Escuela de Ciencias de la Computación e Informática

Programación 2

Profesor: Javier Vásquez

Grupo 2

Tarea Programada 3

Dillian Badilla Mora – B50800
dilliann1995@hotmail.com

José Alberto Soto Li – B46912
josesotli795@gmail.com

Segundo semestre

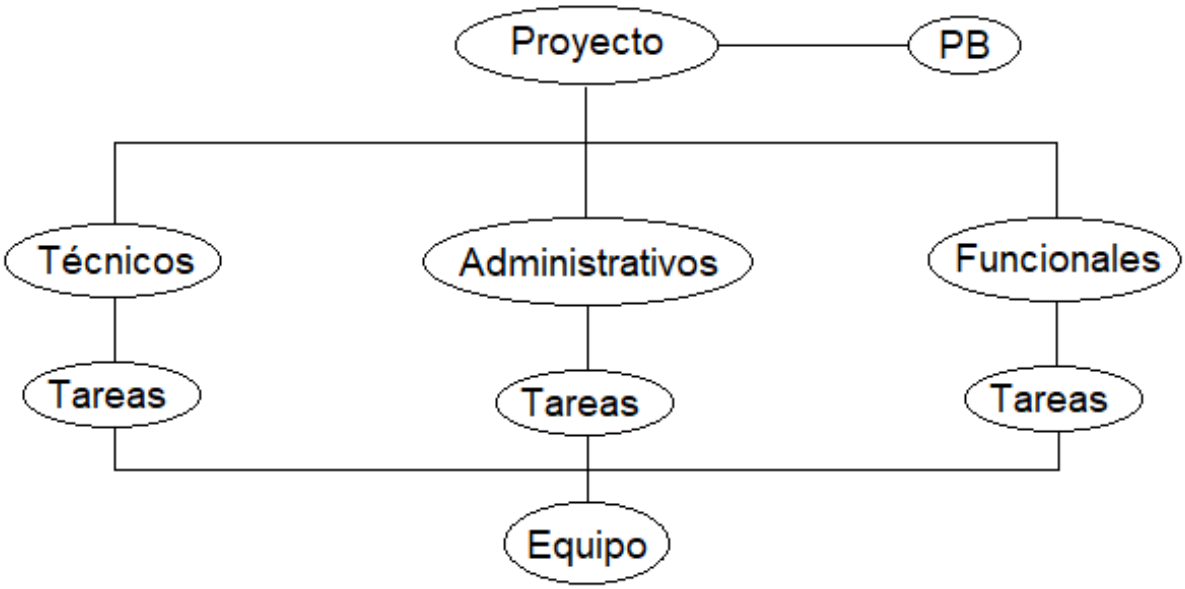
29/10/2017

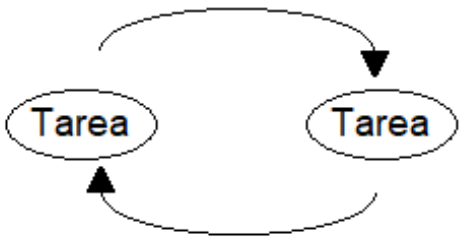
Análisis de implementación:

La limitación más considerable es que el programa se muestra y toma entrada desde la consola y no mediante un interfaz más amigable.

La interfaz le permite al usuario la creación de tareas según su requerimiento (nombre, prioridad, esfuerzo e impacto), además permite la asignación de estas a un integrante del equipo al igual que la relación entre estas.

En el enunciado no se exige la implementación de métodos de modificación ni de eliminación, por lo que si se da un error de entrada y se quiere hacer un cambio se tendría que correr de nuevo el programa.





Para resolver el problema dividimos el programa en seis clases:

Interfaz: define como va a ser la interacción con el usuario. De aquí se toman las entradas y se envían a los diferentes constructores y métodos para la manipulación de la información (menú).

Persona: se utiliza una plantilla para manipular las instancias de esta clase como strings. Estas instancias son a las cuales se les asignan tareas.

Equipo: define al equipo de personas, contiene un vector de objetos Persona (nombre), a los cuales se les asignaran las tareas. Su funcionalidad se divide en los siguientes métodos:

- *agregarPersona(Persona<string>)*: agrega elementos al vector de personas.
- *mostrarEquipo()*: despliega el vector de personas con sus respectivas tareas.

Tarea: define los atributos de las tareas(nombre, prioridad, esfuerzo, impacto, etc...). Su funcionalidad se divide en los siguientes métodos:

- *agregarRelaciones(Tarea)*: guarda la relación entre las tareas elegidas por el usuario.
- *eliminar_dup()*: es un método auxiliar que elimina elementos repetidos en la lista de relaciones.
- *setData(string, int, double, int)*: asigna valores a los atributos de la tarea.
- Se sobrecarga el operador (<) para darle un orden a la lista de tareas.

Requerimiento: la clase requerimientos define la estructura genérica de los tres tipos(técnicos, administrativos y funcionales). Tiene una complejidad que se mide en horas y un vector de tareas. Contiene un único método:

- *agregueTareas(Tarea)*: toma como parámetro una tarea y la agrega a su vector de tareas.

Proyecto: es la clase principal, en ella se encuentran los tres requerimientos (por ende las tareas), el equipo de personas y el PB (lista que abarca las tareas de los tres requerimientos).

Aquí se encuentran los métodos de manipulación más fundamentales (mostrar tareas, asignar tareas, relacionar tareas). Su funcionalidad se divide en los siguientes métodos:

- *agregaPB()*: agrega las tareas de cada requerimiento al vector PB.
- *ordena()*: ordena el PB mediante prioridad.
- *muestraTareas()*: imprime la lista de tareas en el PB.
- *asignarTarea(int, int)*: mediante dos índices, asigna una tarea y una persona.
- *asignarRelacion(int, int)*: mediante dos índices relaciona dos tareas.
- *mostrarRelaciones()*: imprime las relaciones de tareas existentes.