Enseñando y Aprendiendo Programación Orientada a Objetos en los primeros cursos de Programación: la experiencia en la Universidad ORT Uruguay

Ing. Inés Kereki [1]
Universidad ORT Uruguay
e-mail: kereki_i@athenea.ort.edu.uy

Resumen

En este trabajo presentamos una experiencia realizada en la Universidad ORT Uruguay relacionada con la enseñanza inicial de la programación. Habitualmente, en las carreras de Ingeniería en Sistemas se comienza enseñando programación bajo el paradigma estructurado. En 1996, en la Universidad ORT Uruguay, comenzamos la enseñanza de la programación bajo el paradigma de orientación a objetos. Realizamos un análisis de la situación anterior y explicamos y justificamos, desde el punto de vista teórico e institucional, la situación actual. Explicamos cómo implementamos el cambio, describiendo los objetivos del curso, la elección del lenguaje de programación, los contenidos, la metodología y la forma de evaluación. Para analizar los resultados, estudiamos la producción de los alumnos y además les realizamos entrevistas en las cuales obtuvimos su opinión sobre su aprendizaje y la forma de resolver problemas, así como también su actitud hacia el estudio y el uso del computador.
Ofrecemos conclusiones surgidas del análisis de toda esta experiencia y de los datos obtenidos.

Introducción

La enseñanza inicial de la programación está cambiando: de comenzar enseñando programación estructurada a comenzar con programación orientada a objetos. De a poco, los programas académicos de las Universidades están encaminándose a esta relativamente nueva forma de programar.

En 1996 en la Universidad ORT Uruguay se realizó este cambio.

¿Por qué tomamos esta decisión? ¿Cómo la implementamos? ¿Qué resultados obtuvimos? ¿Qué piensan los alumnos? En este trabajo damos respuesta a estos planteos y ofrecemos conclusiones.

¿Por qué empezamos enseñando programación orientada a objetos?

Situación anterior

En planes de estudio anteriores (de carreras similares), los alumnos cursaban 4 semestres bajo el paradigma estructurado. En el quinto semestre se presentaba el paradigma de objetos.

[1]Docente y Coordinadora de Programación I y II en Ingeniería de Sistemas.
Universidad ORT Uruguay
Cuareim 1451. Tel. 902 1505. Fax. 900 2952. Montevideo, Uruguay
Internet: www.ort.edu.uy

El problema que veíamos era que, a esa altura, los alumnos no tenían capacidad de abstraerse de los temas básicos, es decir, ver los sistemas desde otra perspectiva diferente a la de cómo se programa.

Esto lo percibíamos en conversaciones informales con docentes y revisando trabajos producidos por los alumnos. Por ejemplo, en una prueba escrita, cuando se les pedía a alumnos de 5to. semestre que realizaran un seudocódigo para determinado algoritmo, la solución que planteaban era una implementación a nivel de código, escribiendo directamente las sentencias en algún lenguaje.

Hace algunos semestres, ya detectada esta dificultad, se hizo un primer intento de comenzar con la programación orientada a objetos.
El resultado no fue lo esperado, probablemente debido a una inadecuada elección del lenguaje de programación (Pascal). Se pretendió cambiar la forma de aprender a programar pero no se cambió el lenguaje. Tampoco se introdujeron cambios en la forma de dictado de la clase.

Un indicador de esta situación fue el análisis de los trabajos de los alumnos que, en su mayoría, no hacían un buen uso del paradigma de objetos.
También ese cambio planteó dificultades porque había alumnos cursando la carrera con el plan anterior de la materia y el plan nuevo.

Situación actual

En 1996 se creó la carrera de Ingeniería en Sistemas. Esta carrera tiene la finalidad de formar Ingenieros en Sistemas capaces, entre otras habilidades, de:

- diseñar y desarrollar sistemas de alta escala y complejidad;
- desempeñarse con éxito como desarrolladores de software, consultores independientes, líderes de proyectos o gerentes de sistemas y
- adaptarse al constante cambio en la industria e integrarse a equipos multidisciplinarios de investigación en el desarrollo de nuevas tecnologías.

