Ingeniare. Revista Chilena de Ingeniería
ISSN: 0718-3291
facing@uta.cl
Universidad de Tarapacá
Chile

Arganis Juárez, Maritza Liliana; De Luna Cruz, Faustino

Dibujo de series de igual o distinta longitud utilizando un lenguaje de programación disparado por eventos

Ingeniare. Revista Chilena de Ingeniería, vol. 21, núm. 1, abril, 2013, pp. 70-81
Universidad de Tarapacá
Arica, Chile

Disponible en: http://www.redalyc.org/articulo.oa?id=77225903007

Dibujo de series de igual o distinta longitud utilizando un lenguaje de programación disparado por eventos

Drawing series with the same or different length by means of an event-driven computing language

Maritza Liliana Arganis Juárez[1] Faustino De Luna Cruz[1]

Recibido 31 de marzo de 2011, aceptado 9 de noviembre de 2012
Received: March 31, 2011 Accepted: November 9, 2012

RESUMEN

El análisis de datos numéricos en ingeniería demanda el empleo o desarrollo de herramientas que permitan, entre otros cálculos, la elaboración de gráficas donde se presenten una o más funciones. En este artículo se presenta paso a paso el uso de Visual Basic 6® y el control PictureBox® para dibujar una o más series en un mismo gráfico y que pueden ser de igual o distinta longitud o cantidad de datos, para su comparación. Se ejemplifica su aplicación con éxito en el caso del manejo de datos de precipitación de estaciones pluviográficas. La metodología presentada puede utilizarse con Visual Basic 6®, pero el proceso puede codificarse en otros lenguajes de programación comúnmente aplicados como herramientas en la solución de diversos problemas de ingeniería que tienen que ver con la revisión rápida de procesos de generación de datos.

Palabras clave: Gráficos, PictureBox®, serie de datos, Visual Basic 6®, estadística.

ABSTRACT

Numerical analysis data in engineering needs the use or development of software that allows creating the graphical representation of functions, among other calculations. In this article a step by step use of Visual Basic 6® and the control PictureBox® are presented as helping tools to draw one or more series in the same graphic, with the same or different length or data quantity for their comparison. The management of rainfall data from a pluviographic station is presented as an example of this successful application. The methodology here presented can be used with Visual Basic 6®, but the process can be codified in other computing languages commonly applied as tools to solve several engineering problems related to the quick inspection of data generation processes.

Keywords: Graphics, PictureBox®, data series, Visual Basic 6®, statistics.

INTRODUCCIÓN

Al iniciar un proyecto de ingeniería que involucra el análisis de series de tiempo se deben revisar la cantidad y la calidad de los mismos; una inspección gráfica resulta de utilidad para percibir posibles datos anómalos resultado de errores inherentes o propios de los datos [1]. Herramientas como hojas de cálculo llegan a ser de utilidad en estos casos. En otras ocasiones, ya al introducir la información a un procedimiento numérico o estadístico, que se llega a realizar repetidas veces, no resulta práctico pasar de un ambiente del algoritmo de solución a un archivo de texto y luego a una hoja de cálculo para hacer los dibujos por un camino semimanual; en estos casos es más conveniente la elaboración

[1] Instituto de Ingeniería. Universidad Nacional Autónoma de México. Edificio 5 Cub. 403. Av. Universidad 3000 Copilco. C.P. 04360. Coyoacán, México, D.F. E-mail: MArganisJ@iingen.unam.mx; FLunaC@iingen.unam.mx

de gráficos con ayuda del código del lenguaje de programación para que directamente dichos gráficos se muestren al usuario una vez realizado el proceso sin necesidad de utilizar otro software.

El lenguaje de programación Visual Basic 6® cuenta con distintas alternativas para dibujar gráficas, alguna de ellas con la limitante de que si se dibuja más de una serie en el mismo gráfico, todas las series dibujadas deben ser de mismo tamaño, con el fin de llegar a arreglos cuadrados, tal es el caso de la herramienta MsChart®.

