

Patrones de Objetos de Aprendizaje para la Programación¹

Learning Object Patterns for Programming²

Ray Jones

Learning Technology Institute
London Metropolitan University
r.jones@londonmet.ac.uk

Tom Boyle

Learning Technology Institute
London Metropolitan University
t.boyle@londonmet.ac.uk

Resumen

El objetivo de este artículo es mostrar cómo los objetos de aprendizaje existentes, los cuales han demostrado ser exitosos previamente, pueden usarse para derivar patrones que podrían reusarse en el diseño de nuevos objetos de aprendizaje. Esto se demuestra en el contexto de objetos de aprendizaje que fueron diseñados para asistir a alumnos que no habían tenido experiencia previa en la materia en el dominio de la programación de ordenadores.

El aprendizaje de la programación de ordenadores presenta importantes desafíos a los que recién se inician en la computación, por lo que reusar diseños de objetos de aprendizaje exitosos tiene el potencial de beneficios pedagógicos reales.

Los patrones son una técnica de diseño conocida en los campos de la arquitectura y de la ingeniería de software. De un modo similar al que se usa en el diseño de software orientado a objetos, los patrones para el diseño de objetos de aprendizaje pueden derivarse de

¹ Traducción al español: Nora Lizenberg. Adaptación para RED: Miguel Zapata Ros

² Versión original en Inglés, publicada en el *Inderdisciplinary Journal of Knowledge and Learning Objects* – Volume 3, 2007 y disponible en <http://ijklo.org/Volume3/IJKLOv3p019-028Jones.pdf>.

Editor: Alex Koohang

Material published as part of this publication, either on-line or in print, is copyrighted by the Informing Science Institute. Permission to make digital or paper copy of part or all of these works for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage AND that copies 1) bear this notice in full and 2) give the full citation on the first page. It is permissible to abstract these works so long as credit is given. To copy in all other cases or to republish or to post on a server or to redistribute to lists requires specific permission and payment of a fee. Contact Publisher@InformingScience.org to request redistribution permission.

recursos de aprendizaje exitosos; estos patrones pueden luego reusarse en el diseño de otros nuevos.

Este artículo describe objetos de aprendizaje que fueron diseñados para ayudar a los nuevos programadores de ordenadores y cómo se extrajeron patrones de esos objetos de aprendizaje. Esto resulta en un pequeño catálogo de patrones de objetos de aprendizaje que tiene potencial para reusarse en la construcción de nuevos objetos de aprendizaje.

Palabras clave: objetos de aprendizaje, reuso, patrones, plantillas, diseño, autoría

Abstract

This aim of this paper is to show how existing learning objects, that have previously proved to be successful, can be used to derive patterns that could be reused in the design of new learning objects. This is demonstrated in the context of learning objects that were designed to aid the mastery of computer programming by learners who had had no previous experience in the subject.

Learning computer programming presents significant challenges to newcomers to computing so the reuse of successful learning object design has the potential for real pedagogical benefits.

Patterns are a well-known design technique in the fields of architecture and software engineering. In a similar way to their use in object-oriented software design, patterns for the design of learning objects can be derived from successful existing learning resources; these patterns can then be reused in the design of new ones.

This paper describes the learning objects that were designed to aid new computer programmers and how patterns were extracted from those learning objects. This results in a small learning object pattern catalogue that has the potential for reuse in the construction of new learning objects.

Keywords: Learning objects, reuse, patterns, templates, design, authoring.

INTRODUCCIÓN

El objetivo de este artículo es demostrar cómo los patrones de diseño pueden derivarse de los objetos de aprendizaje existentes y de modo que puedan reusarse en el diseño de otros nuevos. Esto puede traer grandes beneficios en el diseño de nuevos recursos ya que pueden basarse en aquellos que ya han demostrado ser efectivos. Reusar diseños efectivos de este modo es beneficioso en dos aspectos: en primer lugar, los nuevos recursos pueden heredar la efectividad de los originales; en segundo lugar, será más

rápido y fácil crear nuevos recursos a partir de un patrón en vez de comenzar desde cero.

