Una propuesta para facilitar la generación de tutoriales

Zulema B. Rosanigo[1]; Alicia Paur[2]; Pedro Bramati[3]; Alfredo Ortega[4]; José P.Cerra[5]

Facultad de Ingeniería – Sede Trelew – U.N.P.S.J.B. Te-Fax (02965) 42 84 02

Resumen

En este artículo se presentan los avances del proyecto de investigación "Construcción de tutoriales basados en componentes reusables", Nro. 383 de la Facultad de Ingeniería de la U.N.P.S.J.B. cuyo objetivo es buscar nuevas alternativas que mejoren la calidad educativa, facilitando al docente la construcción de nuevos tutoriales, del tipo enseñanza "paso a paso", para que el alumno pueda ejercitarse tanto como lo necesite.

Palabras claves

Software educativo – Tutorial – Componente reusable - Framework

Introducción

Ciertas áreas del conocimiento pueden adquirirse por medio de ejercitación, siguiendo una secuencia de pasos específicos. Ejemplo: construcciones geométricas, capacitación de uso de una herramienta, resolución de problemas físicos, matemáticos, lógicos, etc.

Para facilitar este tipo de aprendizaje, los tutoriales son una herramienta de gran ayuda. Normalmente un tutorial para enseñanza del tipo "paso a paso", parte de un enunciado ejemplo y lo desarrolla mostrando las etapas intermedias. Si se quisiera variar el ejemplo implicaría desarrollar nuevamente todas las etapas.

Si queremos lograr un aprendizaje significativo, debemos contemplar las necesidades y características individuales del alumnado, requiriéndose una variedad de ejemplos que cambien según la destreza adquirida por cada alumno y aún así, podría no ser suficiente para alguno de ellos.

Confeccionar tutoriales con estas características, resultaría una tarea muy laboriosa y demandaría demasiado tiempo.

Nuestro proyecto de investigación, denominado "Construcción de tutoriales basados en componentes reusables" tiene por finalidad facilitar al docente la construcción de tutoriales del tipo enseñanza paso a paso, a partir de algunos componentes desarrollados especialmente para el dominio de aplicación, para que el alumno pueda ejercitarse tanto como lo necesite.

---

[1] Ingeniera Civil – Analista Programador Universitario– Magister en Ingeniería de Software - Investigador Cat. III - Profesor Asociado D.E. brosanigo@infovia.com.ar

[2] Analista Programador Universitario - Investigador Cat. V – J.T.P. D.S.E. - apaur@topmail.com.ar

[3] Ingeniero Civil – Investigador Cat. IV - Profesor Titular D. S.E.. bramati@infovia.com.ar

[4] Analista Programador Universitario - Alumno de la Licenciatura en Informática cachito@ortega.net.ar

[5] Alumno de Analista Programador Universitario– jopa@infovia.com.ar

Cada tutorial creado con esta herramienta puede ser usado para la creación de otro, debiéndose adaptar en forma inteligente al nuevo contexto. Esta inteligencia para adaptarse es la que le permitirá variar fácil y dinámicamente los ejemplos según las necesidades del alumno.

Estado del proyecto

Se ha definido una arquitectura flexible para la generación de tutoriales, quedando establecidos los aspectos que tienen mayor impacto en el diseño y definidos los parámetros, cualidades y características de los componentes de manera que resulten reusables.

Entre los requerimientos funcionales que nos hemos propuestos, podemos mencionar:

- Contemplar dos modalidades de uso bien definidas:
  - creación de tutorial (el docente prepara su material didáctico).
  - uso de un tutorial (el alumno ejercita).
- Posibilitar la creación de tutoriales en forma natural y sencilla.
  - implica automatización de tareas.
- Adaptación inteligente a un nuevo contexto.
  - el tutorial que se cree, debe poder convertirse en un nuevo componente del dominio y ser utilizado para la generación de otro tutorial.
  - requiere mecanismo especial de persistencia.
- Permitir la ejecución completa o parcial, y el avance o retroceso según los requerimientos del alumno.

Teniendo en cuenta esto, definimos un marco (framework) para el generador de tutoriales, el cual forma el esqueleto de la aplicación, provee la funcionalidad común, define las abstracciones fundamentales y sus interfaces, establece las interacciones entre los objetos, dejando en determinados lugares espacios en blanco o puntos de articulación (hot spot). Cada espacio se refiere a los aspectos de los tutoriales que pueden variar de una aplicación a otra, y es allí donde el framework debe proveer flexibilidad.

Para realizar una aplicación de generación de tutoriales en un dominio en particular, basta con incluir los componentes que satisfagan las situaciones requeridas por la aplicación. Por ejemplo, si pretendemos crear un generador de tutoriales en el dominio de la Geometría deberíamos incluir componentes que representen los entes geométricos, sus gestores y las operaciones primitivas que pueden realizarse sobre ellos.

La definición de interfaces, tanto de usuario como entre módulos y componentes, cobra un papel sumamente importante en este proyecto, ya que uno de los objetivos que se persigue es "Facilitar la construcción de herramientas educativas utilizando componentes reusables". Y en ese "facilitar la construcción ...", pretendemos lograr una herramienta que pueda ser utilizada por cualquier docente, con o sin conocimientos informáticos, y que mediante mecanismos sencillos de selección y comunicación de componentes existentes (ya desarrollados), pueda generar un nuevo tutorial.

El modelo arquitectónico en que se basa la aplicación es el Model-View-Controller[6], que divide el problema en tres componentes: el *modelo* que contiene el corazón de la funcionalidad, la *vista* que despliega la información al usuario y el *controlador* que maneja la entrada del usuario. La vista y el controlador definen la interfaz gráfica. De esta manera se desacoplan los problemas y se logra mayor reusabilidad.

