C++-OO como lenguaje introductorio a la programación

Fernando LLopis Pascual Ernesto Pérez López
Dept. de Lenguajes y Sistemas Informáticos
Universidad de Alicante
03080 Alicante
e-mail: llopis@dlsi.ua.es, ernesto@dlsi.ua.es

Resumen

En la carrera de Ingeniería Informática de la Universidad de Alicante se han venido utilizando como lenguajes introductorios a la programación el Pascal y el C. Actualmente estamos utilizando lo que llamamos C++-OO [7], que no es más que utilizar el lenguaje C++ pero sin introducir los aspectos referentes a la programación orientada a objetos. En el presente artículo se ve una comparativa de los lenguajes de programación mencionados, indicando nuestra experiencia sobre su uso.

1. Introducción

Quizá una de las cuestiones más debatidas, y que más controversias provoca en la enseñanza de la programación, es la elección del primer o primeros lenguajes de programación. Ello se puede comprobar al observar los diversos lenguajes de programación que se enseñan, o incluso que su estudio no se realice en clases de teoría sino únicamente en las clases prácticas.

Un buen lenguaje, utilizado para la introducción a la programación, debería cumplir las siguientes características:

- Debe ser un lenguaje de propósito general.
- Debe facilitar los hábitos de programación estructurada y programación modular.
- Debe posibilitar al usuario la definición de tipos de datos adicionales.
- Debe existir un estándar de dicho lenguaje.

Además sería recomendable que estuviera difundido, que existieran versiones de dicho lenguaje para los entornos más utilizados por los alumnos (Linux/Unix, Windows) y que sus requisitos mínimos de ejecución no fuesen especialmente elevados. También es positivo que dispusiesen de suficiente documentación y bibliografía.

La evolución y cambios que han sufrido los lenguajes de programación en la enseñanza han sido enormes. De hecho si miramos hacia atrás, no es extraño ver que lenguajes como el FORTRAN, COBOL y BASIC eran utilizados en asignaturas de programación en muchas universidades. Hoy en día, quizá solo el FORTRAN se sigue utilizando en la Universidad en ámbitos muy concretos.

Con respecto a los lenguajes introductorios, cabe indicar que el lenguaje Pascal ha sido el más utilizado [2], no obstante su uso hoy en día está empezando a decaer y su sustituto natural, el Modula-2 (o Modula-3) no ha conseguido ni mucho menos ocupar su lugar.

Pascal se diseñó para ser una herramienta para la enseñanza de conceptos de programación, no obstante herramientas basadas en lenguaje Pascal (más bien en Object Pascal, la versión orientada a objetos de Pascal) tienen actualmente una gran aceptación en la industria (Borland Delphi o su versión para Linux denominada Kylix). Del lenguaje Pascal cabe decir que es un lenguaje sencillo de aprender, y facilita la adquisición de hábitos de programación estructurada y la comprensión de otros lenguajes de programación no tan sencillos.

Quizá los mayores defectos de Pascal son:

- No disponer del concepto de módulos compilados de forma independiente, problema ya resuelto en Modula-2.
- El lugar donde se deben definir las variables del programa principal, tan alejadas de su código y que pueden ser utilizadas como variables globales por las funciones y procedimientos que se declaran a continuación.

- Pascal standard en cierta forma ha sido absorbido por el estándar de hecho, el Turbo-Pascal, tanto es así que hay más libros disponibles de Turbo-Pascal que de Pascal y algunos compiladores de Pascal han asumido algunas de las especificaciones del Turbo-Pascal.
- Otro de los problemas que tiene Pascal, es que difícilmente puede seguir siendo utilizado en otras asignaturas de cursos superiores, como Tipos Abstractos de Datos o Programación orientada a Objetos.

En algunas Universidades se ha optado por el lenguaje C, del cual cabe decir muchas cosas, sobre todo que es un lenguaje que no deja indiferente. Sus detractores dicen de él que es un lenguaje difícil de entender y que dificulta la utilización de la programación estructurada. Sobre el primer aspecto cabe indicar que estoy de acuerdo parcialmente con él, creo que es tan fácil o difícil de entender como el Pascal, exceptuando un aspecto que ciertamente complica mucho su comprensión, como es la gestión de los parámetros por valor y por referencia y la utilización en algunos comandos de la variable o de la dirección de la variable. Es este aspecto el que pienso que crea cierta frustración en los alumnos, y se desmoralizan al no saber claramente cuando deben colocar los símbolos & y * delante de cada variable. Por otro lado, cabe destacar que C tiene la ventaja de disponer una continuación o mejora del mismo, C++, C orientado a objetos, que puede ser utilizada en cursos posteriores.

