Actas de las Jenui, vol. 3. 2018.
Páginas: 327-334

Una metodología para la realización y evaluación efectiva de exámenes de programación usando el ordenador

Sandra Martín
Departamento de Comercio
IES Luis Buñuel
Alcorcón (Madrid)
smartinpena@educa.madrid.org

Mª Teresa González de Lena
Ana Belén Moreno
Ángel Sánchez
Jesús Sánchez-Oro
José F. Vélez
Escuela Técnica Superior de Ingeniería Informática
Universidad Rey Juan Carlos (URJC)
Móstoles (Madrid)
mariateresa.gonzalezdelena@urjc.es
belen.moreno@urjc.es
angel.sanchez@urjc.es
jesus.sanchezoro@urjc.es
jose.velez@urjc.es

Resumen

El uso de ordenadores durante los exámenes de programación es una demanda muy habitual entre los estudiantes universitarios. En este trabajo se presenta una metodología para evaluar a los estudiantes de un curso de programación, que incorpora como novedad principal el uso del ordenador, por parte de los estudiantes, durante el examen.
El trabajo comienza justificando las ventajas del uso del ordenador durante el examen y explicando los retos que introduce. Luego, se describen los diferentes procedimientos que componen la metodología propuesta: prácticas en casa, test unitarios, exámenes prácticos y exámenes tipo test. También se describen en este trabajo dos herramientas que se han desarrollado ad-hoc para complementar dicha metodología. La primera herramienta tiene por objetivo impedir el uso fraudulento del ordenador durante el examen. La segunda herramienta sirve para semiautomatizar la corrección de los exámenes.
Finalmente, en este trabajo se analizan los resultados obtenidos antes y después de la aplicación de la metodología propuesta. De este análisis se deriva que el uso del ordenador durante el examen parece ofrecer una calificación más objetiva de los estudiantes.

Abstract

The use of computers during programming exams is a very habitual demand of the computer grade students. In this paper, a methodology for programming courses is presented, whose main innovation consists of using computers during the exams.
This work begins justifying the advantages of using the computer during the exams and explaining the challenges that it introduces. Later, the procedures of the proposed methodology are described: unit tests, multiple choice questions, practical exams and home work exercises. We also describe two tools that had been developed to support these procedures. The first one avoids a fraudulent use of the computer during the exam. The second one is used to automatize the correction of the exams.
In this work, we analyze the results obtained before and after the application of the proposed methodology. From this analysis, we notice that the use of computers during the exams allow us to get a more objective qualification of the student.

Palabras clave

metodología, examen, ordenador, test unitarios, acceso a Internet.

1. Introducción

Usualmente a los estudiantes de programación se los instruye con contenidos teóricos mediante clases magistrales y libros. También se los capacita de manera práctica con ejercicios que deben resolver con el ordenador en el laboratorio o fuera de él. Según Ala-Mtka [1], mientras los ejercicios prácticos que realizan los alumnos durante el curso suelen hacer uso del ordenador, la forma más habitual de evaluar a los estudiantes de programación prescinde del ordenador.

Al ser las prácticas un trabajo que parcialmente se suele realizar fuera del laboratorio, el profesor puede desconfiar sobre la autoría de las mismas. El profesor puede llegar a pensar que la práctica la ha resuelto un compañero del alumno o incluso un profesional. Quizás es por ello que, las prácticas pueden formar parte de la nota o no, pero habitualmente el porcentaje más importante de la nota de un alumno corresponde a un examen con ejercicios prácticos y cuestiones teóricas que se realiza en papel al final del curso. Con este tipo de exámenes presenciales y en papel se resuelve el problema de la autoría. Sin embargo, la parte práctica de un examen de programación en papel tiene, a nuestro juicio, tres problemas importantes:

- En los exámenes prácticos sobre papel el estudiante no tiene acceso al ordenador, ni a otros medios que ha utilizado para aprender y para resolver los casos prácticos. Básicamente, el alumno siempre ha programado usando un ordenador. Luego, en su vida laboral, también usará el ordenador para resolver este tipo de problemas. Por ello, en los exámenes el alumno se encuentra en una situación no natural, respecto a lo que ha hecho en el pasado o hará en el futuro para resolver problemas similares.
- Como el estudiante no cuenta con herramientas para resolver problemas complejos, en el examen se simplifican en exceso los enunciados, haciéndolos a veces irreales y difíciles de entender. Si se dispusiese del ordenador para resolver el examen, los enunciados podrían ser más realistas, e incluso, incluir código de ejemplo.
- Finalmente, los exámenes manuscritos por los alumnos son difíciles de gestionar, manipular y comprender por el profesor. Esta dificultad tiene diversos motivos. Está el problema de su almacenamiento físico, de la posibilidad de pérdida de un examen y el hecho de que los alumnos cada vez están menos acostumbrados a escribir en papel. Por otro lado, el código de un programa cuando está manuscrito suele ser más difícil de entender que un texto más literario, debido a la falta de contexto inherente al código. Además, cuando se programa es habitual olvidar alguna línea o fragmento de código, y su inclusión en el papel por parte de el alumno (quizás en letra más pequeña en un margen) complica aún más la lectura. El uso del ordenador permitiría eliminar el problema del almacenamiento, la posibilidad de pérdida y facilitaría la comprensión del texto.

