Docencia de Semáforos y Monitores con BACI: un ejemplo de aprendizaje guiado

V.Galiano,O.López,M.Martínez,H.Migallón,P.Piñol,D.Ubeda
Dpto. Física y Arquitectura de Computadores
Universidad Miguel Hernández
03202 Elche
e-mail: {vgaliano,otoniel,mmrach,hmigallon,pablop,ubeda}@umh.es

Resumen

En este artículo se pretende recoger la experiencia realizada en la asignatura 'Sistemas Operativos' de segundo curso de Ingeniería Técnica en Informática de Gestión, impartida en la Escuela Politécnica Superior de Orihuela de la Universidad Miguel Hernández. En concreto la experiencia se ha realizado sobre la parte del temario de la asignatura que cubre los conceptos de concurrencia, exclusión mutua y sincronización de procesos.

La utilización de las nuevas tecnologías no garantiza unos mejores resultados en el aprendizaje de los alumnos. Lo que sí desempeña un papel crucial en la obtención de buenos resultados es, especialmente, la forma en la que la tecnología es puesta en práctica y utilizada [9].

Con este ejemplo proponemos una metodología para la inclusión de las nuevas tecnologías en el aula cuyos objetivos principales son intentar garantizar un buen rendimiento en las prácticas de la asignatura así como obtener una mayor motivación del alumnado, una mejor comprensión de los conceptos teóricos y un uso planificado y guiado en el proceso de enseñanza-aprendizaje.

Para conseguir dichos objetivos se ha puesto en marcha, siguiendo la planificación de dicha metodología, el uso en el aula de una herramienta que simula el efecto de los semáforos, los monitores y la sincronización de procesos llamada BACI (Ben-Ari Concurrent Interpreter)[1].

1. Introducción

Informes recientes han puesto el énfasis en la importancia de las tecnologías digitales para dar respuesta a los antiguos y recientes problemas de la educación [8], pero no siempre tienen suficientemente en cuenta la compleja realidad que la circunda. De este modo es posible identificar dos grandes formas de concebir la utilización de estas tecnologías como factor de innovación educativa. La primera se caracteriza por centrar gran parte del interés y la atención en los sistemas informáticos. La segunda, por situar la problemática de la mejora de la educación en el conjunto complejo de factores que configuran las situaciones de enseñanza y el aprendizaje, que, o se transforman de forma paralela e interactiva, o inhiben la propia innovación.

Al inicio del curso se realizó una encuesta a los alumnos sobre los conocimientos previos necesarios para abordar sin dificultad ciertos aspectos prácticos de la asignatura.

Esta encuesta mostró un perfil en el alumnado con conocimientos demasiado superficiales de programación en C, tanto en entornos Unix-like (como puede ser Linux utilizando las normas POSIX) como en entornos Windows (usando el API Win32). También se descubrió que el porcentaje de alumnos que había instalado Linux en casa alguna vez era bajísimo y mucho menor el que lo utilizaba habitualmente. Adicionalmente, en las aulas disponibles para prácticas se encontraba instalado únicamente Windows. Además, cierta parte del alumnado estaba más interesado en

aprender Linux como simples usuarios, aunque avanzados, que en "perder tiempo" con la teoría de los Sistemas Operativos, ya que a su modo de ver "nunca" se iban a encontrar con la necesidad de poner en práctica dichos conocimientos, bien como diseñadores o como programadores de sistemas. Es decir, se percibió cierta desmotivación que podía impedir la asimilación de algunos conceptos complejos de la asignatura.