Cuando hablamos de futuro, cabe hablar de dos lenguajes de programación, Ada y Java. Ada es un buen lenguaje de programación, que cubre muchos de los requisitos solicitados, pero tiene una complejidad que lo aleja de lo recomendable en un curso de iniciación y además está poco difundido y se dispone de poca bibliografía sobre él. Su viabilidad dependerá de la aceptación del nuevo estándar. Java es un lenguaje de tercera generación con características de programación orientada a objetos, multiplataforma, y especialmente preparado para el desarrollo de aplicaciones en Internet. Java está basado en gran medida en C++, pero con una gestión de memoria menos propensa a errores de programación. No obstante, Java tiene una eficiencia considerablemente menor que C++, debido a ser un lenguaje interpretado o seudointerpretado. Su popularidad ha experimentado un gran crecimiento en los últimos años, de hecho hay propuestas que lo tratan como un buen lenguaje introductorio a la programación.

En la Ingeniería en Informática de la Universidad de Alicante fundamentalmente se habían utilizado dos lenguajes, en los cursos introductorios a la programación, Pascal y C. Los motivos de la elección del primero eran debido a sus buenas características docentes, no obstante fue sustituido por el uso del lenguaje C por la extensión que tenía, para su uso en otras asignaturas, en cursos posteriores.

Actualmente hemos adoptado como lenguaje de programación el C++, ya que tiene una serie de ventajas sobre los lenguajes ya comentados, por un lado asume las ventajas del lenguaje C y disminuye sus inconvenientes en los aspectos más complicados tales como la gestión de parámetros, asemejándose más a Pascal en este aspecto, además ofrece una serie de mejoras que lo hacen más sencillo de utilizar que el C, en aspectos tales como: la entrada/salida, definición de tipos, gestión de memoria dinámica... Por otra parte, permite no trabajar obligatoriamente en programación orientada a objetos entendida como tal. De hecho lo que se va a estudiar del lenguaje C++ va a ser realmente un C++ -00, es decir un lenguaje C "mejorado".

A continuación detallaremos las ventajas e inconvenientes que hemos encontrado al utilizar este lenguaje con respecto a Pascal y C.

2. Comparativa

Tipos de datos básicos

En este aspecto es claro ganador el C++. En principio los tipos de datos básicos que permiten estos lenguajes son muy similares, entero, carácter y real (C dispone de dos tipos de reales). No obstante Pascal además dispone del tipo de datos lógico (boolean), mientras que en C se debe simular. Este tipo lógico ya ha sido incorporado a C++.

Además una mejora del C sobre el Pascal es que completa su definición de los tipos indicados con la posibilidad de utilizar modificadores de tipos (long, short, unsigned) que incrementan notablemente las posibilidades de definición de las variables. Esto puede permitir al alumno pensar y especificar de forma más correcta la información que va a utilizar. Además es útil como introducción a definiciones de tipos más complejas con las que se va a encontrar en otras asignaturas posteriores.

Sentencias de entrada/salida

Las funciones de entrada y salida en C son algo complicadas de utilizar y pueden producir algo de desesperación a los alumnos cuando las usan. El mayor problema que tiene este tipo de sentencias en lenguaje C es que su uso depende del tipo de variables sobre las que se apliquen, complicando mucho su entendimiento y correcta utilización. Mientras que en C++ y Pascal su uso es más sencillo.
Así un fragmento de programa que lea un entero en Pascal, C y C++ podría ser

Pascal

```
readln (a);
write ('El numero es ',a);
```

C

```
scanf ("%d",&a);
printf ("El numero es %d",a);
```

C++

```
cin >> a ;
cout << "el numero es " << a ;
```

Otro aspecto a destacar es que en la sentencia de lectura en C se debe incluir el famoso '&', pero sólo en algunos tipos de variables, ya que en las cadenas no se utiliza. El alumno debe saber la letra que especifica el formato del tipo de variable a leer, si este tipo debe utilizar el '&' o no. Y para empeorar las cosas, si en algún caso no utiliza correctamente el comando puede obtener al ejecutar el programa un mensaje de error tipo *Segmentation fault,* de difícil detección visual y que suele decrementar su amor por el C.