Hay varios estudios que nos animan a introducir el ordenador como herramienta en los exámenes de programación. Por ejemplo, C. Zilles et al. [11] señala que el 75% de los estudiantes encuestados prefieren realizar los exámenes utilizando ordenador. En un trabajo de J. Barros et al. [2] señalan que los exámenes en el laboratorio son percibidos como más justos por los estudiantes, mejorando su motivación. Por otro lado, R. Deloatch et al. [4] señalan que los estudiantes sienten menos ansiedad al realizar exámenes con el ordenador y de hecho lo prefieren.

Sin embargo, el uso del ordenador durante el examen no está exento de problemas. Si los estudiantes disponen de un ordenador durante el examen es posible que lo utilicen fraudulentamente para saltarse las normas del examen. Por ejemplo, si se prohíbe copiar código de los compañeros, un estudiante podría configurar fácilmente un sistema de almacenamiento en la nube para compartir los ficheros. Si se prohíbe realizar búsquedas en Internet, un estudiante podría aprovechar cuando el profesor no lo observe para conectarse. Estos problemas se resolverían si se pudiese eliminar la conexión a Internet durante el examen, pero dicha conexión resulta necesaria para descargar y entregar el examen, y también resulta necesaria en aulas en las que los programas a utilizar están virtualizados en la nube. También podrían resolverse con una gestión adecuada de la red local del aula del examen. Sin embargo, este enfoque puede ser imposible cuando la red es gestionada por un órgano externo.

El uso del ordenador durante los exámenes de programación ya ha sido propuesto en otros trabajos [6, 7, 11]. El uso de test unitarios [3] para mejorar el aprendizaje de los alumnos también ha sido propuesto antes [8, 10]. Sin embargo, en este trabajo incorporamos estos conceptos de una manera novedosa al propio examen de programación. Además, incorporamos herramientas para monitorizar las acciones del alumno durante el examen y para ayudar luego en la corrección. El enfoque propuesto creemos que es válido para aquellos casos en los que existen herramientas (como los test unitarios y los compiladores) y cuando los estudiantes han usado el ordenador durante su entrenamiento. Sin embargo, como señalan J. Sheard et al. [9], también creemos que no existe un único enfoque general para evaluar a los estudiantes de los cursos de programación.

En este artículo se presenta una metodología para evaluar a los estudiantes de un curso de programación. Dicha metodología tiene su base fundamental en permitir el uso del ordenador durante el examen y consta de diferentes herramientas y procedimientos. La metodología que se va a describir se ha utilizado con éxito durante tres cursos académicos en la asignatura de Estructuras de Datos Avanzadas de tercer curso del Grado en Ingeniería Informática de la Universidad Rey Juan Carlos de Madrid.

2. Metodología propuesta

En esta sección se describen los diferentes procedimientos de evaluación, y las herramientas que complementan estos procedimientos.

Los procedimientos de evaluación son:

- Un conjunto de prácticas que el alumno debe resolver por su cuenta fuera del aula.
- Un examen de tipo test que se realiza el último día de clase o el día del examen final. Dicho examen sirve para evaluar conceptos teóricos que no se evaluarían con solo un examen práctico.
- Un examen de tipo práctico que se realiza el día del examen final. En este examen se permite el uso del ordenador, del compilador y de la ayuda que este proporciona.
- Un proceso de evaluación semiautomatizado de las prácticas y de los exámenes prácticos.

Y las herramientas que complementan estos procedimientos son:

- Una aplicación, denominada *ExamWatcher*, que monitoriza las acciones realizadas por el ordenador del alumno.
- Un *script* que gestiona el proceso de corrección integrándolo con el gestor de contenidos de la universidad.

En las siguientes subsecciones se describen en detalle estos procedimientos y herramientas.

2.1. Las prácticas

Durante el curso, el alumno es requerido a resolver 4 prácticas de carácter obligatorio que se corresponden con los 4 grandes bloques de teoría de la asignatura. Cuando el profesor termina de explicar cada uno de estos bloques de teoría, realiza una sesión práctica para explicar el enunciado de la práctica correspondiente. Luego, al principio de cada clase contesta las cuestiones que tengan los alumnos al respecto.