Si se consideran los criterios apropiados para la reutilización y se asegura que los patrones y plantillas incorporan estas características, los recursos de aprendizaje derivadas de los patrones también serán reusables dentro de contextos similares a aquellos para los cuales han sido diseñados.

Como lo indica Goodyear (2005), hay una gran demanda insatisfecha de orientación en diseño, en particular, cuando se da en la forma de ideas personalizadas y reusables. La intención es que la implementación de patrones de diseño como plantillas reusables contribuya a satisfacer esta demanda.

El resto de este trabajo está estructurado en tres partes.

En la primera sección de este artículo se describen los antecedentes de este trabajo en términos de los recursos que se usarán a efectos de la demostración, los criterios para reusar los recursos de aprendizaje y los patrones de diseño.

La segunda sección se ocupa del análisis de los recursos (objetos de aprendizaje) y la identificación inicial de los patrones en ellos.

La tercera sección del artículo describe los patrones más formalmente como un pequeño catálogo de patrones de diseño.

ANTECEDENTES

Los Objetos de Aprendizaje

Los recursos utilizados para este trabajo han sido objetos de aprendizaje muy usados y probados que fueron diseñados para complementar un curso de primer año de un programa de grado en programación. En este curso, los objetos de aprendizaje demostraron ser una valiosa contribución para un avance significativo en el éxito de los estudiantes y una mejora sustancial en los índices de aprobación del curso. Su uso exitoso está bien documentado como parte de una metodología innovadora de aprendizaje distribuido o mixto (blended learning) para la enseñanza de programación a los estudiantes de primer año de computación. (Boyle, Bradley, Chalk, Jones y Pickard, 2003). Los objetos de aprendizaje ganaron un Premio Europeo al Software Académico en 2004 (EASA, 2004).

Los objetos de aprendizaje fueron diseñados por un equipo de expertos en el tema y un desarrollador multimedia, quien lo creó en Flash. Cada objeto de aprendizaje estaba autocontenido y estaba formado por una secuencia de cuadros, los cuales consistían en contenidos multimedia más pequeños, también autocontenidos. El objetivo de cada objeto de aprendizaje era explicar un concepto en particular de programación en el lenguaje de programación Java a través del uso de ejemplos. (Se pueden encontrar ejemplos de estos objetos de aprendizaje en <http://www.londonmet.ac.uk/ltri/learningobjects/examples.htm>.)

Los objetos de aprendizaje principales, su contenido y la secuencia de ese contenido fueron diseñados para implementar un método pedagógico en particular que se describe en la siguiente sección.

MÉTODO PEDAGÓGICO

En el nivel más simple la estructura de los objetos de aprendizaje tiene tres partes, cada una de las cuales se implementa a través de una o más “páginas” multimediales. La primera parte es una introducción, en donde se introduce en el nuevo tema al alumno; la segunda es la parte central del objeto de aprendizaje, en donde el alumno comprenderá en nuevo tema; y la tercera parte el alumno aplica su nuevo conocimiento.

La primera parte del objeto de aprendizaje se implementa en una sola página que contiene una descripción simple del tema a tratar.

La segunda parte del objeto de aprendizaje, el desarrollo, se puede expandir en tres sub-etapas: Captar, Aprender, Comprender.

La sub-etapa de Captar se lleva a cabo a través de una página con dos componentes – un comentario relacionado con un ejemplo familiar, visual del nuevo concepto “abstracto”. El objetivo aquí es captar la atención de los alumnos al brindar un ejemplo familiar y fácilmente comprensible del concepto que se va a tratar.

