



**ESCUELA POLITECNICA NACIONAL**  
**DESARROLLO DE JUEGOS ITERATIVOS**  
**PROYECTO PRIMER BIMESTRE**

**Nombre:** Tatiana Gualpa

**TEMA:** El GDD: De Requerimientos a un Proyecto Gestionable

**OBJETIVOS**

- **Objetivo General:**

Analizar un Documento de Diseño de Juego real para transformar una visión creativa en un plan de desarrollo de software técnico, estructurado y gestionable mediante metodologías Ágiles.

- **Objetivos Específicos:**

1. Realizar una "arqueología de diseño" sobre el GDD de Diablo para identificar los componentes arquitectónicos clave del software.
2. Traducir la narrativa de diseño en artefactos de ingeniería (Épicas e Historias de Usuario) con criterios de aceptación claros.
3. Definir un Producto Mínimo Viable (MVP) priorizando las funcionalidades esenciales para validar el núcleo del juego en una etapa temprana (Sprint 0).

**DESARROLLO**

**FASE 1: ARQUEOLOGÍA DE DISEÑO (Investigación del GDD)**

**1. Juego y Fuente**

- **Juego:** *Diablo* (Pitch Document Original - 1994)
- **Desarrolladora:** Condor Inc. (luego Blizzard North)

**2. Visión Central**

El documento propone un RPG de fantasía oscura isométrica que rompe con la complejidad de los juegos de rol de la época (1994). La visión es crear una experiencia visceral de "hack-and-slash" donde la acción es inmediata y la rejugabilidad es infinita. **Elevator Pitch:** "Un juego de rol donde el jugador desciende a una mazmorra generada aleatoriamente, llena de monstruos y tesoros, con una interfaz simplificada basada en el ratón, enfocándose en la acción rápida y la gratificación constante del botín (loot)."

### 3. Resumen Estructural (Puntos Clave del Documento)

El GDD es conciso (8 páginas) pero define la columna vertebral del género ARPG:

1. **Premisa y Mercado:** Identifica un vacío en el mercado para juegos RPG que no requieran horas de estudio de manuales. Define el público objetivo como jugadores que buscan gratificación inmediata.
2. **Estructura de Niveles (Procedural):** Describe el sistema fundamental de *Random Level Generation*. Explica que ningún nivel será igual dos veces, detallando cómo se ensamblan habitaciones, pasillos y trampas mediante algoritmos para mantener la frescura.
3. **Sistema de Clases y Progresión:** Define los arquetipos (Guerrero, Ladrón, Hechicero) no solo por estética, sino por estadísticas numéricas y árboles de habilidades, estableciendo la base de la lógica matemática del juego.
4. **Interfaz y Control:** Propone una revolución en la UX (Experiencia de Usuario) eliminando la necesidad de escribir comandos de texto, reemplazándolos por un sistema "Point-and-Click" para movimiento, combate e inventario.
5. **Catálogo de Entidades:** Lista jerarquías de monstruos (desde esqueletos básicos hasta demonios mayores) y define la economía de objetos (armas mágicas generadas aleatoriamente).

### 4. Análisis de Ingeniería

Desde la perspectiva de la Ingeniería de Software:

- **Sección más clara y útil: La Lógica de Generación Procedural.** El documento especifica claramente que el mapa no es estático. Esto dicta la arquitectura del backend: necesitamos un motor de "tiles" (baldosas) y un algoritmo (posiblemente basado en grafos o autómatas celulares) que valide la conectividad entre habitaciones antes de cargar el nivel.
- **Sección ambigua/Faltante: El Modelo de Tiempo y Concurrencia.** El documento original proponía que el juego fuera *basado en turnos* (el tiempo avanza solo cuando el jugador actúa). Sin embargo, la visión de "acción rápida" choca con esto. Desde ingeniería, falta una definición clara del

"Game Loop" (bucle de juego): ¿Cómo se gestionan las colisiones y las interacciones de la IA si el juego pasa a tiempo real? También carece de requerimientos no funcionales sobre latencia de red para el modo multijugador sugerido.