Cada práctica consta de 3 o 4 ejercicios que deben resolverse con el ordenador. A su vez, cada ejercicio consta de:

1. El enunciado, compuesto de un texto, código de apoyo y quizás un diagrama.
2. Los test unitarios, que facilitan la verificación del funcionamiento del código desarrollado.
3. Las interfaces de las clases y los métodos públicos que el alumno debe implementar. Suministrar dichos elementos es importante para facilitar la ejecución automática de los test unitarios.

Una posible pregunta de las prácticas podría ser:

> "El estudiante debe implementar el método factorial, que recibe como entrada un número entero *n* mayor que cero y devuelve otro entero correspondiente al producto de todos los números entre *n* y 1."

En este caso, la interfaz en Java que se proporcionaría al alumno podría ser:

```
class MathUtils {
   /**
    * param nun: número al que se le calcula
    * el factorial
    * return: el factorial de nun
    */
  public static int factorial(int num) {
        //Escribe aquí tu código
  }
}
```

Finalmente, un ejemplo de un test unitario para dicha pregunta sería:

```
class MathUtilsTests {
  @test
  static int factorialTest() {
    assertEquals(factorial(5),120);
    assertEquals(factorial(1),1);
  }
}
```

2.2. El examen tipo test

A final de curso, los estudiantes son convocados para realizar un examen de tipo test. Este examen se utiliza para verificar que los alumnos han adquirido los conocimientos teóricos requeridos. En general, en este examen se prefieren las preguntas que requieren que el estudiante resuelva un pequeño problema para poder contestar adecuadamente, aunque también suele haber varias preguntas puramente conceptuales.

2.3. El examen práctico

El examen práctico consta de 3 ejercicios que deben resolverse usando el compilador. Al igual que en las prácticas, cada pregunta consta de: el enunciado y el código de apoyo, los test unitarios que clarifican los enunciados, y las interfaces de las clases y los métodos públicos que el alumno debe implementar.

Todos estos elementos se entregan a los alumnos en formato electrónico, menos los enunciados, que se entregan a los alumnos en papel para dificultar su acceso desde fuera del aula antes de finalizar el examen. Además, se crean dos variantes de los enunciados en papel, para que los compañeros examinados, a izquierda y derecha, tengan enunciados diferentes.

La Figura 1 muestra los subprocesos que componen el examen práctico. Al principio del examen los estudiantes descargan la aplicación *ExamWatcher*, permitiéndoles utilizar el ordenador del aula o su propio ordenador. *ExamWatcher* monitoriza la actividad del alumno durante el examen. Normalmente esta aplicación muestra un panel verde, que torna a rojo si el alumno utiliza el ordenador para comunicarse o para consultar documentación no autorizada. En la subsección 2.5 se explica en detalle esta herramienta.

Tras activar *ExamWatcher*, los alumnos descargan el código asociado al examen, desconectan el ordenador de la red y comienzan a resolver los enunciados.

Cuando finalizan el examen generan un fichero ZIP con el código desarrollado y utilizan el gestor de contenidos de la universidad para enviarlo al profesor.

Un aspecto importante a considerar es la ansiedad ante el examen. Como señalan Delir y Sheard [7], cualquier innovación puede introducir ansiedad entre los estudiantes. Por ello, es importante familiarizar al estudiante con las herramientas que se usarán en el examen en sesiones previas. En este caso hemos aprovechado las sesiones prácticas para introducir los test unitarios, la herramienta *ExamWatcher* y la comunicación con el gestor de contenidos de la universidad.

2.4. El proceso de evaluación

El profesor realiza un proceso de evaluación que consta de los siguientes pasos:

- Descarga del código de cada examen desde el gestor de contenidos de la universidad.

