**Informe de Decisiones de Diseño**

**Fecha:** 13/04/2025  
**Integrantes:**

* Lukas Flores
* David Vasquez

**Introducción**

El presente informe tiene como objetivo describir el proceso de diseño y las decisiones tomadas en el desarrollo de un sistema de evaluación implementado en C++. Este sistema permite gestionar preguntas de evaluación académica mediante operaciones como agregar, actualizar, eliminar, buscar por nivel taxonómico, calcular el tiempo total estimado de resolución y generar evaluaciones automáticas.

La motivación principal de este proyecto es proporcionar una herramienta funcional que facilite la creación y organización de bancos de preguntas, permitiendo su reutilización y control con base en criterios pedagógicos como los niveles taxonómicos de Bloom. El cuerpo del documento se centrará en detallar la estructura del sistema, describir cada componente y justificar las decisiones clave del diseño.

**Descripción de la Solución**

El sistema se compone de tres clases principales: Pregunta, Evaluacion y SistemaEvaluacion

**Modelo y Objetivo de cada Elemento**

— **Clase Pregunta** Representa una unidad básica de evaluación. Contiene atributos como ID, enunciado, nivel taxonómico, tipo de pregunta, tiempo estimado y año.  
 *Objetivo:* encapsular toda la información relevante de una pregunta individual, permitiendo su validación y manipulación.

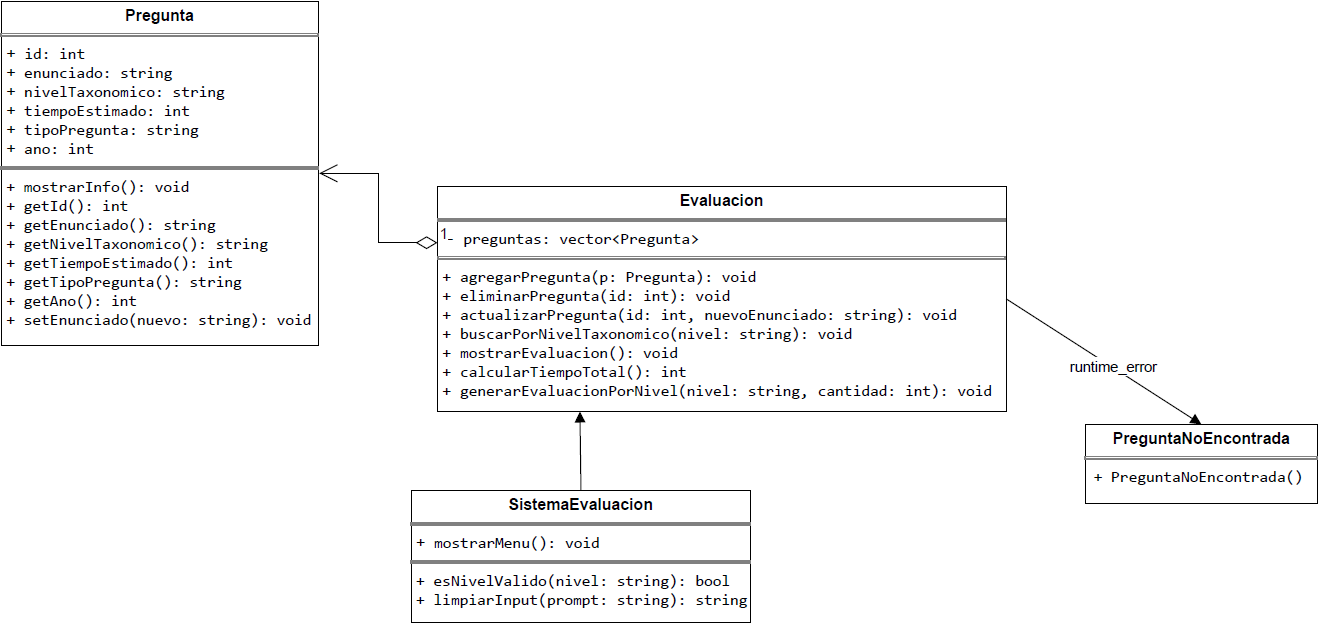
— **Clase Evaluacion** Administra un conjunto de preguntas. Permite agregar, eliminar y actualizar preguntas, así como generar evaluaciones filtradas por nivel taxonómico y calcular el tiempo total.  
 *Objetivo:* actuar como núcleo lógico del sistema que gestiona la colección de preguntas bajo criterios de calidad y consistencia.

— **Clase SistemaEvaluacion** Controla la interacción con el usuario a través de un menú, validando entradas y guiando al usuario por las funcionalidades del sistema.  
 *Objetivo:* proporcionar una interfaz textual clara para facilitar el uso del sistema y reducir errores de entrada.

**Decisiones de Diseño**

1. **Validaciones Preventivas** — **Se validan los niveles taxonómicos contra una lista cerrada para evitar errores semánticos. Además, se impide el ingreso de preguntas repetidas por ID o enunciado dentro de un margen de ±1 año. Estas decisiones aseguran la calidad y unicidad del banco de preguntas.**
2. **Uso de Excepciones** — **Se implementó una excepción personalizada PreguntaNoEncontrada, y se utilizan excepciones estándar como invalid\_argument para gestionar errores de manera controlada, mejorando la robustez del sistema.**
3. **Separación de Responsabilidades** — **Cada clase tiene una responsabilidad clara (modelo, lógica de evaluación, interfaz), siguiendo el principio de diseño de *Single Responsibility*, facilitando el mantenimiento y futuras extensiones.**
4. **Menú Interactivo** — **Se decidió implementar un menú interactivo para manejar múltiples operaciones sin reiniciar el programa. Esto mejora la experiencia de usuario y permite realizar múltiples tareas en una sola ejecución.**
5. **Modularidad y Escalabilidad** — **La estructura del código permite agregar nuevas funcionalidades, como exportación de evaluaciones o categorización más compleja, sin afectar el diseño base.**

**Diagrama de Clases** — **UML**

****

**Conclusión**

El desarrollo del sistema permitió cumplir con los objetivos planteados: gestionar eficazmente un banco de preguntas educativas, validando su integridad y facilitando la generación automatizada de evaluaciones. Las decisiones de diseño, centradas en la claridad, la validación y la modularidad, resultaron fundamentales para alcanzar un sistema robusto, escalable y fácil de usar.

Como reflexión final, el trabajo evidencia la importancia de aplicar principios sólidos de diseño orientado a objetos en la solución de problemas educativos, favoreciendo tanto el desarrollo técnico como la utilidad práctica del sistema.