VPL: Laboratorio Virtual de Programación para Moodle

Juan Carlos Rodríguez del Pino
Departamento de Informática y Sistemas
Universidad de Las Palmas de Gran Canaria
Campus Universitario de Tafira
35017 Las palmas de Gran Canaria
jcrodriguez@dis.ulpgc.es

Enrique Rubio Royo
Departamento de Informática y Sistemas
Universidad de Las Palmas de Gran Canaria
Campus Universitario de Tafira
35017 Las palmas de Gran Canaria
rubio@cicei.com

Zenón J. Hernández Figueroa
Departamento de Informática y Sistemas
Universidad de Las Palmas de Gran Canaria
Campus Universitario de Tafira
35017 Las palmas de Gran Canaria
zhernandez@dis.ulpgc.es

Resumen

En este trabajo se presenta VPL (Virtual Programming Lab), una herramienta software de código abierto que permite la gestión de prácticas de programación en Moodle. Esta herramienta está compuesta de un módulo Moodle, un applet editor de código fuente y un demonio Linux que permite la ejecución remota de programas de forma segura. Pretende ahorrar tiempo y mejorar la gestión general de este tipo de actividades, además de permitir la realización de las prácticas sólo con un navegador. Sus características más destacadas son: la posibilidad de editar el código fuente y ejecutar las prácticas de forma interactiva desde el navegador, ejecutar pruebas que revisen las prácticas y analizar la similitud entre prácticas para el control del plagio. Esta herramienta se ha empleado con éxito en diversas asignaturas durante el año 2009. VPL está disponible para su descarga en http://vpl.dis.ulpgc.es.

1. Introducción

Los estudios de informática, y otros de perfil tecnológico, incluyen asignaturas que incorporan en su metodología la realización de prácticas de programación.

La evaluación de las prácticas, sobre todo en los primeros cursos, puede requerir bastante tiempo y esfuerzo por parte del profesor debido a la cantidad de alumnos, al número de trabajos y a su complejidad, lo que, con frecuencia, impide una realimentación adecuada durante el desarrollo de las prácticas. Ello supone un serio hándicap para el proceso formativo, provocando la pérdida de una parte importante del potencial de aprendizaje que conlleva la realización de una práctica.

La disponibilidad de una herramienta que facilite el seguimiento y la orientación personalizada y continuada del proceso de aprendizaje del alumno contribuye a allanar las dificultades a las que se enfrenta éste. Al profesor, la posibilidad de automatizar gran parte de la evaluación, le permite aplicar su esfuerzo de una manera más productiva.

Los problemas que pretende resolver esta herramienta han estado presentes desde el inicio de la enseñanza de la informática y se han sucedido distintas propuestas de solución. Algunas se han centrado en aportaciones en el ámbito de la evaluación automática del estilo, sintaxis y lógica de los programas, como STYLE [9] y CAP [12]. Otros entornos están especializados en las capacidades de evaluación, como PASS [1, 2, 13]. En algunos casos, para conseguir la ejecución de prácticas, se centran en lenguajes específicos, como PACER [8], que usa un lenguaje propio (ELI) similar a C, y HoGG [7] y ELP [14], específicos para Java. Las herramientas más complejas incluyen capacidades más generales de gestión de cursos, como control de entregas, evaluación y búsqueda de similaridad. En esta línea destacan Ceilidh [3], que ha evolucionado a CourseMaster [4], y BOSS [5, 6].

VPL es la herramienta heredera de GAP [10, 11] creada para superar sus limitaciones, integrándose en una plataforma de enseñanza electrónica, siendo una herramienta de código abierto, escalable, que admite múltiples lenguajes, etc. Se ha ganado en usabilidad, integración y seguridad, y se ha perdido, entre otras

características que podrían ser recuperables, la gestión de entrega de prácticas en equipo.