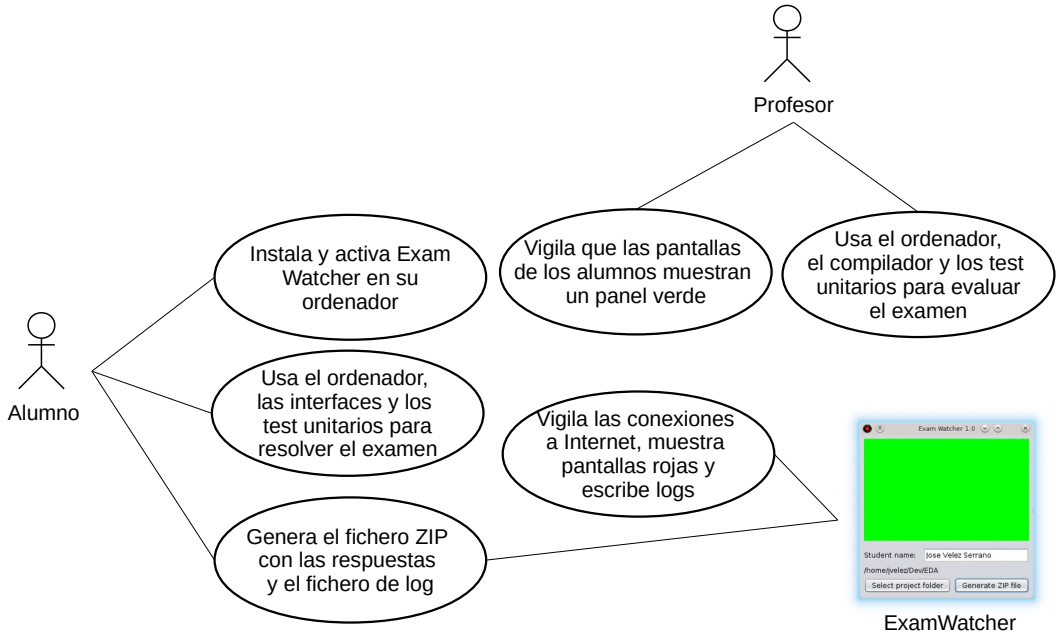


Figura 1: Diagrama de Casos de Uso UML de la metodología propuesta.

- Compilación del código de cada examen.
- Ejecución de los test unitarios asociados a cada pregunta.
- Inspección visual del código.
- Uso de una rúbrica predefinida para puntuar cada apartado del examen (ver Figura 2). Está rúbrica suele tener 3 o 4 categorías (de mal a bien) y un campo de texto para escribir algún comentario de ser necesario.

Este proceso de evaluación es gestionado mediante un *script* que guía al profesor de un paso a otro. Dicho *script* se detalla en la sección 2.6.

Pregunta	Mal	Regular	Bien	Comentario
Apartado 1.a	No hay respuesta o no tiene ningún sentido	El código tiene sentido pero no pasa los test	El código es correcto y pasa los test	No funciona porque recorres la ED al revés.
	0 puntos	0.5 puntos	1 punto	

Figura 2: Ejemplo de rúbrica de corrección. En este caso el alumno recibe 0.5 puntos por el apartado 1.a del examen debido a que no ejecuta los test.

2.5. La aplicación *ExamWatcher*

ExamWatcher[1] es una aplicación que se ha desarrollado para monitorizar las conexiones y las aplicaciones que se ejecutan en el ordenador huésped. La aplicación consiste en una ventana que muestra un panel verde o rojo y varios botones (ver Figura 3). Si el panel presenta color rojo (ver Figura 3a), el ordenador no debe usarse para realizar el examen, mientras que si el color es verde (ver Figura 3b), sí puede usarse.

Durante el examen, *ExamWatcher* realiza continuos test para garantizar que el ordenador cumple las reglas del examen. La comprobación se realiza probando la conexión a diferentes destinos de Internet. Por ejemplo, comprueba las páginas web de diversos servicios como: Dropbox, Mega, OneDrive o Google Drive. Si alguna de las comprobaciones tuviese éxito, el panel cambiaría a color rojo. Este proceso se describe en la Figura 4.

Si el estudiante corrige el problema, *ExamWatcher* pasará a mostrar una pantalla verde de nuevo, pero en un fichero de *log* quedará reflejada la infracción, para que el profesor pueda tomar las acciones pertinentes durante la corrección. Por ello, para evitar posteriores problemas, los estudiantes deben reportar al profesor cualquier incidencia con *ExamWatcher*.

Es muy importante garantizar la visibilidad del panel coloreado, ya que permite al docente comprobar durante el examen que el ordenador está siendo utilizando adecuadamente por el estudiante (ver Figura 5). Para ello, la ventana es modal, situándose siempre por encima de cualquier otra aplicación del escritorio.

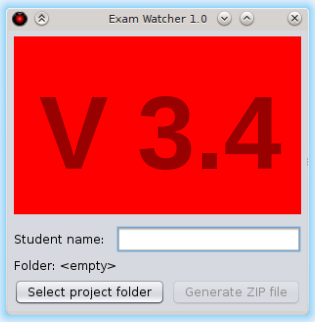
Para evitar que los estudiantes creen un una aplicación *ExamWatcher* falsa, los colores pueden ser configurados por el profesor antes de comenzar el examen y el número de versión puede ser cambiado. Además, el profesor también puede modificar un código *hash* interno que se usa para firmar los ficheros que se entregan. Si el estudiante usase una imitación del *Exa-*

[1]Descargable desde:
https://github.com/jfvelezserrano/ExamWatcher

mWatcher durante el examen, los ficheros generados al finalizar el examen estarían firmados con un código *hash* incorrecto y el fraude se detectaría.

