EXPERIENCIA DE COORDINACION EN DOS ASIGNATURAS DE INFORMATICA

Robert BENAVENTE[1], Ernest VALVENY[2], Jaume GARCIA[3], Ágata LAPEDRIZA[4], Miquel FERRER[5], Gemma SANCHEZ[6]

Centre de Visió per Computador[1,2,3,4,5,6], Departamento de Ciencias de la Computación [2,3,4,5,6]

Universitat Autònoma de Barcelona[1,2,3,4,5,6]

Resumen

En esta comunicación presentamos una experiencia docente en la que se ha coordinado el funcionamiento de dos asignaturas de ingeniería informática. Tradicionalmente, las dos asignaturas, que pertenecen al mismo ámbito de conocimiento pero tratado desde puntos de vista diferentes, habían funcionado de forma independiente. Este hecho provocaba que la mayoría de los alumnos no tuvieran una visión global de ellas y hacía que perdieran el interés por las asignaturas.

La coordinación de las asignaturas tiene como eje central el desarrollo de un proyecto común bajo la metodología del aprendizaje basado en proyectos. Durante el curso, los alumnos deben implementar diferentes partes del proyecto que finalmente integran a partir de los conocimientos adquiridos en ambas asignaturas. Además, esta metodología permite el trabajo de algunas de las competencias necesarias en ingeniería.

Después de tres cursos de experiencia, los resultados han sido muy satisfactorios tanto a nivel cualitativo como cuantitativo. La motivación y la implicación de los alumnos hacia las asignaturas ha aumentado considerablemente y el rendimiento de los alumnos se ha incrementado de forma notable, descendiendo el número de alumnos no presentados y aumentando el número de alumnos que superan las dos asignaturas.

Palabras Clave: *Experiencia docente, coordinación, aprendizaje basado en proyectos, aprendizaje cooperativo, informática.*

1. Introducción

En el nuevo marco del Espacio Europeo de Enseñanza Superior (EEES), la docencia universitaria ya no puede basarse únicamente en la transmisión de contenidos a los alumnos, sino que debe ir más allá para dotar al alumno de una serie de competencias necesarias para el desarrollo de su profesión. Por lo tanto, la adaptación al EEES implica muchos cambios en la organización y la metodología docente. En algunos casos, este proceso de cambio implica que dos o más asignaturas deban coordinarse para alcanzar los objetivos del nuevo marco.

En este trabajo se presenta una experiencia docente en la que se ha coordinado el funcionamiento de las asignaturas *Algoritmos y Programación* (AP) y *Lenguajes de Programación* (LP) de la titulación de ingeniería informática con el objetivo de mejorar el proceso de aprendizaje del alumno. La experiencia se ha llevado a cabo en los cursos 2005-06, 2006-07 y 2007-08, y se enmarca dentro del plan piloto de adaptación al EEES que se ha llevado a cabo en algunas titulaciones de la Universitat Autònoma de Barcelona (UAB) desde el curso 2004-05.

Las dos asignaturas corresponden al mismo ámbito de conocimiento de la informática, el de la programación en lenguajes imperativos de alto nivel, y son impartidas en el primer semestre del primer curso de la titulación. La diferencia entre ellas es que mientras AP es una asignatura con una componente más teórica y que corresponde a un nivel alto de abstracción, LP es más práctica y tiene como principal objetivo que el alumno aprenda a programar en un lenguaje de programación concreto que, en nuestro caso, es el lenguaje C. Por lo tanto, se trata de dos

asignaturas complementarias y, de hecho, podríamos considerar que constituyen un único módulo de introducción a la programación.

Sin embargo, a pesar de tener una relación tan directa en cuanto a contenido, estas dos asignaturas siempre habían funcionado de forma independiente. Este hecho había provocado que durante el proceso de aprendizaje se produjeran situaciones poco eficientes, tales como solapamiento de temarios, repeticiones de temas y actividades (por ejemplo, las prácticas de ambas asignaturas eran muy parecidas) o falta de sincronización entre las explicaciones teóricas del diseño de un algoritmo (en AP) y su implementación práctica en el lenguaje de programación (en LP). Como resultado de estos desajustes, los alumnos tenían dificultades para ver la relación entre AP y LP, y no eran capaces de tener una visión global sobre el contenido de las dos asignaturas y el proceso de desarrollo de una aplicación informática. Estos problemas también provocaban la desmotivación de muchos alumnos y la pérdida de interés por las asignaturas.

Con la participación en el plan piloto de adaptación al EEES consideramos que era una buena oportunidad para mejorar la docencia de estas dos asignaturas y solucionar las deficiencias citadas anteriormente. Así pues, los objetivos de la adaptación de las asignaturas que nos propusimos fueron:

- Conseguir que los alumnos tuvieran una visión global de la programación, desde el inicio (planteamiento de un problema y diseño de un algoritmo) hasta el final (implementación del algoritmo mediante un lenguaje de programación).
- Optimizar el funcionamiento global de las dos asignaturas evitando la repetición de temario y actividades.
- Incentivar la participación e implicación de los alumnos mediante el planteamiento de problemas reales y motivadores.
- Trabajar en ambas asignaturas algunas de las competencias necesarias en unos estudios de ingeniería.

