Corrección automática de ejercicios de estructuras de datos a través de una plataforma de e-learning

Joan Surrell, Imma Boada, Josep Soler, Ferran Prados, Jordi Poch
Dpto. de Informática y Matemática Aplicada. Universidad de Girona
Campus de Montilivi, Edificio P4, 17071 Girona
joan.surrell,imma.boada,josep.soler,ferran.prados,jordi.poch@udg.edu

Resumen

Las estructuras de datos son una de las materias más importantes en las carreras de Informática. En este artículo presentamos el uso y las ventajas que ofrece la utilización de una plataforma de e-learning en el desarrollo de esta asignatura. Entre las ventajas que nos proporciona esta herramienta destacamos la corrección automática y on-line de los distintos tipos de ejercicios relacionados con la materia: estructuras dinámicas, lineales, árboles, etc. Además, permite al profesor conocer en todo momento el nivel de aprendizaje del alumno y detectar rápidamente posibles deficiencias. Con el uso de esta plataforma hemos logrado llevar un control exhaustivo del trabajo del alumno y motivar al alumnado con el objetivo de mejorar los resultados académicos.

Summary

Data structures are one of the most important subjects in Computer Science. In this paper we present an e-learning tool designed to give support at teaching and learning of this topic. We describe the tool, how it is used and the main benefits that provide us. Amongst them, there are the on-line and automatic correction of different types of exercises related to the subject, such as, dynamic structures, sequential or trees; the capability to track the student progress which allows the teacher to know about their deficiencies.

Palabras clave

E-learning, corrección automática, estructuras de datos, programación.

1. Introducción

El proyecto de construcción del Espacio Europeo de Educación Superior (EEES) y su estructuración a través del sistema europeo de créditos (ECTS), propone un nuevo modelo educativo en el cual los planes docentes y las metodologías a usar han de estar basados en el aprendizaje de los estudiantes, en contraposición con la tradicional programación centrada en la enseñanza del profesor. Se pretende un cambio en el modelo educativo universitario donde el estudiante sea el protagonista y donde lo más importante sea lo que debe aprender y como podemos garantizar que lo aprenda.

Conscientes de estos cambios, un grupo de profesores del Departamento de Informática y Matemática Aplicada de la Universidad de Girona nos propusimos desarrollar un sistema que facilitase y a su vez motivase al alumno en el aprendizaje de asignaturas con un gran componente práctico. De todos es conocido el fracaso del sistema tradicional donde un profesor, después de explicar una materia, facilita un listado de problemas a los alumnos, despreocupándose tanto de si los saben hacer, como de si realmente los hacen. Ante este problema decidimos desarrollar una plataforma para asignar problemas a cada alumno. Esta plataforma guarda las soluciones enviadas en la base de datos del sistema y las corrige automáticamente. En caso de ser erróneas se facilitan las indicaciones oportunas para su corrección. De esta forma podemos seguir el trabajo del alumno, detectamos sus deficiencias y además podemos realizar una evaluación continuada. En este artículo se presenta el uso que hacemos de esta plataforma y las ventajas que ofrece la corrección automática en la docencia de la asignatura de "Introducción a las estructuras de datos". Esta es una asignatura de segundo curso de las Ingenierías Técnicas en Informática (Gestión y Sistemas) y suele ser una asignatura con un alto número de abandonos y de suspensos.

El artículo se ha estructurado en siete secciones. En la sección 2 se presenta un breve análisis de otras experiencias relacionadas. En la

sección 3 se describen los contenidos que generalmente se imparten en una asignatura de estructuras de datos. En la sección 4 se describe la plataforma ACME con la que venimos trabajando y los tipos de problemas que soporta. En la sección 5 se presenta como se ha usado la plataforma en la asignatura de Estructuras de Datos de nuestra universidad. En la sección 6 se comenta la metodología usada y como utilizamos la plataforma para el seguimiento y evaluación de los alumnos. Finalmente, en la sección 7 se presentan las conclusiones.