Finalmente, *ExamWatcher* también dispone de un botón que permite generar el fichero ZIP que contiene todo el código que el alumno ha programado durante el examen, el fichero de *log* y un fichero de control firmado con el código *hash*. Una vez que el *ZIP* ha sido generado, el estudiante debe conectarse de nuevo a Internet para subir el fichero *ZIP* al gestor de contenidos de la universidad.

(a)



(b)

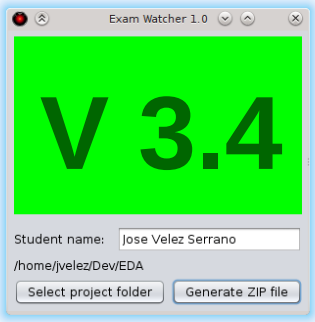


Figura 3: Interfaz gráfica de la aplicación *ExamWatcher*. El panel rojo (a) indica que el ordenador del estudiante está cometiendo una infracción; el panel verde (b) indica que todo parece correcto en su ordenador.

2.6. El script de gestión de la corrección

Para facilitar la labor de corrección de los exámenes y de las prácticas se ha desarrollado un *script* ejecutable que automatiza las partes más tediosas. Así, al comenzar la corrección de los exámenes se lanzan secuencialmente las siguiente acciones:

1. Comprobación automática de copia, mostrando el examen más parecido.
2. Inspección automática del *log*, mostrando las infracciones si las hubiese (mostrando la hora y el detalle de la infracción cometida.
3. Compilación automática del código del examen, ejecución automática de los test unitarios y presentación de resultados. Los test pueden ser los mismos que los suministrados a los estudiantes durante el examen, pero también pueden consistir en una versión más sofisticada.
4. Apertura automática de una ventana con el código del examen para que el profesor pueda realizar una inspección visual del examen.

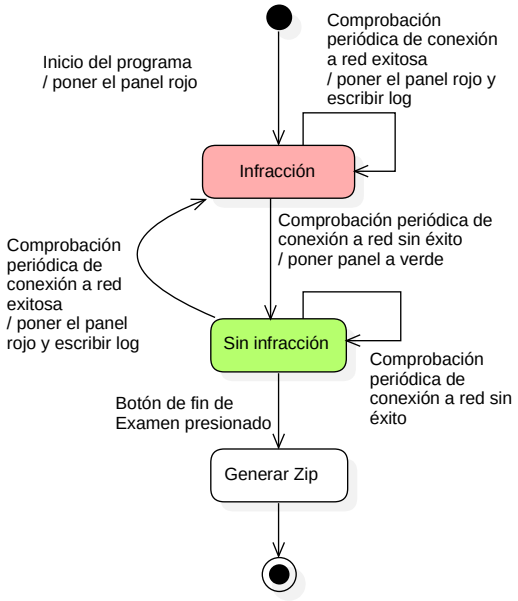


Figura 4: Diagrama de estados UML de la aplicación *ExamWatcher*.

3. Resultados experimentales

La metodología propuesta está siendo usada desde el curso 2015-2016 en los exámenes finales de la asignatura Estructuras de Datos Avanzadas, de tercer curso del grado en Ingeniería Informática de la URJC.

En los cursos previos los exámenes eran similares, pero no se usaba el ordenador durante el examen. Así, los estudiantes resolvían el mismo número de enunciados, con una complejidad similar y con similares porcentajes de valor en cada una de las partes.

3.1. Correlación examen/prácticas

La Figura 6 muestra la relación entre las notas de las prácticas realizadas por el estudiante en su casa y la nota obtenida en el examen. Este análisis se ha realizado para los cursos en los que se produjo la transición metodológica, de manera que se pueden comparar los datos antes y después de implantar la metodología propuesta. El estudio incluye notas de 30 alumnos del curso 2014-2015 y 40 del curso 2015-2016.

Al igual que ocurre en un estudio similar realizado por J. C. Gonzalez et al. [5], las prácticas con ordena-