2. Objetivos

Los objetivos planteados en el desarrollo de esta herramienta son:

1. Implementar una herramienta abierta, con una amplia distribución y capaz de captar aportaciones externas.
2. Cubrir un amplio abanico de posibilidades de uso, incluyendo prácticas presenciales, no presenciales, exámenes en laboratorio, etc.
3. Ser independiente del lenguaje de programación utilizado en las prácticas.
4. Proveer un entorno de desarrollo simple para facilitar el aprendizaje en los primeros cursos.
5. Facilitar la evaluación automática de las prácticas entregadas.
6. Garantizar la seguridad del sistema.

El primer objetivo motiva que el software desarrollado se distribuya bajo licencia GNU/GPL v2 y la elección de Moodle como entorno de destino, por ser una plataforma de e-learning con gran implantación y que da grandes facilidades para la integración de componentes externos.

Durante todo el desarrollo se ha tenido en cuenta no restringir el sistema a un lenguaje de programación concreto, aunque se ha limitado a lenguajes que se ejecuten en sistemas Linux y con interacción textual.

3. Arquitectura del sistema

VPL es un sistema formado por tres componentes software: el módulo Moodle, el sistema cárcel y el editor. El módulo Moodle está escrito en lenguaje PHP y es el núcleo del sistema, pudiendo operar, con funcionalidades reducidas, aún cuando alguno de los otros dos componentes, o ambos, se hallen inoperativos.

El editor de código es un applet Java y permite, desde los navegadores, editar ficheros con código fuente de lenguajes de programación, además de permitir la interacción, por medio de una consola, con la ejecución de programas en un servidor cárcel.

El sistema cárcel está construido usando como núcleo un demonio Linux. Es el encargado de ejecutar programas en un entorno controlado y de forma segura, atendiendo a las peticiones del servidor Moodle. Estas peticiones requieren la ejecución de guiones y ficheros binarios con limitaciones en cuanto a memoria, tiempo, disco usado y número máximo de procesos simultáneos. Para mayor fiabilidad y comodidad de implantación y uso, es recomendable que los sistemas cárcel se constituyan como máquinas virtuales inalterables, de forma que al reiniciarse vuelvan siempre al estado original seguro. Para posibilitar la fácil implantación de sistemas cárcel, en la página web de la herramienta (http://vpl.dis.ulpgc.es), se suministran sistemas cárcel, en forma de máquinas virtuales, preconfiguradas y listas para ser usadas.

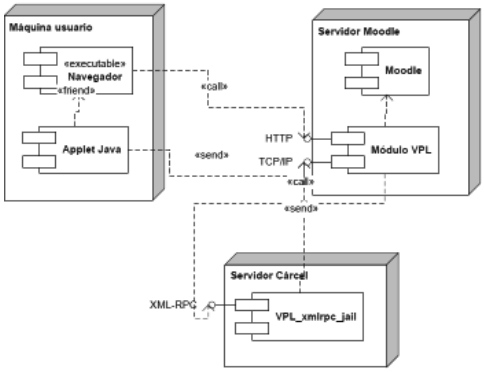

Figura 1. Arquitectura de los componentes software

En la figura 1 se muestra la arquitectura de interacción entre los componentes software. El módulo, núcleo del sistema, está incrustado en un servidor Moodle, los otros dos componentes no interactúan entre ellos, pero sí por medio del módulo. El editor se ejecuta en el navegador interactuando con el módulo mediante acciones en JavaScript. El demonio en el servidor cárcel recibe peticiones del módulo y da respuestas a éste mediante el protocolo XML-RPC [15]. En caso de ejecución interactiva, tanto el editor como el demonio cárcel contactan con el módulo para que éste haga de pasarela reenviando la información de uno a otro.

3.1. Requisitos de una instalación

El modulo VPL requiere una versión 1.9.x de Moodle y PHP5 o superior. Para ejecutar o evaluar una entrega se necesita disponer de al

menos un servidor cárcel. Para disponer de ejecución interactiva se necesita, además, que en la máquina del servidor Moodle se puedan abrir al menos dos puertos —un número mayor es recomendable. Los "scripts" PHP que lanzan la evaluación y sobre todo la ejecución en consola, requieren habitualmente modificar los límites de tiempo en la configuración PHP.

El servicio cárcel necesita un Linux/GNU con kernel 2.6.18 o superior. La compilación del servidor cárcel requiere un compilador de C++. Los script de instalación y ejecución están escritos para distribuciones Red Hat o compatibles y requieren tener instalado xinetd.