2. Experiencias relacionadas

Se han llevado a cabo muchas experiencias consistentes en utilizar entornos virtuales para complementar la docencia presencial de distintas materias. Ya en el ámbito de las estructuras de datos y como experiencias presentadas en estas jornadas podemos citar [1-5]. Aunque en distintos contextos, el denominador común de todos estos trabajos es facilitar el aprendizaje de la materia. En [1] se centran en la generación de recursos educativos a ser usados, en [2] se presenta un entorno colaborativo para la docencia online de programación que permite la implementación del Aprendizaje Basado en Proyectos (ABP). En la misma línea de ABP en [3] se presenta una herramienta para la monitorización de una práctica para esta asignatura. En [4] se presenta una herramienta pensada ya para la evaluación continua y que permite la corrección automática de los ejercicios. En [5] se presenta una plataforma abierta que proporciona mecanismos para realizar una evaluación completa de un ejercicio de programación realizado en C o en Java para una asignatura de programación básica.

La corrección automática de programas informáticos es un tema sobre el que se han realizado numerosos estudios y proyectos. La finalidad de estos es reducir el tiempo de corrección de las prácticas de estas materias y por otro lado facilitar al alumno la validación de sus ejercicios. La estrategia de corrección automática más usada consiste en ejecutar el programa desarrollado por el alumno sobre un conjunto de test de pruebas. Cada uno de estos test está formado por unas entradas y la salida esperada. En caso que el programa del alumno pase todos los test de pruebas el programa se considera correcto. En esta línea destacamos los trabajos [6-8] que permiten la corrección y evaluación de forma automática de programas escritos en un determinado lenguaje de programación. Nuestro sistema sigue la metodología de estos trabajos permitiendo tanto la corrección como la evaluación continua de programas escritos en diferentes lenguajes. La descripción de la primera versión de nuestro sistema de corrección de ejercicios de programación se detalla en [9].

3. Estructuras de datos

Las asignaturas de Estructuras de Datos suelen impartirse en un segundo nivel de la carrera de Informática después de haber cursado una materia introductoria de programación básica. Los alumnos que acceden a este segundo nivel conocen las bases de la programación (recorridos, búsquedas, ordenación, acciones/funciones, paso de parámetros, etc.) y aspectos elementales de estructuras de datos (tablas y tuplas). En la asignatura de Estructuras de Datos se presentan las estructuras dinámicas encadenadas, las estructuras lineales (pilas, colas y listas) y los árboles binarios. A partir de estos conceptos pueden desarrollarse estructuras más complejas (diccionarios, árboles de búsqueda, grafos, etc.) y los algoritmos asociados a su tratamiento.

La forma tradicional de presentación de la materia dispone de sesiones teóricas en las que se exponen los conceptos del tema ilustrados con ejemplos clarificadores y se complementan con sesiones prácticas de problemas de clase para asentar los conceptos presentados, así como el desarrollo de uno o más programas de tamaño mediano en los que el alumno puede ver la aplicación práctica de los conocimientos adquiridos. El desarrollo de la práctica suele hacerse en lenguajes imperativos o con una componente de orientación a objetos (C, C++, Ada, Java) y en la teoría/problemas se acostumbra a usar un lenguaje algorítmico o una versión simplificada del mismo lenguaje de programación.

Esta materia es acumulativa, pues cada nuevo tema se basa en lo desarrollado en temas anteriores. Difícilmente un alumno comprenderá los temas finales de la materia (algoritmos sobre grafos o estructuras de datos compuestas) si no ha

asimilado los conceptos básicos (estructuras dinámicas y árboles binarios). Al tratarse de conceptos eminentemente prácticos, la resolución de problemas y la implementación práctica de programas que aplican los aspectos desarrollados en las sesiones teóricas, son elementos esenciales para asimilar la materia. Por esta razón la asimilación de los primeros temas condiciona mucho la superación de la asignatura. Sin embargo la organización clásica de la materia no permite al profesor conocer la evolución del grupo debido a la masificación y al desconocimiento del grado de éxito que tienen en la resolución de problemas (si llegan a resolverlos).

Queremos resaltar que en este tipo de asignaturas el trabajo personal del alumno y la resolución de problemas es fundamental. Por ello, el uso de plataformas de soporte a la docencia nos permite mejorar la docencia y el seguimiento del trabajo del alumno. Dadas las características de la plataforma ACME y las necesidades de las asignaturas de Estructuras de Datos podemos obtener muchos beneficios con su aplicación. A continuación, presentamos las principales características de la plataforma ACME.

