Evaluación continua con CAP, un Corrector Automático de tareas de Programación

Óscar Sapena, Mabel Galiano, Natividad Prieto, Marisa Llorens

Departamento de Sistemas Informáticos y Computación
{osapena,mgaliano,nprieto,mllorens}@dsic.upv.es

***Resumen*—El objetivo de este artículo es presentar un recurso TIC de corrección automática para su uso en el seguimiento y la evaluación de asignaturas de Programación en Java de los primeros cursos del Grado de Informática. Las principales ventajas que presenta con respecto a otros ya existentes son: a) guía al alumno durante la resolución de tareas de programación y b) permite al profesor afrontar con objetividad y eficacia el seguimiento y la evaluación del alumnado. Se proporcionan también resultados sobre su utilización en el curso 2011/12, así como sobre el grado de satisfacción de alumnos y profesores.**

***Palabras Claves*— Corrección automática, evaluación continua, Programación en Java, recurso TIC de aprendizaje.**

I. INTRODUCCIÓN

En este artículo se presenta CAP, un Corrector Automático de Programas, como una herramienta básica para hacer sostenibles las tareas de evaluación continua y seguimiento de tres asignaturas de programación de los primeros cursos del Grado de Informática. Este corrector ha permitido poner en práctica algunas de las ideas que, tanto con Bolonia como sin ella, los autores de este trabajo tenían sobre su labor docente y que podrían resumirse como sigue:

- La mejor forma de aprender a programar es programando.
- El alumno debe convertirse en el centro del proceso de enseñanza-aprendizaje. El profesor, si quiere asumir su papel en este nuevo esquema, debe aplicar nuevas estrategias para un seguimiento y evaluación objetivos y eficaces de sus alumnos.

Al finalizar el curso pasado, los autores ya habían experimentado con diversas técnicas que, en mayor o menor medida, implementaban estas ideas en el marco de Bolonia [10]; concluían entonces que era imprescindible el uso de las tecnologías de la información en los procesos de tutorización y evaluación del alumnado, si bien era necesario analizar y elegir entre el amplio abanico de metodologías docentes aquéllas que no sólo fueran efectivas sino que además pudieran ser usadas con confianza tanto por profesores como por alumnos. Al final de este curso, los autores creen haber encontrado en CAP una herramienta para sustentar esta conclusión.

En la sección II se describen el desarrollo y los usos de CAP y, en las secciones III y IV, se aportan y analizan una serie de datos para justificar el interés de esta herramienta.

II. ESTRATEGIAS DE INNOVACIÓN CON CAP

El corrector CAP se presenta como una herramienta dual, pues pretende servir tanto al alumno como al profesor en un proceso asimismo dual como el de la enseñanza-aprendizaje. Por ello, tanto a nivel de diseño como de uso, se despliegan dos ventanas:

- La del administrador, donde el profesor diseña actividades para la consecución de los objetivos formativos de la asignatura y la evaluación de su enseñanza-aprendizaje; por motivos obvios, el alumno no tiene acceso a esta ventana.
- La del corrector propiamente dicho, a la que accede el alumno para realizar las tareas de programación propuestas, bajo la tutela subyacente del profesor que las ha diseñado.

En lo que sigue se explican con mayor detalle todos los aspectos que caracterizan esta dualidad de CAP: los principios de funcionamiento del corrector automático, cómo se han introducido y aplicado las estrategias de innovación por parte del profesor y, finalmente, las ventajas que proporciona su uso a profesores y alumnos.

A. Principios de funcionamiento del corrector

Al igual que la mayoría de correctores automáticos existentes [4, 7, 12, 13], el funcionamiento de la herramienta CAP sigue, en líneas generales, el esquema que se describe en la Fig. 1.