En este documento se describe la elaboración, paso a paso, de gráficas de funciones expresadas en forma tabular, que pueden tener la misma longitud o longitudes diferentes y que pueden dibujarse en un mismo plano coordenado a partir de la herramienta PictureBox® de Visual Basic en su versión 6.0, la cual resulta de mucha utilidad en problemas prácticos de ingeniería. El control PictureBox se ha utilizado, por ejemplo, para calcular el área, perímetro, longitud y ancho de hojas de especies vegetales [2]; los PictureBox se suelen utilizar para abrir imágenes en un software [3]. K. Mock [4] utilizó el control PictureBox para representar la imagen principal dentro de un sistema de visualización de las mareas de cuencas; también utilizó un PictureBox para representar la variación de la marea contra el tiempo; otros ejemplos del uso del PictureBox han sido en el diseño de un software que ayuda a poner en marcha un motor de un sistema de diseño de dibujo [5] y para visualizar la imagen tridimensional de redes de fractura y trayectorias de filtración en rocas [6].

En este trabajo se presenta el caso particular del manejo de esta herramienta aplicado a datos de precipitaciones máximas y medias, tratadas como tormentas anuales, pertenecientes a 49 estaciones pluviográficas.

ANÁLISIS TEÓRICO

Lenguaje de programación Visual Basic 6.0

Su antecesor es el BASIC (Beginner's All-purpose Symbolic Instruction Code), el cual fue desarrollado por John Kemeny y Thomas Kurts en el Dartmouth College en 1964 para enseñar programación a principiantes; el Lenguaje Visual Basic está catalogado dentro de los lenguajes disparados por eventos que llevaron a la programación orientada a objetos a otro nivel [7]. Las instrucciones se reemplazan por íconos o símbolos, cada ícono representa un objeto o una función de programación o un procedimiento. Instrucciones como dar click o arrastrar fueron programadas en el ícono, de manera que el programador no necesitaba programar tantas instrucciones. Este lenguaje, utilizado tanto por usuarios finales como programadores profesionales, fue introducido por Microsoft en 1992, como Visual Basic para Windows 3 y posteriormente surgieron las versiones de Visual Basic 4 para Windows 95 y Visual Basic 5.0 para Windows 95 y office 97 y Visual Basic 6.0.

Control PictureBox

García de Jalón, Rodríguez y Brazález [8] definen al PictureBox como el control gráfico más potente y general de Visual Basic 6.0. Es un tipo de formulario reducido debido a que puede contener imágenes y controles tales como botones, shapes, labels, cajas de texto, etc.

Elementos básicos para crear un gráfico con PictureBox en Visual Basic

Dibujo de una serie de datos

Para elaborar un Picturebox en visual basic con una serie de datos o una función y=f(x), se requiere el conjunto de pareja de valores (x, y), los cuales se pueden dar:

- En forma explícita, indicándolos en un arreglo, que puede ser tipo vector o tipo matriz e indicando el valor específico de la variable x y de la variable y, por ejemplo:

$$x(1)=0,\ x(2)=1,\ x(3)=2,\ x(4)=5,\ x(5)=6,\ldots,x(n)=a;\ y(1)=0,\ y(2)=2,\ y(3)=4,\ y(4)=10,\ y(5)=12,\ldots y(n)=2a$$

- También los datos de y se pueden dar en forma de ecuación, conocido previamente el valor de x que también se puede dar de esa forma por ejemplo:

$$x(i+1)=i,\ i=0,1,2,\ldots.n$$
$$y(i+1)=2*x(i),\ i=0,1,2,\ldots n$$

Los datos (x, y) pueden provenir de un proceso anterior.

Para poder dibujar la serie se requiere abrir un proyecto en Visual Basic (Figura 1).