4. La plataforma ACME

Para presentar la plataforma ACME en primer lugar describiremos las características principales de la misma. A continuación, presentaremos los diferentes tipos de problemas que soporta centrándonos en los que se usan en la asignatura de Estructuras de Datos.

4.1. Características de la plataforma

El desarrollo de la plataforma ACME se realizó con el fin de satisfacer los siguientes objetivos:

- Disponer de un repositorio único de problemas base de distintos tipos, donde los profesores puedan introducir y compartir sus problemas. Los problemas se catalogan según las materias, asignaturas, temáticas y nivel de dificultad de cada uno de ellos.
- A ser posible cada problema base dispone de varios enunciados y de parámetros variables, de forma que combinándolos se puedan hacer múltiples variaciones de un mismo problema. Cada problema base lleva asociado las pautas para su corrección automática. Por ejemplo, un problema de programación lleva asociado varios test de pruebas con unas entradas y las salidas esperadas para cada una de ellas.
- En cada asignatura en que se utiliza la plataforma, el sistema ofrece a cada alumno un cuaderno de problemas distinto. Este cuaderno está organizado por temas en los que se añaden los distintos problemas a medida que avanza el curso. El alumno envía soluciones a los problemas asignados y ACME los corrige automáticamente.
- El sistema guarda todas las soluciones enviadas por el alumno pudiendo ser consultadas por el profesor.
- En función del número de soluciones enviadas hasta obtener la solución correcta, el profesor puede hacer una primera valoración de las habilidades del alumno.
- El sistema debe soportar la corrección automática y on-line de los distintos tipos de problemas. Cada tipo de problemas tiene su módulo corrector específico.
- Todas las funcionalidades de la plataforma se deben realizar desde cualquier navegador.

A modo de ejemplo en las figuras siguientes se muestran dos interfaces habituales de trabajo de la plataforma ACME. En la Figura 1 puede verse, para un grupo de alumnos, el estado de resolución de los problemas de un determinado tema. También podemos ver el número de soluciones enviadas hasta llegar a la solución correcta. La Figura 2, visualiza el historial de un alumno, mostrando el número de tema / problema y los errores cometidos. Seleccionando un problema pueden verse las distintas soluciones enviadas.

Comentar que en su última versión ACME ya está totalmente integrado en la plataforma estándar Moodle, de forma que fácilmente se puede implantar en los centros que dispongan de ella. Desde esta plataforma los profesores pueden escoger en el apartado de actividades cualquier problema ACME.

4.2. Tipos de problemas

La estructura modular de la plataforma ACME facilita la integración de nuevos correctores de problemas con lo cual es muy fácil ampliar la tipología de problemas soportados.

Para la incorporación de un nuevo tipo de problema al sistema, sólo hay que definir la interfaz adecuada, la estructura y pautas de

corrección de los problemas y evidentemente desarrollar el corrector pertinente.