La sub-etapa de Aprender introduce el constructo de código objetivo como tercer componente. Para nuestros propósitos, se puede entender “aprender” como el punto en el cual el alumno obtiene una visión holística del concepto que se introduce. El objetivo de esta sub-etapa es ver (aprender) el efecto global del código antes de ir a los detalles. Se usa una animación para ilustrar el efecto global de ejecutar el código.

En la Comprensión es cuando el nuevo concepto se vincula a un nivel más lógico. Esto se tipifica como un análisis paso a paso del nuevo concepto o idea. La sub-etapa de Comprender se lleva a cabo a través de un cuadro que provee de un recorrido animado por el código del programa ilustrando el efecto de cada paso.

Los tres componentes se sincronizan a través del uso de botones incrustados dentro del comentario textual.

El objeto de aprendizaje finaliza con un ejercicio de construcción andamiada en donde los alumnos pueden, bajo condiciones de apoyo, crear activamente un fragmento de programa que ilustra el concepto que se está aprendiendo. Esto facilita la transferencia a las condiciones en la “vida real” en las cuales el alumno necesitará usar un programa editor/compilador estándar para construir y probar el código.

Es interesante notar que, a diferencia de otras metodologías, la última parte del objeto de aprendizaje no se ve como una prueba de cierre sino que más bien es una captación activa que apoya la transición hacia aplicar el nuevo concepto a una situación en la vida real.

Aunque este método está relacionado con el Aprendizaje Experiencial de Kolb (*Experiential Learning*, 1985), se desarrolló empíricamente y, como se mencionó anteriormente, ha demostrado ser una contribución efectiva para mejorar el éxito de los estudiantes en aprender habilidades de programación.

Es importante destacar que aunque anteriormente brindamos una explicación simplista de aprender y comprender que es suficiente para los propósitos de este artículo, Kolb (1985) da una explicación más detallada de estas ideas.

REUTILIZACION DE LOS OBJETOS DE APRENDIZAJE

Entre los argumentos a favor de reusar los recursos de aprendizaje se incluyen los económicos (Downes, 2001) y los de calidad (Jones, 2004). Económicamente, la reutilización de recursos en vez de su reinversión debe ahorrar tanto tiempo como esfuerzo y, por lo tanto, dinero. Y desde un punto de vista de calidad, cuanto más se reuse un recurso es más probable que sea de alta calidad simplemente porque más gente habrá estado expuesta a él y tendrá la oportunidad de brindar su opinión sobre él.

Sin embargo, para que los recursos de aprendizaje sean efectivamente reusados, deben estar diseñados con la idea de la reutilización: deben ser cohesivos y desacoplados de otros recursos (Boyle, 2003). En otras palabras, un objeto de aprendizaje reusable debe ser relativo a un único tema (es decir, cohesivo) y no estar vinculado innecesariamente a recursos externos (es decir, desacoplado), por lo que un objeto de aprendizaje debe ser “una unidad de aprendizaje independiente y autónoma” (Polsani, 2003).

REUTILIZACIÓN DEL DISEÑO

Existe otro tipo de reutilización, además de la simple reutilización del contenido: la reutilización del diseño. Éste es el tipo de reutilización central en el diseño y la programación orientada a objetos (*Object-Oriented Programming OOP*). En OOP, los objetos consisten en dos partes: los datos y las operaciones que pueden realizarse con esos datos. Por ejemplo, en un programa de computación relativo al empleo de personal, podrían haber objetos que representaran a empleados individuales. Tales objetos contendrían datos tales como el nombre del empleado, su salario y otros detalles personales, como ser la cantidad de horas trabajadas. Las operaciones que este objeto podría contener podrían ser darle al empleado un aumento en la paga, o calcular el salario mensual. Los objetos que representan cada uno de los empleados individuales de una compañía contienen el mismo tipo de datos y las mismas operaciones; lo que cambia son los valores específicos de los datos para cada empleado. De ese modo, el diseño de los objetos de cada empleado es el mismo.