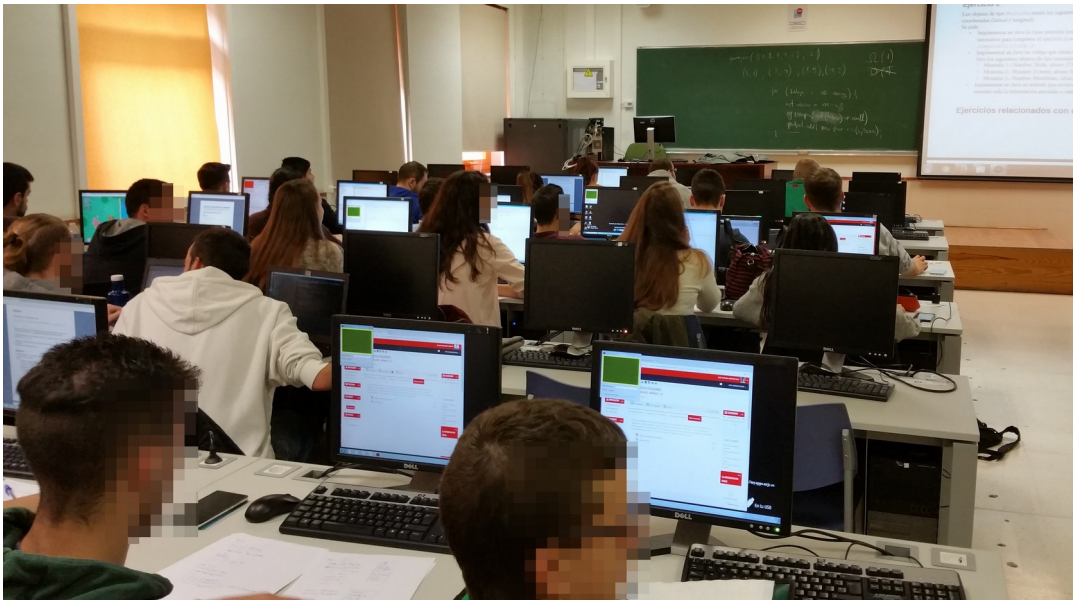


Figura 5: Instantánea de un momento en el transcurso de un examen. Nótese que la mayoría de los estudiantes tienen el panel del *ExamWatcher* en verde, aunque hay un panel rojo en la esquina superior izquierda.

dor y el examen final en papel no muestran una correlación muy acusada. En este caso, la correlación tiene un valor de 0,33. Sin embargo, cuando el examen final se realiza usando el ordenador esa correlación aumenta hasta 0,54. Creemos que este aumento de correlación podría tener múltiples orígenes: por un lado, podría deberse a que el alumno durante el examen usa las mismas herramientas a las que se ha habituado durante las prácticas; podría deberse a que los enunciados del examen estén más medidos por el profesor, ya que debe ser resuelto usando el ordenador; también podría deberse a que el alumno puede comprobar el funcionamiento de sus respuestas, gracias a la posibilidad de ejecución; finalmente, también podría deberse a la mejora en la compresión de los enunciados, derivada de la existencia de los test unitarios.

3.2. Encuesta de opinión

Para conocer la opinión de los estudiantes respecto al cambio en el sistema de evaluación, se ha realizado una encuesta anónima a los que han utilizado este sistema por primera vez. El Cuadro 1 muestra los resultados obtenidos por 44 de los 60 alumnos que se presentaron a la convocatoria de Enero de 2018.

Primero analizaremos los resultados sin diferenciar entre los estudiantes que han conseguido superar el examen y los que no. Respecto al uso de la herramienta *ExamWatcher* para la detección de usos ilícitos del ordenador durante el examen, se puede comprobar que la gran mayoría de los estudiantes (95%) considera que su uso es sencillo, de manera que no se producirán errores en la entrega derivados del mal uso de la herramienta. Además, un porcentaje muy alto de los estudiantes (93%) considera que les gustaría poder utilizar el ordenador en otras asignaturas de programación, lo que refleja una sensación positiva de la aplicación de esta metodología.

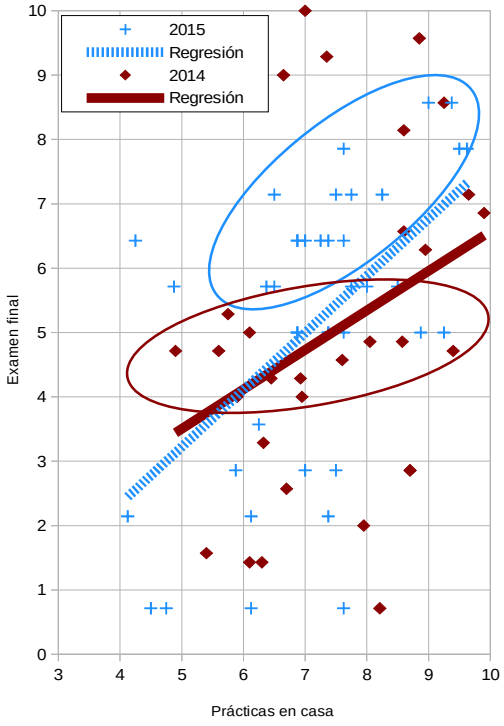


Figura 6: Correlación entre la nota de los trabajos fuera de clase y la nota de los exámenes sin ordenador (2014-2015) y con él (2015-2016).