El hecho de participar en el plan piloto proporcionaba cierta flexibilidad para reorganizar las asignaturas redistribuyendo recursos, modificando contenidos e incluyendo nuevas metodologías docentes. Sin embargo, las experiencias del plan piloto debían ceñirse al plan de estudios vigente, de forma que se debía mantener la independencia de las dos asignaturas y contemplar que, por ejemplo, un alumno estuviera matriculado de una de las asignaturas pero no de la otra. Esto implicaba que el proceso no podía ser una simple fusión de las dos asignaturas, sino que se debía coordinar su funcionamiento. En este contexto, se propone a los alumnos el desarrollo de un proyecto común a lo largo del curso como elemento central de la coordinación entre las dos asignaturas [1]. De esta forma se consigue que AP y LP compartan un mismo hilo argumental a la vez que los alumnos adquieren una visión más global de los fundamentos de la programación.

En este trabajo presentamos el proceso de reorganización de las asignaturas que se siguió para conseguir coordinarlas y solucionar así los problemas que provocaba su funcionamiento anterior. En los siguientes apartados detallamos las metodologías utilizadas en la docencia de las asignaturas, los resultados obtenidos en este proceso y las conclusiones que hemos extraído de la experiencia después de tres cursos de aplicación de la coordinación de las asignaturas.

2. El proceso de coordinación

En el momento de iniciar el proceso de adaptación, en el curso 2005-06, había un total de 318 alumnos matriculados en AP y 291 en LP. La distribución de la docencia según su tipología (teoría, problemas o prácticas) era la siguiente:

Algoritmos y Programación tenía un total de 6 créditos distribuidos en 3 de teoría, 1.5 de problemas y 1.5 de prácticas. Los alumnos estaban distribuidos en 4 grupos para teoría y problemas (con aproximadamente 80 alumnos en cada grupo) y 14 de prácticas (con un máximo de 24 alumnos en cada grupo).

Lenguajes de Programación tenía un total de 4.5 créditos distribuidos en 1.5 de teoría y 3 de prácticas. Los alumnos estaban distribuidos en 4 grupos para teoría (con entre 70 y 75 alumnos por grupo) y 11 de prácticas (con un máximo de 28 alumnos en cada grupo).

El proceso de coordinación de las asignaturas se debía realizar con los recursos docentes disponibles, ya que a pesar de participar en el plan piloto de adaptación al EEES no se contaba con recursos adicionales.

2.1 Redistribución de contenidos

Al iniciar la adaptación se realizó un análisis de los contenidos y las necesidades de cada asignatura para llegar a una nueva distribución que facilitara el funcionamiento y la coordinación de las asignaturas. Dado que las dos asignaturas pertenecen al mismo bloque temático, nos marcamos como objetivo llegar a un único temario que incluyera todas las etapas del ciclo clásico de la programación: análisis, diseño, codificación y prueba (Ver Fig. 1).

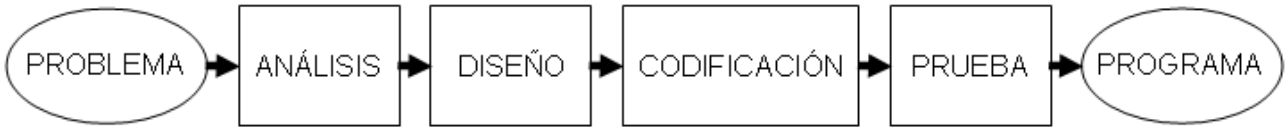

Fig. 1 Esquema del ciclo clásico de la programación

Como resultado del análisis realizado se identificaron cuatro contenidos básicos:

- Conocimientos teóricos básicos sobre algorítmica
- Análisis y síntesis de algoritmos para resolver problemas
- Conocimientos básicos del lenguaje de programación C
- Codificación de algoritmos en lenguaje C

Teniendo en cuenta la naturaleza de las dos asignaturas, se consideró que los dos primeros puntos, que corresponden a las partes más relacionadas con la algorítmica, se debían incluir en AP, mientras que los dos últimos puntos, que corresponden a la parte más aplicada del desarrollo de un programa, se debían incluir en LP. De esta forma, las primeras etapas de la programación clásica se tratan exclusivamente en AP, mientras que las dos últimas etapas se trabajan exclusivamente en LP (Fig. 2).

