La programación en grupos de trabajo

Maria Isabel Alfonso Galipienso
Francisco José Mora Lizán
Francisco Martínez Pérez
Departamento Ciencia de la Computación e Inteligencia Artificial
(email: eli@dccia.ua.es, mora@dccia.ua.es, fmartine@dccia.ua.es)

Resumen

De todos es conocida la importancia que tiene el desarrollar aplicaciones dentro de un grupo de trabajo, y lo difícil que puede resultar. Aquí trataremos sobre tres tipos de organización seguida en las prácticas de Ingeniería del Software II en diferentes años, para permitir que el alumno tome conciencia de esta problemática y sepa afrontar con éxito el futuro. Para cada una de ellas expondremos los aspectos positivos y negativos así como las herramientas utilizadas para su realización.

1 Introducción

Durante el curso 1992-93 se comenzó a impartir por primera vez la asignatura Ingeniería del Software 2 en la carrera de Ingeniería Informática en la Universidad de Alicante. Ya desde el primer momento nos planteamos el enfocar las prácticas de forma que abordáramos la construcción de un sistema software "grande", es decir, un sistema que por su envergadura necesitara ser desarrollado mediante el esfuerzo conjunto de varias personas. De esta forma el alumno vive la experiencia de trabajar dentro de un grupo de trabajo, descubriendo así los beneficios y problemas que de ello se derivan, al tiempo que les mostramos diferentes métodos y soluciones.

Evidentemente no disponemos del tiempo necesario para la realización de un proyecto "real" con las dimensiones mencionadas, pero hemos intentado simularlo y trasladar dicha situación "real" a la escala de tiempo (un cuatrimestre) y personal (alumnos) a la que nos vemos limitados.

Para ello hemos organizado los grupos de prácticas de diferentes formas a lo largo de varios años. En cada una de ellas hemos encontrado una serie de ventajas e inconvenientes que iremos exponiendo a continuación. También mostraremos los métodos y herramientas utilizadas para cada una de las fases de desarrollo de una aplicación software (Análisis, diseño, implementación, prueba, documentación, etc...) así como de las herramientas utilizadas para la comunicación y organización de los grupos de trabajo.

2 Organización de los grupos

La asignatura Ingeniería del Software 2 tiene una carga de 6 créditos (3 teóricos más 3 prácticos) con carácter cuatrimestral. Las prácticas se planifican en 15 sesiones de 2 horas (una sesión por semana) para cada alumno. El número de alumnos por sesión oscila entre 15 y 20.

Hasta el momento hemos experimentado con los siguientes tipos de organizaciones:

1. COOPERATIVA: Todos los alumnos de una sesión trabajando con un único proyecto.
2. ROTACIONAL: Los alumnos de una sesión se dividen en diferentes grupos, cada uno con una instancia diferente del mismo proyecto.
3. MODULAR: Los alumnos de una sesión de dividen en diferentes grupos y abordan una serie de pequeños problemas que se van resolviendo en pocas sesiones.

A continuación se detalla cada una de las aproximaciones, mostrando algunas de sus ventajas, inconvenientes, problemas y soluciones.

2.1 Organización cooperativa

Este tipo de organización es la que más se asemeja al mundo real. Se trata de desarrollar una aplicación entre un grupo de 10 personas, de forma que cada uno de los documentos desarrollados durante las prácticas tenga un responsable (Plan de proyecto,

especificación de requisitos, documento de diseño, plan y procedimiento de prueba y manual de usuario). Además se nombrarán responsables (cargos) para las siguientes tareas: Implementación, integración, comunicación, y responsable de pruebas y gestor de configuraciones.

Si el grupo está formado por más de 10 personas, se subdividirán los cargos se crea convenientes en dos sub-responsabilidades claramente delimitadas. La idea es que cada uno de los alumnos, de forma totalmente individual, asuma un puesto de responsabilidad y actúe como "director" de esa tarea, encargándose de organizarla y revisarla de la forma que considere más oportuna.

Al mismo tiempo, cada uno de los alumnos tomará parte en la realización de un conjunto determinado de actividades, actuando así como individuo que forma parte de un colectivo que sigue las directrices de un líder (cargo).

Con este planteamiento y con el fin de que las prácticas fuesen amenas implementamos una serie de juegos a elegir por el grupo (guerra de barcos, damas, stratego y lingo).