En OOP el diseño de un objeto se define en una *clase*. Una clase es como una plantilla que define los datos individuales que un objeto contendrá (pero no su valor) y las operaciones que podrían realizarse con los datos. En un programa orientado a objetos los objetos individuales podrían crearse automáticamente, según se necesiten, a partir de una clase previamente definida. Este es el tipo más común de reutilización en OOP: no es la reutilización de contenidos sino más bien la reutilización del diseño, es decir, de la clase usada en un programa, tantas veces como sea necesario, para crear objetos nuevos. Aún más: una vez que una clase ha sido definida, puede almacenarse en un repositorio y reusarse en un programa de computación diferente.

La aplicación de este tipo de reutilización a los objetos de aprendizaje permitiría que un diseño exitoso se usara para producir nuevos objetos de aprendizaje que compartieran la misma estructura y métodos que los originales. Esta estructura podría, y debería por supuesto, incluir criterios para la reutilización de contenidos descritos en la sección previa.

Otro método para el diseño orientado a objetos es el uso de Patrones de Diseño. Los Patrones son similares en función a las clases pero su alcance es mayor, de modo que un patrón de diseño podría incorporar varias clases. Una vez más, los Patrones son un método de reutilización de diseño.

PATRONES

Los profesionales expertos usan su experiencia en la resolución de problemas pasados para seguir construyendo y crear nuevas soluciones en nuevas situaciones. “Si hay algo que los diseñadores expertos saben que no hay que hacer el resolver cada problema

desde los rudimentos. En su lugar, reusan soluciones que han funcionado en el pasado. Cuando encuentran una buena solución la usan una y otra vez. Tal experiencia es parte de lo que los convierte en expertos.” (Gamma, Helm, Johnson y Vlissides, 1995). Estas soluciones reusables a las que Gamma y los otros hacen referencia se conocen como patrones de diseño.

“Cada patrón describe un problema que sucede una y otra vez en nuestro entorno, y luego describe la solución central para ese problema de tal como que puedes usar esta solución un millón de veces, sin ni siquiera tener que hacerla de igual modo dos veces.” (Alexander, Ishikawa y Silverstein, 1977). Aquí, Alexander y los otros se referían a los patrones en arquitectura y planeamiento urbano pero las soluciones centrales en cualquier área del diseño se pueden describir en términos de patrones.

Son éstos los patrones que se extraerán de los objetos de aprendizaje referidos anteriormente para reusarlos en crear nuevos recursos de aprendizaje. El objetivo al hacer esto es ayudar a los diseñadores de objetos de aprendizaje brindándoles un conjunto de ideas de diseño en forma estructurada que articule claramente el problema de diseño con su solución (Goodyear, 2005). Esto ayudará a permitir que el diseñador produzca recursos de aprendizaje efectivos de un modo eficiente.

ANÁLISIS DE LOS RECURSOS EXISTENTES

Como se mencionó anteriormente, los objetos de aprendizaje del proyecto Java implementaron un método pedagógico particular que resultaba en una secuencia simple de “páginas” de contenido multimedial. Cada una de estas páginas era de un tipo o diseño especial y atendía a un propósito particular. Cada objeto de aprendizaje estaba formado por una secuencia de páginas similar. El control del usuario sobre la secuencia consistía en botones de avance (ir a la página siguiente), retroceso (ir a la página anterior) y rebobinar (ir a la primera página).

En un objeto de aprendizaje típico, la primera página era una página de HTML que brindaba el título y una descripción breve del tema a cubrir. Esto estaba seguido de páginas de distintos tipos, desde textos e imágenes estáticas simples a páginas formadas por secuencias sincronizadas de animaciones y texto.

A continuación, se indica el formato general de cada secuencia y una descripción de cada página en la secuencia.

1. La página de título

La página de título indica el nombre y una descripción breve del objeto de aprendizaje. También hay un enlace *inicio* en la página que ofrece una forma de empezar la

secuencia. Esto representa la primera parte del proceso pedagógico descrito anteriormente.