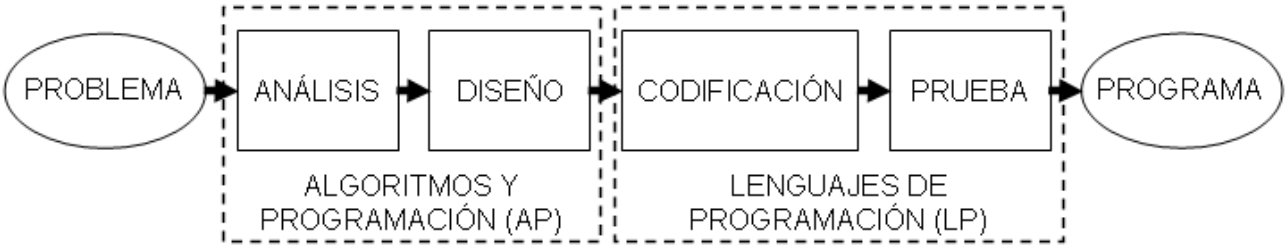

Fig. 2 En Algoritmos y Programación se trabajan los contenidos relacionados con las dos primeras etapas del ciclo clásico de la programación, mientras que en Lenguajes de Programación se trabajan los correspondientes a las dos últimas etapas.

Cada grupo de contenidos se desarrolla utilizando el tipo de docencia con la que creemos que se puede obtener el máximo aprovechamiento por parte del alumno. Así, la docencia de las dos asignaturas se imparte mediante:

1- **Clases magistrales (en AP).** Se presentan los contenidos teóricos del tema, de forma general y abstracta. Para explicar los algoritmos correspondientes se utiliza un pseudocódigo que no corresponde a ningún lenguaje de programación concreto.

2- **Seminarios de problemas (en AP).** Se presenta a los alumnos una serie de problemas a analizar y deben proponer el algoritmo (en pseudocódigo) adecuado para su solución.

3- **Seminarios de programación (en LP).** Se explica la sintaxis de las instrucciones del lenguaje C que se corresponden con las estructuras vistas en teoría y se proponen ejercicios en los que algún algoritmo en pseudocódigo debe ser traducido al lenguaje C.

4- **Prácticas de Laboratorio (en LP).** Se proponen una serie de programas que se deben implementar en el ordenador aplicando los conocimientos teóricos del tema y del lenguaje C adquiridos en las clases magistrales y seminarios anteriores.

A partir del análisis de contenidos, se redistribuyeron los recursos para la docencia de las dos asignaturas. Durante los tres cursos en los que se ha llevado a cabo la experiencia, en AP se han impartido 4 grupos de teoría de 3 créditos cada uno y 8 grupos de problemas de 3 créditos cada uno. En LP se han impartido 8 grupos de seminario de 1.5 créditos cada uno y 11 grupos de prácticas de 3 créditos cada uno. La Tabla 1 resume la redistribución de la carga docente:

Tabla 1 Resumen de la redistribución de recursos docentes entre las dos asignaturas coordinadas. (a) Situación inicial. (b) Distribución final

	Algoritmos y programación			**Lenguajes de programación**		
	Grupos	**Créditos**	**Total**	**Grupos**	**Créditos**	**Total**
Teoría	4	3	12	4	1.5	6
Problemas	4	1.5	6			
Prácticas	14	1.5	21	11	3	33
Total			**39**			**39**

(a)

	Algoritmos y programación			**Lenguajes de programación**		
	Grupos	**Créditos**	**Total**	**Grupos**	**Créditos**	**Total**
Clase Magistral	4	3	12			
Seminarios de Problemas	8	3	24			
Seminarios de Programación				8	1.2	9.6
Prácticas				11	3	33
Total			**36**			**42.6**

(b)

2.2 Planificación temporal

Aunque el proyecto común representa uno de los principales resultados del aprendizaje del estudiante, hay que tener en cuenta que tanto AP como LP son asignaturas del primer semestre de primer curso y que el alumno llega sin conocimientos previos de programación. Por lo tanto, se debe dedicar un tiempo significativo, especialmente en los dos primeros meses del curso, para

presentar los fundamentos de la programación. Esto se hace mediante las clases de teoría de AP y los seminarios de AP y LP.

El temario de las dos asignaturas se divide en ocho temas. Para cada tema, se dedica una clase de cada tipo de las detalladas en el apartado anterior. La secuencia de clase magistral, seminario de problemas, seminario de programación y prácticas de laboratorio correspondiente a cada tema, se coordina entre las dos asignaturas para asegurar que el orden no se rompa en ningún caso. De este modo, el seminario de programación y la práctica de laboratorio siempre se imparten la semana siguiente a la semana en la que se impartieron la clase magistral y el seminario de problemas correspondiente. En la Fig. 3 se muestra un esquema de la organización temporal de las clases de las dos asignaturas.