La tarea de organización de las practicas con este enfoque no ha sido nada fácil debido al alto número de personas implicadas: en este caso el profesor actúa como un super-director del proyecto.

Además, existe una cierta especialización de los alumnos en determinadas tareas, como son los cargos, de forma que por ejemplo tenemos que un alumno puede haber practicado correctamente las actividades conducentes a la realización de la integración del sistema, pero por otra parte no ha practicado lo suficiente otras actividades que desempeñan otros cargos, como pruebas o gestión de configuraciones. Es decir, no todos los alumnos practican las mismas cosas, aunque todas ellas se plantean en las clases teóricas.

Por otro lado, y a pesar de que algunos proyectos no fueron terminados completamente por falta de tiempo (concretamente cuatro sobre doce grupos en total), la experiencia resultó altamente satisfactoria para los alumnos, que comprobaron lo esencial de una buena gestión y comunicación efectiva entre los miembros de un equipo para realizar con éxito un proyecto software. Es decir, consiguieron practicar lo que se les explicó en teoría, logrando así los objetivos marcados.

Un problema que observamos en este tipo de organización es el control del trabajo realizado por cada persona, encontrándonos en algunos casos con alumnos con falta de motivación que retrasan al grupo (como se ha mencionado cuatro de los 12 grupos no consiguieron un producto totalmente acabado). También nos encontramos con la figura opuesta, que es la persona que quiere hacerlo todo y no deja participar al resto. Por lo tanto habrá que reforzar en este tipo de organización la idea del individuo como integrante participativo del grupo, por encima de los intereses individuales. A pesar de esto la propia dinámica del grupo tiende a detectar y corregir estas anomalías, teniendo que intervenir el tutor en pocas ocasiones y a petición expresa de los alumnos.

2.2 Organización rotacional

La organización rotacional es la más difícil de entender. Se trata de que los alumnos de una sesión se subdividan en varios grupos (en este caso 3), cada uno de los cuales debe desarrollar una instancia de un determinado proyecto (el mismo para los tres, con objeto de facilitar la organización). Cuando un grupo acaba una fase del proyecto "entrega" el trabajo realizado al grupo siguiente, y "recibe" el trabajo realizado por el grupo anterior (para ello los grupos han sido ordenados previamente). Esquemáticamente y suponiendo que creamos 3 grupos (A,B y C) y 3 instancias de un proyecto (X,Z,Y), el desarrollo de los trabajos se realizará de la siguiente forma:

	Grupo A	Grupo B	Grupo C
Análisis	X	Y	Z
Diseño	Z	X	Y
Implementación	Y	Z	X
Prueba	X	Y	Z
Integración	Z	X	Y
Documentación	Y	Z	X

En nuestro caso planteamos la gestión de un VideoBank, cuya especificación fue entregada a los alumnos (asumimos que las diferentes técnicas y metodologías utilizadas tanto en especificación

como en análisis y diseño son ya conocidas por el alumno, habiendo adquirido dichos conocimientos en las asignaturas AESI: Análisis y Especificación de Sistemas de Información e IS1: Ingeniería del Software 1 respectivamente, y que son prerequisitos de Ingeniería del Software 2). Así, según este enfoque, en cada sesión había 3 videobanks distintos (ya que la especificación dada no era totalmente completa, debiendo completarla los alumnos).

Dentro de cada grupo se asignaron una serie de roles, siendo cada persona responsable de una faceta (planificación, análisis, diseño, documentación, implementación, integración y comunicación), aunque todos trabajaban en todas las fases.

La experiencia fue muy interesante ya que eliminábamos el problema de la especialización de los alumnos en tareas concretas, haciendo que todos practicasen todo. Además, se incrementaba la necesidad de una mayor coordinación y comunicación efectiva para terminar con éxito el proyecto.

Sin embargo, y precisamente por este aumento de conflictos potenciales debidos a la falta de coordinación y comunicación, el mayor problema con el que nos enfrentamos fue el de los retrasos en las fechas de entrega. Cuando un grupo se retrasaba, el siguiente grupo que debía de recoger el resultado de su trabajo no podía continuar. Para resolver este problema aconsejamos 3 posibles soluciones:

- Introducir holguras, cosa que no es muy sencilla pensando en el tiempo en que tenemos para el desarrollo de las prácticas (un cuatrimestre).
- Otra solución es proporcionar al alumno la documentación correspondiente a las primeras fases (especificación, análisis y diseño).
- Una tercera posibilidad consiste en reducir el número de rotaciones, por ejemplo reducirlas a la mitad.