2. La página de concepto

En esta página el objetivo es introducir el concepto y explicarlo en un contexto con el cual el alumno ya esté familiarizado.

La página contiene dos ventanas verticales: en la de la derecha hay una descripción textual del concepto, y en la de la izquierda hay una animación que ilustra el concepto. Hay un botón en la ventana de texto que inicia la animación.

Esta página implementa la primera fase de la parte de “captar” en el proceso pedagógico.

3. La página de ejemplo

A continuación se demuestra el concepto en el contexto de lenguaje Java de programación. La página consiste en tres ventanas: una ventana contiene el programa Java; la segunda incluye una imagen gráfica que ilustrará la operación del programa Java; y la tercera es una ventana de texto que tiene una explicación del programa. Hay un botón dentro de la ventana de texto que inicia la animación.

Esta página implementa la segunda fase de la captación: es aquí en donde el alumno “aprende” el problema.

4. La página de explicación detallada

Aquí se da el mismo ejemplo que en la página anterior pero en este caso el código de programa se separa en líneas (o pequeñas secciones) por vez, en el orden de ejecución del programa.

Nuevamente, hay un botón en la ventana de texto que inicia la animación pero esta vez todas las ventanas están animadas. La animación ilustra la ejecución del código resaltando las partes apropiadas del código que se están ejecutando. Al mismo tiempo la animación también se divide en partes y la ventana de texto cambia a medida que cada parte describe la acción de una línea o sección del código de programa en particular.

Entonces, hay tres partes de la página que están en ciclos sincronizados: el código, el comentario y la animación.

Este es la parte final del proceso de captación, en donde el objetivo es la comprensión.

5. La página de práctica

Esta página final propone un problema al alumno. Para resolver el problema el alumno debe elegir fragmentos del código de programa a fin de construir un programa válido o un fragmento de programa. La página está construida en tres partes: en la parte superior hay instrucciones textuales para el alumno. En la parte inferior hay líneas de código de programa que deben ser seleccionadas. En el medio hay inicialmente un área en blanco en donde el programa resultante se mostrará.

Para construir el programa el alumno simplemente cliquea en las líneas de código correctas en orden. Éstas luego se copian en el área central de la página en donde se construye el fragmento completo de programa. El alumno recibe una devolución sobre las opciones que elige de modo que cualquier error pueda corregirse.

Esta página es el ejercicio andamiado al que se refiere el método pedagógico.

Puede verse a partir de esto que hay cinco tipos básicos de página. Una secuencia típica consiste de al menos cinco pasos como los descritos anteriormente. Sin embargo, en algunos objetos de aprendizaje los pasos 3 y 4 se repetían con distintos ejemplos de programación y, a menudo, había más de una página de práctica. Por lo tanto, un objeto de aprendizaje típico puede consistir en una secuencia de más de cinco pasos.

OBJETOS DE APRENDIZAJE COMO PATRONES

Uno de los objetivos principales de este trabajo es mostrar que los patrones de diseño pueden derivarse de objetos de aprendizaje existentes y exitosos de modo que puedan reusarse en el diseño de nuevos objetos de aprendizaje. Estos nuevos recursos incorporarían materiales nuevos o existentes y abarcarían temas distintos del original.

A partir del análisis anterior, se puede detectar un patrón en la secuencia de páginas que conforman cada objeto de aprendizaje. Los tipos individuales de páginas también pueden considerarse patrones ya que describen una actividad particular del alumno.

De este modo un patrón principal puede considerarse como una simple secuencia de páginas que introducen un nuevo concepto de programación. Este patrón puede considerarse como un patrón compuesto (es decir, formado por otros patrones).