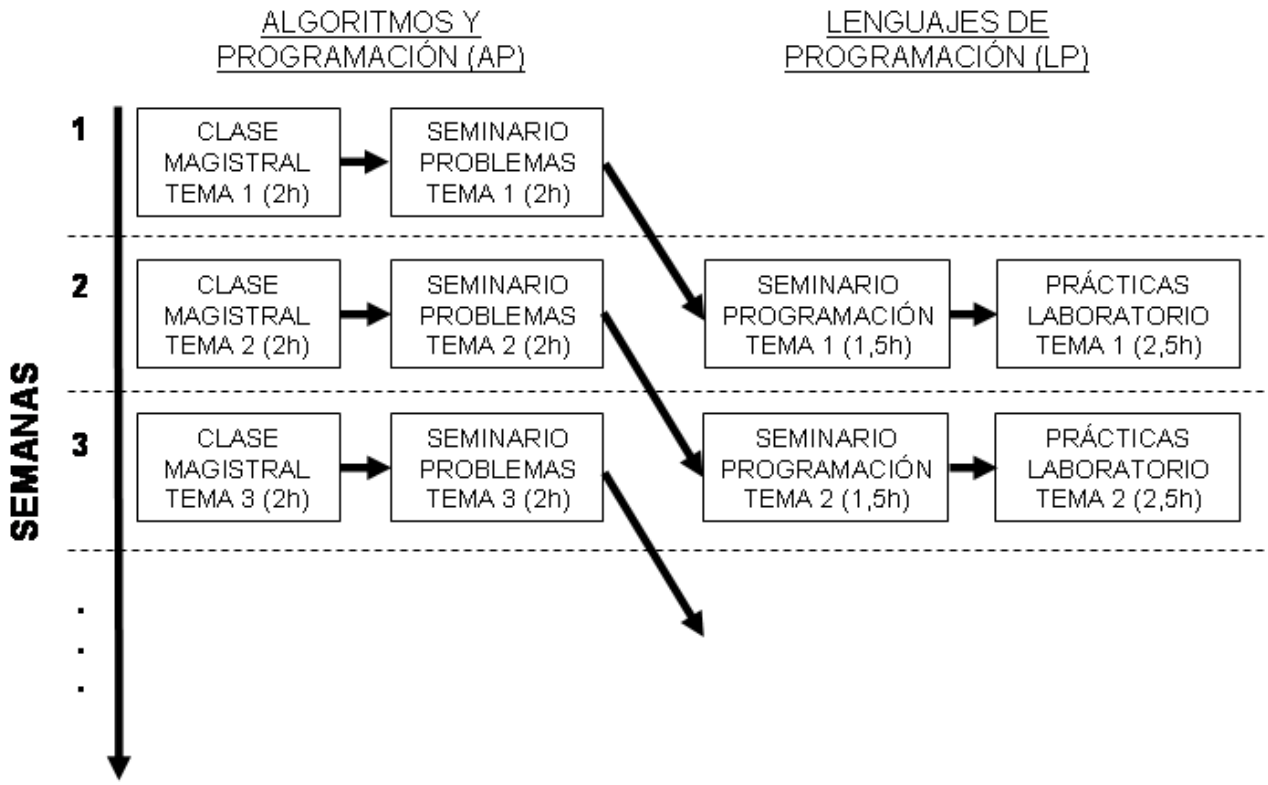

Fig. 3 Planificación temporal de las dos asignaturas después de la coordinación de contenidos.

La coordinación de los calendarios es una de los puntos clave de la experiencia, ya que con esta programación se consigue que el alumno siga una secuencia de aprendizaje coherente, pasando de la mayor abstracción de las clases magistrales a la mayor especialización de las sesiones de prácticas.

Estas sesiones de clase presencial, necesarias para introducir los conceptos teóricos y prácticos de la programación, están relacionadas también con el proyecto final del curso para integrar así, todo el proceso de aprendizaje del alumno. Tanto en los seminarios como en las prácticas de laboratorio, la mayoría de los problemas propuestos pertenecen a pequeñas partes del proyecto final, de forma que a medida que los van resolviendo están desarrollando diferentes partes del proyecto. De esta forma el alumno se ve obligado a ser constante en el desarrollo del trabajo y durante todo el curso tiene la ayuda del profesorado para ir avanzando.

Por otra parte, después de los dos primeros meses de clase, una vez introducidos los fundamentos de la programación, el trabajo personal del alumno se debe dedicar básicamente al desarrollo del proyecto, tanto para integrar las partes realizadas a lo largo de las sesiones previas de prácticas como para implementar la parte restante y más compleja del proyecto que sirve para integrar todos los conceptos que se han explicado previamente. En esta parte los alumnos

cuentan siempre con la ayuda de los profesores que tutelan el proceso de desarrollo de la parte final del proyecto.

Para poder hacer un seguimiento del trabajo del estudiante, se programan tres entregas evaluables durante el curso. Estas entregas son incrementales, de forma que cada una se basa en el trabajo realizado en la entrega anterior. De esta forma, las entregas también tienen una función tutelar ya que el profesor, además de evaluar el trabajo realizado, puede aprovechar para corregir errores u orientar al estudiante respecto a como continuar el trabajo. Cada entrega tiene una parte que corresponde a AP y otra que corresponde a LP. En la parte correspondiente a AP se pide al estudiante que haga el diseño genérico del algoritmo, mientras que en la parte de LP se le pide una implementación en C. Puesto que dentro del ciclo de programación el diseño de un algoritmo debe ser previo a su implementación en un lenguaje concreto, las entregas de AP son siempre anteriores a las entregas de LP. De esta forma, el estudiante ve la utilidad real de la fase de diseño como paso previo a la implementación.