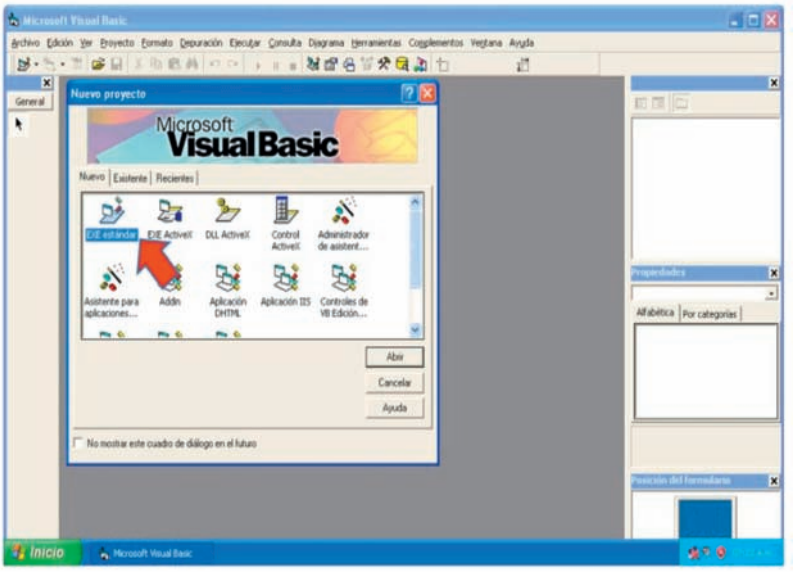

Figura 1. Apertura de un proyecto en Visual Basic.

Posteriormente la forma que aparece (Form1) puede ampliarse arrastrando la esquina inferior derecha con el mouse, en dicha forma se selecciona el ícono de PictureBox y se coloca en la hoja, también se requieren por lo menos dos botones de comando, el de inicio para dibujar y el de salida del programa; el nombre del PictureBox se indica en la ventana de propiedades (al darle click en el mismo aparecen sus propiedades del lado izquierdo), de manera similar las propiedades se muestran para cada botón de comando.

En la Figura 2 aparecen estos elementos antes de darles un nombre particular y en la Figura 3 después de asignarle su nombre, en el caso de los botones de comandos, el texto del botón de comando se indica en la propiedad Caption.

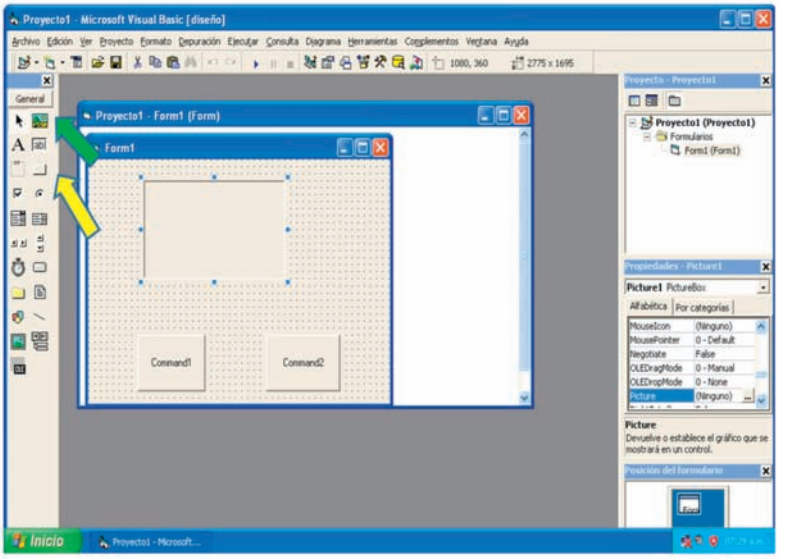

Figura 2. Disposición del PictureBox y botones de comando.