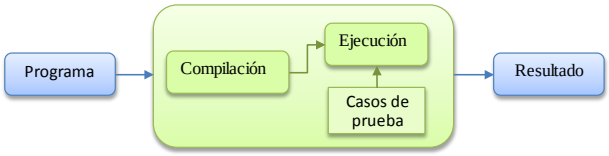

Figura 1. Funcionamiento básico de un corrector automático

El sistema corrector compila el programa y lo ejecuta, para ponerlo a prueba con unos pocos juegos de datos, comparando el resultado obtenido con el previsto [3, 5, 8, 11]. Ahora bien, CAP se diferencia de estos otros correctores en dos aspectos importantes, que los autores consideran los dos grandes inconvenientes de estos. A saber:

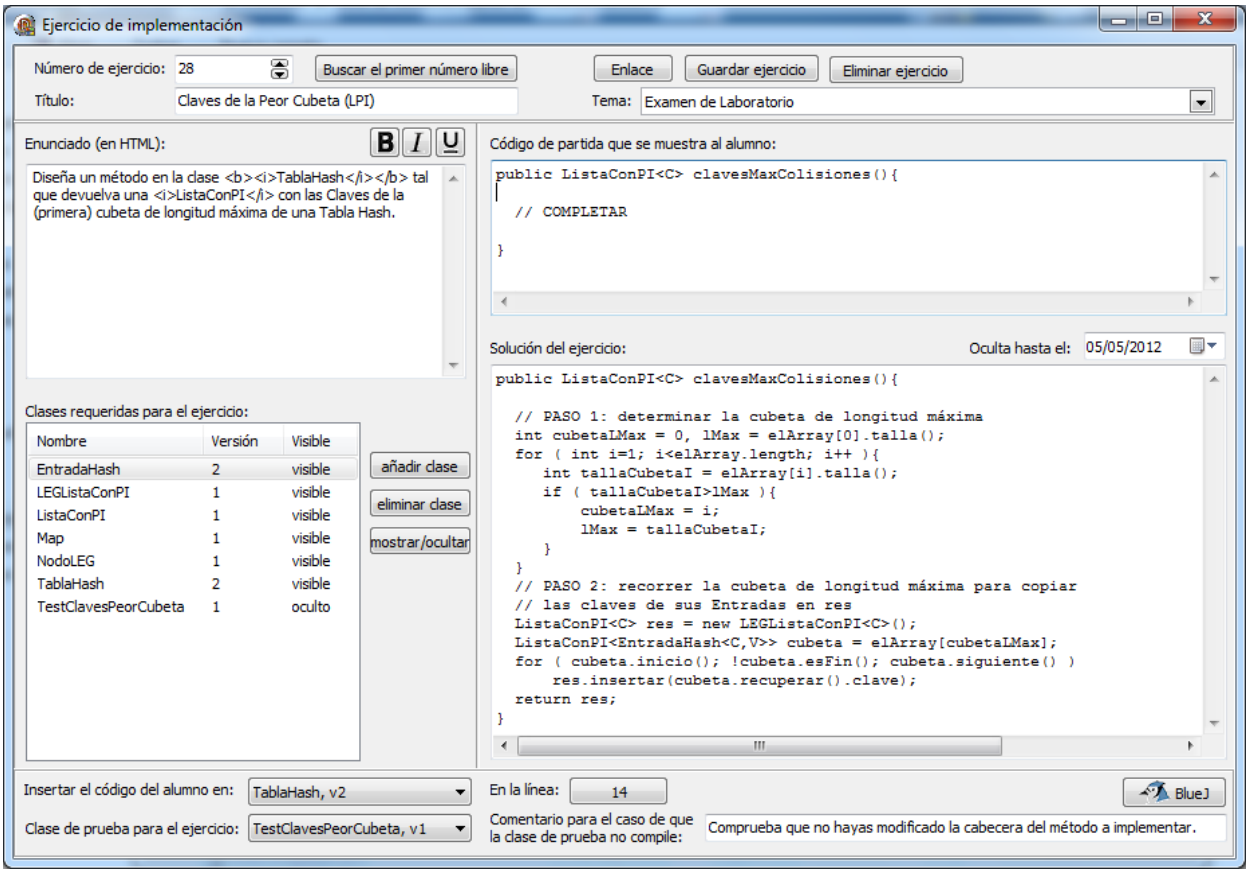

Figura 2. Elementos para el diseño de una tarea en CAP

- No impone una disposición fija para la definición de los problemas, que obligue a que los programas a desarrollar tengan que recibir unos datos de entrada y generar una salida en un formato muy determinado.
- No es un mero oráculo que sólo indica si la solución propuesta es o no correcta; al contrario, proporciona al estudiante una retroalimentación modulada, i.e. que puede variar a criterio del profesor que diseña un ejercicio en función del uso que va a hacer de este. Así, si se trata de una actividad sumativa en la que se desea un bajo nivel de tutela, la salida del corrector puede ser la habitual de un oráculo (correcto/no correcto). En cambio, si se trata de una actividad formativa, el corrector puede desplegar una gran variedad de comentarios y sugerencias que guíen al alumno en la consecución del objetivo.

CAP posee estas dos características porque ha sido desarrollado como un applet Java. Ello no solo facilita su integración en cualquier sitio web sino que, a diferencia de otros lenguajes como el C o C++, le permite hacer uso del mecanismo Java *reflection* [9], que permite obtener información sobre clases y objetos durante la ejecución de un programa. Así, con CAP, se puede corregir cualquier tipo de programa, clase o método (completo o parcialmente implementado) desarrollado en Java, aunque no requiera ningún dato de entrada o no genere ningún tipo de salida.