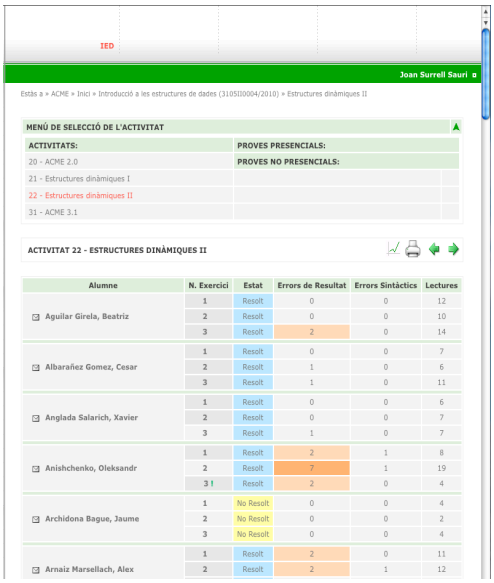


Figura 1. Estado de resolución de los problemas

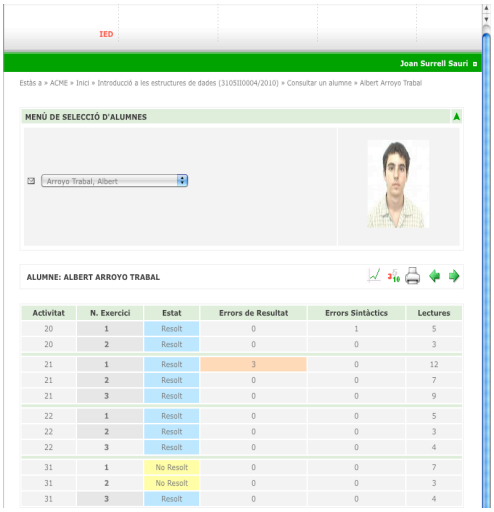


Figura 2. Historial de un alumno

En la actualidad la plataforma corrige problemas estándar, es decir los que se encuentran en cualquier plataforma como son los tipos test, elección múltiple o rellenar blancos. Además integra correctores automáticos para materias científicas/técnicas específicas de carreras universitarias. De esta manera ACME permite en estos momentos la corrección de:

- Cualquier problema que requiera un planteamiento matemático, desde derivadas, integrales, ecuaciones diferenciales, hasta problemas de física, economía, química, etc. [10].
- Corrección de ejercicios de programación escritos en distintos lenguajes informáticos y también en pseudo-código [9].
- Ejercicios de bases de datos, desde la corrección de diagramas entidad-relación, de esquemas de BD relacionales, sentencias SQL, álgebra relacional, etc. [11].

5. Aplicación de ACME en la asignatura de Estructuras de Datos

En una asignatura de estructura de datos es esencial que los alumnos aprendan a gestionar la información en memoria interna. Deben saber crear estructuras de tamaño variable, insertar datos, borrar, buscar y modificar. Además deben conocer las ventajas de las distintas maneras de estructurar los datos, así como los algoritmos particulares de cada representación.

Podemos considerar, por tanto, que para un alumno es fundamental ejercitarse en estructuras dinámicas y sus algoritmos característicos. Deben conocer las posibilidades de las estructuras lineales y el estudio de los árboles binarios con toda la problemática que conlleva la recursividad. La consolidación de estos aspectos básicos es fundamental para poder afrontar con éxito el estudio de técnicas más complejas (diccionarios de datos, técnicas de dispersión, árboles balanceados, grafos). La corrección de los problemas relacionados con esta materia requiere mucha dedicación por parte del profesor por lo cual no se pueden plantear tantos como sería deseable. Es en este punto donde la plataforma ACME nos ayuda.

A continuación se describen los tipos de problemas que permite la plataforma y que son usados en la asignatura de Estructuras de Datos.

5.1. Problemas de programación básica

Los alumnos que acceden a la asignatura de Estructura de datos ya han cursado una asignatura de programación en el nivel anterior. Con objeto de consolidar los conocimientos adquiridos anteriormente, se proponen dos o tres ejercicios de programación básica en las sesiones iniciales de la asignatura. De esta forma los alumnos pueden repasar conceptos adquiridos como son recorridos, búsquedas, codificación de clases y a la vez familiarizarse con el entorno ACME, en el caso de que no lo hayan usado anteriormente. El profesor aprovecha estas sesiones iniciales para avanzar suficiente materia en las sesiones teóricas para poder realizar los programas que realmente pertenecen a la materia. Podemos considerar que en esta fase ACME se usa como herramienta para la consolidación de conceptos básicos que se suponen ya adquiridos.

5.2. Problemas de estructuras dinámicas

Cuando ya se ha introducido suficiente materia de la asignatura, el estudiante debe enfrentarse a los primeros problemas básicos de estructuras de datos. Para ello se le plantean dos conjuntos de ejercicios con dificultad creciente en los que debe demostrar su capacidad para trabajar con estructuras de datos dinámicas. En estos ejercicios iniciales el programa principal lo proporciona el propio enunciado, así como la representación de la estructura de datos con la que se trabajará. El alumno debe codificar sólo los métodos indicados de la clase que encapsula la estructura dinámica. Una vez codificados los métodos por parte del alumno, y verificados en su ordenador, debe enviarlos mediante la página web del problema para que el sistema ACME pueda determinar si son correctos o no.