Buschmann, Meunier, Rohnert, Sommerlad y Stal (1996) definen a los patrones de diseño en parte como describiendo un problema recurrente de diseño particular que surge en contextos particulares de diseño y que presenta un esquema ya probado para su solución. Los componentes del objeto de aprendizaje descritos aquí fueron extraídos de recursos de aprendizaje pre-existentes y exitosos, y se describen de una forma

tecnológicamente neutral de modo que puedan considerarse como patrones de diseño. Sin embargo, Buschmann et al también sugieren que los patrones deberían describirse en una forma consistente y estructurada en términos de su contexto (la situación que origina el problema), el problema mismo y la solución a tal problema. El consejo sigue la guía de Alexander et al (1997), quienes describe sus patrones de una forma similar y refieren a una colección de tales patrones como un lenguaje de patrones.

También es importante que los componentes de los objetos de aprendizaje que sean definidos usando estos patrones tengan muy en cuenta la necesidad de reuso y adaptabilidad. De este modo, estos aspectos deben ser tenidos en cuenta en cualquier documentación de un patrón.

Por lo tanto los patrones identificados hasta ahora deben documentarse como se describe anteriormente.

Un catálogo de Patrones “Objetos de Aprendizaje para Programación”

Las siguientes descripciones forman un pequeño catálogo de patrones para conceptos de aprendizaje de programación de computación.

Nuevo concepto de programación

Contexto: Los alumnos frecuentemente encuentran dificultades en la programación de computación. Si no se comprenden estos conceptos se dificulta el progreso en el aprendizaje de la carrera.

Problema: Para aprender un nuevo concepto de programación y ser capaz de aplicar este concepto en un programa, es muy valioso comenzar desde una posición familiar para el alumno. Una vez cómodo con el concepto, se pueden dar ejemplos de programación en el lenguaje que se elija. Tales ejemplos deben ser explicados paso a paso de modo que el alumno puede empezar a aplicar el nuevo concepto en sus propios programas. Finalmente, el alumno necesita poder practicar escribir programas propios, pero para ganar confianza en hacerlo es útil brindar un entorno sencillo en el cual el alumno pueda practicar sus conocimientos antes de enfrentar las complejidades de un entorno de programación real.

Solución: Usar una secuencia de páginas multimedia para implementar cada una de las etapas de aprendizaje descriptas anteriormente.

Página de Título

Contexto: El alumno está por aproximarse a un recurso de aprendizaje que no ha visto anteriormente.

Problema: Es útil que el alumno sea advertido de lo que viene para que se prepare y reciba instrucciones sobre cómo usar el recurso.

Solución: la página de título brinda un espacio para un título y una descripción del recurso.

Página de Concepto

Contexto: se le presentará al alumno un nuevo concepto de programación

Problema: Los conceptos de programación son abstractos y a veces de difícil comprensión para el alumno. Introducir un concepto similar de la vida diaria o de un tema que se sabe será familiar para el alumno resulta de ayuda para introducir el nuevo concepto.

Solución: Ofrecer un recurso que describe el concepto familiar y luego lo relaciona con el concepto de programación.

Página de Ejemplo

Contexto: El alumno está familiarizado ahora con el concepto a cubrir pero no está familiarizado con su particular implementación en el lenguaje de programación.

Problema: El alumno desconoce cómo se usará el nuevo concepto en el lenguaje que se está aprendiendo.

Solución: Se ofrece un ejemplo de implementación del concepto en el lenguaje que se está aprendiendo junto a un comentario y una animación que muestra lo que sucede cuando se ejecuta el ejemplo en el código de programa.

Página de explicación detallada

Contexto: Cuando se aprende un nuevo concepto de programación es valioso poder ver un programa ejemplo, o un fragmento de programa y tener su operación explicada de modo interactivo.

Problema: Para ilustrar adecuadamente el concepto o estructura de programación es útil “correr” el ejemplo. En una clase guiada por un docente esto se logra fácilmente, pero para el auto-estudio un recurso dedicado necesita ser diseñado de modo que permita que el alumno controle el ritmo y progreso a través del proceso de “corrido”.

