Generadores de Interfaces de Usuario: QT Designer, NetBeans y Windows Forms Designer.

Alejandro Alvarez B.
Universidad de Costa Rica, Escuela de Ciencias de la Computación e Informática
San José, Costa Rica
barricada_xxi@yahoo.com

AND

Esteban Valerio R.
Universidad de Costa Rica, Escuela de Ciencias de la Computación e Informática
San José, Costa Rica
estebanvalerio@gmail.com

Abstract

In this article, a brief introduction to the Graphic User Interfase Creators it's done. They're introduced as an example of Fourth Generation Languages. Then, for three of them (QT Designer Windows Form Designer, and NetBeans GUI Builder), their features are shown. Finally, some of the advantages and disadvantages of the those tools are enumerated.

Keywords: 4GL, GUI Creators, GUI, NetBeans, QT Designer, Windows , Visual Estudio (Windows Forms), Form Generator.

Resumen

En este artículo se hace una breve introducción a los Generadores de Pantallas. Se presentan los mismos como un ejemplo de Lenguajes de Cuarta Generación. Luego se exponen las características de tres de ellos: el QT Designer, el Windows Form Designer y el NetBeans GUI Builder. Finalmente, se enumeran algunas de las ventajas y desventajas que presentan dichas herramientas.

Palabras clave: 4GL, Diseñador de Interfaz Gráfica de Usuario, Interfaz Gráfica de Usuario, NetBeans, QT Designer, Windows Forms, Visual Estudio (Windows Forms), Generador de Formularios

1. Introducción

En la actualidad es común que los usuarios de computadoras estén acostumbrados a utilizar programas con interfaces gráficas. Estas interfases se utilizan usualmente en un rango muy amplio de programas que va desde los sistemas operativos hasta aplicaciones específicas como reproductores o juegos. El objetivo final de dichas interfaces es que los programas finales sean mucho más amigables y fáciles de usar para los usuarios.

No obstante, la necesidad utilizar interfaces gráficas en los programas impone una carga extra en los diseñadores y desarrolladores de software, pues deben incorporarlas en sus aplicaciones aún en etapas muy tempranas del desarrollo del mismo. Por ejemplo, las interfaces gráficas son muy utilizadas en el desarrollo de prototipos para obtener y validar los requerimientos de las aplicaciones en la etapa de análisis del proceso de desarrollo de software.

Ahora bien, muchos lenguajes de programación no proporcionan librerías estándar para el manejo gráfico y, aún en aquellos en que se proporcionan dichas librerías, el paso de todos los parámetros necesarios para crear y colocar los

diversos elementos en la pantalla es muy complicado y requiere un gran esfuerzo por parte de quien implementa dichas interfaces.

Para solventar esta necesidad, surgieron una serie de herramientas especializadas de software que permiten crear esas interfaces gráficas en forma sencilla reduciendo el número de parámetros que debe introducir el usuario y proporcionando un alto nivel de abstracción para el diseño de las mismas. Estas herramientas son denominadas creadores de interfases gráficas de usuario o diseñadores de pantallas.

En este artículo se hace una breve introducción a dichas herramientas. Para ello, en primer lugar, se expone el concepto de interfaz gráfica de usuario, a continuación se explica qué son los diseñadores de interfaz gráfica de usuario, luego se muestran las características de tres de ellos: el QT Designer, el NetBeans y el Windows Forms Designer y, finalmente, se exponen las ventajas y desventajas de los diseñadores, en general.

2. Definición de una interfaz grafica de usuario

En Ciencias de la Computación, dentro de la disciplina de la Interacción Persona Computador, se define la **GUI** (*Graphics User Interface,* Interfaz Gráfica de Usuario), como el medio de interacción entre un usuario y un sistema informático que se realiza mediante el lenguaje visual [1]. Esta interfaz debe proveer al usuario un ambiente agradable y sencillo para el correcto entendimiento y ejecución del programa.