---

[6] Buschmann F., Meunier R., Rohnert H., Sommerland, P., Stal, M.*Pattern-Oriented Spftware Architecture: a system of patterns.* Ed. Wiley 1996

Se identificaron, definieron e implementaron los componentes comunes a todos los tutoriales. Se están desarrollando componentes específicos del dominio de la Geometría que representan entes geométricos: Punto, Recta, Arco y componentes que modelan transformaciones o cálculos: intersección entre figuras, trazados, etc.

Se ha desarrollado un prototipo bastante completo de la aplicación y se están realizando pruebas, las cuales hasta el momento son satisfactorias.

Para asegurar que la arquitectura propuesta es flexible y reusable, se están haciendo pruebas con componentes de dos dominios distintos, el de la Geometría y el de la Aritmética, y desarrollados por diferentes personas. Hasta ahora las pruebas resultaron muy satisfactorias, ya que se pudo cambiar de dominio de aplicación sin necesidad de modificación en los componentes comunes del generador de tutoriales. Esto valida que las interfaces de comunicación entre componentes han sido bien establecidas y respetadas.

Puesto que un paso de un tutorial a crear puede ser tanto un componente primitivo (ya desarrollado) como un tutorial creado con anterioridad por el docente, nos enfrentamos a la situación en que no sólo el alumno debe aprender, sino que el sistema que estamos diseñando también debe "aprender" a utilizar los tutoriales creados, de la misma manera que sabe utilizar las primitivas. Esto implica que:

a. cada tutorial debe tener un mecanismo de persistencia.
b. cada tutorial debe autoejecutar los pasos en función de las entradas del nuevo contexto en que se ejecuta y las salidas de los pasos anteriores.

Este es un aspecto fundamental para garantizar la reusabilidad de un tutorial. Se analizaron diferentes aproximaciones de solución: objetos serializables, base de datos de órdenes y propiedades, pero se descartaron por no satisfacer todos los requerimientos. En este momento se está probando el modelo que creemos que cumple con todos los requerimientos y al que denominamos "Paso Inteligente", ya que representa un paso del tutorial y encapsula el conocimiento y provee los mecanismos que le permiten adaptarse "inteligentemente" al ser utilizados en nuevos contextos.

Herramientas y lenguaje de programación utilizados

La principal ventaja de Internet sobre otros medios de comunicación es la interactividad y uno de los recursos que permite explorar mejor la interactividad en la web es el lenguaje Java, a través de aplicaciones pequeñas y seguras (applets). Con Java se pretende buscar una solución de alta calidad del producto final pero manteniendo un costo de desarrollo razonable.

Una manera de ayudar a cumplir con este objetivo es maximizar el reuso y posibilidad de evolución. El reuso produce reducción de tiempo y costo e incrementos de calidad, en la medida que el desarrollador pueda encontrar, utilizar y adaptar al nuevo contexto, aquellas soluciones que ya han sido probadas y usadas exitosamente.

Se trabaja con Java 1.4 que provee, entre otros:

- Java2D: Utilizada para la representación de todos los componentes.
- JavaBeans: Estándar que deben cumplir todos los componentes.
- Reflexión: Utilizada para obtener información y manejar a los componentes. En las últimas versiones se reemplazo en gran medida el uso de reflexión por el uso de instrospectores de objetos provistos por las librerías de manejo de Beans.
- Clases para el manejo de XML: Método estándar para almacenamiento de datos.

Conclusiones

- La posibilidad de individualizar el proceso de enseñanza-aprendizaje que nos ofrece la computación, permite optimizar las capacidades de cada individuo al no tener que imponerles a todos los estudiantes el mismo ritmo:
    - Se evita el aburrimiento en los alumnos más hábiles o rápidos.
    - Se evita frustración y decepción en dicho proceso para quienes tienen mayores dificultades.
- Con la utilización de un tutorial, el alumno sigue el proceso "paso a paso" en la resolución de un problema.
    - Se resuelve la problemática generalizada de encontrar la resolución impresa en un solo paso, en donde se han obviado todos los procesos intermedios.
- Con este tipo de herramientas, el docente puede crear tutoriales en forma amigable y sencilla, y el alumno puede aprender y practicar a su propio ritmo y necesidad.

Bibliografía

[1] Buschmann F., Meunier R., Rohnert H., Sommerland, P., Stal, M. *Pattern-Oriented Spftware Architecture: a system of patterns.* Ed. Wiley 1996

[2] Cooper, James W. - *Java Design Patterns: A Tutorial*, 1998 – Addison Wesley

[3] Eckel, Bruce. *Thinking in Java* 2nd ed. ISBN 0-13-027363-5- Prentice Hall

[4] Gamma, Eric; Helm, Richard; Johnson, Ralph and Vlissides, John, *Design Patterns. Elements of Reusable Software*, Addison-Wesley, 1995

[5] Johnson, R. *Documenting frameworks using patterns* – OOSPLA'92

[6] Kristof, Ray –Satran, Amy, *Diseño interactivo*. Ediciones Anaya Multimedia 1998.

[7] Pressman, Roger S*., Ingeniería de software: Un enfoque práctico*. Ed. Mc Graw Hill. 1997. Cuarta edición.

[8] Zulema B. Rosanigo ; Alicia Paur ; Pedro Bramati ; Alfredo Ortega ; José P.Cerra *Un framework para generación de tutoriales*. Actas VIII Congreso Argentino de Ciencias de la Computación (CACIC 2002) Buenos Aires.

[9] Rosanigo, Z.B., Paur, A., Bramati, P. *Metodología de desarrollo de software educativo*. Actas de VI Congreso Internacional de Ingeniería Informática ICIEY2K Fac. de Ingeniería, U.B.A. - Buenos Aires – 2000