Solución: Usar una metodología de “multi-frame”: un programa ejemplo, o un fragmento de programa se muestra en un cuadro en donde su ejecución se simula resaltando cada sección del código en el orden de ejecución. En la segunda ventana, el efecto de la ejecución del programa se demuestra a través de una animación, y en la tercera se brinda una explicación de la ejecución.

La animación ilustra el código que se está ejecutando resaltando las partes apropiadas del código que se están ejecutando. Al mismo tiempo, la animación también se fracciona y la ventana de texto cambia con cada paso para describir la acción de la línea en particular o sección del código de programa. Así, hay tres partes de la página que transcurren a través de una sincronización: el código, el comentario y la animación.

El alumno controla la operación a través de un botón en la ventana de explicación: esto inicia la animación sincronizada.

Página de Práctica

Contexto: El alumno está ahora familiarizado con el concepto y cómo se implementa en el lenguaje de programación que se está aprendiendo. Sin embargo, los alumnos deben poder escribir sus propios programas y esto habitualmente requiere mucha atención al detalle: un simple error de escritura puede causar un error.

Problema: Los programadores noveles a menudo encuentran que los estrictos requisitos de sintaxis son un obstáculo para la comprensión. Errores simples de sintaxis hacen que el programa se vuelva inútil y esto puede causar frustración. Esta frustración y el tiempo necesario para corregir errores de sintaxis no tienen estrictamente que ver con el aprendizaje del nuevo concepto, pero reduce la efectividad de la experiencia de aprendizaje.

Solución: Brindar un método simple de crear un programa sintácticamente correcto sin la necesidad de escribir un programa real. Esto evita el problema de los errores de sintaxis. Los alumnos pueden estar ahora seguros de haber entendido el nuevo concepto y están en mejor posición para crear sus propios nuevos programas.

Conclusión y Trabajo Futuro

El análisis de los objetos de aprendizaje originales ha resultado en la identificación de un patrón pedagógico que representa un marco simple en el cual una serie de actividades pueden ubicarse. Las actividades mismas son también patrones y se implementan como plantillas en las cuales se puede insertar el contenido apropiado.

Estos patrones representan un lenguaje patrón simple para objetos de aprendizaje que apoyan la enseñanza y el aprendizaje de nuevos conceptos de programación. También ofrecen guía reusable y personalizable en el diseño de nuevos recursos de aprendizaje que son una contribución a las demandas insatisfechas referidas anteriormente. Pueden usarse para producir recursos totalmente nuevos o, alternativamente, recursos existentes basados en patrones pueden personalizarse para, por ejemplo, actividades de reordenación o para incluir nuevas actividades que aumenten o reemplacen las existentes. Esta personalización brinda la oportunidad para los objetos de aprendizaje basados en estos patrones de adaptarse a distintos requerimientos.

Parte del trabajo resultante de este estudio será diseñar herramientas de autor de objetos de aprendizaje que utilicen los patrones antes descritos y que permitan que los diseñadores creen rápidamente nuevos recursos a partir de los patrones que han sido descubiertos, o personalizar los objetos de aprendizaje existentes para sus propios propósitos (Boyle, 2006).

A pesar de que los patrones, aquí, sólo están relacionados con el diseño de objetos de aprendizaje para apoyar la enseñanza y el aprendizaje de programación, se puede vislumbrar que técnicas similares podrían usarse para producir un lenguaje de patrones y herramientas que apoyen otras áreas temáticas, diferentes tecnologías y diferentes metodologías pedagógicas. Por ejemplo, Koohang y Harman (2005) vinculan fuertemente el aprendizaje en línea y la pedagogía constructivista, como también lo hace Nash (2005). Los diseños de objetos de aprendizaje que están basados en principios constructivistas pueden ser excelentes fuentes de patrones de diseño que permitirían a los autores de objetos de aprendizaje producir más fácilmente recursos de aprendizaje pedagógicamente sólidos. Adicionalmente, Cebeci y Tekdal (2006) discuten el uso de podcasts como objetos de aprendizaje y concluyen en la necesidad de rediseñarlos para su uso instruccional; el potencial de uso de los patrones de objetos de aprendizaje que incorporen estructuras pedagógicas apropiadas es claro.