Nuestra intención inicial era desarrollar las prácticas de la asignatura en C con las normas POSIX y pensamos que dicha aproximación es recomendable para esta asignatura si el perfil del alumnado lo permite. A parte de las carencias técnicas, se observó cierta pasividad en los alumnos respecto a la participación en el aula y la motivación en general; este comportamiento provoca la falta de "feedback" por parte del alumno hacia el profesor lo que le sitúa en una posición en la que éste actúa meramente como transmisor de conceptos, pero sin la confianza de que se hayan asimilado los conocimientos pretendidos. Esta situación puede ocasionar que el docente reitere una y otra vez de diversas maneras los mismos conceptos, pudiendo incluso ser contraproducente en el sentido de que el alumno acabe confundido y/o aburrido. El "feedback" del alumno es necesario para el docente, pues gracias a él, el profesor detecta el nivel de aprendizaje que se produce en el aula y puede ajustar sus exposiciones. Aún más, se puede estimar el resultado de la evaluación y que no se produzcan diferencias entre las notas estimadas y las reales.

Se trata de una asignatura cuatrimestral, con un contenido denso en conceptos que deben quedar bien afianzados en el alumno. Normalmente las prácticas relacionadas con los conceptos teóricos intentan profundizar en éstos a la vez que generan experiencia en el alumno. Lo ideal es realizar las prácticas de esta parte de la asignatura utilizando los mecanismos que un sistema operativo real pone a disposición del programador de sistemas (véase [7][2][3][4][10]). Intuimos, no obstante, que siguiendo la aproximación docente basada en POSIX o Win32 seguramente no cumpliríamos con las expectativas cuantitativas y/o cualitativas que entendemos necesarias para estos temas. Supusimos que perderían más tiempo con las problemáticas típicas de la inexperiencia en el lenguaje de programación que en centrarse en la comprensión de dichos conceptos. Una vez dominado el lenguaje de programación con el que se desarrolla, se concentra la atención hacia la resolución del problema y no hacia la sintaxis de éste. Igualmente pensamos que para que los conceptos quedaran suficientemente claros, los ejercicios que deberían hacer en las prácticas debían tener un nivel de complejidad medio-alto y esto era necesario mantenerlo. Había que buscar una alternativa que asegurara, en cualquier caso, que el tiempo dedicado a llevarla a cabo fuese el mismo que el que se emplearía sin ella, pues el tiempo de cada bloque ya estaba fijado en la programación de la asignatura.

2. Presentación de la asignatura

La experiencia en concreto se ha realizado en la asignatura 'Sistemas Operativos'. Se trata de una asignatura cuatrimestral de 7,5 créditos distribuidos en 4,5 créditos teóricos y 3 créditos prácticos. Dicha asignatura se imparte en el primer cuatrimestre a razón de tres horas de teoría semanales y dos de prácticas. El programa de la asignatura se presenta a continuación desglosando el tema de Gestión de Procesos. Como se puede ver en el siguiente esquema, el temario es extenso, con lo que la experiencia debía garantizar no consumir más tiempo que el inicialmente programado a la parte en la que se aplica. El tiempo disponible para la teoría de esta parte era de 7,5 horas, distribuidas en cinco sesiones de hora y media, tal y como podemos ver en la figura 1.

1. Fundamentos de los sistemas operativos
2. Gestión de procesos
   - Introducción
   - Vida de un proceso
   - Estados de un proceso
   - Creación y terminación de procesos
   - Intercambio Estructuras de control

- Control de procesos
- Modos de ejecución del procesador
- Creación de procesos
- Cambio de proceso
- Cambio de contexto
- *Concurrencia-Sincronización*
- Planificación de procesos

3. Gestión de memoria
4. Gestión de ficheros
5. Gestión de entrada-salida

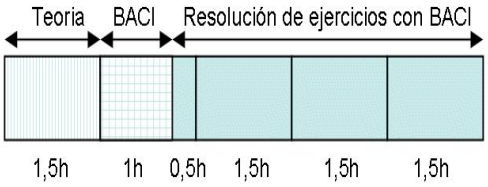

Figura 1: *Distribución de las clases*

3. Motivación metodológica

Nuestra propuesta de desarrollo de esta parte de la asignatura la basamos en las estrategias para desarrollar expectativas positivas de aprendizaje descritas por J.L Castillejo en [5]:

- *Es necesario que el alumno tenga continuadas experiencias de éxito.* Para ello el profesor plantea prolemas muy sencillos utilizando BACI, pero con errores fácilmente detectables por el alumno, lo que provoca pequeños debates en el aula para resolverlos. Como efecto añadido, el alumno va asimilando la sintaxis de BACI.