Estructuras de control

Las estructuras de control que disponen ambos lenguajes son muy similares, ambos disponen de estructuras de selección simple (if) y múltiple (case o switch-case). Quizá el formato de esta estructura que el lenguaje C realiza sea más propenso a producir errores, ya que es relativamente sencillo no utilizarla correctamente, debido al uso que se debe hacer de la sentencia break, la cual marca el final de las acciones para un caso determinado.

Ejemplo

```
//Programa de Bush o Puttin para gestionar
//las relaciones con el exterior

cin >> a;
switch (a) {
  case  1 : cout << "Feliz navidad amigo" ;
  case 2 : EnviarMisilesAtomicos();
 }
```

Es curioso comprobar el funcionamiento de este programa cuando uno de los líderes desee enviar felicitaciones de navidad.

Además el lenguaje C dispone de otro operador de selección (?) que quizá sea muy cómodo para programadores avanzados pero un poco complicado para utilizar en un curso introductorio.

Las estructuras iterativas disponibles en ambos, de condición inicial (while) o de condición final (repeat / do-while) son muy similares, la de contador (for) tiene muchas más prestaciones en lenguaje C, aunque es ligeramente más compleja.

Dentro de este apartado cabe hablar de la problemática que supone, para los que se inician en el lenguaje C, la utilización de las condiciones en las estructuras iterativas y de selección. C permite basar sus consideraciones para saber si una expresión es cierta o falsa en base al valor de la expresión (0 falso, distinto de 0 cierto). Esto puede obligar a definir un tipo lógico para facilitar el entendimiento del problema. Además el operador de igualdad que utiliza, el "= =" y dado

el hecho que expresiones como "while (a = 3)" o "if (b=0)" son consideradas como correctas (se asigna el valor a la variable y después se comprueba dicho valor para evaluar si la condición se cumple o no), provoca una gran cantidad de errores y bastante confusión en los alumnos, incluso aunque se haga especial hincapié cuando se explican en clase.

Así un programa del tipo

```
cin >> a ;

if ( a = 2)
  cout << "Has introducido un 2";
else
  cout << "No has introducido un 2";
```

Puede llevar mucho tiempo a un alumno (y a un no-alumno) detectar donde está el fallo inexplicable.

Ambos lenguajes disponen de operadores (nada recomendables dentro de la programación estructurada) que permiten cambiar la secuencia de ejecución del programa (goto en ambos y break y continue en C). Dentro de un curso introductorio a la programación estructurada no se deberían estudiar o en todo caso únicamente comentar su existencia y recomendar su no utilización. Hay que exceptuar el caso de la sentencia break cuando se utiliza en la estructura de selección múltiple (switch-case).

Tipos de datos estructurados

Los lenguajes en cuestión soportan los principales tipos de datos estructurados, tablas (arrays), registros, punteros y ficheros. La definición es muy similar, pero no así la utilización. En cierto modo Pascal trata las variables de tipo estructurado como variables de tipo básico, permitiendo por ejemplo asignar directamente arrays del mismo tipo (aunque es muy restrictivo cuando se trata de arrays compatibles pero no del mismo tipo), no así C cuyas operaciones permitidas se basan en cierta medida en la forma en la que implementa los tipos de datos estructurados (no permite asignar con el operador igualdad las tablas, sino que obliga a utilizar funciones especificas para realizar dicha tarea).

El C++ al incluir el tipo string ya permite utilizar operaciones de tipo asignación y comparación directa de algunos tipos estructurados.

Uno de los típicos problemas con los que se enfrentaba el alumno al utilizar cadenas en C era la obligatoriedad de utilizar funciones para su tratamiento.

```
char cadena1[15], cadena2 [15];

if (strcmp (cadena1,cadena2) == 0)
   printf ("Son iguales ");
```

Utilizando strings

```
string cadena1, cadena2;

if ( cadena1 == cadena2)
  cout << "son iguales"
```

Programación modular

Ambos lenguajes permiten la definición y utilización de subprogramas. No obstante Pascal se definió en base a una compilación monolítica, no existen los conceptos de módulos compilados de forma independiente (este problema ha sido subsanado en las implementaciones del lenguaje modernas, pero a costa de limitar su portabilidad). En el otro lado, C facilita la utilización de funciones y bibliotecas de funciones. Además en Pascal se confunde el diseño del lenguaje con la implementación subyacente, obligando a utilizar la directiva forward para declarar la cabecera de subprogramas que van a ser referenciados antes de que su código se haya definido, para dar cabida a los compiladores de un solo paso.