El lenguaje visual de las **GUI** representa la información y acciones disponibles dentro de una aplicación por medio de un conjunto de imágenes y gráficos lo suficientemente sencillos y manejables, con la meta ideal de que cualquier usuario, dentro de un dominio cultural determinado, sin necesitad de demasiada experiencia previa, sea capaz de llevar a cabo las tareas normales dentro de la aplicación.

La interfaz grafica de usuario es la evolución de los antiguos programas de línea de comando hasta lo que se ve en nuestros días en los que se empieza a experimentar con el uso de interfaces en tercera dimensión [7]. Es más, no es aventurado señalar que actualmente, dentro de la mente de un usuario promedio, una aplicación de software sin interfaz gráfica o con una mala interfaz gráfica está destinada a morir en el olvido.

La revolución en el diseño de las interfaces dentro de las principales aplicaciones del mercado se comenzó a dar a mediados de los ochentas con la aparición de Apple en el mercado computacional, su sistema operativo grafico provocó un fenómeno tan fuera de serie. De hecho las bases establecidas por dichos sistema operativo para las interfaces gráficas, aún están en vigor. Luego comenzaron a aparecer las versiones de la competencia por parte de Microsoft, empezando con el sistema operativo Windows 3.1 hasta la ultima versión de Windows, el Vista en cualquiera de sus presentaciones [1].

En este artículo dará una breve introducción a varias herramientas **4GL** (*Fourth Generation Languages*, Lenguajes de Cuarta Generación) para desarrollar **GUI** (*Graphics User Interfaces,* Interfaces Gráficas de Usuario).

3. Definición de los diseñadores de GUI

Los **diseñadores de interfases gráficas de usuario**, son herramientas de software que permiten generar interfaces gráficas de usuario mediante un lenguaje cercano al lenguaje natural. En general, proveen una abstracción de las librerías de diseño gráfico, para desarrollar en forma más sencilla las interfaces.

Los diseñadores de **GUI** modernos utilizan, a su vez, una interfaz gráfica para comunicarse con el desarrollador. Es decir, el desarrollador crea su interfaz mediante el lenguaje gráfico proveído por la herramienta (paletas de componentes, acciones de “arrastrar” y “soltar”, presión de botones, selectores de colores, etc). No obstante, dichas herramientas también requieren el uso de texto para ajustar en forma fina el comportamiento y las propiedades de los componentes.

Los creadores de GUI se consideran un tipo de lenguajes de cuarta generación porque esa abstracción tan amplia permite que el desarrollador al diseñar la interfase se concentre más en qué es lo que quiere hacer más que en cómo lograrlo. Es interesante recalcar acá que el lenguaje utilizado por esas herramientas es el lenguje visual o gráfico, lo que permite que las interfaces se construyan mediante una especie de analogía de colocar elementos sobre una mesa de trabajo. Para el desarrollador el diseño de la pantalla se vuelve un problema de "acomodar" las piezas en el lugar correcto más que determinar cuales son todos los parámetros que requieren cada uno de los componentes de la interfaz gráfica.

4. Ejemplos de Diseñadores de GUI

En esta sección expondremos algunas características relevantes de tres

4.1 QT Designer

QT Designer es una herramienta para el desarrollo de formularios y presentaciones gráficas para las aplicaciones. Permite acelerar el desarrollo de interfaces de alto rendimiento, a la vez que proporciona una forma fácil de diseñar interfaces gráficas de usuario avanzadas generando el código fuente para las mismas, lo que permite al desarrollador ajustarlo a sus necesidades.

Este generador de interfaces fue creado inicialmente por la empresa TROLLTECH para trabajar en varias distribuciones Linux. No obstante, actualmente puede instalarse en otras plataformas como Windows y Mac OsX.

El QT Designer utiliza como base la librería gráfica de QT, que ha sido transportada a diversas plataformas, lo que permite que el código generado por el QT Designer pueda ser utilizado en diversas plataformas. Además, el QT funciona sólo o asociándose a algunos entornos de desarrollo integrado como Visual Studio .Net o Eclipse. Esta herramienta provee características muy poderosas como la previsualización de la interfaz, soporte para widgets y un editor de propiedades bastante poderoso.