3. Metodología docente

Como ya se ha explicado anteriormente, el hilo argumental que permite tener una visión global de los contenidos de las dos asignaturas es un proyecto común de programación. Para el desarrollo del proyecto se ha aplicado la metodología de aprendizaje basado en proyectos [2], que anteriormente ya se ha aplicado con éxito a otras asignaturas de ingeniería [3][4]. Por otra parte, en los seminarios de problemas y programación, los alumnos trabajan en equipo y se utiliza la metodología de aprendizaje cooperativo [5][6]. En la Fig. 4 se muestra un esquema que resume el ámbito de aplicación de cada metodología y la relación entre ellas.

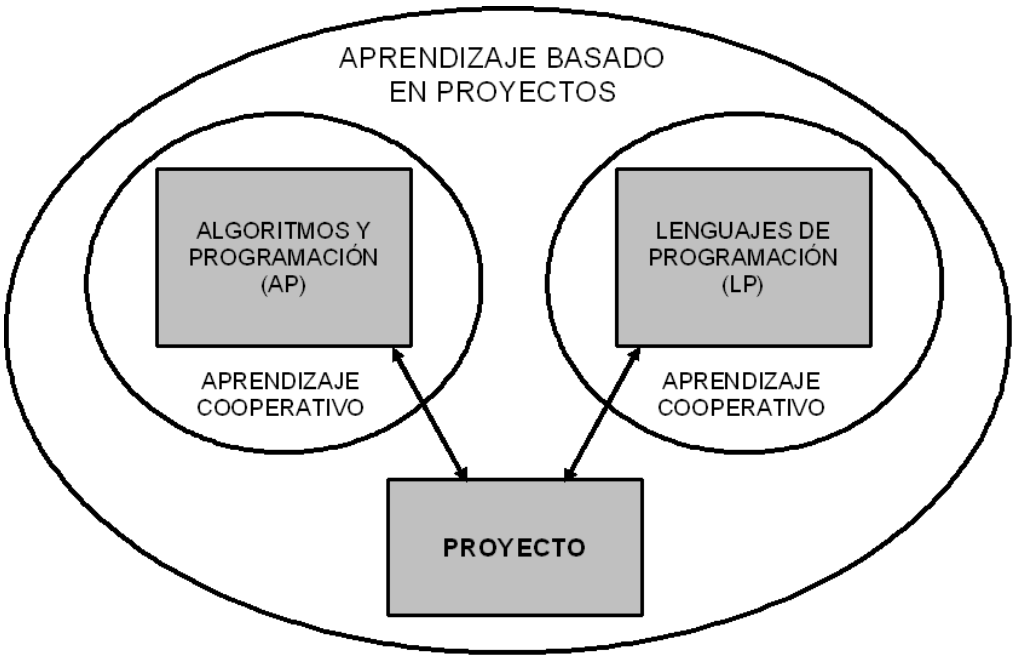

Fig. 4 Esquema de la aplicación de las diferentes metodologías docentes en las dos asignaturas.

En los apartados siguientes se detalla la forma en la que se ha aplicado cada metodología así como el sistema de evaluación que se ha seguido en las dos asignaturas.

3.1 Aprendizaje cooperativo en los seminarios

Tanto en los seminarios de problemas de AP como en los de programación de LP se aplica la metodología de aprendizaje cooperativo [5][6]. El trabajo generalmente se realiza en grupos de dos o tres alumnos, aunque en algunas ocasiones se realizan actividades en las que los grupos se fusionan en la parte final de la actividad para obtener grupos más amplios de cuatro o cinco alumnos. En estos seminarios, se propone a los alumnos una serie de actividades o problemas en

las que cada miembro del grupo debe realizar una parte del trabajo que posteriormente se combina con las partes del resto de integrantes del grupo para obtener la solución final. En otras actividades, los alumnos deben analizar separadamente un mismo problema y a continuación discutir las soluciones propuestas para llegar a un consenso y entregar una única solución por grupo. Una explicación más detallada de la aplicación del aprendizaje cooperativo a los seminarios de problemas y de programación se puede encontrar en [7].

Al final de cada seminario, los alumnos deben entregar la solución a la actividad propuesta en la sesión, que siempre contiene una parte evaluable a nivel de grupo y otra a nivel individual. De esta forma se puede realizar una evaluación continua del proceso de aprendizaje de los alumnos. Por otra parte, la aplicación de esta metodología permite el trabajo de algunas de las competencias que se quieren proporcionar en estas asignaturas, tales como la capacidad de trabajo en equipo, la capacidad de análisis y síntesis, y las habilidades comunicativas.

3.2 El proyecto común

La realización del proyecto común permite que los alumnos puedan desarrollar un proyecto de cierta complejidad, que engloba todos los contenidos del curso. Al inicio del curso se presenta a los alumnos un enunciado genérico que plantea las características básicas del proyecto a realizar. El enunciado tiene la suficiente libertad para que los alumnos puedan tomar sus propias decisiones de diseño y lo deban analizar en profundidad para concretar los detalles de implementación y escoger las estructuras de programación más adecuadas. Además, el enunciado debe corresponder a un problema lo más real y motivador posible. Por este motivo, en general, se proponen versiones simplificadas de juegos conocidos. Para la realización del proyecto, los alumnos trabajan en grupos de dos personas que entregan un único trabajo.

