard to add game logics

- Webhook requires a separate "state mgmt" by external service.
- It's hard to write a big code by uploading JS.
  In addition, it makes version control complicated.

GBaaS Weakness

✓ Not for non session-based games

- Lack of API supporting open world style games.
  This is not “lack of implementation”, rather “design issue”.

✓ Hard to add game logics

- Webhook requires a separate “state mgmt” by external service.
- It’s hard to write a big code by uploading JS.
  In addition, it makes version control complicated.

✓ Hard to support MMO

- Session-based API inevitably uses broadcasting, which doesn’t scale
- Scoped transmission depends on context and game logic.

SDK-based Approach Strength & Weakness

✓ Strength

- Easy to add game logic.
- Not limited to game genres and types.
- Easy to debug things.
- Thus, good for games with complex logics or MMO.

SDK-based Approach Strength & Weakness

✓ Strength

- Easy to add game logic.
- Not limited to game genres and types.
- Easy to debug things.
- Thus, good for games with complex logics or MMO.

✓ Weakness

- More work even for bootstrapping.

Conclusion

✓ GBaaS is super convenient for not worrying about hosting.

✓ GBaaS provides “simplified”, high-level logical components.

✓ Operational components should be concerned when choosing GBaaS.

Conclusion

✓ GBaaS is super convenient for not worrying about hosting.

✓ GBaaS provides “simplified”, high-level logical components.

✓ Operational components should be concerned when choosing GBaaS.

✓ GBaaS may suffer difficulties from simplified logical components

✓ From its pros/cons, GBaaS is suitable for single player, session-based multiplayer

Conclusion

✓ GBaaS is super convenient for not worrying about hosting.

✓ GBaaS provides “simplified”, high-level logical components.

✓ Operational components should be concerned when choosing GBaaS.

✓ GBaaS may suffer difficulties from simplified logical components

✓ From its pros/cons, GBaaS is suitable for single player, session-based multiplayer

✓ SDK-based approach may have more work to kick-start.

✓ But it has pros in terms of variety of game systems it can support.

✓ Also, easier to debug with less blackbox designs.

THANKS!

Great Technology For Great Games, iFunFactory

DK Moon

dkmoon@ifunfactory.com

www.ifunfactory.com