Lic./Ing. en Sistemas – Universidad CAECE – Año 2021 2do.C
Algoritmos y Estructuras de Datos II

Trabajo Práctico Grupal. Desarrollo en JAVA

Enunciado del problema

Una empresa de desarrollo de software pretende un sistema para la **gestión** de sus **proyectos**.

Se informa que:

- Un proyecto se puede basar en uno o más requerimientos, los cuales pueden ser funcionales o no funcionales. Los requerimientos funcionales poseen un tiempo estimado (en hs.). La duración estimada del proyecto está dada por la suma de las estimaciones de los requerimientos a cumplir, más una estimación adicional propia. Un requerimiento es exclusivo de cada proyecto, no puede ser atendido por más de un proyecto.
- Para cada proyecto se estima un presupuesto en pesos, discriminado por área: análisis, desarrollo y testeo.
- Los requerimientos no funcionales (RNF) tienen un costo fijo que es asignado al área de desarrollo. Un RNF puede estar compuesto por uno o más subrequerimientos no funcionales, que poseen las mismas características que los RNF; en tal caso el costo del RNF surge de la suma de los subrequerimientos que lo componen.
- Cada proyecto tiene asignado un equipo de trabajo, con integrantes que tienen diferentes roles: 1 líder y varios analistas, desarrolladores y testers. Un integrante puede participar en diferentes proyectos. Cada rol tiene atributos propios que lo diferencian (ej. lenguajes en los que participa el desarrollador, tipo de analista (funcional o no funcional), tester dominio ó interfaz usuario, etc)
- Para cada integrante se registran semanalmente la cantidad de horas trabajadas en cada proyecto.
- El valor de la hora trabajada de cada integrante depende del rol que cumple, la antigüedad, y características propias de cada rol: desarrolladores juniors y seniors, analistas funcionales y no funcionales, etc.
    - Cada rol tiene como atributo el valor de la hora.
    - Para todos los roles, la antigüedad incrementa el valor de la hora en un 0.5% por año.
    - Para los desarrolladores, se agrega por hora un valor de acuerdo a la categoría, además de un 1% del valor hora por cada lenguaje que domina.
    - Para los testers, se agrega un 2% al valor hora por cada alcance testeado.
    - Para los analistas no funcionales se agrega un 5% al valor hora del rol.
- El costo del líder se prorratea en proporciones iguales en las 3 áreas.