Aunque la idea del aprendizaje basado en proyectos es que los alumnos vayan adquiriendo los conocimientos necesarios para el desarrollo del proyecto a medida que los van necesitando, el hecho de que los alumnos no tengan ningún conocimiento previo de programación, hace que no se les pueda dejar absoluta libertad y se deba llevar a cabo una tarea de tutela con ellos. Por este motivo, y también con el objetivo de optimizar el tiempo disponible, tanto en los seminarios como en las prácticas de laboratorio, la mayoría de los problemas propuestos corresponden a partes del proyecto final. Además, el enunciado abierto proporcionado al inicio del curso se va detallando un poco más a medida que se deben implementar algunas de las partes.

De esta forma, también se potencia la constancia en el trabajo durante el curso, ya que si el alumno va realizando todos los trabajos de los diferentes seminarios y sesiones de laboratorio, va a tener acabada una parte considerable del proyecto cuando llegue la parte final del curso que se debe dedicar casi exclusivamente al desarrollo del proyecto.

Aparte de la entrega final del proyecto, también se establecen dos entregas parciales que junto con la entrega final, son incrementales, es decir, que cada entrega incluye el trabajo entregado anteriormente más alguna parte nueva. Con estas entregas también se ejerce la función tutelar del profesor ya que los errores o deficiencias que se detectan en los proyectos pueden ser solucionados para las siguientes entregas.

Como ejemplo de algunos de los proyectos que se han implementado en los últimos cursos, podemos citar la implementación del popular juego del Tetris o del juego de la ruleta. En la Fig. 5 mostramos un ejemplo de proyecto realizado por alumnos.

3.3 Sistema de evaluación

El sistema de evaluación para las dos asignaturas es prácticamente el mismo, aunque considerando para el cálculo de la nota, sólo la parte del proyecto correspondiente a cada asignatura. La planificación que se ha presentado, permite realizar un seguimiento del proceso de aprendizaje a lo largo de todo el curso y dar un peso muy importante a los trabajos y las entregas parciales realizados durante el curso. Aunque las pruebas escritas individuales todavía tienen un peso importante (40% de la nota final), todo el trabajo realizado a lo largo del curso supone el 60% restante. De este 60%, la mitad (30% de la nota final) corresponde a la entrega final del proyecto común (el diseño del proyecto se computa en AP y el resultado de la implementación se computa en LP). Finalmente, el 30% restante corresponde al trabajo de evaluación continuada realizado en

los seminarios de problemas (para la nota de AP) y en los seminarios de programación (para la nota de LP).