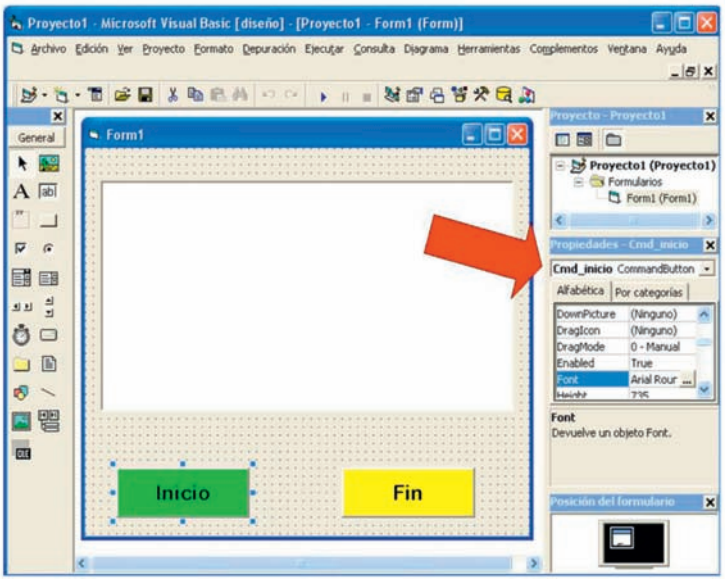

Figura 3. Personalización de botones de comando.

Para poder dibujar la serie se requiere tener previamente dimensionados los vectores, de preferencia declarándolos como variables globales en un módulo o dimensionados en la forma donde se encuentre el PictureBox dentro del proyecto; puede ser que estas variables se hayan obtenido dentro de un módulo entonces deben aparecer declaradas como globales en el mismo.

Se pueden indicar con Labels los valores de las escala horizontal y vertical de la serie a dibujar; lo anterior se puede hacer manualmente si previamente se conoce su valor máximo y mínimo en cada caso; se indican algunos de estos valores con un incremento seleccionado y se colocan distribuidas en las márgenes inferior e izquierda del PictureBox, posteriormente en el menú formato, espacio vertical se le da igualar y también en espacio horizontal se le da igualar, adicionalmente se puede indicar con Labels el nombre de las variables, un ejemplo se ilustra en la Figura 4.

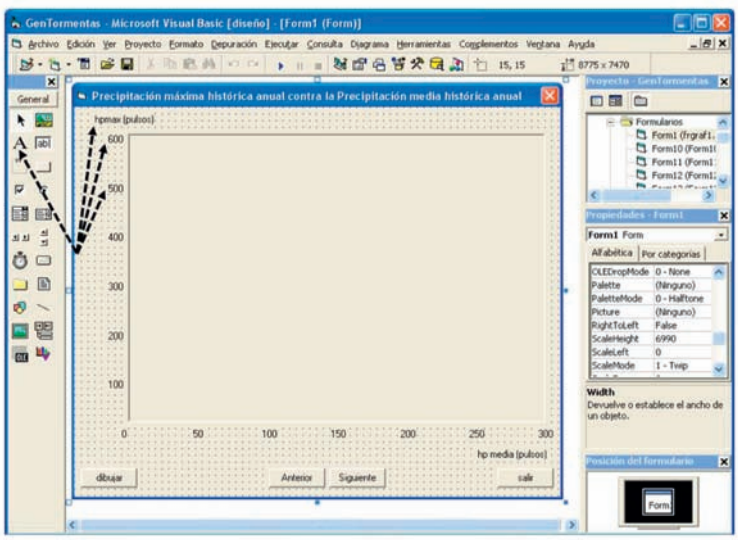

Figura 4. Ejemplo de colocación de labels para indicar la escala de la serie a dibujar.

Para graficar la serie en el código del botón de comandos que indique dibujar, se le proporciona información correspondiente a los valores iniciales y finales de los datos, para que el PictureBox tome el tamaño correspondiente, también la serie se puede dibujar como puntos o bien con líneas; en la Figura 5 se presenta el código del botón inicio para el caso del dibujo de puntos de las parejas de valores de las variables denominadas precipitación media y precipitación máxima (reportados en pulsos). A continuación se detalla el código que aparece en la Figura 5.

```
Private_Sub cmddibujar_Click()
TIE=300
Q_MAX=600

Pic_Hid.ScaleWidth=TIE
Pic_Hid.ScaleTop=Q_MAX
Pic.Hid.ScaleHeight=-Q_MAX
Pic_Hid.ScaleLeft=0
Pic_Hid.Refresh

´dibuja puntos
DAT_REG = (ntes)
Pic_Hid.DrawWidth = 4
For i = 1 To DAT_REG
Pic_Hid.PSet (hpmet(i), hpmax(i)), QBColor(4)
Next
End Sub
```

