Generación automática de pruebas de validación de prácticas de sistemas distribuidos

Enrique A. de la Cal Marín, Marco A. García Tamargo, María José Suárez-Cabal
Dpto. de Informática
Universidad de Oviedo
Campus de Viesques, S/N 33204 Gijón
{delacal, marco, cabal}@uniovi.es

Resumen

La corrección de prácticas de sistemas distribuidos es una tarea costosa para el profesor que evalúa dichos ejercicios. Esta tarea puede ser sistematizada mediante la ejecución de tests automatizados cuya elaboración representa una alta inversión de tiempo para cada práctica a corregir. Así surge la necesidad de desarrollar una herramienta que, a partir de pequeñas especificaciones proporcionadas por el usuario, genere fácil y rápidamente dichos scripts libres de errores de sintaxis. Con este fin, se ha desarrollado una herramienta web flexible, sencilla de utilizar y que permite ahorrar tiempo dedicado a la elaboración de scripts.

1. Motivación y objetivos

En general el proceso de elaboración de prácticas de laboratorio en las que existe alguna componente de programación conlleva un alto consumo de tiempo en dos aspectos:

- la elaboración manual de los mecanismos de prueba, ya sean scripts de prueba, archivos de entrada/salida, etc.
- la propia aplicación de las pruebas sobre los ejercicios entregados por los alumnos y la valoración de los mismos.

Dichas tareas consumen un tiempo proporcional al número de alumnos que se evalúan y que suele ser alto. Por ello parece bastante evidente que ese tiempo puede reducirse con el fin de invertirlo en otras tareas más provechosas.

Existen múltiples trabajos en la literatura en los que se aborda dicha problemática, [7][8][9], pero ninguno se adapta a la evaluación de prácticas de sistemas distribuidos basados en mecanismos de bajo nivel. Este trabajo presenta una herramienta para la generación automatizada de scripts de prueba de prácticas de sistemas distribuidos. La cual, como se mostrará a lo largo del presente trabajo, supone un ahorro sustancial en el tiempo de evaluación empleado por los profesores.

Este trabajo se estructura en las siguientes secciones: en el apartado 2 se recoge un repaso de los trabajos existentes en la literatura sobre tests automáticos de prácticas, en la sección 3 se analiza la propuesta realizada, finalizando con las secciones de resultados numéricos y conclusiones.

2. Estado del arte

Existen diversos trabajos en la literatura correspondientes a profesores de universidades españolas que vienen empleando herramientas de corrección automatizada [10] para la evaluación de ejercicios prácticos de programación. La mayoría de estos trabajos emplean técnicas de prueba de caja blanca como el interesante trabajo de J.C. Rodríguez [9] donde se le permite al alumno tener una revisión adelantada sobre la corrección en cuanto a la indentación, comentarios, etc. de su práctica.

Otros trabajos que emplean técnicas de caja blanca, como [11], se centran en lenguajes de programación concretos, como C++, Ada o Scheme. El tipo de programación que se realiza en el desarrollo de sistemas distribuidos de bajo nivel suele ser una programación compleja y bastante difícil de encajar en esquema clásicos de validación. Por ello suele recurrirse a técnicas de prueba basadas en casos (caja negra). Existen herramientas generalistas pero una herramienta más ad-hoc puede ahorrar bastante tiempo de configuración.

3. Propuesta

3.1. Contexto de desarrollo

La herramienta propuesta se ha desarrollado tomando como referencia las necesidades de una asignatura de sistemas distribuidos impartida en 5º curso de ingeniería informática. Se trata de una

asignatura con 9 créditos de Prácticas de Laboratorio que está organizada en dos bloques cuatrimestrales:

- 1º cuatrimestre: Mecanismos de bajo nivel para la construcción de aplicaciones distribuidas (mecanismos IPC y Threads) [1][2][3].
- 2º cuatrimestre: Tecnologías de desarrollo de aplicativos Web (PHP, J2EE) [4][6].

Para evaluar los conocimientos adquiridos por el alumnado, cada alumno deberá realizar y superar al final de cada cuatrimestre un examen tipo test sobre la materia impartida, así como entregar una práctica basada en un enunciado publicado por los profesores.

Cada una de estas prácticas supone el 45% de la nota final, siendo la nota del examen tipo test el 10% restante.