Otras características importantes que presenta el corrector CAP son las siguientes: maneja una base de datos MySQL en la que se almacena tanto la biblioteca de tareas de programación como las estadísticas de utilización y los resultados obtenidos por los alumnos; comparte el sistema de autenticación de la intranet de la UPV, lo que le permite identificar directamente al alumno que realiza el ejercicio, aunque también puede utilizarse sin que el usuario se haya identificado en la plataforma ("Usuario invitado") y, por tanto, sin guardar los resultados obtenidos.

B. Rol del profesor: diseño de tareas y análisis de resultados

El cometido principal del profesor es elegir aquellos ejercicios en los que se plasman de forma significativa los objetivos concretos de un tema o una práctica, y generar a partir de ellos un conjunto de casos de prueba que permitan señalar y, en su caso, ayudar a solventar las dudas y carencias que el alumno pueda tener sobre dichos objetivos, tanto si es consciente de ello como si no.

El profesor debe ponerse en el papel de alumno y, en base a su experiencia, tratar de prever los errores que el alumno puede cometer, cuantificarlos para que la respuesta que reciba el alumno no sólo sea bien o mal sino que le sirva como estímulo para mejorar por si mismo su solución.

En concreto, el diseño de una tarea, tal como se muestra en la Fig. 2, requiere rellenar una serie de ítems (título, tema y enunciado de la tarea, entre otros) y, principalmente, la generación de una serie de clases (visibles o no para el alumno) necesarias para la resolución del ejercicio; entre ellas destaca el test que deberá pasar el código del alumno cuando