Si analizamos el método de evaluación, el 71% de los alumnos considera que disponer de test unitarios que especifiquen el comportamiento del código que han de desarrollar les ha clarificado el enunciado del examen, reduciendo su ambigüedad y, por tanto, minimizando el número de respuestas erróneas debidas a malentendidos. Para reducir el 17% de alumnos que consideran que los test no les han ayudado, se propone la realización de más sesiones prácticas incluyendo test unitarios, para que se familiaricen con su funcionamiento y analicen las ventajas de poder utilizarlos. Por otra parte, un 83% de los alumnos considera que la evaluación ha sido más justa respecto a exámenes en los que no se utiliza el ordenador. Este resultado puede ser debido a que el uso conjunto de test unitarios y rúbrica de evaluación especifica muy claramente qué requisitos son necesarios para superar la prueba previamente a la realización de la misma.

El uso del ordenador en el examen puede producir confusiones derivadas del entorno, notificaciones, problemas técnicos, etc., por lo que consideramos necesario preguntar al estudiante si tuvo alguna distracción que pudiera causar peores resultados en su examen. La respuesta general (98%) ha sido negativa, por lo que no se produjeron distracciones durante el desarrollo de la prueba.

Gracias a la utilización del ordenador, los enunciados de la prueba se parecen en mayor medida a las prácticas realizadas durante el curso. Sin embargo, es necesario confirmar que este método de evaluación permite aumentar la similitud entre las prácticas realizadas y el examen. La respuesta ha sido positiva en general (85%), por lo que se puede concluir que la evaluación en el ordenador ha hecho que para el estudiante la prueba sea más similar a las prácticas de lo que lo era la prueba tradicional en papel.

Las pruebas tradicionales en papel solían generar descontento en algunos alumnos debido a que no se sentían cómodos programando en papel, y por lo tanto, no podían plasmar todos los conocimientos adquiridos. El objetivo de la última pregunta es confirmar que la utilización del ordenador ha mejorado la satisfacción del estudiante respecto a ser capaz de mostrar todo lo aprendido en la evaluación de la asignatura, lo que se confirma con el 90% de los estudiantes.

En este estudio, también se ha realizado el análisis teniendo en cuenta únicamente la opinión de aquellos estudiantes que ya habían cursado la asignatura otros años y no habían podido superarla. La importancia de estos resultados radica en que estos estudiantes se han enfrentado a las pruebas tradicionales en papel y, por lo tanto, su opinión es muy valiosa para conocer la relevancia del paso a ordenador. En este caso, se observa que se mantiene la tendencia general del Cuadro 1. Sin embargo, hay algunas respuestas que cabe destacar. En primer lugar, el porcentaje de alumnos a los que no les ha ayudado utilizar test unitarios ha aumentado (del 17% al 20%). Esto puede ser debido a que son estudiantes que tradicionalmente se han examinado en papel, por lo que sería interesante realizar más sesiones prácticas para que estos se adapten al nuevo sistema. Por otra parte, es interesante destacar que la totalidad de los encuestados considera que el ordenador no ha producido distracciones y que la valoración global del cambio es muy positiva.

Pregunta	**Indiferente**	**No**	**Sí**
¿Consideras que el sistema utilizado para evitar el uso ilícito del ordenador (*ExamWatcher*) era muy complejo?	0	95	5
¿Te gustaría que en otros exámenes de programación de la carrera se utilizase el ordenador?	5	2	93
¿Consideras que disponer de test unitarios durante el examen te ha ayudado a entender mejor las preguntas?	12	17	71
¿Consideras que la corrección que ha realizado el profesor es más justa, o está mejor justificada, respecto a otros exámenes de programación que no usaban el ordenador?	5	12	83
¿Consideras que el uso del ordenador en el examen te ha distraído a la hora de resolver los problemas?	0	98	2
¿Consideras que los problemas del examen eran parecidos a los que has resuelto en las prácticas?	0	15	85
¿Consideras que al usar el ordenador durante el examen has podido demostrar mejor tus conocimientos?	0	10	90

Cuadro 1: Resultados en porcentaje de la encuesta anónima realizada a los alumnos al respecto de la metodología de evaluación propuesta.

Finalmente, también se ha realizado el análisis separando aquellos alumnos que no han conseguido superar la asignatura, con el objetivo de analizar y mejorar la implementación de la prueba. En este caso, se mantiene la preferencia de utilizar el ordenador en las pruebas entre los estudiantes, además de que también consideran que los test unitarios son una herramienta útil para entender mejor las preguntas de la prueba.

El resultado más relevante de esta parte del análisis se encuentra en las dos últimas preguntas. En concreto, los estudiantes que no han superado la prueba consideran que la dificultad del examen ha sido superior a la de las prácticas. Además, aunque la mayoría (67%) cree que gracias al ordenador han podido plasmar mejor sus conocimientos, hay un porcentaje considerable de alumnos (33%) que no cree que el ordenador haya ayudado a ello. Entre las medidas propuestas para mejorar estos resultados se encuentra el aumento del número de sesiones prácticas y la realización de uno o varios simulacros de examen, para

que los estudiantes se enfrenten al entorno que verán el día de la prueba sin la presión de la evaluación.