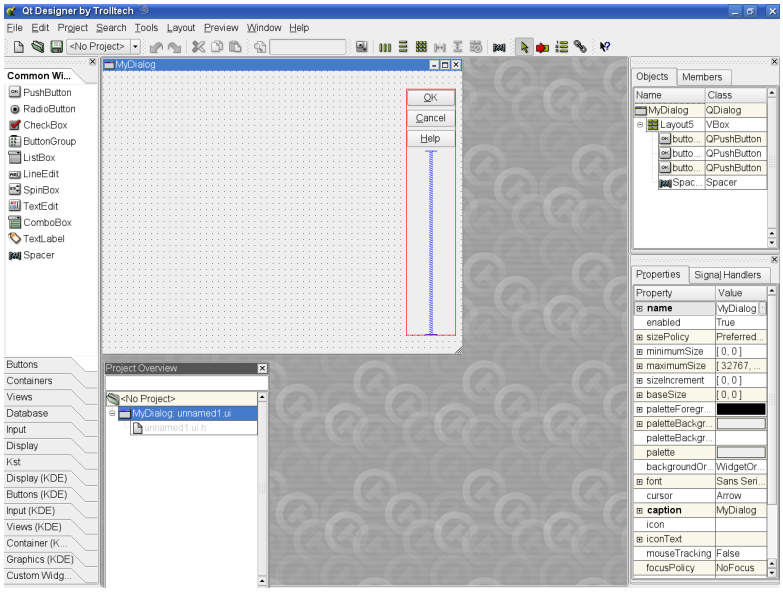

Imagen 1: Diseñador de Interfases QT

Qt y Qt Designer tienen dos opciones principales de licenciamiento: Licencia Comercial y Licencia OpenSource. La licencia comercial (que debe ser pagada) se usa cuando el desarrollador está creando aplicaciones de su propiedad para la

distribución comercial de las mismas. La licencia Open Source es gratuita pero implica que el desarrollador se ve obligado a liberar su código fuente y hacer una distribución gratuita de su aplicación. [14]

La librería QT es una librería implementada en C++ y orientada a objetos. Actualmente hay enlaces para otros lenguajes como Python o Perl y, en el caso de java, se ha desarrollado un **API** (*Aplication Programming Interface*, Interfaz para Programación de Aplicaciones) para poder utilizar esta librería C++ en aplicaciones Java.

Los conceptos básicos de la librería son los widgets (objetos), los slots (o señales) y los eventos.

Los widgets son contenedores que pueden contener widgets en cualquier cantidad de niveles. El widget "padre" de dicha jerarquía puede ser cualquiera.

El QT Designer crea archivos (.ui) que contienen la especificación de la interfaz, provee una serie de plantillas predeterminadas para crear los proyectos.

La interfaz para la creación de las GUI se basa en el uso de una paleta que clasifica los diversos objetos que incluye muchos de los widgets de la librería QT y además permite la adición de nuevos widgets (aún creados por el usuario).

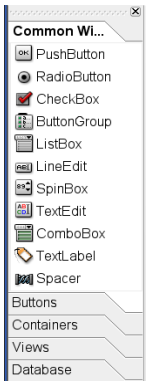

Imagen 2: Paleta de Selección de widgets del QT Designer

El diseño de la interfaz es muy sencillo pues el creador de Interface utiliza el estilo “Selecciona y dibuja”. Es decir, el usuario escoge de la paleta un tipo de widget y luego lo pone sobre el formulario, dibujándolo en la posición deseada y estableciendo el tamaño deseado.

La ventana de propiedades permite cambiar el diseño estándar de un widget para personalizarlo. Todos los cambios que se realizan en la interfaz, se van almacenando en el fichero .ui creado al incio. Dicho archivo guarda la descripción de la interfaz en formato XML.

