Trabajo Opcional
La Torre de Babel - Programar en Grupo

Pablo Sánchez

Dpto. Matemáticas, Estadística y Computación
Universidad de Cantabria
Santander (Cantabria, España)
p.sanchez@unican.es

Objetivos

Objetivos

1. Ser capaz de aplicar los conceptos aprendidos sobre un lenguaje de programación, de tipo estructurado u orientado a objetos, (teóricamente) desconocido por el alumno.
2. Aprender a despreciar la sintaxis de un lenguaje de programación.
3. Aprender a trabajar en grupo.

Trabajar en Grupo

Máxima del Trabajo en Grupo

Tiene sentido si el valor del conjunto debe ser superior a la suma de las partes.

1. El grupo produce resultados de mayor calidad que si los individuos trabajan individualmente.
2. Cada individuo saca beneficio del hecho de pertenecer al grupo.
3. Todos los miembros del grupo desarrollan una tarea distinta que es beneficiosa para el conjunto del grupo.
4. Precondición: Todos los miembros del grupo poseen algún tipo de capacidad, habilidad y conocimiento que le permite desarrollar una tarea beneficiosa para el grupo.

Trabajar en Grupo en Programación

1. **Programación por pares**, usado comúnmente en *programación extrema* [Beck, 1999].
2. Programar por turnos, jugando al abogado del diablo.

El Abogado del Diablo

1. El objetivo del abogado del diablo es encontrar tantos fallos, de cualquier tipo, como sea posible (hacerse una lista de elementos a comprobar).
2. La idea es que es mucho mejor encontrar los errores antes de entregar el producto, que arreglarlos (si aún fuese posible) cuando éstos se producen.
3. El papel de abogado del diablo debe cambiar por turnos.
4. El resto de compañeros han de aceptar de buen grado al abogado del diablo.
5. El abogado del diablo debe desempeñar su papel diligentemente.

Esquema Recomendado de Trabajo (I)

1. Decidir en grupo el TAD y la técnica de implementación a usar, de acuerdo a las expectativas y capacidades de cada grupo.
2. Un miembro del grupo diseña las estructuras de datos necesarias para implementar el TAD.
3. El compañero ejerce de abogado del diablo; busca posibles debilidades de las estructuras teniendo en cuenta las operaciones a implementar y propone soluciones.
4. Se discuten las soluciones, se aprueban si procede y se intercambian los roles.
5. Se repite el proceso desde el paso 2 hasta que ambos estén satisfechos con el diseño.

Esquema Recomendado de Trabajo (II)

1. En grupo, se ordenan las operaciones de forma que se puedan ir implementado de forma secuencial.
2. Se dividen las operaciones en bloques de forma que cada bloque tenga más o menos la misma carga de trabajo. Los bloques pueden ser impares.
3. Por cada bloque, un compañero implementa y el otro ejerce de abogado del diablo.
4. Los papeles se van intercambiando de forma alternativa por cada bloque.

Esquema Recomendado de Trabajo (III)

1. El objetivo del abogado del diablo es encontrar tantos fallos como sea posible, tanto de programación como de documentación.
2. El abogado del diablo debe diseñar concienzudamente los casos de prueba y archivar los programas de prueba para dichos casos (ej. test1, test2, etc.).
3. En caso de fallo, el compañero más capacitado para ello soluciona el fallo y el abogado del diablo pasa de nuevo los casos de prueba que correspondan.
4. Cuando el abogado del diablo da el visto bueno, el compañero que implementó hace de abogado del diablo de abogado del diablo.

Ejemplo de Lista de Comprobaciones

1. Funciona para estructuras vacías, con un elemento y varios elementos.
2. No es posible acceder a punteros nulos ni acceder a posiciones no válidas de un vector si se cumplen las precondiciones.
3. Todas las llamadas a métodos cumplen las precondiciones.
4. Si se satisfacen las precondiciones, ningún método lanza una excepción.
5. Todas las funciones están comentadas de forma adecuada, precisa y concisa.
6. No hay faltas de ortografías ni errores tipográficos (ni acentos ni ñ's).
7. Entiendo todo el código, bien porque su funcionalidad es obvia o porque se deduce de los comentarios proporcionados.
8. Los mensajes que se muestran por pantalla son claros, precisos y concisos; y están libres de errores tipográficos y ortográficos (salvo acentos y ñ's).

Edición de un Programa por Varios Usuarios

1. Es preciso controlar las versiones y el acceso concurrente a los ficheros de código [Hass, 2003] (esperar a 4º).
2. Sólo un miembro del grupo tiene el control sobre el código y puede modificarlo.
3. No borrar los cambios, simplemente guardarlos entre comentarios.
4. Si se introducen elementos nuevos, marcar los elementos nuevos.
5. Identificar cada cambio con un número de cambio e identificar el autor del cambio.
6. Mantener un índice de cambios donde se guardan los cambios hechos y el motivo de dicho cambio.
7. Por cada cambio, volver a ejecutar de nuevos los tests.
8. Si el cambio se aprueba, se puede eliminar la traza del mismo.
9. Cada cierto tiempo, guardar versiones estables del trabajo realizado.

Referencias

Beck, K. (1999).
Extreme Programming Explained: Embrace Change.
Addison-Wesley Professional.

Hass, A. M. J. (2003).
Configuration Management Principles and Practice.
Addison-Wesley Professional.