La gran ventaja de la organización rotacional es la evaluación por parte de los alumnos del trabajo realizado por sus compañeros, fomentando así las revisiones entre grupos y potenciando la comunicación efectiva entre ellos.

2.3 Organización modular

Este tipo de organización nos ha servido para eliminar problemas de atrasos en las diferentes fases del proyecto software. Por otro lado nos ha permitido utilizar prácticas diseñadas de forma precisa para conseguir los objetivos en determinadas fases. Como ya veíamos en otro tipo de organizaciones utilizabamos el mismo problema de principio a fin y nos encontrabamos con que el problema no se adecua bien para practicar en una determinada fase.

La planificación de las sesiones de trabajo estuvo en función de las sesiones teóricas [1]:

Teoría	Práctica
Administración de proyectos	Comentario sobre la planificación de WinWord 1.0 [2]
	Establecimiento de Plantilla Documento Planificación.[3]
	Especificación de una aplicación VideoClub.
	Herramientas PERT GANT.
Estimación de costes.	Aplicación del método de COCOMO al proyecto propuesto.
Administración de Recursos Humanos	Diseño del documento de Pruebas de selección y organización de personal.
	Diseño y realización de pruebas de seleección
Gestión de Calidad	Estudio de un articulo sobre la Calidad del Software [4]
Mejora de Procesos	Establecimiento de métricas para obtener información del proceso software.
	Establecimiento de métodos de análisis de la información obtenida.
La Búsqueda de Defectos	Diseño de pruebas de funcionales, estructurales y de interfaz.
Verificación y Validación	Ejecución de las pruebas de unidad.
	Integración del sistema.
Mantenimiento de Software	Diseño de plantillas de petición de cambios.

	Establecimiento de mecanismo de mantenimiento.
Gestión de configuraciones	Uso de las herramientas de construcción del sistema y de gestión de versiones
Reingeniería	Utilización de una herramienta de reingeniería.

En esta aproximación, una de las ventajas más claras frente a las anteriores organizaciones consiste en que todos los alumnos realizan exactamente las mismas tareas, con lo que no hay especializaciones. Además permite una mayor flexibilidad a la hora de reforzar aspectos concretos con ejemplos de desarrollo más adecuados en cada caso.

Como inconveniente disminuye notablemente la necesidad de comunicaciones entre los miembros del grupo, de forma que los alumnos experimentan menos con la resolución de problemas derivados de una comunicación no efectiva, que en un caso real suelen ser habituales.

3 Herramientas y métodos utilizados

Para la organización cooperativa y rotacional de grupo, se necesitaban de forma especial herramientas que facilitasen la comunicación e intercambio de información, así como resultaba fundamental el establecer una forma efectiva de visibilidad y control de cambios de la información. Para ello experimentamos con dos opciones: en una de ellas se utilizó la herramienta LaTeX que nos permitía una crear plantillas de documentos de forma sencilla y rápida de forma que se estandarizasen los formatos a utilizar por todos; en la segunda optamos por utilizar formatos html debido a que ademas, se favorecía bastante la accesibilidad actualizada de la documentación generada por todos los alumnos, teniendo en cuenta que se sucedían distintos cambios en la misma durante la realización de las prácticas.

A la hora de realizar estimaciones se ha contado con una herramienta software de libre distribución, que implementa el método algorítmico de COCOMO, de forma que resulta bastante cómodo el uso de las fórmulas del método. Dicho software permite utilizar tanto el modelo básico, como el intemedio y el detallado, así como posibilidad de generar automáticamente informes de resultados obtenidos, los cuales se pueden integrar fácilmente en la documentación general, bien sea utilizando LaTeX o formatos html utilizando el visor de Netscape y/o Mosaic.

Para la creación de agendas hemos optado por utilizar Microsoft Project, que permite de forma sencilla el manejo de redes de tareas y posibilita la organización y visualización de los datos (actividades, tiempos, precedencias, recursos y costes), en distintos formatos de manera automática (redes de tareas, estructura de descomposición de trabajos, calendario...), así como generación automática de fichero de resultados.