Qt ofrece soporte para la localización de aplicaciones, es decir, el proceso de adaptar una aplicación a diversas culturas. No obstante el soporte ofrecido se refiere únicamente a brindar diversas traducciones para los elementos textuales de la interfaz. Aún así ofrece una herramienta de traducción automática muy conveniente: el Qt Linguist

4.2 NetBeans

El “constructor” de interfases de usuario del IDE Netbeans (conocido anteriormente como el “Proyecto Matisse”) es un módulo del Entorno de Desarrollo Integrado NetBeans.

Dicho sistema de software es un proyecto de código abierto que, antes de la versión 5.5 se distribuía bajo la Licencia Pública de Sun y, a partir de dicha versión, es distribuido mediante la licencia CDDL (*Common Development and Distribution License*) [3]. Este proyecto está patrocinado por Sun Microsystems aunque se inició originalmente como un proyecto estudiantil [4].

Este editor de interfases gráficas está orientado hacia la librería gráfica Swing de Java. Es decir, que únicamente produce código fuente para Java.

Además de dicho código fuente, el Generador de intefase produce unos archivos ".form" escritos en fomato XML que contienen las especificaciones de los diversos componentes gráficos del usuario. Si bien estos archivos no necesitan ser distribuidos junto con el código para compilar las interfases, sin ellos es imposible para el editor gráfico reconstruir el diseño de la pantalla para una edición posterior. Por ello tampoco el editor puede generar un archivo ".form" para clases GUI no creadas por NetBeans (aunque hay herramientas en desarrollo para ello) [5].

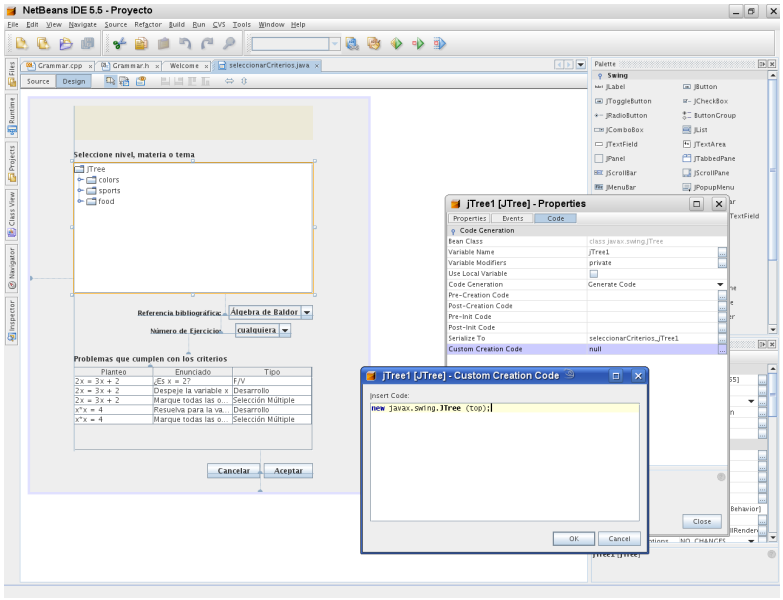

Imagen 3: Vista de Diseño del NetBeans GUI Builder

Este editor de pantallas proporciona los diversos componentes Swing en su paleta de componentes (aunque dicha paleta de componentes es extensible, por lo que se pueden agregar nuevos componentes compatibles).

Tiene dos modos de vista diseño y código fuente. En el modo de vista de diseño se pueden arrastrar y soltar los diversos componentes, mientras que en el modo de vista de código fuente se puede ver y editar el código fuente. No obstante, en este último aspecto es preciso realizar una observación importante. No es posible editar el código fuente generado automáticamente, el cual es bloqueado. No obstante, el usuario sí puede agregar código fuente propio.