- *En el diseño de instrucción deben aparecer con especificidad y claridad los requisitos para lograr los objetivos propuestos.* Puesto que el objetivo era que realizaran prácticas complejas para los conceptos de exclusión mutua y sincronización, se presentaron, tras finalizar la exposición teórica de dichos conceptos, las prácticas obligatorias que debían realizar. Una vez expuestos los objetivos, el alumno percibe un alto grado de dificultad en dichas prácticas. Es entonces cuando le contamos que logrará realizarlas sin tanta dificultad, pues se le va a presentar durante algunas sesiones teóricas problemas en los que se resuelven diversos aspectos que deben utilizar para solucionar dichas prácticas.

- *Se eleva el nivel de expectativas en la medida que se permite un autocontrol de los resultados y se usan refuerzos de atribución que permitan conectar el éxito con su habilidad y esfuerzo personal.* Durante las sesiones teóricas en las que se presentan ejercicios a resolver se crean grupos de alumnos que deben solucionar dichos ejercicios; de esta forma se fomenta además de la competitividad, la interrelación grupal en el aula con el fin de conseguir una mayor participación, puesto que el refuerzo del grupo es mayor que la inhibición propia de un solo individuo. Mediante refuerzos de atribución según se van resolviendo los ejercicios de dificultad creciente, conseguimos que el grupo perciba la sensación de éxito derivada de su habilidad y esfuerzo, afrontando así con mayor ánimo y seguridad las prácticas.

Para potenciar las estrategias expuestas anteriormente, nos hemos basado en algunos elementos didácticos que sugieren C. Coll y otros y, J. Gimeno en [6][11]. Estos autores argumentan que el diseño y la planificación de la enseñanza debería prestar atención simultáneamente a cuatro dimensiones:

- *Los contenidos a enseñar*

  Son los contenidos propios del temario de esta parte de la asignatura como son la exclusión mutua y sincronización de procesos usando semáforos y monitores entre otros mecanismos.

- *Los métodos de enseñanza*

  Basándonos en las tres estrategias metodológicas descritas anteriormente, hemos

puesto en práctica la planificación metodológica que se detalla en el próximo apartado, haciendo uso de herramientas tecnológicas en el aula para potenciar el proceso de enseñanza-aprendizaje.

- *La secuenciación de los contenidos*

  La secuenciación de los contenidos se desarrolla en la metodología propuesta introduciendo primeramente ejemplos sencillos e incrementando posteriormente el nivel de dificultad y contenidos progresivamente.

- *La organización social de las actividades de aprendizaje.*

  Se ha fomentado la participación en grupos dentro del aula, resolviendo los ejercicios planteados con el fin de aumentar la motivación y la participación.

4. Planificación metodológica

Para plasmar las estrategias metodológicas comentadas, se ha realizado la siguiente planificación que se muestra gráficamente en la figura 2:

Presentación teórica de semáforos, semáforos binarios y monitores.

Se realiza la presentación teórica de los conceptos de concurrencia, exclusión mutua, sincronización, semáforos, semáforos binarios y monitores. Esta exposición teórica se realiza de la forma más concisa y clara posible, introduciendo algún ejemplo, pero sin dedicar excesivo esfuerzo en que el alumno sea capaz, con dichas exposiciones, de resolver cualquier problema. Esto se realizó en la primera sesión (véase figura 1).

Presentación de las prácticas obligatorias a realizar en el laboratorio.