## **FASE 2: TRADUCCIÓN A HISTORIAS (Desglose de Requerimientos)**

He desglosado la visión en **13 Épicas Técnicas**, incluyendo notas sobre lo que implican a nivel de código para demostrar profundidad.

**Formato:** Como [ROL], quiero [ACCIÓN], para [BENEFICIO].

1. **Épica (Core - Generación de Mapas):** "Como Sistema, quiero generar la distribución de la mazmorra (paredes, pisos) mediante un algoritmo aleatorio al cargar un nivel, para asegurar que la topografía sea única en cada partida."
  - *Nota técnica:* Requiere algoritmo de validación de caminos (Pathfinding check) para evitar habitaciones inaccesibles.
2. **Épica (Core - Movimiento):** "Como Jugador, quiero desplazar a mi personaje a la coordenada (x,y) haciendo clic izquierdo en el suelo, para navegar por el entorno isométrico."
  - *Nota técnica:* Implementación del algoritmo A\* (A-Star) para evitar obstáculos dinámicos.
3. **Épica (Combate - Hitbox):** "Como Jugador, quiero atacar a un enemigo haciendo clic sobre su sprite, para iniciar una tirada de daño basada en mis estadísticas."
  - *Nota técnica:* Detección de colisiones y cálculo de RNG (Random Number Generator) vs. Defensa del enemigo.
4. **Épica (Datos - Stats del Jugador):** "Como Sistema, quiero gestionar los atributos base (Fuerza, Magia, Destreza, Vitalidad) y recalcularlos al subir de nivel, para determinar la eficacia del personaje en combate."
5. **Épica (Inventario - Grid System):** "Como Jugador, quiero una interfaz de inventario basada en cuadrícula (grid) donde los objetos ocupan diferente espacio (1x1, 1x3, 2x2), para gestionar estratégicamente qué equipo llevar."
  - *Nota técnica:* Lógica de arrays bidimensionales para gestión de espacio disponible.
6. **Épica (Economía - Loot Generation):** "Como Sistema, quiero generar objetos con propiedades aleatorias (sufijos y prefijos mágicos) al matar un enemigo, para recompensar al jugador con equipo único."

- *Nota técnica:* Tablas de probabilidad (Loot Tables) y base de datos de modificadores de objetos.
- 7. **Épica (IA - Comportamiento Agresivo):** "Como IA de Enemigo, quiero calcular la distancia al jugador y trazar una ruta hacia él cuando entre en mi radio de visión, para atacarlo cuerpo a cuerpo."
- 8. **Épica (Visual - FoV):** "Como Sistema, quiero oscurecer las áreas del mapa que no están en la línea de visión directa del jugador (Fog of War), para crear tensión y atmósfera de exploración."
  - *Nota técnica:* Algoritmo de Raycasting o Shadowcasting en tiempo real.
- 9. **Épica (UI - Feedback Vital):** "Como Jugador, quiero ver representaciones visuales claras de mi Salud y Maná (esferas rojas y azules), para monitorear mi estado sin leer números constantemente."
- 10. **Épica (Magia - Sistema de proyectiles):** "Como Jugador, quiero lanzar hechizos (ej. Bola de Fuego) que viajen en línea recta y exploten al impacto, para dañar enemigos a distancia."
- 11. **Épica (Persistencia - Save/Load):** "Como Jugador, quiero guardar el estado exacto de mi personaje e inventario en el disco local, para poder retomar la progresión en otra sesión."
- 12. **Épica (Interacción - NPCs):** "Como Jugador, quiero interactuar con personajes no jugables en el pueblo (Tristram) para comprar/vender objetos y reparar equipo dañado."
- 13. **Épica (Loop - Muerte):** "Como Sistema, quiero reiniciar el nivel y dejar el cuerpo del personaje con su equipo en el lugar de muerte, para penalizar el fallo y motivar la recuperación del cadáver."