En la Figura 3 se muestra un ejemplo de problema. Como puede verse el enunciado del problema no es simplemente un texto sino que también se adjuntan detalles del código, como debe hacerse la entrada y salida de datos, la declaración de la estructura a representar así como representaciones gráficas que ayuden a la interpretación del problema. El alumno puede descargarse el código del problema desde la propia web, debe completarlo, verificarlo y enviarlo para su corrección por parte del sistema.

Para determinar el buen funcionamiento del problema, el profesor adjunta diversas secuencias de entrada de datos y sus correspondientes salidas que permiten verificar el funcionamiento del problema. Estas secuencias no son visibles por parte de los alumnos. Cuando un alumno envía un problema para su corrección el sistema compila el código, lo ejecuta y aplica las secuencias de verificación. Como resultado de este proceso, el sistema proporcionada al alumno de este tipo de problemas una respuesta que puede ser de tres niveles:

- Errores de compilación: no suelen ser habituales si el estudiante ha verificado inicialmente el problema en su ordenador, aunque pueden darse por confusión en el envío de fichero u otras circunstancias.
- Errores de ejecución: son los más habituales y suelen darse en casos no previstos por los alumnos (estructuras vacías, borrado del primer o último elemento,…). El sistema visualiza al alumno un ejemplo de entrada, la salida esperada y la obtenida, marcando con colores rojo y verde las diferencias detectadas. En la Figura 4 se muestra un ejemplo del mensaje de respuesta que recibe el alumno ante una solución incorrecta. Como puede verse se muestra en verde la solución correcta y en rojo el resultado obtenido por el alumno.
- Correcto: todos los test proporcionados con el enunciado son superados, el sistema acepta como correcto el problema.

EXERCICI: Omplir estructura de l'activitat Estructures dinàmiques II

Es disposa d'una estructura dinàmica circular simplement encadenada amb sentinella i un punter a l'inici tal i com es representa a la figura que hi ha al final del text. Implementar en C++ els mètodes nElements que mira quants elements hi ha a l'estructura i OmplirFinal que posa un conjunt de valors en ordre creixent al final de l'estructura. Els valors a posar són els enters consecutius des de 1 fins al valor indicat pel paràmetre. Usar les operacions anteriors en el següent programa (main.cpp):

```
#include <iostream>
#include "estructuraDinamica.h"
using namespace std;

void main() {
    estructuraDinamica e;
    int primer, anterior, n;

    cin >> n;
    while (n != 0) {
        e.OmplirFinal(n);
        cin >> n;
    }
    e.Llistar();
    cout << "N.ELEMENTS: " << e.nElements() << endl;
}
```

Cal tenir en compte que OmplirFinal no verifica si ja existeixen els elements afegits. De fet, en crides successives, es repetirian els valors afegits. El fitxer de capçalera és el següent (estructuraDinamica.h):

```
#ifndef TAD_estructuraCircularSentinella_h
#define TAD_estructuraCircularSentinella_h

struct node {
    int dada;
    node *seguent;
};

class estructuraDinamica { // Circular Sentinella
    node * inici;
public:
    estructuraDinamica();
    void OmplirFinal(int n);
    int nElements() const;
    void Llistar() const;
};
#endif
```

El constructor per defecte i el mètode Llistar són els següents:

```
estructuraDinamica::estructuraDinamica() {
    inici = new node; inici->seguent = inici; inici->dada = 0;
}

void estructuraDinamica::Llistar() const {
    if (inici != inici->seguent) {
        node * p = inici->seguent;
        while (p != inici) {
            cout << p->dada << " "; p = p->seguent;
        }
    }
    cout << "S:" << inici->dada << endl; // llista el sentinella
}
```

Cal descarregar el codi adjunt al problema.

Figura 3. Problema de estructuras dinámicas

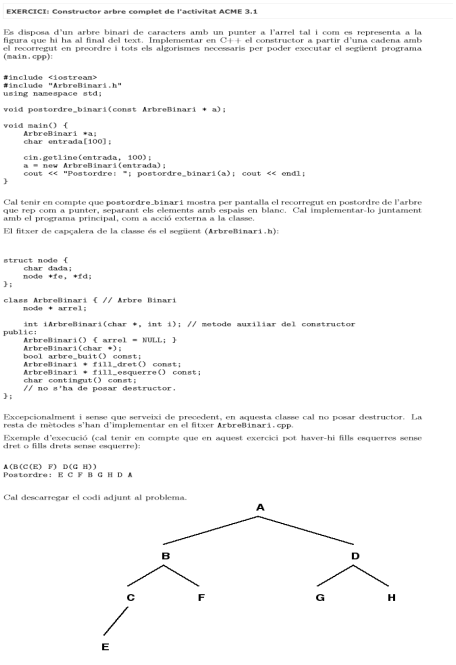
Figura 4. Mensaje de respuesta.