La carrera de Ingeniero en Sistemas tiene una extensión de 5 años.
El requisito de ingreso es haber cursado y aprobado 6to. año de Ingeniería del Bachillerato Diversificado.

En el currículo se incluyen materias vinculadas a bases de datos, diseño de sistemas, estructuras de datos y algoritmos, ingeniería de software, programación y matemáticas. En particular, en los dos primeros cursos de Programación se enseña con el enfoque de objetos (utilizando Smalltalk). En el tercer y el cuarto semestre se utiliza C y C++ como lenguaje.

Fue un excelente momento para empezar (desde el comienzo mismo de la carrera) con el paradigma de programación orientada a objetos.
Adicionalmente, no se crearía problemas con alumnos en diferentes etapas de la carrera pues todos estarían empezando.

Luego de haber realizado un rastreo sobre el tema detectamos que en pocas universidades se están realizando experiencias de este tipo. Se relevó información vía Internet acerca de los programas de carreras similares de varias universidades y se confirmó que comienzan con el enfoque de programación estructurada.

Como *justificación teórica* a la conveniencia de utilizar programación orientada a objetos podemos citar varios autores.

En el paradigma de objetos, el objetivo del diseño pasa de modelar el comportamiento del mundo a modelar objetos que existen en el mundo y sus comportamientos individuales. La arquitectura de la aplicación está mucho más de cerca a la estructura del problema.

Como indica Rumbaugh (1991), el diseño orientado a objetos es una nueva forma de pensar en los problemas usando modelos sobre conceptos del mundo real. Lo fundamental es construir el objeto, el cual combina estructuras de datos y comportamientos como una entidad única.

Los objetos son relativamente fáciles de definir, realizar y mantener porque reflejan la modularidad natural de una aplicación. Con el refuerzo de la herencia, esos objetos pueden usarse de nuevo en aplicaciones futuras y así reducir la cantidad de código nuevo que debe escribirse.

Pressman (1992) refiere que, a diferencia de otros métodos de diseño, el diseño orientado a objetos da como resultado un diseño que interconecta los objetos de datos y las operaciones, de forma que modulariza la información y el procesamiento en vez de sólo la información. La naturaleza del diseño orientado a objetos está ligado a tres conceptos básicos: abstracción, modularidad y ocultación de la información.

La abstracción, según una definición citada por Pressman (1992 op.cit.), le permite a uno concentrarse en un problema al mismo nivel de generalización, independientemente de los detalles irrelevantes de bajo nivel. También el uso de la abstracción permite trabajar con conceptos y términos que son familiares al entorno del problema, sin tener que transformarlos a una estructura no familiar.

Refiere Winblad (1993) que las principales ventajas de la programación orientada a objetos son la mejora de la gestión de la complejidad y el aumento de la productividad del programador. Además mejora la flexibilidad y utilidad del software resultante.

Köllig (1996) señala que en los últimos 10 años ha habido un desplazamiento de los lenguajes procedurales a los orientados a objetos. Además afirma en su artículo que el mejor lugar para enseñar objetos es en un curso introductorio en primer año.

En resumen, según Winblad (1993 op.cit.), el enfoque de la programación orientada a objetos proporciona un nivel superior de abstracción de la programación a través de la construcción de objetos. Los lenguajes orientados a objetos simplifican el modelo de programación, hacen más exacta la solución y dan normalmente como resultado un menor número de líneas con un código menos complejo.

Como *justificación institucional*, debemos destacar algunas características importantes de nuestra universidad que favorecieron el cambio. En particular:

- se siente un alto grado de colaboración entre los docentes;
- se fomenta el intercambio entre los docentes de los resultados de los alumnos;
- la dirección está involucrada directamente en el proceso;
- los docentes tienen participación directa en el cambio;

- la evaluación del nuevo plan se hizo en el grupo de trabajo con los docentes y
- la planificación de los programas se llevó a cabo por los docentes del área.