Se pone en conocimiento de los alumnos los enunciados de las prácticas obligatorias que deben realizar en el laboratorio, cuya complejidad es media-alta. Entendidos los enunciados, se pasa una encuesta a los alumnos cuyo objetivo es establecer un nivel de referencia para poder medir de alguna forma las mejoras obtenidas por la aplicación de la metodología: Las preguntas de la encuesta fueron:

Figura 2: *Esquema metodológico*

- Evalúa dificultad de la práctica de 1 a 5
- ¿Cuántas horas estimas que necesitas para realizar cada práctica?

  a. Menos de 4 b. Entre 4 y 6 c. Más de 6
- ¿Cuánta ayuda crees que necesitarás del profesor?

  a. Mucha b. Bastante c. Poca d. Nada
- ¿Crees que te resultará difícil encontrar los errores en el programa?
- ¿Crees que los conceptos de concurrencia, semáforos y monitores son complejos de adquirir sin implementar programas?

Tras evaluar los resultados de la encuesta, vimos que la mayoría de los alumnos pensaban que les resultaría muy difícil realizar la práctica, tal y como habíamos previsto. La presentación de las prácticas y la encuesta se realizaron al comienzo de la segunda sesión (véase figura 1).

Presentación de BACI

Se hace una presentación del entorno de programación concurrente gráfico JBACI aprovechando que se muestra el primer ejemplo de programación concurrente, de manera que se les explica los menús y las diferentes opciones del programa. Esta parte corresponde a la segunda sesión.

Presentación y resolución de los problemas

En esta parte se presentan problemas resueltos con BACI en el aula de teoría mediante el uso de un cañón conectado al ordenador donde el profesor utiliza el simulador y transparencias para ir explicando los conceptos teóricos. El hecho de utilizar el lenguaje BACI en clase de teoría, consigue que el alumno vaya familiarizándose con su sintaxis. También se consigue que el alumno concrete los conceptos abstractos en algo más tangible y pierda el miedo inicial a esta parte de la asignatura. Se empiezan las simulaciones con problemas muy sencillos implementados con semáforos, semáforos binarios y monitores cuyo único objetivo es afianzar los conocimientos teóricos y llamar la atención del alumno para conseguir que piense que no es tan difícil como parecía a priori y que se sienta capaz de ir participando en los siguientes ejercicios planteados. Para cada problema presentado se discute en clase el enunciado y una vez entendido el problema se presenta el código en BACI. En unos casos este código corresponde a la solución y en otros se presenta una solución incompleta o con errores que se resuelven "en directo", preferiblemente como consecuencia de un debate provocado acerca de cómo solucionarlo.

Planteamiento de la práctica propuesta

Cuando el nivel de dificultad de los problemas que se estén presentando sea el apropiado y, cuando el momento lo requiera, se presenta una práctica propuesta para subir nota asignable a un único alumno. El momento en que se presentan las prácticas propuestas variará. Unas veces tras la exposición de un problema típico que se puede modificar fácilmente para que se convierta en una práctica propuesta y otras cuando no se lo esperan para llamar la atención del alumno. Adicionalmente, cuando se termine el tiempo asignado a esta parte de la asignatura se presentan suficientes prácticas propuestas, pero sólo para un 30 % de los alumnos que habitualmente asisten a clase. De esta forma se fomenta la competitividad.

Finalización del ciclo

Cuando el tiempo destinado a esta parte del temario se ha agotado y, tras la realización de las prácticas, es el momento de volver a realizar la encuesta al alumnado con el objetivo de poder medir las diferencias respecto a la encuesta inicial y analizar sus resultados.

- Evalúa la dificultad de la práctica de 1 a 5
- ¿Cuántas horas te ha llevado realizar la práctica?

  a. Menos de 4 b. Entre 4 y 6 c. Más de 6
- ¿Cuánta ayuda has necesitado del profesor?

  a. Mucha b. Bastante c. Poca d. Nada
- ¿Te ha resultado difícil encontrar los errores en el programa con ayuda del simulador BACI?

  a. Mucho b. Bastante c. Poco d. Nada
- ¿Has visto con más claridad los conceptos de concurrencia, semáforos y monitores al implementarlos y probarlos con el simulador BACI?