En este tema se plantean al alumno entre 5 y 7 problemas repartidos en dos módulos (2 o 3 de algoritmos básicos de estructuras dinámicas y 3 o 4 de algoritmos más complejos). Se realizan este tipo de problemas durante 3 semanas del curso.

5.3. Problemas de árboles binarios

Las estructuras de datos basadas en árboles merecen un capítulo aparte pues en ellas confluyen dos aspectos de gran dificultad para los alumnos: estructuras dinámicas y recursividad. Por esta razón el enfoque es parecido al de las estructuras lineales, pero con una dificultad progresiva de tratamiento. Cada problema dispone de 3 enunciados iguales, pero con datos de naturaleza cada vez más compleja: árboles con sólo dos niveles, árboles con diversos niveles pero con dos hijos en cada nivel y árboles generales que pueden tener diversos niveles con 1 o 2 hijos en cada nivel. Esta presentación de forma progresiva permite al estudiante refinar un algoritmo recursivo en caso que tenga dificultad en diseñarlo de forma completa.

Por lo que respecta al resto de los aspectos, estos problemas no se diferencian de los de estructuras dinámicas. El estudiante dispone de parte de código en el enunciado, debe completar uno o varios métodos de la clase, puede probar el código en su ordenador antes de enviarlo y, cuando lo envía, recibe una respuesta del sistema en los mismos 3 niveles descritos anteriormente (error de compilación, error de ejecución y correcto). En la Figura 5 se muestra un ejemplo de enunciado de árboles binarios. En él se puede observar el programa principal (incompleto), indicaciones de cómo debe proporcionarse la salida, la declaración de la clase a desarrollar y los métodos a implementar.



EXERCICI: Constructor arbre complet de l'activitat ACME 3.1

Es disposa d'un arbre binari de caracters amb un punter a l'arrel tal i com es representa a la figura que hi ha al final del text. Implementar en C++ el constructor a partir d'una cadena amb el recorregut en preordre i tots els algorismes necessaris per poder executar el següent programa (main.cpp):

```
#include <iostream>
#include "ArbreBinari.h"
using namespace std;

void postordre_binari(const ArbreBinari * a);

void main() {
    ArbreBinari *a;
    char entrada[100];

    cin.getline(entrada, 100);
    a = new ArbreBinari(entrada);
    cout << "Postordre: "; postordre_binari(a); cout << endl;
}
```

Cal tenir en compte que postordre_binari mostra per pantalla el recorregut en postordre de l'arbre que rep com a punter, separant els elements amb espais en blanc. Cal implementar-lo juntament amb el programa principal, com a acció externa a la classe.

El fitxer de capçalera de la classe és el següent (ArbreBinari.h):

```
struct node {
    char dada;
    node *fe, *fd;
};

class ArbreBinari { // Arbre Binari
    node * arrel;

    int iArbreBinari(char *, int i); // metode auxiliar del constructor
public:
    ArbreBinari() { arrel = NULL; }
    ArbreBinari(char *);
    bool arbre_buit() const;
    ArbreBinari * fill_dret() const;
    ArbreBinari * fill_esquerre() const;
    char contingut() const;
    // no s'ha de posar destructor.
};
```

Excepcionalment i sense que serveixi de precedent, en aquesta classe cal no posar destructor. La resta de mètodes s'han d'implementar en el fitxer ArbreBinari.cpp.

Exemple d'execució (cal tenir en compte que en aquest exercici pot haver-hi fills esquerres sense dret o fills drets sense esquerre):

```
A(B(C(E) F) D(G H))
Postordre: E C F B G H D A
```

Cal descarregar el codi adjunt al problema.