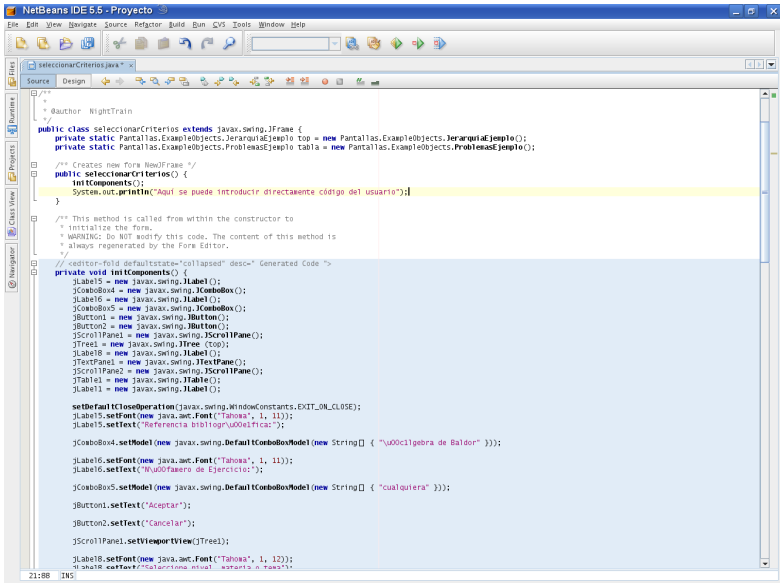

Imagen 4: Vista de Código del NetBeans GUI Builder

Este generador de pantallas, para todos los componentes, permite seleccionar y modificar las propiedades de éstos. Además permite al usuario introducir manualmente su propio código en muchas de éstas propiedades para reemplazar el código por defecto que introduce el NetBeans. Esto permite un grado importante de personalización de los componentes. Aunque, desgraciadamente, las modificaciones realizadas en forma manual no pueden verse en la vista de diseño, por lo que es preciso compilar y ejecutar el código para comprobar el resultado del código manual.

También es importantes señalar que al introducir código fuente manualmente, es posible dejar al componente en un estado no consistente, lo que será oportunamente avisado por NetBeans para que el usuario lo corrija. No obstante, en caso de que no haya una corrección satisfactoria, cuando se almacena la interfaz gráfica, puede que el componente infractor completo sea descartado, aspecto que hay que tomar en cuenta y tratar con precaución.

Al arrastrar y soltar los componentes el NetBeans trata de alinearlos automáticamente con los componentes previamente generados, lo que es muy útil si la alineación automática es correcta, pero muchas veces modifica la posición de todos los componentes anteriores, lo que puede ser muy frustrante.

También es importante señalar que el Desarrollador de Interfases Gráficas de NetBeans ofrece apoyo para la internalización y la accesibilidad en sus componentes, para esto último importante citar también el Java® Accesibility Helper que reporta los problemas que puede tener una aplicación para personas con discapacidades.

Es importante señalar que el generador de GUI del NetBeans (y en general todo el IDE) ha mostrado un sorprendente desarrollo desde su inicio y cada vez se le agregan nuevas funcionalidades. Por ello, si bien tiene algunos aspectos "complicados", posee muchas características interesantes y permite generar interfaces agradables en poco tiempo, además de que se encuentra respaldado por el principal desarrollador del lenguaje Java, como lo es Sun Microsystems.

El hecho de generar código que usa librerías estándar de Java dota a las interfases realizadas en dicho editor de una gran portabilidad, pues únicamente dependen de que exista una **JVM** (*Java Virtual Machine*) apropiada. No obstante, el usar este editor (al menos por el momento) implica utilizar como lenguaje de desarrollo de la interfaz a Java.

4.3. Windows Form Designer

El Windows Form Designer es la herramienta de interfaz de usuario de Visual Estudio que permite agregar elementos a un *Form* (Formulario) [8]. El elemento y los formularios son parte de Windows Forms, que es el nombre que recibe el **API** para la interfaz gráfica de usuario incluída como parte de la infraestructura .NET de Microsoft.

El hecho de que los Windows Forms se basen en la infraestructura .NET permite que dichos controles puedan ser usados en cualquiera de los lenguajes de programación que sigan la *CLS* (Common Languaje Specification) [11][10], porque debemos recordar que en la arquitectura .NET todos ellos se traducen a un lenguaje intermedio común. No obstante, es importante señalar que los componentes de las Windows Forms son dependientes de la plataforma, pues dependen fuertemente del Win32 API, por lo que para portarlas a una nueva plataforma (p.e. OSX o Linux) sería preciso redefinir las mismas [11] aunque ya en dicha plataforma se pudiera utilizar el *CLR* (Common Languaje Runtime) y el CLS de .NET [10].