Para una interacción completa con el sistema se necesita un navegador con JavaScript y soporte para applets Java 1.5. El editor y el código JavaScript que le acompaña se han probado con éxito en Mozilla FireFox 3.5, MS Internet Explorer 8.0, Opera 10.51 y Chrome 4.1.

4. Características del sistema

4.1. Integración con Moodle

El módulo suministra las características habituales de las actividades Moodle como: copias de seguridad y restauración de datos, reinicio de un curso, uso del libro de calificaciones, establecimiento automático de evento de fin de plazo de entrega, control de acceso de usuarios basado en roles, listado de alumnos según grupos, registro de accesos, etc.

En el caso de la copia de seguridad es de destacar que se ha decidido almacenar sólo la última entrega de cada práctica. Si se almacenasen todas las versiones, el tiempo y tamaño requeridos sería superior a lo admisible cuando los cursos tuviesen un número de alumnos, actividades o reentregas moderadamente grande. Esta limitación es achacable al formato de compresión usado por Moodle para realizar las copias de seguridad.

4.2. Control de entrega

El módulo dispone de múltiples opciones para controlar cuándo, qué, cómo y desde dónde se hace entrega de los ficheros requeridos. Se puede limitar el periodo de entrega y es posible establecer cuándo se puede acceder a la descripción de la tarea a realizar. Se debe establecer el número máximo de ficheros por entrega. Opcionalmente se puede indicar el tamaño máximo de cada fichero entregado. Se pueden establecer nombres para los ficheros a subir. Los ficheros a los que se les dé nombre son obligatorios en la entrega, el resto de ficheros, hasta llegar al máximo, son opcionales.

El mostrado de la descripción de la actividad, la edición de ficheros en el navegador y la subida directa de ficheros se pueden restringir a las redes que se especifiquen y/o proteger con una clave que se establezca. Esta opción permite manejar de forma sencilla el uso de la herramienta para la realización de pruebas presenciales en el laboratorio.

Además, es posible introducir contenidos iniciales en los ficheros a entregar. Esto permite que cuando el alumno accede inicialmente al editor, éste estará precargado con los contenidos establecidos.

El alumno puede recuperar en formato zip los ficheros entregados almacenados en el sistema. También es posible visualizarlos en una página web, mostrándose con resaltado sintáctico acorde al lenguaje.

El sistema almacena las entregas y los ficheros guardados desde el editor. Los usuarios con rol de calificador pueden ver estas versiones previas. Por razones de espacio y eficiencia no se almacenan todas las versiones, eliminando las que se hayan producido en medio de dos entregas dentro de un periodo configurable.

La filosofía de recepción de prácticas del sistema es que durante el periodo correspondiente se pueden realizar tantas entregas como se desee y, sólo después, se hará una evaluación definitiva de la última. Cada subida de ficheros o cada modificación y guardado en el editor es una entrega. Con cada entrega, el sistema realiza evaluaciones automáticas y muestra al alumno los resultados obtenidos para que le sirvan de realimentación cara a las subsiguientes entregas.

4.3. Edición de entrega

Es posible editar los ficheros a entregar o los ya entregados desde el propio navegador usando el componente editor. El editor permite las operaciones típicas de edición como seleccionar, copiar, cortar y pegar, permite deshacer y rehacer modificaciones y buscar y reemplazar (véase la figura 2). Además, posibilita editar varios

ficheros a la vez, cumpliendo las restricciones de nombres y número de ficheros establecidos en la actividad. También realiza el resaltado sintáctico para C, C++, Java, Ada, Pascal, Scheme, Prolog, Fortran y bash script.