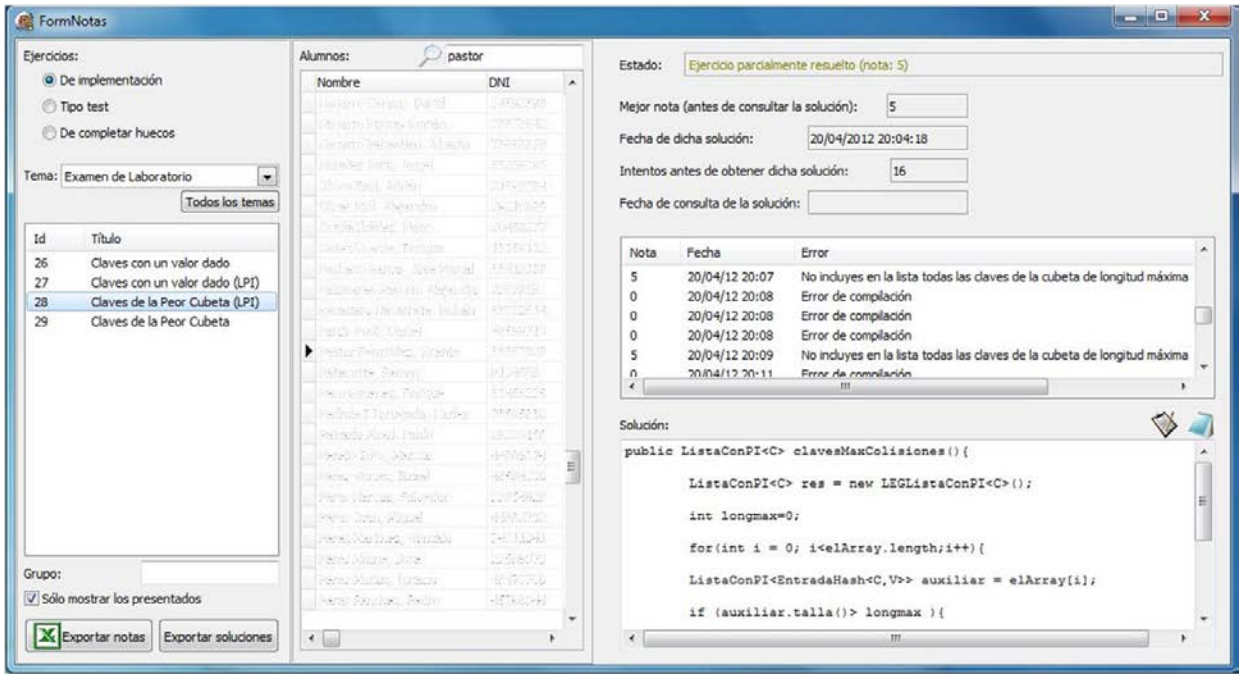

Figura 3. Elementos para evaluar una tarea en CAP

se ubique en la línea y clase indicadas por el profesor. Además, este puede añadir una solución oficial al problema y hacerla visible a partir de una fecha determinada.

En la Fig. 3 se observan los elementos que permiten al profesor el seguimiento y evaluación de la resolución de la tarea por parte de los alumnos. Concretamente, puede consultar la siguiente información de un alumno o de un grupo:

- Número de intentos realizados para resolver el problema planteado.
- Fecha en la que ha consultado la solución oficial del problema, en caso de que lo haya hecho.
- Calificación, fecha y código de la mejor solución obtenida por el alumno.
- Histórico de las notas y de los errores cometidos por el alumno en cada uno de los intentos.

Subrayar finalmente que el profesor puede obtener estadísticas sobre las tareas planteadas mediante la herramienta: número de alumnos que han participado, calificación media, lista de errores más frecuentes, número medio de intentos para encontrar la solución, etc. Constituyen por ello una valiosa fuente de realimentación para el profesor que, en base a ellas, puede seleccionar con mejor criterio las tareas que propone y mejorar los casos de prueba que usará para su corrección.

C. Rol del alumno: aprendizaje autónomo y autoevaluación

En la Fig. 4 se muestra la ventana a través de la que el alumno interactúa con CAP. En su parte izquierda figuran: el nombre del estudiante si está identificado en PoliformaT, o “Usuario invitado” si no lo está; asignatura y tema en el que se sitúa el ejercicio; la lista de clases adicionales que el alumno puede consultar para resolver el ejercicio. La parte derecha del corrector se divide, a su vez, en otras tres: el enunciado del problema, un editor de código donde el alumno escribe la solución y un cuadro de texto que muestra los errores cometidos junto con la calificación obtenida. Concretamente, hay tres tipos de errores que el corrector puede detectar y valorar:

- Errores de compilación: ocurren cuando el código del alumno contiene errores sintácticos y son exactamente los mismos que proporciona la máquina Java en cualquier otro entorno. Un código que no puede compilarse no puede ejecutarse para su validación, por lo que recibe una calificación de cero.
- Errores en tiempo de ejecución: son errores que aparecen durante la ejecución del código del alumno. La gestión de excepciones de Java le permite al corrector identificar el origen del problema, dar pistas para su resolución y evaluarlo en consecuencia. Un caso especial son la aparición de cálculos infinitos, que son calificados con un cero ya que impiden aplicar los casos de prueba al código del alumno.
- Errores lógicos: ocurren cuando la solución del alumno genera un resultado inesperado, es decir, que no cumple las indicaciones del enunciado. Los casos de prueba pueden detectar en qué medida la solución proporcionada se desvía de la solución deseada y calificarla en consecuencia.

Tal y como puede observarse en la Fig. 4, los mensajes de error que muestra CAP a la hora de validar el código le ofrecen al alumno una guía eficaz para poder solucionarlos, tanto por la información adicional que ofrece para los errores de ejecución, sobre los que la máquina Java daría una información escueta, como para los errores lógicos que no detectaría la máquina Java y que un oráculo se limitaría a calificar de incorrectos. El alumno, además, puede guardar su última solución, por ejemplo para continuar en otro momento, y, en su caso, consultar la solución oficial del problema si no ha logrado resolverlo correctamente.

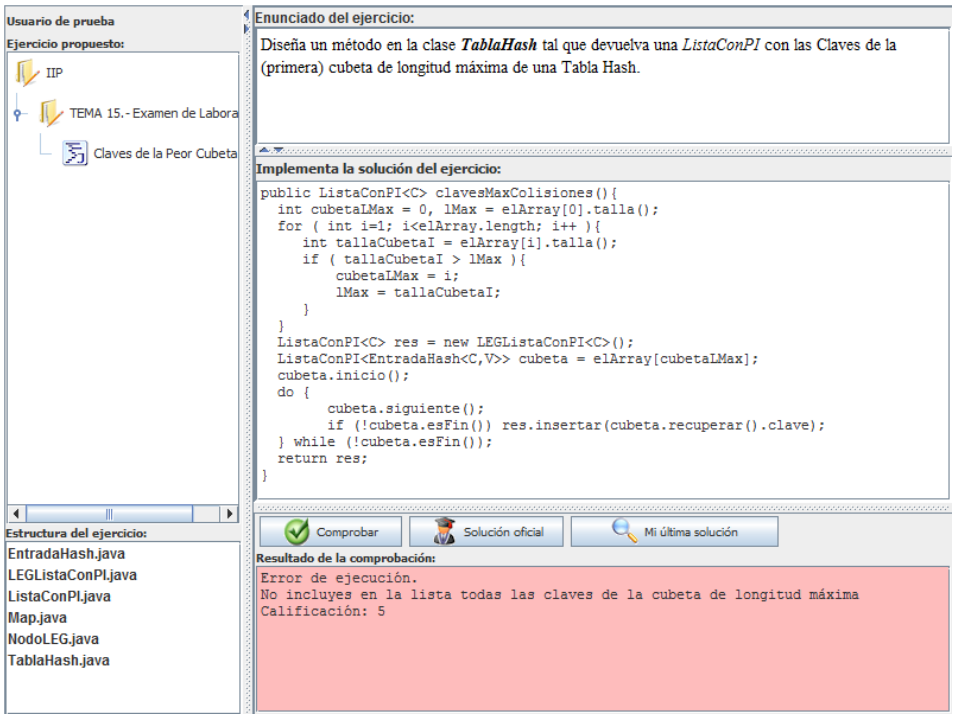

Figura 4. Ventana de CAP para el alumno

III. Resultados y Discusión

Durante el curso 2011/12 se le han suministrado al corrector del orden de 50 tareas (13 para evaluación de prácticas, algunas con sus versiones en valenciano e inglés, y 28 para actividades de seguimiento) de tres asignaturas de primer y segundo curso del Grado de Informática: Introducción a la Informática y la Programación, Programación y Estructuras de Datos y Algoritmos; al menos 630 alumnos han realizado 2 ejercicios utilizando CAP y 150 más de 10.