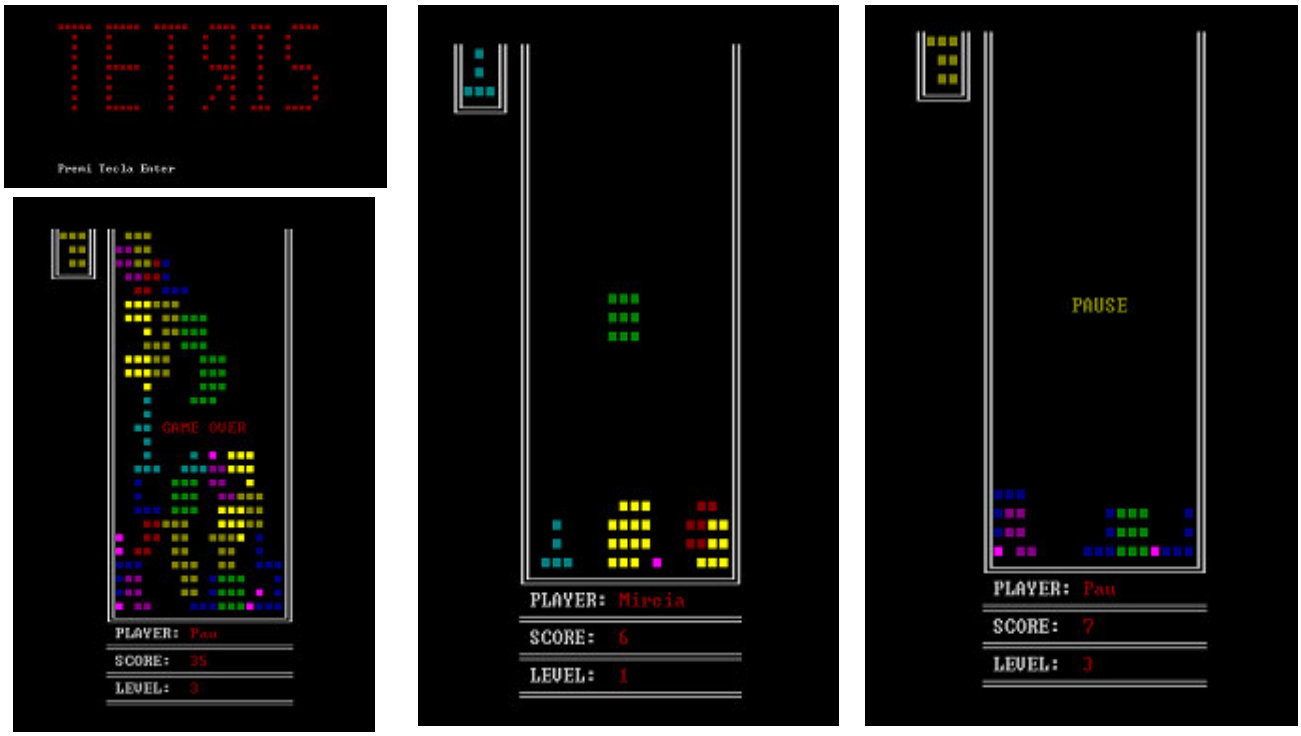

Fig. 5 Ejemplo de proyecto realizado por los alumnos: el juego del Tetris.

4. Resultados

Los resultados de la coordinación de las dos asignaturas han sido muy satisfactorios. El principal resultado es que se ha optimizado el funcionamiento de ambas de forma que se han solucionado los problemas detectados anteriormente tales como los solapamientos de temario o los desajustes temporales que hacían que, en algunos casos, se trataran de forma práctica en LP algunos temas que todavía no habían sido vistos a nivel teórico en AP. Esta mayor coordinación ha permitido que los alumnos tengan una visión global de las dos asignaturas de forma que les resulta más fácil relacionar los contenidos teóricos y abstractos de AP con su aplicación práctica en LP. Este hecho ha supuesto un incremento del interés de los alumnos que se refleja, por una parte, en un aumento de la asistencia a clase y por otra, en una mayor participación e implicación de los alumnos en el desarrollo de las clases. Por otra parte, el cambio de metodología asociado a los cambios en la organización de las asignaturas ha permitido trabajar algunas de las competencias fijadas para la titulación de ingeniería informática.

A nivel cuantitativo, se han comparado los resultados globales de las asignaturas con los resultados del curso anterior a la reorganización (2004-05) y se ha observado que, en primera convocatoria, el porcentaje de no presentados ha descendido del 39.31% al 11.56% en AP y del 38.83% al 20.81% en LP. El porcentaje de aprobados ha subido del 20.44% al 41.62% en AP y del 20.27% al 35.84% en LP. Además, la nota media de los alumnos presentados ha mejorado en 2.05 puntos (de 5.07 a 7.12) en AP y en 0.22 puntos (de 7.66 a 7.88) en LP. Por último cabe destacar la elevada correlación entre los aprobados de las dos asignaturas, puesto que el 75.32% de los aprobados en AP son alumnos que también aprueban LP. Este resultado se incrementa al hacer la correlación en sentido inverso: el 78.37% de los aprobados de LP también aprueban AP. La Tabla 2 muestra un resumen de estos resultados.

Respecto a la adquisición de competencias, mediante la metodología presentada se trabajan algunas de las competencias transversales que se han marcado como básicas en nuestra titulación. El desarrollo de las actividades permite trabajar la capacidad para el trabajo en equipo

(al trabajar con diferentes compañeros a lo largo del curso e incluso en una misma sesión), la capacidad de análisis y síntesis (al tener que comparar diferentes soluciones, propias y de los compañeros, para decidir la mejor solución) y las habilidades comunicativas (al tener que discutir los pros y contras de las diferentes soluciones, explicar el trabajo propio a los demás y defender las soluciones propuestas). Por otra parte, también se trabajan otros valores como la responsabilidad y la confianza en uno mismo cuando, por ejemplo, la evaluación el resultado global del grupo depende del resultado individual de una persona en un momento concreto, de forma que todos los componentes del grupo deben cooperar para mejorar el resultado final.

Tabla 2 Resultados cuantitativos de la experiencia de coordinación después de tres cursos de aplicación. Comparativa entre los resultados en primera convocatoria del último curso de la experiencia y el curso anterior a su inicio.

	Algoritmos y programación		**Lenguajes de programación**	
	2004-2005	**2007-2008**	**2004-2005**	**2007-2008**
Nº matriculados	318	173	291	173
% No Presentados	39.31	11.56	38.83	20.81
% Aprobados	20.44	41.62	20.27	35.84
Nota media (presentados)	5.07	7.12	7.66	7.88

5. Conclusiones

En este trabajo se ha presentado una experiencia de adaptación de dos asignaturas de ingeniería informática al EEES, en la que la reorganización de los contenidos y la inclusión de metodologías activas de aprendizaje han obtenido muy buenos resultados. El objetivo de la coordinación de las dos asignaturas fue, desde un principio, conseguir que los alumnos tuvieran una visión global de ambas asignaturas y así incrementar la participación e implicación de los alumnos en su proceso de aprendizaje de forma que el nivel de conocimientos y competencias mejorara respecto a los obtenidos anteriormente. En este aspecto, hay que destacar que los resultados han sido muy satisfactorios. La coordinación y la aplicación del aprendizaje basado en proyectos y del aprendizaje cooperativo han dinamizado las clases y han favorecido la implicación de los alumnos en su proceso de aprendizaje, hecho que se ha reflejado en un aumento considerable de la asistencia a clase y, por consiguiente, del seguimiento del curso. El resultado final ha sido una mejora muy importante del rendimiento de los alumnos que se ha visto reflejado en el aumento del número de presentados, el número de alumnos que superan la asignatura, y lo que creemos que es más importante, un aumento del nivel de conocimientos.