Por lo dicho anteriormente, parece más sencillo utilizar programación modular en C, pero hay otro aspecto fundamental que en Pascal se ha gestionado de forma mucho más comprensible (que no más eficiente) y éste es la definición de parámetros de las funciones por valor y por referencia. En Pascal se define en la cabecera la forma de utilizar la variable y la utilización de la variable dentro del cuerpo del procedimiento es totalmente transparente mientras que en C se obliga en el cuerpo del subprograma a conocer si

la variable fue recibida por valor o por referencia, debiendo utilizar en muchos momentos los operadores '*' y '&' provocando gran confusión a los alumnos. No obstante el lenguaje Pascal es quizá excesivamente estricto en la definición de tipos de datos estructurados a la hora de expresar los parámetros, lo que provoca además la poca versatilidad de uso de algunos subprogramas.

C++ soluciona en parte este problema, definiendo de forma más clara el tema de los parámetros por valor y por referencia.

Así se ha pasado del

```
/*Función que efectúa el producto de dos números*/
void multiplicar (int a, int *producto) {
   *producto = *producto * a;
 }

void main( )  {
  int a,b;
  cin>> a;
  cin >> b;
  multiplicar (a,&b);
}
```

a uno mucho más sencillo y de tipo Pascal que se utiliza en C++

```
/*Función que efectúa el producto de dos números*/
void multiplicar (int a, int &producto) {
   producto = producto *a;
 }

void main( )  {
  int a,b;
  cin>> a;
  cin >> b;
  multiplicar (a,b);
}
```

3. ¿Por qué no POO?

Hay muchas propuestas que defienden el uso de la programación orientada a objetos como mejor forma de introducir a los alumnos en la programación[1] [8]. Por contraste de opiniones, incluso con profesores que imparten la asignatura de Programación Orientada a Objetos parece claro que este paradigma no es trivial y puede ser bastante difícil de comprender. Quizá no queda claro si esto se debe a que los alumnos han empezado a "pensar" siguiendo la programación imperativa y el cambio a este paradigma es más complicado de lo que parece. No obstante parece que la forma de trabajo que propone la programación imperativa está más cerca del razonamiento humano (aunque evidentemente esta afirmación es muy discutible) y es el motivo que nos hace decantarnos todavía por el modelo tradicional.

4. Conclusión

El uso del C++-OO ha sido muy bien aceptado por los alumnos. Estos tenían grandes dificultades en comprender y utilizar algunos conceptos de C, sobre todo en lo referente al tema de paso de parámetros y uso de los famosos operadores '&' y '*'. No obstante preferían el C al Pascal porque consideraban a este un lenguaje "capricho" de los profesores y de poca utilidad cuando ambos se impartieron en un mismo curso. Bien es cierto que el lenguaje que se utiliza en las asignaturas de cursos posteriores es C o C++. Las peticiones e inquietud de los alumnos ahora están empezando a dirigirse a Java, debido a las noticias que reciben de ellos por el imparable efecto Internet. (Aunque Microsoft ya ha ofertado su competencia a Java a través de su C#).

Nuestro planteamiento futuro es valorar más en profundidad la posibilidad y conveniencia de que Java sea el lenguaje utilizado aunque nuestras dudas son muy serias al respecto.

Referencias

[1] Arnow, D.; Weiss, G. *Introducción a la programación con Java.* Addison-Wesley 2000

[2] Berlanga Llavori, Rafael y otros. *Introducción a la programación con Pascal.* Publicacions de la Universitat Jaume I. 2000

[3] Corbi, A.; Llopis, F et all. Fundamentos de Programación. Volumen I : Metodología.

Servicio de Publicaciones de la Universidad de Alicante, 1998

[4] Corbi, A.; Llopis, F et all. Fundamentos de Programación. Volumen II : Lenguajes Servicio de Publicaciones de la Universidad de Alicante, 1998

[5] Deitel, H; Deitel, P. Como programar en C/C++ Prentice Hall, 1999.

[6] Feuer,A; Gehani,N. A Comparison of the programming Languajes C and Pascal. ACM Computer Surveys Vol 14 Núm1. 1982.

[7] LlopisF.; Pérez, E. ; Ortuño, F. *Introducción a la programación.* Algoritmos y C/C++. Publicaciones de la Universidad de Alicante. 2000

[8] Mansfield, K; Antonakos, J. An introduction to Programing using C++. Prentice-Hall 1997.

[9] Pratt,T.;Zelkowitz,M. Lenguajes de programación. Diseño e implementación. Tercera edición Prentice Hall, 1998