Figura 5. Problema de árboles binarios

Debido a la mayor dificultad que presentan los problemas de árboles se realizan menos ejercicios (sólo 3 o 4) y se desarrollan durante un período de 3 semanas.

5.4. Aplicación a otros problemas

La plataforma permite abordar la resolución de problemas simples como los usados en estructuras de datos básicas y otros más complejos que puedan evaluarse comparando sus entradas y salidas. Para problemas que requieren un conjunto de clases para su resolución, deben testearse individualmente las clases y, cuando ya funcionan todas ellas, realizar un test final del conjunto. Esta posibilidad puede aplicarse en el caso en que las clases son conocidas de antemano, pero no si la resolución del problema implica la determinación de las clases. Este tipo de problemas deben considerarse como problemas de diseño, no de programación, pues el diseño es el aspecto más importante a evaluar. El módulo ACME de programación no es aplicable a problemas de diseño.

6. Metodología usada

El uso de la plataforma ACME en esta asignatura ha pasado por diferentes etapas.

Las primeras experiencias son del curso 2004/05. En esta etapa inicial el uso de la plataforma ACME en la asignatura de Estructura de Datos estaba centrado en la parte de estructuras dinámicas y punteros. Se asignaban 3 problemas a los alumnos y estos debían resolverlos en unos límites de tiempo preestablecidos. Los primeros resultados fueron satisfactorios por lo cual se amplió el uso de la plataforma progresivamente a árboles binarios y a estructuras lineales. La idea de partida era usar la plataforma para resolver problemas sobre estos aspectos de forma exclusiva.

La introducción de la plataforma no implicó grandes cambios en la metodología docente. El proceso era simple: el profesor proponía un conjunto de problemas del tema. Los alumnos disponían de un período limitado de tiempo para resolverlos, pasado el cual eran evaluados por el sistema según el número de errores que la resolución del problema hubiera comportado.

Inicialmente sólo se disponía de unos pocos ejercicios de estructuras dinámicas, que fueron ampliándose con el tiempo. A continuación se desarrollaron ejercicios introductorios de árboles binarios, para completarlo finalmente con ejercicios de estructuras lineales. Aunque la experiencia parecía positiva por lo que respecta a la participación de los alumnos, no así la adquisición de conocimientos, pues algunos aspectos de la programación no pueden aprenderse con la sola verificación de las entradas y salidas de los programas. Estas limitaciones nos llevaron a una segunda fase que implicó un cambio en la metodología.

La experiencia adquirida en la primera fase de aplicación de ACME en la asignatura nos llevó a agrupar las estructuras dinámicas con las lineales. Hasta el momento estas estructuras se trataban de forma independiente. La forma de evaluación también se modificó. Se definió un sistema de evaluación híbrido. Por una parte había un conjunto de problemas que eran avaluados por el entorno ACME. Por otra parte, los profesores evaluaban unos pocos ejercicios presentados en papel (2 o 3 ejercicios). Las dos evaluaciones se complementaban.

El mismo sistema se adoptó para el caso de los árboles. Así pues, los ejercicios iniciales se desarrollaban en el entorno ACME y se complementaban con los ejercicios que debían presentarse en papel y que eran corregidos por el profesor.

Este nuevo sistema de evaluación nos presenta diferentes ventajas, entre ellas:

- El alumno resuelve de forma autónoma un elevado número de problemas.
- El profesor sólo debe evaluar un reducido número de los mismos.
- Los problemas que recibe el profesor han pasado varios test de prueba que permiten asegurar un funcionamiento mínimo y, por tanto, cierto tipo de errores ya se han eliminado.
- A pesar de que el entorno ACME funciona de forma autónoma, el alumno puede consultar al profesor la causa de los errores que le detecta el sistema y éste puede interaccionar con el mismo código con el que trabaja el alumno.
- La presentación de algunos ejercicios en papel ayuda a disminuir la inevitable copia de ejercicios. Aunque ACME dispone de un sistema de comparación de ficheros que permite detectar ejercicios iguales o bastante

parecidos, no se ha usado. Debe ser el propio alumno quien se responsabilice de su aprendizaje. Presentar ejercicios originales en la corrección automática permite al alumno afrontar con más garantías los ejercicios de evaluación manual y los exámenes. ACME es una herramienta de aprendizaje más que de evaluación. La evaluación sólo se proporciona para motivar al alumno.