Señalada esta limitación que hereda de la infraestructura .NET, podemos señalar que el Windows Form Designer es una herramienta de diseño de interfaces gráficas muy robusta y sencilla. En realidad, provee todas las características señaladas para las herramientas anteriores y las mejora.

Por ejemplo, en el aspecto de la internacionalización de la interface, no sólo permite asignar locales para la traducción, sino que también permite cambiar la disposición de los elementos para que se dispongan en el formulario de derecha a izquierda en vez de izquierda a derecha (con unas pocas excepciones) [12]

Esta herramienta también permite editar las diversas propiedades para modificación del comportamiento por defecto de los componentes y agregar código personalizado a los controles. Si bien, también en este caso es preciso ejecutar el programa para poder ver los efectos de dicho código sobre las Forms.

En donde esta herramienta verdaderamente sobresale es en las capacidades que tiene para asistir en la disposición de los componentes ya que permite alinearlos, igualarles el ancho o el alto, procesarlos como grupo, etc. Estas propiedades son muy deseables y permiten ajustar la forma de la interface en forma rápida y sencilla.

Una diferencia importante con las herramientas mencionadas anteriormente es que el diseñador no es código abierto y es propiedad de Microsoft. Aunque Microsoft permite la descarga gratuita del .NET Framework y una versión simplificada del Visual Studio llamada Visual Studio Express Edition (aunque sólo para la versión 2005, por ahora) [13]. Esto implica que al utilizar este diseñador de pantalla uno está asumiendo el riesgo de aceptar que el soporte y desarrollo de la herramienta seguirá estando en manos de Microsoft, a diferencia de las herramientas de código abierto, donde se puede delegar el soporte (y el desarrollo) en manos de cualquier persona o compañía.

5. Ventajas de los diseñadores de GUI

Los diseñadores de pantallas reseñados nos permiten ilustrar las ventajas propias de los mismos en general:

- Permiten desarrollar rápidamente una **GUI**. Especialmente si se considera el tiempo que se requeriría para realizar dicha interfaz mediante instrucciones del lenguaje de programación, especialmente si este último no brinda librerías estándar o instrucciones para el manejo gráfico.

- Si la herramienta de diseñado de GUI es de uso muy extendido, las **GUI** producidas por ésta seguirán un formato gráfico semejante, lo que hará que para los usuarios finales sea intuitivo utilizar la nueva aplicación, si han usado con anterioridad alguna otra aplicación que sigan ese estándar gráfico.

- Los generadores de pantalla reseñados utilizan un lenguaje gráfico para diseñar las interfaces (junto con algunas propiedades que deben introducirse en forma textual). Es decir, los componentes se crean presionando botones, arrastrando componentes, etc. Este lenguaje es muy sencillo de usar y muy adecuado para el diseño gráfico de las intefaces.

- El lenguaje gráfico utilizado para la generación de interfaces normalmente sigue muy de cerca o es igual al lenguaje gráfico que ofrecerán las GUI por dicha herramienta diseñadas. Esto permite que, al desarrollar las interfases gráficas, el desarrollador se familiarice a su vez con el uso de los distintos componentes que puede utilizar en su aplicación y pueda evaluar el efecto que los mismos producirán en sus usuarios finales.

- El uso extendido de los generadores de pantallas ha hecho que, de facto, el lenguaje gráfico utilizado por estos sea muy semejante. Por ejemplo, los diversos generadores de GUI reseñados en el presente artículo ofrecen componentes gráficos equivalentes con nombres semejantes (menús, botones, etiquetas, listas seleccionables y otros). Esto ha producido que cada vez más haya un grupo importante de usuarios para los que dicho lenguaje es conocido, lo que permite que puedan deducir intuitivamente el uso de los componentes aunque hayan sido desarrollados por diferentes herramientas de diseño de GUI.