### **FASE 3: MAPA DE PROYECTO (Estimación y Priorización)**

Para gestionar este alcance, aplicamos un enfoque incremental. No podemos hacer un sistema de magia complejo si el personaje no puede caminar.

### **FASE 1: PROTOTIPO "WALKING SKELETON" (EL MVP)**

*Objetivo:* Validar la viabilidad técnica del motor isométrico y el ciclo básico de interacción. Si esto falla, el proyecto se cancela.

- **Épica 1: Core - Generación de Mapas (Versión simplificada):** Crear un piso plano con paredes perimetrales.

- **Épica 2: Core - Movimiento:** Implementar el clic-para-mover y la cámara isométrica.
- **Épica 3: Combate - Hitbox (Básico):** Poder hacer clic en un cubo estático (enemigo dummy) y que este desaparezca (muera).
- **Épica 9: UI - Feedback Vital:** Mostrar la barra de vida bajando si el enemigo "toca" al jugador.
- **Épica 13: Loop - Muerte:** Pantalla de "Game Over" simple.

**Resultado:** Un "juego" donde caminas, matas un cubo y puedes morir. Feo, pero funcional.

## **FASE 2: ALPHA JUGABLE (Features Centrales y RPG)**

*Objetivo:* Implementar la lógica matemática ("Math-heavy phase") y la persistencia. Aquí es donde el juego se convierte en un RPG.

- **Épica 4: Datos - Stats del Jugador:** Implementar las fórmulas de daño reales (Fuerza \* Arma).
- **Épica 7: IA - Comportamiento Agresivo:** Los enemigos ya no son cubos estáticos; persiguen al jugador (Pathfinding).
- **Épica 1: Generación de Mapas (Completa):** Algoritmo real de mazmorras con habitaciones y pasillos.
- **Épica 5 y 6: Inventario y Loot:** Los enemigos sueltan items y puedes equiparlos para subir tus stats.
- **Épica 11: Persistencia:** Poder cerrar el juego y no perder el nivel del personaje.

**Resultado:** Una experiencia de juego completa ("Vertical Slice"). Se puede jugar de principio a fin, pero le falta pulido.

## **FASE 3: BETA Y PULIDO (Nice-to-Have y Atmósfera)**

*Objetivo:* "Juice" (Sensación de juego), gráficos, sonido y características complejas no bloqueantes.

- **Épica 8: Visual - FoV (Iluminación):** Implementar las sombras dinámicas (crítico para la atmósfera de Diablo).
- **Épica 10: Magia - Sistema de proyectiles:** Añadir la complejidad de hechizos y partículas visuales.
- **Épica 12: Interacción - NPCs:** Añadir el pueblo, tiendas y diálogos.

- **Audio:** Efectos de sonido (pasos, espadaos, gritos de monstruos) y música ambiental.
- **Multijugador:** (Si el tiempo lo permite) Sincronización de dos clientes en una misma partida.

**Resultado:** El producto final listo para lanzamiento (Gold Master).

## CONCLUSIONES

El análisis del GDD de *Diablo* demuestra que una visión ambiciosa debe ser "troceada". Intentar programar la generación procedural compleja, la IA avanzada y el sistema de inventario simultáneamente conduciría al fracaso del proyecto. La definición del MVP en la Fase 3 es crítica para el éxito.

El ejercicio evidenció cómo párrafos descriptivos ("una mazmorra oscura y cambiante") deben traducirse en requerimientos técnicos rígidos ("Algoritmo de generación de tiles con validación de caminos"). Esta traducción es la responsabilidad principal del ingeniero de requisitos.

Al analizar el documento bajo la lupa de ingeniería, se detectaron riesgos técnicos no explícitos en el diseño, específicamente el cambio de "Turn-based" a "Real-time", lo cual impacta drásticamente la arquitectura del motor de juego y la gestión de eventos.

## Bibliografía

Brevik, D. (1994). *Diablo Pitch Document*. Condor Inc. Recuperado de: [diablo\\_pitch.pdf](#)