Desde un punto de vista metodológico la integración de ACME en la asignatura no supuso grandes cambios. Fue necesario preparar el material y los problemas para poder trabajar a través de la plataforma. Desde el punto de vista académico los resultados que se obtuvieron también son satisfactorios. Como hitos destacables remarcamos el mejor aprovechamiento de la asignatura por parte de los alumnos y la mayor asistencia a las sesiones teóricas y de laboratorio. Además, conseguimos un mejor rendimiento académico, tal como muestran los datos de la Tabla 1 en la que se comparan los dos últimos cursos respecto al último curso (2003/04) en el que todavía no se usaba la plataforma ACME. En esta Tabla para cada curso diferenciamos entre los alumnos que han seguido la evaluación continuada de los que han realizado un único examen final. En ambos cursos puede observarse que los alumnos que optaron por la evaluación continua en la que se usaba ACME como parte de la evaluación obtuvieron mucho mejor resultado que los que optaron por el examen final. A remarcar que los alumnos que seguían esta última opción eran los que no podían asistir regularmente a clase.

Curso	Evaluación	Aptos	No aptos
2003/04		75	104
2008/09	Continua	58	15
	Ex. Final	3	39
2009/10	Continua	63	33
	Ex. Final	3	34

Tabla 1. Relación de alumnos que superan la asignatura en función de seguir evaluación continua o examen final

7. Conclusiones

En este documento se ha descrito el uso de la plataforma ACME como herramienta de corrección automática de problemas en una asignatura de estructura de datos. Se ha descrito la experiencia docente realizada desde el curso 2004/05 en la Universidad de Girona con objeto de potenciar las bases de la asignatura, especialmente los aspectos relacionados con las estructuras dinámicas y los árboles binarios. Como valoración final de la experiencia descrita se resalta la mejor consolidación de los conocimientos por parte de los alumnos, una mayor asistencia a las sesiones de teoría y laboratorio así como una mejora significativa del porcentaje de alumnos que supera la materia.

Referencias

[1] Sarasa A. *Generación de recursos educativos para la enseñanza de la asignatura de Estructuras de datos y de la información en un campus virtual.* JENUI 2004.

[2] Gallego M., Gortázar F. *EclipseGavab, un entorno de desarrollo para la docencia online de la programación.* JENUI 2009.

[3] Fiol G., Fiol C., Miró M. *La práctica monitorizada: una herramienta válida en el aprendizaje activo de la asignatura Estructuras de la Información*. JENUI 2007

[4] Moltó G., Galiano M., Herrero C., Prieto N., Sapena O. *Uso de herramientas TIC para la mejora de la interacción profesor-alumno, la evaluación continua y el aprendizaje autónomo.* Jornadas de Innovación UPV 2009: Metodologías Activas para la Formación en Competencias, 2009.

[5] Romero F.P., Serrano_J., Pérez de Inestrosa H. *CUESTOR: Una nueva aproximación integral a la evaluación automática de prácticas de programación.* JENUI 2010

[6] Higgins C., Gray G., Symeonidis P. and Tsintsifas A. *Automated assessment and experiences of teaching programming*. ACM Journal of Educational Resources in Computing. Vol 5(3), pp 1-21, 2005.

[7] Joy M., Griffiths N. and Boyatt R. *The BOSS on-line submission and assessment system*.

ACM Journal on Educational Resources in Computing. Vol 5(3). 2005.

[8] Saikkonen R., Malmi L. and Korhonen A. *Fully Automatic Assessment of Programming Exercises.* Proc. of Conference on Innovation and Technology in Computer Science Education (ITiCSE). pp 133-136, 2001.

[9] Boada I., Soler J., Prados F., Poch J. *A teaching/learning support tool for introductory programming course.* IEEE Proc. 5th International Conference on Information Technology based Higher Education and Training ITHET 2004.

[10] Prados F., Boada I., Soler J., Poch J. *Automatic Generation and Correction of Technical Exercises.* Proc. International Confernce on Engineering and Computer Education ICECE 2005.

[11] Soler J. *Entorno virtual para la docencia y la evaluación automática en Bases de Datos.* Tesis doctoral. 2010.