¿Cómo implementamos la enseñanza de programación orientada a objetos en los primeros semestres?

El lenguaje elegido fue Smalltalk V. Algunas razones de esta elección fue la cualidad de ser el lenguaje más puro orientado a objetos (por lo cual no ofrece salidas laterales al paradigma) y de ser lo suficientemente simple como para un primer semestre.

Hubo varios aspectos a definir: objetivos del curso, contenidos, metodología, evaluación.

Los **objetivos** incluyen la capacitación del estudiante en los conceptos fundamentales del paradigma de objetos.

Los **contenidos** involucran temas básicos del paradigma: análisis, diseño y programación orientado a objetos. Temas tales como: herencia, modularidad, polimorfismo, diseño, re-uso, son tratados en el curso. Como libro de guía utilizamos 'Discovering Smalltalk' de LaLonde (1994).

La **forma de enseñar** cambió respecto a la forma tradicional de dar clase. Se introdujo el uso del TV Elite (PC con un televisor adicional y software que permite ampliar y resaltar elementos de la pantalla). En la clase se dispone de este equipo en forma permanente. Los temas se exponen e inmediatamente se muestran y prueban. Como agregado adicional, los alumnos, al finalizar la clase, se llevan en diskette todo lo realizado durante la misma, con la ventaja extra de que pueden concentrarse en atender a fondo lo que está pasando durante la clase sin la urgencia de copiar o tomar notas.

También se incluyeron clases en laboratorios, donde cada alumno tenía un PC disponible y una guía de trabajo detallada para seguir.

Se implementó además un WEB académico, en el cual se incluyó todo el material de prácticos del curso y listas de discusión de temas. El uso del correo electrónico permitió, además de fomentar la comunicación entre los estudiantes y docentes, la solución de problemas y dudas.

La **evaluación** del curso se realizó a través de parciales y trabajos obligatorios.
Los parciales son individuales y se realizaron en horas de clase. Los trabajos obligatorios se podían realizar en equipos de dos estudiantes y tenían una duración de, aproximadamente, un mes. El primer trabajo del primer semestre fue una jerarquía simple, para acercar a los alumnos a los conceptos de herencia y variables de clase e instancia. El segundo trabajo de ese semestre fue agregar al primero más elementos. El objetivo fue mostrar el re-uso de código.

En el segundo semestre, el primer trabajo fue una situación real, en la cual los alumnos tenían que completar la información sobre el dominio para poder decidir qué implementar, con la idea de presentarles un problema habitual con el cual nos enfrentamos los ingenieros de sistemas: definir los requerimientos del sistema, la funcionalidad, los límites. Para ello, debieron consultar a los usuarios del futuro sistema, en este caso, este papel fue representado por los docentes.

El segundo trabajo consistió en modificar los requerimientos del primero, de acuerdo a sugerencias por ellos mismos presentadas. La calidad del diseño, la importancia de una buena documentación y todos los demás aspectos que hacen a un buen sistema fueron realmente sentidos y comprendidos por los alumnos.

Con el objetivo, entre otros, de mostrar que no sólo se diseña y programa para uno mismo, se realizó otro trabajo más que denominamos Corrida Cruzada. En esta instancia se entregó a cada equipo el trabajo de otro equipo. Debieron presentar un informe sobre todos los aspectos: documentación, calidad del diseño, lógica, funcionamiento, etc.

Al final de cada semestre, se realizó un examen individual.

¿Qué resultados obtuvimos?

En los aspectos formales, los resultados fueron muy buenos. Al revisar los trabajos de los alumnos (obligatorios, parciales, exámenes) notamos una gran calidad en todo sentido: desde el análisis y diseño hasta la implementación y documentación.