Una vez evaluado el cuestionario, la mayoría de los alumnos coinciden en que les ha resultado menos difícil de lo que inicialmente habían supuesto. También han respondido de forma satisfactoria en cuanto al uso del simulador BACI.

5. BACI

5.1. ¿Qué es BACI?

Los conceptos tales como sección crítica, concurrencia y las técnicas de sincronización entre procesos son temas importantes en informática. Además, debido al auge de la computación paralela y distribuida, entender la concurrencia y sincronización de procesos se hace más

necesario que nunca para los alumnos. En general, para obtener un conocimiento robusto de los conceptos teóricos, es aconsejable cierta dosis de experimentación. En concreto, para asimilar completamente los conceptos mencionados, es necesario experimentar con la programación concurrente. BACI es una opción para obtener esta experiencia con programación concurrente, que facilita el entendimiento de los conceptos de sección crítica, concurrencia y sincronización. BACI viene de Ben-Ari Concurrent Interpreter. Se trata de un compilador de lenguaje $C--$, un dialecto restringido de lenguaje C++, que genera código objeto interpretable (PCODE). El compilador y el intérprete fueron originariamente procedimientos en un programa escrito por M. Ben-Ari, basados en el compilador original de Pascal de Niklaus Wirth. Ben-Ari tomó el lenguaje Pascal-S y le añadió construcciones de programación concurrente tales como la construcción *cobegin... coend*y la variable tipo semáforo con operaciones *wait y signal*. BACI simula la ejecución de procesos concurrentes y soporta las siguientes técnicas de sincronización: semáforos enteros, semáforos binarios y monitores.

5.2. ¿Por qué BACI?

Como hemos comentado, el nivel de programación en lenguaje C de los alumnos era suficientemente bajo como para cuestionarnos el uso de las normas POSIX en las prácticas de la asignatura, ya que pensamos que perderían más tiempo resolviendo problemas relacionados con la inexperiencia en el lenguaje, que centrándose en los contenidos. Por este motivo y los siguientes decidimos utilizar el simulador BACI:

- Simplicidad en la sintaxis de BACI. Un subconjunto muy limitado de C++
- Entorno multiplataforma. Permite que cualquier alumno pueda instalarlo fácilmente en casa, independientemente del sistema operativo de que disponga, pues existen las versiones en JAVA, linux/unix y ms-dos.
- Se trata de un software de distribución gratuita y se distribuye con el código fuente.
- El entorno JBACI utilizado es un entorno gráfico muy sencillo de utilizar y permite la ejecución paso a paso, pudiendo realizar operaciones de depuración y obtener el valor de las variables tanto globales como las internas de los semáforos y monitores. Esto no es sencillo de realizar en una aproximación basada en POSIX para un alumnado sin experiencia.
- BACI actualmente es una herramienta muy extendida y usada en otras universidades para la docencia de esta parte de la asignatura [1].

5.3. Ejemplo de problema simulado en clase

Para poner un ejemplo de cómo se han presentado los problemas a los alumnos, se ha capturado una imagen de JBACI. JBACI es un entorno gráfico de desarrollo realizado en JAVA que utiliza BACI en su núcleo pero que ofrece un interfaz más completo y cómodo para el usuario. En la página web de BACI [1] se puede acceder a este entorno.

En la figura 3 se muestra una captura de la ventana de Edición del entorno de desarrollo JBACI, que es el que se ha usado en clase para mostrar los problemas. El código de la imagen corresponde al primer problema propuesto, donde se utilizan dos procesos cooperantes para conseguir un objetivo, sumar 20, donde cada proceso aporta 10 a la suma total. Se aprovecha el ejemplo para presentar la forma de inicializar en BACI los *arrays*, la sintaxis del bucle *for*, la sentencia *cobegin*, cómo imprimir en la consola con *cout* y cómo obtener el identificador de un proceso con la función *wich-proc()*. Pero sobre todo se llama la atención de los alumnos en cómo se puede dividir una tarea en la ejecución de procesos cooperantes para realizarla. En el ejercicio se pretende demostrar cómo, si no se protege mediante exclusión mutua la variable *total*, que es utilizada por los procesos cooperantes,