3.2. Descripción del paradigma de ejercicio a evaluar

La práctica del primer cuatrimestre suele constar de varios procesos multi-hilo que se intercomunican y sincronizan entre sí. Estos procesos suelen estar parametrizados y ser deterministas, es decir, ante una entrada concreta producen una salida concreta. Esta salida normalmente suele ser en forma de fichero de texto. A través de los parámetros que reciben en la línea de comando se puede, entre otras cosas, modificar el número de hilos de un pool, el tamaño de las colas circulares, etc., lo cual permite luego al profesor realizar diversas pruebas de caja negra comparando los resultados obtenidos con los esperados.

3.3. Problemática de la corrección manual

La evaluación de ejercicios de programación hasta el curso 2002-2003 se realizaba dos pasos:

- En un primer paso se realiza una verificación de su funcionamiento mediante pruebas de aceptación de usuario (caja negra) que usualmente suelen ser simples si el proceso es manual.
- En un segundo paso se suele revisar el código (caja blanca), tarea que suele determinar la calificación final.

Dado el número de alumnos de la asignatura (una media de 100 en los 3 últimos cursos) y el tiempo que toma la realización de este tipo de pruebas, se decidió automatizar en parte los pasos necesarios para realizarlas (extracción de los ficheros, compilación, lanzamiento de cada test, comprobación de resultados, generación de informes, etc.) mediante el uso de scripts Unix [5].

A continuación se listan las etapas de evolución del proceso de evaluación de prácticas:

- Etapa1. Las prácticas se evalúan lanzándolas manualmente.
- Etapa2. La configuración de las pruebas se realizaba mediante la concepción íntegra y ad-hoc de una serie de scripts de lanzamiento automático de las prácticas.
- Etapa3. Se comenzaron a reutilizar los scripts de prácticas anteriores como plantillas para las nuevas prácticas.

Tabla 1. Tabla de tiempos invertidos en etapas 1-3

Tiempo invertido (en minutos)	Etapa1 (Sólo archivos de E/S)	Etapa2 (Scripts desde 0)	Etapa3 (Scripts Plantillas)
Configuración de la pruebas	120	300	180
Evaluación del alumno	60	10	10

El empleo de scripts de lanzamiento automático ha ahorrado una parte muy importante del tiempo de preparación, ejecución y comprobación de resultados de las diversas pruebas (ver Tabla 1). Aún así, se consume gran parte del tiempo de los profesores en adaptar los scripts de una convocatoria a otra (Fase 3) así como a la comprobación de los resultados y sintaxis de los scripts y archivos de E/S.

Por ello, el equipo de profesores de la asignatura ha elaborado una aplicación que permite generar de forma automática y libre de errores de sintaxis, los scripts de lanzamiento de las diversas pruebas, así como los ficheros de resultados con los que se realizarán las comparaciones.

Esta aplicación, ha disminuido todavía más el tiempo de preparación y validación de dichos scripts de prueba.

3.4. Herramienta

La herramienta se ha concebido como una aplicación Web desarrollada en PHP y JavaScript con nivel de accesibilidad AAA.

Una vez autentificado, el profesor accede a una pantalla con las opciones del menú principal desde el cual se puede acceder a las dos grandes

funcionalidades de la aplicación: la configuración de los scripts y la gestión de usuarios.

Figura 1. Opción Definir Prueba-I

Desde la opción Scripts se pueden seleccionar las operaciones principales de la misma, a saber:

- Definir/Eliminar Prueba
- Definir ficheros de entrada/resultados
- Generar archivos.

Figura 2. Opción Definir Prueba-II

Desde la opción Definir Prueba (Figura 1 y Figura 2) se pueden configurar los parámetros más importantes que necesitamos conocer para poder generar los scripts de lanzamiento de procesos y comprobación de resultados:

- Nombre de la prueba.
- Path base de lanzamiento de los scripts en la máquina donde se ejecutarán los mismos.
- Número de pruebas de caja negra a realizar.
- La definición de los ejecutables que intervienen en cada prueba así como los parámetros que se le pasan a los mismos desde la línea de órdenes.

Una vez superada esta primera etapa, el usuario puede pasar a definir los ficheros de entrada desde la opción del mismo nombre del menú Scripts.
Desde esta opción (Figura 3) el usuario puede definir para cada fichero de entrada:

Figura 3. Definir los ficheros de entrada

- El número de campos.
- El delimitador usado para separar cada campo.
- El tamaño máximo de cada línea.
- El número máximo de líneas.
- El tipo de cada campo.