4. Conclusiones

En este trabajo hemos presentado una metodología para evaluar a los estudiantes de un curso de programación avanzada que incorpora como novedad principal el uso del ordenador durante el examen.

La metodología propuesta incluye la realización: de prácticas, que se apoyan en test unitarios; de exámenes prácticos, con enunciados similares a los de las prácticas; y de exámenes de tipo test, para evaluar el conocimiento teórico. Además, incorpora dos herramientas que se han desarrollado ad hoc: un software que vigila las acciones del alumno mientras realiza el examen, y una herramienta que ayuda al profesor gestionando la corrección.

Se han analizado los resultados obtenidos antes y después de la aplicación de la metodología propuesta en diferentes cursos de programación realizados en la Universidad Rey Juan Carlos, obteniéndose que el uso del ordenador durante el examen proporciona unas calificaciones en el examen con mayor correlación con las de las prácticas. También, en el análisis de una encuesta que se ha realizado a los alumnos tras el examen, se observa que los alumnos valoran muy positivamente el uso del ordenador en el examen.

Como trabajos futuros se está estudiando extender la funcionalidad de *ExamWatcher* para que sea útil en otros tipos de exámenes que no sean de programación, así como aumentar la integración de las diferentes aplicaciones desarrolladas.

Agradecimientos

Agradecemos desde estas líneas la financiación aportada por el proyecto TIN2014-57458-R.

Referencias

[1] Kirsti M. Ala-Mutka. A Survey of Automated Assessment Approaches for Programming Assignments. Computer Science Education, 15, 83-102, 2005.

[2] João Paulo Barros, Luís Estevens, Rui Dias, Rui Pais, Elisabete Soeiro. Using Lab Exams to Ensure Programming Practice in an Introductory Programming Course. ACM SIGCSE Bulletin - Procs. of the 8th annual conference on Innovation and technology in computer science education, 35-3, 16-20, ACM, 2003.

[3] Yoonsik Cheon, Gary T. Leavens. A Simple and Practical Approach to Unit Testing: The JML and JUnit Way. Procs. of the 16th European Conference on Object-Oriented Programming, 231-255, Springer-Verlag, 2002.

[4] Robert Deloatch, Brian P. Bailey, Alex Kirlik. Measuring Effects of Modality on Perceived Test Anxiety for Computer Programming Exams. Procs. of the 47th ACM Technical Symposium on Computing Science Education, 291-296, ACM, 2016.

[5] J. C. González, L. Arriero, C. Benavente, R. Fraile, J.I. Godino-Llorente, J. Gutiérrez, D. Osés, V. Osma-Ruiz. A case study: Final exam versus continuous assessment marks for electrical and electronic engineering students. ICERI2008 Conference, 2008.

[6] Norman Jacobson. Using On-computer Exams to Ensure Beginning Students' Programming Competency. SIGCSE Bulletin, 32-4, 53-56, ACM, 2000.

[7] Pari Delir Haghighi, Judy Sheard. Summative Computer Programming Assessment Using Both Paper and Computer. Procs. of the 2005 conference on Towards Sustainable and Scalable Educational Innovations Informed by the Learning Sciences: Sharing Good Practices of Research, Experimentation and Innovation, IOS Press, 67-75, 2005.

[8] Diete Pawelczak. Benefits of a Testing Framework in Undergraduate C Programming Courses. Procedia - Social and Behavioral Sciences, 228, 215-221, 2016.

[9] Judy Sheard, Simon, Angela Carbone, Daryl D'Souza, Margaret Hamilton. Assessment of Programming: Pedagogical Foundations of Exams. Procs. of the 18th ACM Conference on Innovation and Technology in Computer Science Education, 141-146, ACM, 2013.

[10] Ian Utting, Allison Elliott Tew, Mike McCracken, Lynda Thomas, Dennis Bouvier, Roger Frye, James Paterson, Michael Caspersen, Yifat Ben-David Kolikant, Juha Sorva, Tadeusz Wilusz. A Fresh Look at Novice Programmers' Performance and Their Teachers' Expectations. Procs. of the ITiCSE Working Group Reports Conference on Innovation and Technology in Computer Science Education-working Group Reports, 15-32, ACM, 2013.

[11] Craig Zilles, Robert Timothy Deloatch, Jacob Bailey, Bhuwan B. Khattar, Wade Fagen, Cinda Heeren. Computerized testing: A vision and initial experiences. 122nd ASEE Annual Conference and Exposition, American Society for Engineering Education, 2015.