Figura 2. Editor applet Java en funcionamiento en Mozilla Firefox

4.4. Ejecución de la práctica

El sistema está diseñado para posibilitar la ejecución de programas desde el navegador. Esto se concreta en tres posibles acciones: ejecutar, depurar y evaluar. Estas tres acciones se pueden realizar desde la página de edición. La acción evaluar también se puede ejecutar automáticamente en el momento de la subida de ficheros al sistema o a petición del calificador en la página de calificación. Las acciones ejecutar y depurar tienen un comportamiento similar llevando a la ejecución interactiva del programa en una consola controlada por el editor. La evaluación conlleva una ejecución sin interacción, siendo su salida procesada para obtener una calificación comentada. La tarea a realizar en cada una de las tres acciones se define en un "script shell" que prepara un fichero con un nombre concreto para su posterior ejecución. Un ejemplo de configuración de ficheros de ejecución se puede ver en la figura 3.

La acción de evaluación es la que posibilita realizar una evaluación automática con realimentación.

Para una elaboración más sencilla de los "scripts" y programas de evaluación y ejecución estos pueden basarse en otra instancia de VPL, tomando sus ficheros y añadiendo código a los scripts originales. Esto permite tener actividades, no visibles para los alumnos, que contengan los ficheros comunes a otras actividades, eliminando duplicidades y facilitando cambios y adiciones.

Para acotar la ejecución de pruebas y programas, se pueden establecer límites a los recursos empleados: tiempo, memoria, tamaño de ficheros y número de procesos. Estos límites se pueden establecer globalmente para todo el sistema y, particularmente, para cada actividad, limitando, a su vez, las actividades que se basan en ella.

Figura 3. Definición de ficheros de ejecución

4.5. Ayuda a la evaluación

El sistema dispone de diversos elementos que pueden ayudar al proceso de evaluación. La evaluación parte de un listado de alumnos que permite el acceso a la página específica de calificación de cada entrega, ver figura 4. En la página de calificación, después de guardar una revisión, la siguiente práctica revisable se puede cargar automáticamente, mejorando la navegabilidad durante la evaluación.

Los comentarios que introduce el evaluador como explicación de la calificación permiten aplicar cierto formato. Si la línea comienza por un guión se entiende que es un título de comentario,

si comienza por “>>” es un texto preformateado y si comienza por http se genera un hiperenlace. Si en el texto se encuentra una secuencia de la forma “fichero:número”, durante el mostrado se genera un hiperenlace a la línea del fichero correspondiente. En una línea de título se puede añadir, al final, una cifra entre paréntesis. Esta cifra representa la penalización que supone ese error. Si la escala de calificaciones es numérica la nota se puede calcular a partir de esta información, ver figura 5.

Figura 4. Listado de entregas por alumnos

Figura 5. Página de calificación

En cada actividad, los títulos de los comentarios son recopilados para poder ser reusados, pudiendo agregarse a las subsiguientes revisiones con un clic. Esto facilita la coherencia de la revisión del conjunto de las prácticas, sobre todo, cuando esta se realiza por varios profesores.

Por otro lado, el sistema dispone de la posibilidad de indicar que se tome como calificación final la nota propuesta por la evaluación automática. Esta opción permite crear actividades evaluables automáticamente y sin intervención de un revisor humano.

4.6. Control de autoría

El sistema dispone de mecanismos para verificar la autoría de las prácticas y dificultar su plagio. Permite analizar la similitud entre prácticas atendiendo a diferentes métricas cuya exposición escapa a los objetivos de este trabajo. Los resultados se ordenan de mayor a menor similitud y se permite la visualización "a dos bandas" de las prácticas similares resaltando los parecidos.

Es posible añadir marcas de agua al código descargado lo que puede suministrar valiosa información sobre el posible empleo de ficheros descargados y cedidos a un tercero.

Otra medida que dificulta el plagio es la posibilidad de no permitir la subida de ficheros, obligando a que las entregas sean por medio del editor, impidiendo, además, que éste acepte pegados de código externo, pero permitiendo las operaciones “cortar”, “copiar” y “pegar” entre documentos en el editor. Esta medida puede ser especialmente útil si desea usar el sistema para pruebas presenciales en un laboratorio.

4.7. Variaciones

Una característica interesante que suministra el sistema es la posibilidad de añadir variaciones a una actividad. Las variaciones se emplean para personalizar la actividad. Son asignadas aleatoriamente a cada usuario. Al usuario se le informa en la descripción de la actividad de la variación o variaciones que le han sido asignadas.