Figura 4. Generar archivos - I

Lo mismo puede hacerse a la hora de describir los ficheros de resultados (Scripts->Definir ficheros de resultados).

Figura 5. Generar archivos – II

Finalmente, desde la opción Scripts->Generar archivos se accede a una pantalla (Figura 4 y Figura 5) desde la cual se puede introducir información sobre el sistema operativo en la que se ejecutarán los scripts, la shell que se utilizará para interpretarlos así como la generación y descarga de los mismos.
Las opciones relativas a la gestión de usuarios sólo están disponibles para los profesores y son las típicas de cualquier sistema que contemple el

acceso de usuarios mediante cuentas: altas, bajas y edición de usuarios.

4. Resultados numéricos

Tras poner en funcionamiento la herramienta propuesta hemos obtenido los siguientes tiempos en la evaluación de prácticas de alumno:

- Configuración de las pruebas: aproximadamente 60 minutos (archivos de entrada y salida).
- Evaluación del alumno: 10 minutos

Tabla 2. Tabla de ratios de ahorro de tiempos respecto a Etapa1 (integra manual) y respecto a la Etapa inmediata anterior

RatioE1 /RatioEi-1	E1	E2	E3	E4
Configuración de las pruebas	-	-250 /-250	-150 /40	50 /66
Evaluación del alumno	-	83,3 /0	83,3 /0	83,3 /0
Ahorro por alumno	-	82,87 /82,87	83,1 /1,17	83,3 /1,18

En la Tabla 2 se recogen los ratios de ahorro de tiempo tanto en la tarea de preparación de los scripts como en la evaluación de las prácticas de los alumnos. Se ha tenido en cuenta un grupo de 100 alumnos. Cuanto más cercano a 0 sea el ratio mayor es el ahorro. Se puede observar que la herramienta propuesta (E4) obtiene el mayor de los ahorros de tiempo tanto respecto a la Etapa 1 (E1) como a su inmediata anterior (E3), principalmente en el paso de configuración de pruebas.

5. Conclusiones y trabajos futuros

Podemos decir que hemos logramos un ahorro significativo del tiempo invertido por alumno (un 83,3% de ahorro respecto al mecanismo manual de la Etapa 1) y aportado una solución generalista al parque de herramientas existentes.

Desde el punto de vista del alumno, entre los beneficios que aporta la herramienta presentada están el disponer de la calificación de su práctica en un tiempo más breve, así como reducir los errores en los parámetros de entrada que puedan influir negativamente en su evaluación.

Por otro lado consideramos que la propuesta puede ser ampliada para el uso por parte del alumno incluyendo las siguientes opciones:

- Auto-evaluación previa a la corrección con baterías de pruebas diseñadas por ellos mismos.
- Auto-evaluación de la práctica mediante las baterías oficiales proporcionadas por los profesores en la convocatoria correspondiente.

Referencias

[1] Brown, C., UNIX Distributed Programming, Prentice-Hall, 1994.

[2] Coulouris, G. et alii., Sistemas distribuidos, Conceptos y Diseños, Addison-Wesley, 2001.

[3] Kleiman, S., Shah, D., Smaalders, B., Programming With Threads, Prentice-Hall, 1996.

[4] Tittel E, et alii., Fundamentos de programación con HTML & CGI, Ed. Anaya Multimedia, Madrid 1996.

[5] Stevens, W. Richard, Advanced Programming in the UNIX Environment, Addison-Wesley, 2005.

[6] Hall, M., Brown, L., Core Servlets and JavaServer Pages, Prentice Hall and Sun Microsystems Press, 2003.

[7] J. Busquets, d. Gil, r. Perez y . Busquets, Entorno para la evaluación automática de prácticas de laboratorio en tiempo real, VIII Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica, 2008.

[8] Y. Farran Leiva, J. López Reguera y C. Hernández Rivas, Plataforma de evaluación automática de programas, Revista Ingeniería Informática, ISSN 0717-4195, Nº. 11, 2005.

[9] J.C. Rodríguez, M. Díaz, Z. Hernández, J.D. González. Hacia la Evaluación Continua Automática de Prácticas de Programación. JENUI, 2007.

[10] MOODLE. Moodle is a course management system (CMS). http://moodle.org/

[11] Ala-Mutka, K.; Uimonen, T. and Järvinen, H. Suppporting Students in C++ Programming Courses with Automatic Program Style Assessment. Journal of Information Technology Education. Vol. 3, pp 245-262. 2004.