Para implementar los sistemas también hemos probado con distintos lenguajes de alto nivel, primero utilizando C y C++, aprovechando que los alumnos ya tienen experiencia en estos lenguajes debido a asignaturas que han cursado con anterioridad a Ingeniería del Software 2, permitiéndoles así centrarse en los aspectos prácticos derivados exclusivamente de la parte teórica del curso. Posteriormente, optamos por el uso del lenguaje Ada, debido a su facilidad para especificar de forma clara los problemas, y su capacidad para poder separar dicha especificación (que tiene que ser visible a todo el mundo) de los detalles internos de implementación. Hay que recordar que, especialmente para los dos primeros tipos de organización es necesario establecer de forma clara las funcionalidades e interfaces de las distintas componentes del problema. A pesar de ser un lenguaje que los alumnos no han utilizado en otras asignaturas, y el esfuerzo extra que supone el aprendizaje de un nuevo lenguaje, los alumnos se muestran satisfechos con la experiencia.

Para la realización de las pruebas se han utilizado los métodos de caja negra de partición equivalente y de caja blanca de prueba de caminos. En este caso el proceso ha tenido que realizarse de forma manual o semi-automatizada con herramientas propias que los alumnos han creado para efectuar las pruebas lo más eficiente posible.

Para integrar el sistema se han seguido distintas estrategias en función del diseño efectuado en cada proyecto, y se ha hecho uso de la herramienta make para construir apropiadamente el sistema.

También se han usado las herramientas RCS y SCCS para practicar con la gestión de versiones y releases. Los alumnos han valorado positivamente la utilización de este software, y "lamentan" no haberlas conocido antes.

Finalmente, y como ejemplo de herramienta de reingeniería se ha optado por PowerDesignor, que permite este tipo de actividades, y generación automática de informes obtenidos. Una vez más hemos constatado el interés de los alumnos por este tipo de software.

4 Conclusiones

Hemos intentado reflejar nuestras experiencias y esfuerzos realizados en el diseño de la organización de las prácticas de la asignatura de ingeniería del software 2, exponiendo las ventajas e inconvenientes de los distintos enfoque se han seguido. La idea es que los alumnos trabajen de la forma más realista posible como integrantes de un grupo de trabajo que tiene un objetivo común: el éxito del proyecto software.

Hasta ahora, hemos utilizado tres aproximaciones diferentes. La aproximación que hemos denominado cooperativa, en la que aparecen las figuras de responsables de tareas, de forma que el alumno actúe como líder en ciertos aspectos del desarrollo, encargándose por tanto del seguimiento y control de los mismos. Esto es bueno, pero no todos los alumnos pueden practicar las mismas tareas, por tanto, para paliar ese efecto de la especialización, se opta por un enfoque denominado rotacional, en el que además, se incrementa la necesidad de comunicación efectiva de los alumnos. Por otro lado vemos que el problema que se presenta es el de posibles retrasos en la ejecución de las prácticas. Como alternativa, presentamos la aproximación modular, en la que reduce la necesidad de comunicaciones y por lo tanto, el riesgo de retrasos no deseados.

Las dos primeras organizaciones son más parecidas a la vida real, pero por ello mas complejas siendo difícil de organizar y llevar un seguimiento y evaluación de los alumnos por parte de los profesores. Por otro lado, en estas aproximaciones la mayoría de los alumnos manifiestan una actitud positiva respecto a las prácticas con resultados visibles por el resultado de las mismas. Por otro lado el tercer enfoque permite el intensificar y reforzar aspectos concretos de la teoría de forma sencilla, teniendo en cuenta la limitación temporal clara con la que nos enfrentamos.

Por otro lado, la disponibilidad de herramientas automáticas adecuadas posibilitan el desarrollo efectivo de las distintas aproximaciones facilitan una mejor comprensión de los métodos de trabajo.

De cualquier forma, pensamos que aquí no se agotan las posibilidades y que es necesario seguir investigando con nuevos métodos que permitan una enseñanza efectiva de la Ingeniería del Software.

Bibliografía

[1] Somerville I. *Software engineering.* Addison-Wesley 1.996

[2] Steve McConnell. *Desarrollo y Gestión de Proyectos Informáticos.* Steve McConnell. McGraw Hill 1.997

[3] Roger S.*Pressman. Ingeniería del Software. Un enfoque práctico.* McGraw Hill. 1.993

[4] Ian, H. G. Woodman.Relationship between the activities of the software process and the quality of the software product. Research Report/94/171. University Strathclyde 1.994