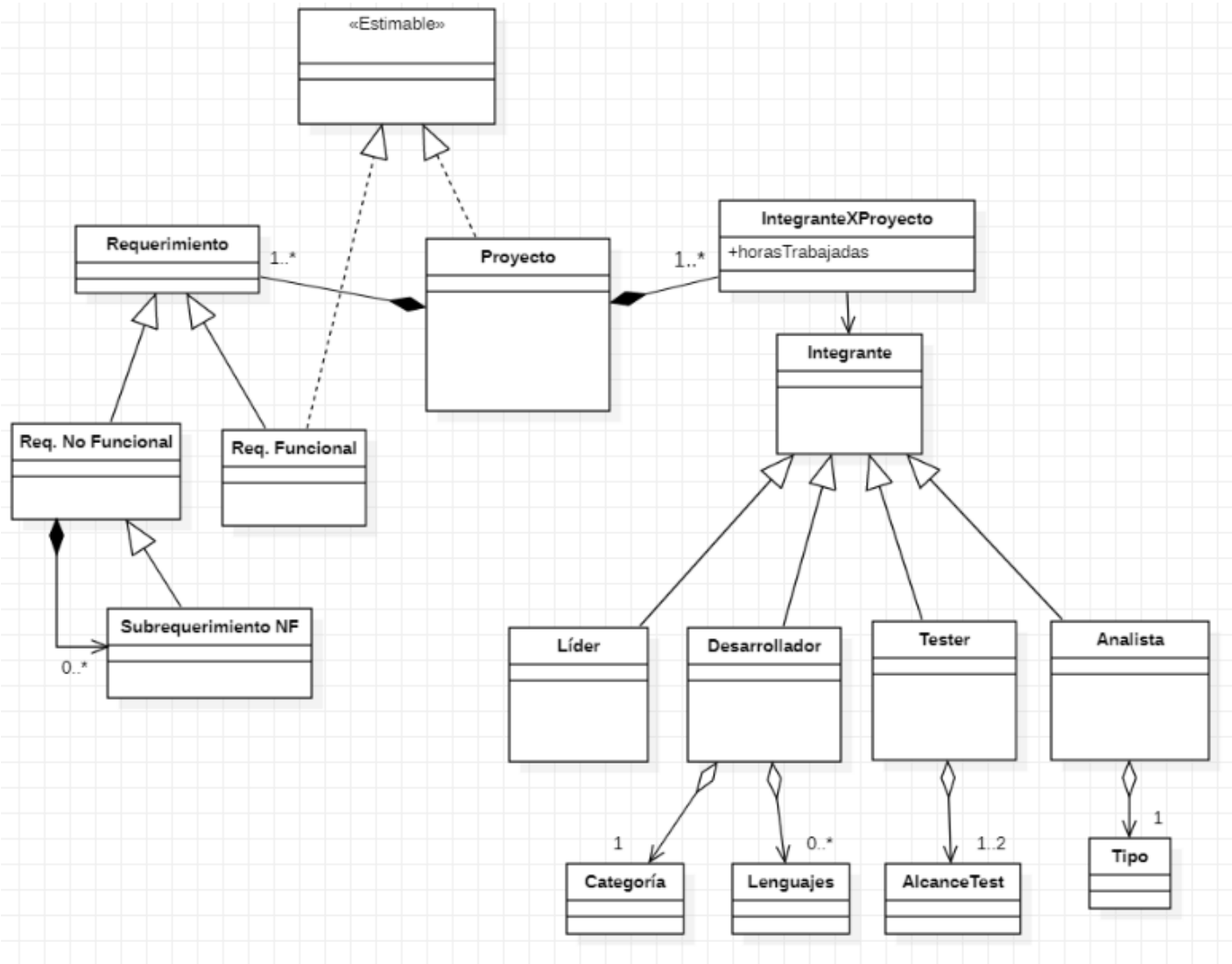
Requerimientos del Sistema

- Registre la carga inicial de cada proyecto, con los requerimientos funcionales y no funcionales que tiene asociado, estableciendo fechas de inicio y finalización, y las estimaciones en tiempo y pesos correspondientes. Esta carga inicial podrá ser realizada desde archivos XML, JSON, o de texto.
- Asigne y quite integrantes a un proyecto en los diferentes roles.
    - Quitar un integrante a un proyecto, implica mantener las horas trabajadas en el mismo previamente.
- Registre las horas trabajadas por un integrante en los diferentes proyectos en los que esté asignado. Para esta funcionalidad, se puede optar por implementarla de manera interactiva o a partir de archivos (XML, JSON, txt)
- Permita la consulta de un proyecto, mostrando su estado actual, el equipo de trabajo asignado, y la comparación entre el costo por área presupuestado y el real, mostrando los desvíos en porcentajes.

- Genere los siguientes reportes (en archivos de texto):
    - Ranking de proyectos. Un reporte ordenado por costo monetario presupuestado descendente, y otro por porcentaje de desvío entre lo presupuestado y lo real.
    - Detalle mensual de las horas trabajadas por los integrantes de un rol determinado.
- Realice las validaciones necesarias para que la información sea consistente (ej. completar datos obligatorios del proyecto, rango de fechas válidas para el proyecto, tiempos e importes positivos, etc.)

Sugerencias y comentarios

- Diagrama de clases sugerido:



- La **interfaz de usuario** puede ser elegida por el grupo (caracter, gràfica (AWT y Swing), web (html + Servlets ó JSP).
- La **persistencia** se implementará mediante serialización (clásica o XML)

- Considerar:

- una clase GestorSoft que centralice la operatoria solicitada y contenga colecciones de proyectos, integrantes, etc.
- el uso de las clases ***Contenedoras*** provistas por Java para administrar listas y conjuntos.
- el uso de clases específicas para el manejo de los reportes.
- el uso de *Enum* para los valores discretos.
- aplicar el lanzamiento de **excepciones** en las validaciones de las clases del dominio, con el objetivo de desacoplarla de la Interfaz de Usuario.
- el uso de la **interface** *Estimable* para objetos que impliquen implementar estimaciones de tiempos.

Condiciones de Aprobación Trabajo Práctico Java

- Conformar un **grupo** de 3 personas. Según el número de alumnos del curso se permitirán 2 grupos de 2 alumnos o un grupo de 4 alumnos.
- Implementar la totalidad de la **funcionalidad** solicitada en el enunciado del problema.
- Aplicar indefectiblemente en la solución los siguientes conceptos de la Programación orientada a Objetos: **encapsulamiento**, **polimorfismo**, **herencia**, **clases abstractas**.
- Cumplir estrictamente con el **cronograma de entregas** (parcial y final) según el siguiente detalle:

	Fecha
Entrega Parcial	05/10/2021
Entrega Final – 1ra. Fecha	26/10/2021
Entrega Final – 2da. Fecha	15/11/2021

- En la Entrega Final:
  - presentar copia digital (por mail, tarea aula virtual, etc.) de los **archivos fuentes** (.java) y la **documentación** en **HTML** generada automáticamente con *javadoc.* Todos los grupos deberán hacer la entrega al comienzo de la clase, no permitiéndose modificaciones de último momento. El grupo que no lo cumpla no podrá presentar el TP en esa fecha.
  - La cátedra podrá proponer un **lote de datos** para la prueba del sistema.
  - Deberán estar presentes **todos** los integrantes del grupo pues se hará una primera evaluación oral del trabajo presentado. El integrante que no lo estuviera se considerará **fuera del grupo** y deberá realizar su propio TP.
  - La **nota** del trabajo práctico es **individual**, basada en la participación en la resolución y defensa del trabajo práctico, y en los conocimientos conceptuales exhibidos en las entregas.
- Otros **conceptos** que incidirán en la **aprobación** del trabajo práctico son:
  - Reutilización adecuada del código.
  - Eficiencia en los algoritmos (ej: búsquedas, ordenamientos)
  - Bajo acoplamiento entre interfaz y lógica de dominio
  - Empaquetamiento criterioso de las clases.
  - Utilización de operadores, métodos y técnicas propias del lenguaje Java (ej: manejo de errores con excepciones propias, métodos *compareTo*, *toString*)
  - Validaciones de ingresos de datos y consistencia de la información.
  - Código prolijo, claro y correctamente comentado. (ej: nombres representativos, crear variables e instancias necesarias, sobrecargar métodos)