El sistema pasa a los script de ejecución, mediante variables de entorno, las variaciones asignadas a cada usuario, permitiendo personalizar las pruebas a realizar para cada uno.

Las variaciones permiten construir actividades más versátiles y diversas, posibilitando que no todos los alumnos tengan que realizar exactamente la misma actividad.

4.8. Seguridad

La seguridad es un elemento clave en VPL. Es de destacar que el módulo Moodle, aunque trata la ejecución y evaluación de entregas, en ningún caso ejecuta código externo, ya sea proveniente de un administrador o de un estudiante. El único código que se ejecuta en el servidor es el del propio módulo. Las ejecuciones y pruebas siempre se realizan en un servidor cárcel. Otro aspecto es la seguridad del almacén de entregas, en el cual se ha eliminado el acceso mediante la vía habitual en Moodle de forma que para recuperar una entrega se emplea un sistema de descarga propio que controla adecuadamente el acceso, devolviendo la entrega en forma de fichero comprimido.

El editor no tiene acceso a ningún recurso de la máquina local, como consecuencia, no puede leer información de ella o alterarla.

El servicio cárcel centra la seguridad en ejecutar las peticiones en una cárcel "chroot". La instrucción "chroot" se encarga de cambiar el directorio raíz del sistema de ficheros. Para ello, se crea un sistema de ficheros ficticio que no permite ejecutar programas que supongan escalar privilegios. En cada ejecución se cambia a un usuario virtual seleccionado aleatoriamente entre un conjunto de disponibles. Al finalizar cada ejecución se eliminan todos los ficheros creados por dicho usuario.

Que el código fuente de VPL esté disponible libremente redunda en mayor confiabilidad ya que se puede verificar el comportamiento del sistema en todo momento.

5. Uso de VPL

El módulo VPL se ha venido usando para la gestión de prácticas en el segundo cuatrimestre del curso 2008/2009 en Estructuras de Datos II, asignatura del segundo curso de las titulaciones de Ingeniería Técnica de Sistemas, Ingeniería Técnica de Gestión e Ingeniería en Informática, además de en el primer cuatrimestre del curso 2009/2010 en las asignaturas de Metodología de la Programación II de las mismas titulaciones. Ambas asignaturas tenían en torno a 140 alumnos.

Las tablas 1 y 2 muestran, para cada una de las asignaturas, el número promedio de entregas de cada práctica que hizo cada alumno. El número es alto ya que, al utilizar el editor integrado, el sistema se emplea como almacén para las prácticas en proceso de desarrollo. Salvo por la primera práctica de la segunda asignatura, el valor mostrado tiene una tendencia claramente descendente, lo cual podría reflejar un dominio creciente de las técnicas de programación por parte del alumno. La anomalía reseñada para la primera práctica puede ser indicativa de que la misma presenta una dificultad inferior a la media.

Práctica	Alumnos que entregan	Media de entregas
1	141	84,95
2	131	97,67
3	124	83,95
4	114	40,17
5	97	28,42

Tabla 1. Entregas en el segundo cuatrimestre del curso 2008/2009

El número de alumnos que entregan las prácticas es también decreciente, debido en parte a que para superar estas asignaturas no es necesario realizarlas todas.

Práctica	Alumnos que entregan	Media de entregas
1	140	51,13
2	137	86,25
3	127	78,96
4	105	50,93

Tabla 2. Entregas en el primer cuatrimestre del curso 2009/2010

6. Conclusiones

Con este trabajo se pone a disposición pública una herramienta que se ha mostrado eficiente en la gestión de prácticas y pruebas presenciales de programación.

Dicha herramienta se integra perfectamente con Moodle, la popular plataforma de e-learning, lo que facilita su aprovechamiento por gran número de usuarios.

Su usabilidad, versatilidad y, sobre todo, la posibilidad de crear actividades fácilmente reusables y compartibles, le confieren una gran potencialidad que se podrá ver reforzada en el futuro por la creación de un repositorio de

actividades (objetos de aprendizaje) reusables de acceso público.

El componente cárcel confiere al sistema una alta seguridad al controlar de forma estricta la ejecución de las prácticas a evaluar.