Por otra parte, la utilización de la metodología presentada permite trabajar la adquisición de algunas competencias imprescindibles para un ingeniero informático. También consideramos que la metodología permite la evaluación indirecta de estas competencias, ya que el nivel de adquisición de estas competencias se refleja en el resultado final del alumno (en nuestro caso el proyecto), evitando así una evaluación directa de las competencias que puede ser muy difícil de realizar.

Sin embargo, la aplicación de esta metodología ha tenido algunos inconvenientes. Por una parte, el aumento de la carga de trabajo para el profesor es considerable, ya que al trabajo de preparación de las actividades hay que sumar el tiempo para su evaluación. Por otra parte, la metodología propuesta resulta incompatible con algún tipo de estudiantes que no pueden tener un seguimiento continuo de las clases, de forma que los alumnos que quedan descolgados en algún momento del curso son difícilmente recuperables. Otro de los problemas que debemos considerar es la presencia de alumnos que sólo están matriculados en una de las dos asignaturas. En el caso de los alumnos sólo matriculados en AP, el caso no es excesivamente problemático puesto que

puede realizar todo el diseño del proyecto sin implementarlo, pero en el caso de los alumnos de LP, se les debe proporcionar una ayuda extra proporcionándoles alguna parte del diseño para que puedan realizar la parte de implementación en C que es la que les corresponde por estar matriculado de LP.

Actualmente se están considerando posibles formas de solucionar estos problemas. Para reducir el volumen de trabajo que supone la evaluación de todos los trabajos de evaluación continuada estamos estudiando cómo se podría aplicar la autoevaluación o la evaluación cruzada entre alumnos [8][9]. Quizá no sea factible aplicarlo a todas las actividades, pero su aplicación a una parte de ellas reduciría considerablemente la carga de trabajo de los profesores. Por lo que respecta a los casos de alumnos matriculados en sólo una de las asignaturas, una posible solución sería la creación de grupos específicos de dos alumnos (uno de AP y otro de LP) para que puedan desarrollar el proyecto completo, de forma que cada alumno realizaría la parte correspondiente a la asignatura de la que está matriculado.

6. Referencias

[1] LAPEDRIZA, A.; GARCIA, J.; VALVENY, E.; BENAVENTE, R.; FERRER, M.; SANCHEZ, G. "Una Experiencia d'Aprenentatge basat en Projectes en l'àmbit de la Informàtica". *V Jornades d'innovació docent*. Bellaterra, 2008.

[2] BRANDFORD, J.D.; STEIN, B.S. *The IDEAL problem solver.* W.H. Freeman and Co., 1993.

[3] MARTÍ, E.; GIL, D.; JULIÀ, C. "Una Experiencia de PBL en la Docencia de la Asignatura de Gráficos por Computador en Ingeniería Informática". *IV Congreso Internacional de Docencia Universitaria e Innovación*. Vol. 1, Barcelona, 2006, pp. 375.

[4] MARTINEZ, F.; HERRERO, L.C.; GONZALEZ, J.M.; DOMINGUEZ, J.A. "Project based Learning Experience in Industrial Electronics and Industrial Applications Design". *Proceedings of the Intenational seminar on innovative teaching and learning in engineering education*, Valladolid, 2006, pp. 299-312.

[5] JOHNSON, D.W.; JOHNSON, R.T.; SMITH, K.A. *Active Learning: Cooperation in the college classsroom.* Edina MN: Interaction Book Company, 1998.

[6] CARRASQUER, P.; DE LLAMA, A.; GIBERT, F.; ORGAZ, N.; PARELLA, S.; REYNAL, N.; SOLÀ, X. *Aprenentatge cooperatiu en ciències socials.* Servei de publicacions IDES-UAB, Col. Eines d'Innovació docent en Educació Superior, Bellaterra, 2006.

[7] FERRER, M.; BENAVENTE, R.; VALVENY, E.; GARCIA, J.; LAPEDRIZA, A.; SANCHEZ, G. "Aprendizaje Cooperativo aplicado a la Docencia de las Asignaturas de Programación en Ingeniería Informática". *VIII Jornada sobre aprendizaje cooperativo y I Jornada sobre innovación docente*. Lleida, 2008, pp.41-46.

[8] VALERO, M.; DIAZ DE CERIORIPALDA, L. "Autoevaluación y co-evaluación. Estrategias para fomentar la Evaluación Continuada". *I Congreso Español de Informática*. Granada, 2005.

[9] JIMENEZ, M.J.; FORCADA, S.; ALVAREZ, M.J.; NAVARRO, T.; SABATER, A.; RALLÓ, M. "Cuestionarios de autoevaluación a través de intranets docentes: una herramienta para el aprendizaje". *15º CUIEET Libro de resúmenes*. Valladolid, 2007, pp.1819-1829.