El criterio de corrección de estos trabajos fue definido por pautas a utilizar por todos los docentes de la materia. Por ejemplo, para los trabajos obligatorios, estas pautas incluyen, entre otros, los siguientes aspectos:

- funcionamiento del sistema o programa;
- documentación;
- lógica;
- manual del usuario;
- listado (comentarios, indentación, nombres de variables) y
- presentación general

Los resultados fueron:

- aprobación de curso: 86% (valor levemente superior a los cursos similares a de carreras anteriores);
- aprobación de examen en el primer período: 70% (antes: 65%) y
- nivel de deserción de 1er. a 2do. semestre: 5% (antes: 25%).

¿Qué piensan los alumnos?

En el marco de una investigación que realizamos sobre cómo aprenden a programar los alumnos que comienzan su aprendizaje de la programación bajo el paradigma de objetos, hemos hecho entrevistas a los alumnos. La pauta de la entrevista trató acerca de cómo se han sentido en el curso y sobre cómo resuelven los problemas. Para esto último se les planteó un ejercicio y se les pidió que indicaran los pasos que seguían para su resolución. Adicionalmente, se les consultó sobre las actitudes que les fomenta el estudio de programación orientada a objetos y sobre cómo les resultó el uso del TV Elite.

A la pregunta "¿Cómo te sentiste en el curso de Programación I de Ingeniería?", los alumnos respondieron, (casi todos), que se sintieron bien durante los cursos. Gran cantidad de ellos tenían cierta experiencia con programación estructurada.

La mayoría de los que se encontraban en esta situación destacó que le resultaba más fácil el enfoque orientado a objetos. Muchos de ellos explicaron que trataron de hacer un paralelismo entre las dos formas, pero que les complicaba, por lo que decidieron 'hacer borrón y cuenta nueva'.

Destacaron, en general, la importancia de ver las cosas a nivel global, del poder que sentían que tenía y de la amplitud que se les presentaba.

Respecto a la forma de resolver problemas, muchos de ellos indicaron que trataban de separar lo importante primero y que iban tomando notas o apuntes de la letra para ubicar las clases. En su mayoría, se puede detectar una estrategia 'top-down' de resolución.

La actitud hacia el estudio fue positiva. La mayoría de ellos citaron que sienten que no hay límites, que investigan, que leen libros y que, cuando disponen de tiempo, 'navegan' en el Smalltalk para ver que más hay. También buscan en Internet información.

El uso del TV Elite fue muy provechoso según las respuestas dadas por los alumnos. El principal aspecto que destacan es la sensación de seguridad que da ver las cosas en el momento. También resaltan la agilidad que brinda a la clase y la importancia de poderse llevar lo hecho en clase en un diskette para seguir viéndolo y profundizando.

Conclusiones

La experiencia de 1996 fue muy positiva. Por supuesto, no es posible ofrecer conclusiones generales del tipo de: 'toda la enseñanza de la programación se simplifica si se empieza con programación orientada a objetos' puesto que recién se dictó el primer año de esta carrera y actualmente se está dictando el segundo.

Estamos convencidos de que este enfoque simplifica el aprender a programar y que los alumnos también lo perciben así. Todo el 'feedback' recibido (producción de los alumnos, encuestas, entrevistas) apoya esta apreciación.

Está siendo objeto de estudio la evolución de los alumnos, es decir, cómo les está resultando en los cursos posteriores de la carrera aprender programación estructurada luego de saber programar con objetos.

Referencias bibliográficas:

Köllig, M; Rosenberg, J. (1996). En: Proceedings of the 27th. SIGCSE Technical Symposium on Computer Science Education. Marzo 1996: 83-87, 190-194. USA: SIGS Publications.

LaLonde, W. (1994). *Discovering Smalltalk*. USA: The Benjamin/Cummings Publishing Company.

Pressman, R. (1992). *Ingeniería del Software. Un Enfoque práctico*. 3era. ed. USA: Mc. Graw Hill.

Rumbaugh, J.; Blaha, M.; Premerlani, W.; Eddy, F.; Lorensen, W. (1991) *Object-Oriented Modeling and Design*. USA: Prentice Hall.

Winblad, A.; Edwards, S.; King, D. (1993). *Software orientado a objetos*. México: Addison-Wesley Iberoamericana S.A.