# Arquitectura Base de un MMO – Documento Fundacional

## Propósito del documento

Este documento define la **arquitectura técnica base**, los **principios de diseño** y el **roadmap por features** para el desarrollo de un MMO. Está pensado para: - Comenzar como proyecto personal - Servir como base clara para sumar colaboradores - Evitar errores clásicos de arquitectura en MMOs

No es un documento de marketing ni de tecnologías de moda. Es un **documento operativo**.

---

## 1. Principios no negociables

1. **Servidor autoritativo**
   El cliente no decide daño, posiciones, drops ni RNG.

2. **Estado caliente en memoria**
   El mundo vive en RAM. La base de datos es persistencia, no runtime.

3. **Hot path simple y local**
   Combate, movimiento e IA ocurren dentro del mismo proceso.

4. **Arquitectura híbrida**

5. Simulación: monolito lógico

6. Servicios de soporte: desacoplados

7. **Latencia > throughput**
   La cola larga de latencia es más peligrosa que el promedio.

---

## 2. Arquitectura General (Macro)

### Capas del sistema

- **Cliente**: render, input, predicción local
- **Edge / Login**: autenticación y sesiones
- **Game Server**: simulación autoritativa del mundo
- **Persistencia**: estado durable
- **Servicios auxiliares**: chat, analytics, admin

### Vista lógica

Cliente → Game Server → (DB / Cache / Event Stream)

---

# 3. Stack Tecnológico Elegido

### Hosting

- Game Server: Bare metal o VMs dedicadas
- Servicios secundarios: Kubernetes / Nomad

### Lenguajes

- Game Server: **Go**
- Servicios web: Go / Kotlin
- Cliente: C++ (Unreal Engine recomendado)

### Comunicación

- Cliente ↔ Game Server: UDP confiable custom (+ TCP fallback)
- Servicios internos: gRPC
- Eventos de dominio: Kafka / Pulsar

### Datos

- Base de datos principal: PostgreSQL
- Cache / KV: Redis
- Event stream: Kafka (fuera del hot path)

---

# 4. Diseño del Game Server

### Componentes internos

- World Manager (shards / zonas)
- Tick loop (100–250 ms)
- Combat Engine
- AI Engine (FSM / Behavior Trees)
- Drop Engine
- Network Layer

### Concurrencia

- Un goroutine por zona
- Mensajes serializados
- Sin locks internos

---

# 5. Arquitecturas por Feature

### 5.1 Creación de personaje

- Servicio separado
- HTTP / gRPC
- Validación y persistencia

- El personaje se carga en memoria solo al login al mundo

---

### 5.2 PvE / PvP / Raids

- Todo ocurre dentro del Game Server
- Misma lógica base para PvE y PvP
- Raids = entidades complejas
- Kafka solo emite eventos finales (BossKilled, PlayerDied)

---

### 5.3 Chat y mensajería

- Servicio separado
- WebSocket
- Redis Pub/Sub
- Tipos: global, party, clan, whisper

---

### 5.4 Drops y loot

- RNG server-side
- Drop decidido al morir la entidad
- Persistencia solo al recoger

---

### 5.5 Eventos del mundo

- Scheduler interno
- Boss spawns
- Eventos temporales
- Persistencia mínima

---

### 5.6 Sistemas críticos adicionales

- Anti-cheat server-side
- Analytics async
- Recovery por snapshots
- Economía y auditoría de trades

---

## 6. Roadmap por Entregas (Feature-based)

### FASE 0 – Fundaciones

- Conexión cliente-servidor
- Movimiento sincronizado
- Mundo vacío

### FASE 1 – Mundo persistente mínimo

- Spawn
- Reconexión
- Zona única

### FASE 2 – Combate básico

- Mobs
- Targeting
- Auto-attack
- HP y muerte

### FASE 3 – Escala y estrés

- Concurrencia
- Profiling
- Lag simulation

### FASE 4 – Skills y estados

- Cooldowns
- Buffs / debuffs
- Casting

### FASE 5 – Drops e inventario

### FASE 6 – Social mínimo

- Chat
- Party
- Trade

### FASE 7 – Mundo vivo

- Respawns
- Bosses
- Eventos

### FASE 8 – PvP

- Reglas PK
- Penalidades

### FASE 9 – Hardening

- Persistencia optimizada
- Recovery
- Monitoring

---

## 7. Qué NO hacer

- Microservicios para combate
- Kafka en tiempo real
- DB como estado vivo
- Kubernetes para simulación
- Eventual consistency en gameplay

---

## 8. Uso del documento

Este documento es: - La referencia arquitectónica principal - La base para onboarding técnico - El punto de partida para decisiones futuras

Debe evolucionar, pero **los principios no cambian**.