6. Desventajas de los diseñadores de GUI

Como se ha indicado, los diseñadores de pantallas tienen grandes ventajas. No obstante, también su uso presenta importantes inconvenientes:

- Como toda herramienta de 4GL, los diseñadores de pantallas son altamente parametrizados. Como lo que se busca es indicar el qué hay que hacer y no el cómo, la el diseñador de pantallas implícitamente reduce el dominio de lo que el lenguaje de programación y las librerías subyacentes pueden lograr. Por ejemplo, que los botones fueran redondos en vez de rectangulares, utilizar componentes con animaciones 3D, etc.

- Si bien los diseñadores reseñados permiten establecer código antes y después de la creación de los componentes, esto no es diferente a realizar la tarea directamente en el lenguaje de programación e incluso puede ser más engorroso porque hay que determinar como "deshacer" u obviar el código que el generador crea por defecto.

- Todos los generadores están asociados a un (o unos) lenguajes de programación predeterminados (p.e. el Netbeans con Java) e inclusive pueden estar asociados a una infraestructura o plataforma determinada (p.e. Visual Estudio con .NET Framework). Esto hace que estas poderosas herramientas no puedan ser aplicadas en forma general, para resolver problemas con lenguajes de programación no soportados. Lo que obliga a buscar una herramienta para dicho lenguaje (lo implica aprender lenguaje gráfico de desarrollo) o, en el peor caso, tener que hacer la interfaz manualmente en el lenguaje de programación.

- Estas herramientas reducen la interfaz con el usuario a una interfaz gráfica, no obstante existen muchos medios de percepción (p.e. sonido) que podrían utilizarse para brindar accesibilidad a personas con alguna necesidad especial (daltónicos, problemas visuales, etc.) Y, en todo caso, si cualquier persona percibe una información por diversos sentidos, se da un procesamiento más eficiente de la misma a nivel cerebral. El uso de una herramienta de creación de GUI le impone al desarrollador sólo una forma de interfaz y brindan poco soporte para la creación de otras interfaces (braille, sonido, etc.) y la mezcla de interfaces producidas por diversas herramientas provoca un problema complejo de programación y sincronización.

- Otro aspecto importante es que, si bien los creadores de interfases gráficas ofrecen algún grado de apoyo a la internacionalización de sus aplicaciones, éste casi que se refiere a brindar la traducción de los elementos textuales. El problema con ello es que una internalización verdadera de una interfaz gráfica debe tomar en cuenta también la necesidad de cambiar de posición los elementos de la interfaz (p.e. Para las culturas en que se acostumbra a leer de derecha a izquierda) o, incluso, el diverso significado que se le da a los colores en diferentes culturas [6].

En general cuando utiliza un diseñador de GUI, el desarrollador de software se está atando al paradigma de lenguaje gráfico que dicho generador provee. Y si dicho desarrollador requiere algo que extienda o esté fuera de dicho paradigma va a tener que realizar un esfuerzo importante para poder lograrlo.

7. Conclusiones

Los diseñadores de interfaces son herramientas 4GL que permiten crear una abstracción de un lenguaje de programación 3GL y unas librerías gráficas para proporcionar al desarrollador una forma fácil y rápida de crear interfaces gráficas de usuario para sus aplicaciones.

El Qt Designer es un creador de interfaces gráficas para la librería QT, provee un gran nivel de portabilidad tanto a nivel de lenguajes de programación soportados (aunque sea mediante ligas) como plataformas sobre las que puede correr (y para las que puede diseñar). No obstante, carece de características avanzadas de integración con los lenguajes de programación y plataformas soportados como lo sería el introducir código en los widgets.

El NetBeans GUI Builder es un diseñador de pantallas integrado dentro del NetBeans IDE. Crea código Java que utiliza componentes Swing. Esto hace que el código resultante sea muy portable, pues únicamente requiere un entorno de ejecución Java adecuado. No obstante, eso implica también una gran dependencia del lenguaje de programación. La integración con el entorno de desarrollo permite un alto grado de personalización de los componentes, que conlleva incluso la capacidad de introducir código *ad hoc* en los mismos y la posibilidad de desarrollar interfaces avanzadas, no obstante es un poco difícil de manejar.