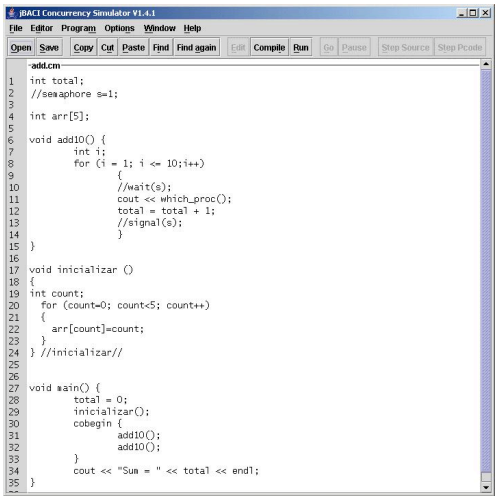

Figura 3: *Editor código BACI*

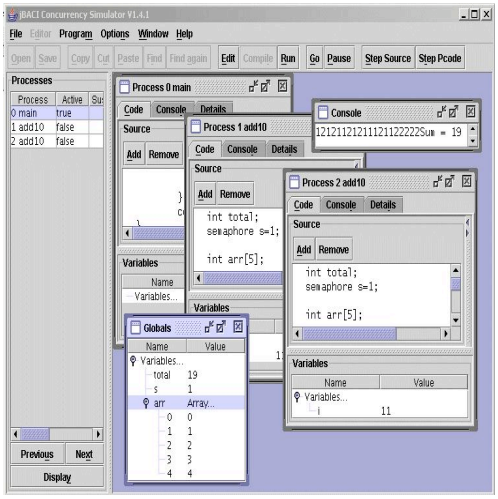

Figura 4: *Pantalla ejecución BACI*

se pueden producir resultados erróneos por no garantizar la exclusión mutua en la lectura y actualización de la variable *total*, es decir, la instrucción $total = total + 1$ debería ejecutarse sin interrupción. En la imagen aparecen comentadas las sentencias de declaración del semáforo que se usa para establecer la sección crítica y las sentencias *wait y signal* que la implementan. En el ejemplo que se muestra a los alumnos, estas instrucciones comentadas, no aparecen. Se ejecuta el programa varias veces y se observa que cada vez da un resultado distinto y que normalmente no se obtiene el resultado esperado, 20.

En la figura 4 se observan varias pantallas del entorno JBACI, donde podemos ver el proceso principal *main* y los dos procesos cooperantes *Process 1 add10* y *Process 2 add10*.

Cada una de estas ventanas tiene el código completo de los procesos *main* y *add10*. Este código se puede ejecutar sin pausa o paso a paso y observar sus efectos. En la ejecución paso a paso se puede observar cómo el código que se ejecuta es de uno u otro proceso, además se puede observar en la ventana de la consola cómo se van mostrando las impresiones y en la ventana de variables globales los valores de las variables y el valor interno de los semáforos conforme se ejecuta el código. Cuando se produce un cambio de proceso se puede observar este hecho en la lista de procesos(a la izquierda de la imagen) indicando en cada momento qué proceso está activo. Cuando se ha explicado el entorno de desarrollo, se forman grupos de trabajo en clase y se propone que sólo pueda dar la solución el portavoz del grupo. Se entrega el código en papel a cada grupo (a la vez que se sigue mostrando en pantalla) y se propone que cada grupo busque una solución con lo que se ha visto en teoría. La deliberación para este ejemplo es muy corta y enseguida, o incluso antes de terminar de repartir el código, algún grupo ya había aportado la solución correcta. Esta forma de hacer se repite con todos los problemas propuestos y, como hemos indicado, conforme la complejidad aumenta, el tiempo de deliberación se incrementa igualmente. El profesor va dando pistas a todos los grupos y estos mantienen la tensión de ser ellos los primeros que den con la solución. Se han comentado anteriormente los beneficios de utilizar esta técnica de motivación.