Un ejemplo significativo que pone de manifiesto la utilidad de CAP, así como la información que proporciona, es la tarea sobre el algoritmo de Dijkstra que se utilizó para evaluar a una parte de los alumnos de EDA: 89 alumnos durante hora y media realizaron un total de 1243 intentos (13,97 como media por alumno), obteniendo una nota media de 5,78. Sin tener en cuenta los 624 intentos que dieron error de compilación, el error más frecuente fue que añadían al resultado más vértices de los esperados. A la vista de estos datos, no es difícil deducir el esfuerzo que hubiera supuesto realizar esta misma evaluación sin utilizar el corrector, con métodos más tradicionales en los que los mismos profesores atienden y evalúan al mismo número de alumnos en el mismo tiempo. Piénsese también cuál hubiera sido el grado de objetividad que el profesor hubiera podido mantener a lo largo de este proceso.

Por otro lado, se ha diseñado una encuesta para conocer la opinión de los alumnos sobre el uso del corrector. Esta encuesta sólo se ha pasado en algunos de los grupos, obteniendo 106 respuestas, aproximadamente el 50% de los alumnos encuestados. En la Tabla I se muestran los resultados de la misma.

Tabla I. Resultados de la encuesta sobre CAP

Enunciado	De acuerdo
Creo que evaluar un ejercicio utilizando el corrector es más exigente que la evaluación del mismo en papel.	68%
El uso del corrector me ayuda a detectar y solventar algunas dudas y errores que en un ejercicio escrito igual no me surgirían.	79%
La realización de ejercicios con el corrector favorece mi participación en clase.	55%
Creo que el uso del corrector me puede ayudar a superar la asignatura con éxito.	72%
Valora el interés de disponer de una biblioteca de ejercicios a resolver para cada uno de los temas.	89%
Aunque no se utilizara como herramienta de evaluación, yo lo usaría para resolver los ejercicios que se proponen en clase y preparar el examen.	82%
Al margen de las limitaciones que presenta, valora la eficacia del corrector a la hora de aprender y preparar la asignatura.	73%
Al margen de las limitaciones que presenta, valora la eficacia del corrector como herramienta de evaluación.	59%

A la vista de los resultados anteriores, globalmente, se puede concluir que el alumno tiene una opinión positiva del uso del corrector aunque lo valoran mejor como herramienta de autoaprendizaje que de evaluación; en esta percepción

puede haber influido el resultado obtenido en la evaluación.

Cabe señalar también que más de un 80% de los alumnos consideran muy interesante poder disponer de una biblioteca de ejercicios a resolver para cada uno de los temas y que lo utilizarían aunque no influyera en su calificación. A pesar de que opinan que evaluar un ejercicio utilizando el corrector es más exigente que la evaluación del mismo en papel, más de un 70% creen que el uso del corrector les ayuda a detectar y solventar algunas dudas y errores que en un ejercicio escrito no les surgirían y, en definitiva, a superar la asignatura con éxito. Asimismo, sólo el 55% de los encuestados opinan que el corrector favorece su participación en clase, probablemente porque solo el 40% han utilizado el corrector presencialmente en clases de teoría.

Algunas de las sugerencias más interesantes que nos han proporcionado los alumnos son las siguientes:

- Mejorar las condiciones de acceso a la herramienta desde fuera de la Universidad.
- Refinar los mensajes de error que les sirven como guía a la hora de resolver un ejercicio.
- Mejorar el editor de código (resaltado de sintaxis, indentación, etc.)
- Posibilitar la realización de los ejercicios fuera de los plazos de entrega indicados por el profesor.

Además de coincidir con la necesidad de realizar estos cambios, los autores piensan que es necesario mejorar la seguridad de la herramienta para evitar copias durante las evaluaciones. En este sentido se ha pensado trabajar en dos líneas: monitorización del puesto de trabajo en actividades sumativas y uso de algún sistema de detección de plagio como JPlag [6]. También sería de gran utilidad integrar la herramienta en PoliformaT [2], la plataforma de formación de la UPV, para que los resultados queden reflejados automáticamente en el libro de calificaciones del alumno que esta proporciona.