El editor, por su propia naturaleza, tiene reducidas las capacidades de edición, autocompletado y ayuda de que disponen los entornos de desarrollo actuales. La carencia más destacable, que deberá resolverse en un futuro próximo, al menos para los lenguajes de programación más usados, es la ausencia de un entorno de depuración amigable.

Las variaciones permiten personalizar las actividades. Aunque actualmente son asignadas automáticamente, en el futuro podría permitirse su elección por cada usuario.

Agradecimientos

Es necesario agradecer al Departamento de Informática y Sistemas de La Universidad de Las Palmas de Gran Canaria el ceder la máquina que hospeda el servidor web VPL (http://vpl.dis.ulpgc.es) y el servidor demo de Moodle con el módulo VPL instalado (http://demovpl.dis.ulpgc.es/moodle), además de posibilitar y mantener la conexión a internet.

Referencias

[1] Choy, M., S. Lam, CK Poon, FL Wang, YT Yu, y L. Yuen. «Towards Blended Learning of Computer Programming Supported by an Automated System.» *Blended Learning*, 2007: 9.

[2] Choy, M., U. Nazir, CK Poon, y YT Yu. «Experiences in using an automated system for improving students' learning of computer programming.» *Lecture notes in computer science* (Springer) 3583 (2005): 267.

[3] Foubister, S.P., GJ Michaelson, y N. Tomes. «Automatic assessment of elementary Standard ML programs using Ceilidh.» *Journal of Computer Assisted Learning* 13 (1997): 99-108.

[4] Higgins, Colin A., Geoffrey Gray, Pavlos Symeonidis, y Athanasios Tsintsifas. «Automated assessment and experiences of teaching programming.» *J. Educ. Resour. Comput.* (ACM) 5 (2005): 5.

[5] Joy, Mike, Nathan Griffiths, y Russell Boyatt. «The boss online submission and assessment system.» *J. Educ. Resour. Comput.* (ACM) 5 (2005): 2.

[6] Luck, M., y M. Joy. «A secure on-line submission system.» *Software-Practice and Experience* 29 (1999): 721-40.

[7] Morris, D.S. «Automatic grading of student's programming assignments: an interactive process and suite of programs.» *Frontiers in Education, 2003. FIE 2003. 33rd Annual.* 2003. S3F-1-6 vol.3.

[8] Palakal, Mathew J., Frederick W. Myers, y Carla L. Boyd. «An interactive learning environment for breadth-first computing science curriculum.» *SIGCSE Bull.* (ACM) 30 (1998): 1-5.

[9] Rees, Michael J. «Automatic assessment aids for Pascal programs.» *SIGPLAN Not.* (ACM) 17 (1982): 33-42.

[10] Rodríguez-del-Pino, JC, M. Díaz-Roca, Z. Hernández-Figueroa, y JD González-Domínguez. «Hacia la Evaluación Continua Automática de Prácticas de Programación.» *Actas de las XIII Jornadas de enseñanza Universitaria de la Informática. Teruel*, 2007: 179-186.

[11] Rodríguez-del-Pino, JC, Z. Hernández-Figueroa, JD González-Domínguez, y M. Díaz-Roca. «Experiencia de uso y características del programa Gestión Automática de Prácticas (GAP).» *Actas Conferencia IADIS Ibero-Americana.* 2005. 35-42.

[12] Schorsch, Tom. «CAP: an automated self-assessment tool to check Pascal programs for syntax, logic and style errors.» *SIGCSE '95: Proceedings of the twenty-sixth SIGCSE technical symposium on Computer science education.* ACM, 1995. 168-172.

[13] Thorburn, Gareth, y Glenn Rowe. «PASS: An automated system for program assessment.» *Computers \& Education* 29 (1997): 195-206.

[14] Truong, Nghi, Peter Bancroft, y Paul Roe. «A web based environment for learning to program.» Australian Computer Society, Inc., 2003. 255-264.

[15] Winer, Dave. *XML-RPC Specification.* 1999. http://www.xmlrpc.com/spec (último acceso: 12 de 2 de 2010).