El Windows Form Designer integrado al IDE del Visual Studio es un diseñador de pantallas, a la vez, robusto y sencillo de utilizar. Está atado a los Windows Forms del .NET Framework, lo que le otorga al desarrollador un alto grado de independencia del lenguaje de programación, pero conlleva una menor portabilidad entre plataformas diversas, pues requiere que la plataforma maneje tanto el CLR de .NET como un equivalene al API Win32 de Windows. Además, es una herramienta privada y sin código fuente disponible.

El uso de herramientas de diseño de GUI trae importantes ventajas y desventajas. Las ventajas principales son el desarrollo rápido y la estandarización de componentes, las desventajas principales se refieren a la restricción del dominio funcional con respecto a los lenguajes y librerías subyacentes y la necesidad de ajustarse al paradigma proporcionado por la herramienta.

El desarrollador de software debe conocer estas herramientas porque es necesario sopesar si para un proyecto determinado pueden ser aplicables, porque pueden aumentar el tiempo de desarrollo en forma importante, aunque no sin un costo como se ha examinado en este artículo.

8. Referencias Bibliográficas

[1] *Autor Desconocido, Interfaz Gráfica de Usuario*, en Wikipedia, URL: http://es.wikipedia.org/wiki/GUI (Fecha de última modificación: 06/06/2007).

[2] Autor Desconocido, Netbeans, URL: http://www.kwwidgets.org/Wiki/KWWidgets/Projects/UIDesigner/Application/PreviousWork/NetBeansStudyResults (Fecha de creación: 13/04/2006).

[3] Autor Desconocido, Legal Materials, URL: http://www.netbeans.org/about/legal/index.html (Fecha de consulta: 17/06/2007).

[4] Autor Desconocido, A Brief history of NetBeans, URL: http://www.netbeans.org/about/history.html (Fecha de consulta: 17/06/2007).

[5] Autor Desconocido, Netbeans User FAQ, URL: http://wiki.netbeans.org/wiki/view/NetBeansUserFAQ (Fecha de consulta: 17/06/2007).

[6] SILBAGRAPHICS, The Meaning of Colours, URL: http://www.sibagraphics.com/colour.php (Fecha de consulta: 18/06/2007).

[7] Autor Desconocido, Graphical user Interface, URL: http://en.wikipedia.org/wiki/Graphical_user_interface (Fecha de última modificación: 15/06/2007)

[8] Autor Desconocido, *Windows Form Designer,* Windows Forms Reference, URL: http://msdn2.microsoft.com/en-us/library/e06hs424(VS.80).aspx (Fecha de consulta: 14/06/2007)

[9] Autor Desconocido, *Windows Forms*, en Wikipedia, URL: http://en.wikipedia.org/wiki/Windows_Forms (Fecha de última modificación: 24/05/2007)

[10] Schemitz, John, .Net Architecture, URL: http://www.midnightbeach.com/dotNetArchitecture.2002.html (Fecha de creación 12/09/2002)

[11] Utley, Craig, *Using ActiveX Controls with Windows Forms in Visual Studio .NET*, Upgrading to Microsoft . NET, URL:http://msdn2.microsoft.com/en-us/library/ms973200.aspx (Fecha de creación: Noviembre 2001).

[12] Autor Desconocido, *How to Create Mirrored Windows Forms and Controls*, Windows Forms Programming, URL: http://msdn2.microsoft.com/en-us/library/xwbz5ws0(vs.80).aspx (Fecha de consulta: 14/06/2007).

[13] Autor Desconocido, Frequently Asked Questions, URL: http://msdn.microsoft.com/vstudio/express/support/faq/ (Fecha de consulta: 16/06/2007)

[14] TrollTech, Licencing Overview, URL: http://trolltech.com/products/qt/licenses/licensing/licensingoverview (Fecha de consulta: 16/06/2007)