Así el alcance de los patrones de objetos de aprendizaje es potencialmente más amplio que el estudio de caso citado aquí. El trabajo futuro apuntará a la técnica en otras áreas temáticas y enfoques pedagógicos.

Artículo finalizado en 2007

Jones, Ray y Boyle, Tom (2007). Patrones de Objetos de Aprendizaje para la Programación. RED, Revista de Educación a Distancia. Número Monográfico X – 30 de noviembre de 2009. Número especial dedicado a “Patrones de eLearning y Objetos de Aprendizaje Generativos”. Consultado el [dd/mm/aaaa] en <http://www.um.es/ead/red/M10>

Referencias

- Alexander, C., Ishikawa, S. & Silverstein, M. (1977). A pattern language: Towns, buildings, construction. Oxford: Oxford University Press.
- Boyle, T. (2003). Design principles for authoring dynamic, reusable learning objects. *Australian Journal of Educational Technology*, 19 (1), 46-58
- Boyle, T., Bradley, C., Chalk, P., Jones, R. & Pickard P. (2003). Using blended learning to improve student success rates in learning to program. *Journal of Educational Media (Special Edition on Blended Learning)*, 28 (2-3), 165-178
- Boyle, T. (2006). The design and development of second generation learning objects. E. Pearson & P. Bohman (Eds.), *Proceedings of Ed-Media 2006 World Conference on Educational Multimedia, Hypermedia & Telecommunications* (pp. 2-11), June 26-30, 2006; Orlando, Florida.
- Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., & Stal, M. (1996). *Pattern-oriented software architecture*. John Wiley.
- Cebeci, Z. & Tekdal, M. (2006). Using podcasts as learning objects. *Interdisciplinary Journal of Knowledge and Learning Objects*, 2, 47-57. Disponible en <http://www.ijklo.org/Volume2/v2p047-057Cebeci.pdf>
- Downes, S. (2001). Learning objects: Resources for distance education worldwide. *International Review of Research in Open and Distance Learning*, 2 (1).
- EASA. (2004). European Academic Software Awards (EASA) website. Última consulta: enero 18, 2007. Disponible en <http://www.bth.se/lab/easa.nsf>
- Jones, R. (2004). Designing adaptable learning resources with learning object patterns. *Journal of Digital Information*, 6 (1), Artículo No. 305.

Kolb, D. (1985). *Experiential learning: Experience as the source of learning and development*. Prentice Hall.

Koohang, A. & Harmon, K. (2005). Open source: A metaphor for e-learning. *Informing Science Journal*, 8, 76-86. Disponible en: <http://inform.nu/Articles/Vol8/v8p075-086Kooh.pdf>

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995). *Design patterns: Elements of reusable object-oriented software*. Addison-Wesley.

Goodyear, P. (2005). Educational design and networked learning: Patterns, pattern languages and design practice. *Australian Journal of Educational Technology*, 21(1), 82-101. Disponible en <http://www.ascilite.org.au/ajet/ajet21/goodyear.html>

Nash, S. (2005) Learning objects, learning objects repositories, and learning theory: Preliminary best practices for online courses. *Interdisciplinary Journal of Knowledge and Learning Objects*, 1, 217-228. Disponible en: <http://ijklo.org/Volume1/v1p217-228Nash.pdf>

Polsani, P. (2003). Use and abuse of reusable learning objects. *Journal of Digital Information*, 3 (4), Artículo No. 164, 19 febrero. Disponible en: <http://jodi.ecs.soton.ac.uk/Articles/v03/i04/Polsani/>