IV. Conclusiones

Los autores tienen la sensación de haber conseguido implementar, y no solo escribir, lo que en [14] se define como un objetivo didáctico: "… *expresa con claridad lo que esperamos que el alumno haya aprendido al acabar el curso. Informa sobre el resultado o el cambio esperado en el alumno como consecuencia del proceso de enseñanza-aprendizaje (conoce lo que no conocía, entiende lo que no entendía, hace lo que no sabía hacer...)*". En otras palabras, con CAP se puede conseguir evaluar el proceso de enseñanza-aprendizaje o "*comparar lo deseado con lo realizado*" [1].

Agradecimiento

Los autores agradecen a los alumnos que han utilizado esta herramienta y a los que han contestado a la encuesta de satisfacción.

Referencias

[1] M.E Alfaro. Aspectos prácticos del proceso de programación y evaluación. Documentación Social. Nº 81. Madrid, 1990.

[2] J. Busquets, D. Roldán, S. Martínez y D. Del Blanco. PoliformaT: una estrategia para la formación on-line en la Educación Superior. Virtual Educa, 2006. http://ihm.ccadet.unam.mx/virtualeduca2006/pdf/177-DRM.pdf.

[3] C. Daly y J. Waldron. Introductory Programming, Problem Solving and Computer Assisted Assessment. Proceedings of the 6th CAA Conference, pp. 95 – 104, Loughborough, 2002.

[4] C. Daly. RoboProf and an introductory programming course. Proceedings of ITiCSE '99, pp. 155 – 158, ACM Press, 1999.

[5] J.A. Gómez. Portal Web de Autocorrección de Programas para Programación de Primer Curso. En II Jornadas Nacionales de Intercambio de Experiencias Piloto de Implantación de Metodologías ECTS: Aplicaciones prácticas de la Convergencia Europea, Badajoz, 2007. https://clocalprog.dsic.upv.es/autocorreccion/

[6] JPlag - Detecting Software Plagiarism. https://www.ipd.uni-karlsruhe.de/jplag/

[7] J. P. Leal y F. Silva. Mooshak: a Web-Based multi-site programming contest system. Software Practice and Experience 33, pp. 567 – 581, 2003.

[8] L. Llana, E. Martín y C. Pareja. Correctores automáticos de programas: Implantación realista en la docencia universitaria. Actas del II Seminario de Investigación en Tecnologías de la Información Aplicadas a la Educación (SITIAE 2008), pp. 31 – 50. Universidad Rey Juan Carlos, 2008.

[9] Oracle. The JavaTM Tutorials. Trail: The Reflection API, 2011. http://docs.oracle.com/javase/tutorial/reflect/index.html.

[10] N. Prieto, M. Llorens, G. Moltó, J.A. Gómez, M. Galiano y C. Herrero. Uso de Metodologías Activas en la Implantación de IIP en el Grado de Informática en la UPV. En XVII Jornadas de Enseñanza Universitaria de la Informática (JENUI 2011), pp. 53 – 60, 2011.

[11] J.C. Rodríguez, M. Díaz, Z. Hernández y J.D. González. Hacia la Evaluación Continua Automática de Prácticas de Programación. En XIII Jornadas de Enseñanza Universitaria de la Informática (JENUI 2007), pp. 179 – 186, 2007.

[12] J.C. Rodríguez, Z. Hernández, J.D. González y M. Díaz. Experiencia de uso y características del programa Gestión Automática de Prácticas (GAP). Actas Conferencia IADIS Ibero-Americana, pp. 35 – 42, 2005.

[13] G. Thorburn y G. Rowe. PASS: an Automated System for Program Assessment. Computers Education 29(4), pp. 195 – 206, 1997.

[14] M. Valero y M. Espona. Material del taller "Adaptación de asignaturas al EEES". Universitat Politècnica de València, 2010.