6. Problemas encontrados

No podemos olvidar que, tras la introducción teórica, el alumno se encuentra con muchos conceptos complejos e interrelacionados entre sí de golpe. Es pues en este momento cuando el alumno siente la necesidad de ver cómo estos conceptos abstractos que acaba de ver se pueden implementar y de qué manera tener algunos ejemplos al respecto que le clarifiquen dichos conceptos.

Para ello es necesario, obviamente, tener preparados todos los programas escritos, compilados, ausentes de errores y prevista la estrategia en la exposición para hacerla de forma que se haga hincapié en los conceptos más importantes de la teoría, así como su secuenciación. Pero también queremos evitar que el alumno se abrume con la presentación de código de ejemplos complejos, ya que simultáneamente tiene que entender el problema y seguir con claridad el código fuente que lo resuelve (su sintaxis y el enfoque en la resolución del problema).

7. Conclusiones

Nuestra experiencia nos ha proporcionado una gran satisfacción al ver que el grado de asimilación de los conceptos presentados ha sido mayor de lo esperado. Su aplicación tuvo un efecto positivo, incrementando la participación y atención del alumno y fomentó la competitividad.

Como principales logros obtenidos podemos enumerar que:

- En general el alumnado ha perdido el "miedo" a la participación en clase.
- Los alumnos han solicitado bastantes más ejercicios propuestos para subir nota que los que teníamos preparados. Es más, ha habido alumnos que han solicitado por propia iniciativa y sin beneficio adicional en la nota, más de una práctica propuesta porque según ellos "programar los semáforos nos ha ayudado mucho a entenderlos, por eso queremos también que nos mande una práctica de monitores para terminar de entenderlos".
- Se ha mejorado la comprensión de la teoría presentada.
- Los alumnos llegan a las prácticas más preparados y aprovechan mejor el tiempo de la práctica.
- Se han obtenido mejores resultados que en otros temas de la asignatura.

Haber conseguido esta actitud en el alumnado ha hecho que el esfuerzo empleado en hacer viable nuestra idea haya merecido la pena.

Como conclusión final queremos remarcar que esta metodología podría ser aplicada a la docencia de cualquier otro área siempre que se disponga de la herramienta tecnológica necesaria.

Referencias

[1] *Página principal de BACI: http* : $//www.mines.edu/fs_home/tcamp/baci/$.

[2] J. Carretero, F. Garcia, P.M. Anasagasti, and F. Pérez. *Sistemas Operativos: una visión ampliada.* MC Graw Hill, 2001.

[3] J. Carretero, F. Garcia, and F. Pérez. *Prácticas de Sistemas Operativos: de la base al diseño.* MC Graw Hill, 2002.

[4] J. Carretero, F. Garcia, and F. Pérez. *Problemas de Sistemas Operativos: de la base al diseño.* MC Graw Hill, 2003.

[5] J.L Castillejo. *Pedagogía Tecnológica.* Ceac, 1987.

[6] C. Coll, J. Palacios, and A. Marchesi. *Desarrollo psicológico y educación.* Alianza, 1991.

[7] F.M. Márquez. *Unix : Programación avanzada.* ra-ma, 1996.

[8] OCDE. *Learning to Change Embracing: ICT at Schools.* 2001.

[9] P. Rivière. Los negocios del multimedia en la escuela. *Le monde Diplomatique*, pages 27–28, 04 1998.

[10] Kay A. Robbins and S. Robbins. *Unix: Programación práctica. Guía para la concurrencia, la comunicación y los multihilos.* Prentice Hall Hispanoamericana, 1997.

[11] J. Gimeno Sacristán. *Autoconcepto, Sociabilidad y Rendimiento escolar.* MEC, 1976.