En el ejemplo presentado en la Figura 5 el PictureBox se llama Pic_Hid, la variable TIE dará el valor de la dimensión de la base del PictureBox, la variable Q_MAX proporciona la altura del PictureBox; lo anterior se asigna en las propiedades ScaleWidth y ScaleTop; para dibujar el origen del sistema coordenado en la esquina inferior izquierda es conveniente asignarle a la propiedad ScaleHeight el valor negativo de la altura dada al PictureBox (en este caso –Q_MAX). Para indicar el inicio de la escala izquierda (ScaleLeft) se le asigna cero o el valor en el que se desee que inicie y es conveniente indicar la propiedad Refresh. Para asegurar el dibujo de la serie, la propiedad AutoRedraw debe estar en la opción True.

Para el dibujo de puntos se debe contar con el número de datos a dibujar, en este ejemplo aparece con la variable DAT_REG y aparece igualado a una variable donde previamente se asignó el total de datos, allí se le puede proporcionar directamente el número de datos. Posteriormente se le asigna al ancho del dibujo (DrawWidth) un valor; en el ejemplo se indica un valor de 4. Posteriormente, si los vectores de las variables a dibujar ya tienen datos en el momento del proceso, basta utilizar un ciclo para hacer el dibujo desde el primer hasta el último dato; para indicar que se dibujen puntos se utiliza la propiedad Pset y entre paréntesis los

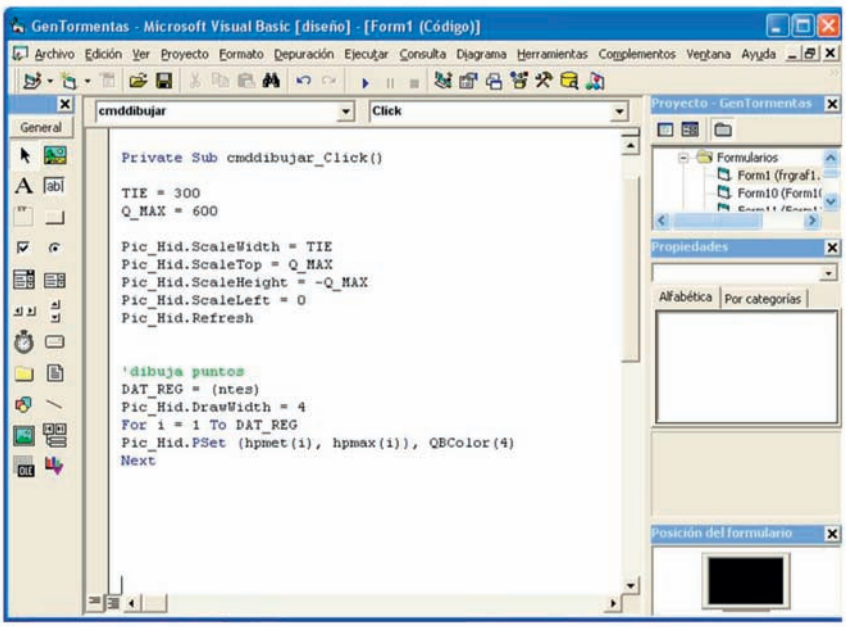

Figura 5. Ejemplo de colocación de código para dibujar la serie por medio de puntos.

nombres de los vectores que contienen a la variable del eje horizontal y a la variable del eje vertical; posteriormente se indica el color de los puntos con la instrucción QbColor(N°) y entre paréntesis el número del color utilizado (Tabla 1). Con este código, al darle click al botón dibujar, aparecerán dibujados los puntos de la serie (Figura 6).

Tabla 1. Números para la instrucción QBColor (N°)

N°	Color	N°	Color
0	Negro	8	Gris
1	Azul	9	Azul claro
2	Verde	10	Verde claro
3	Aguamarina	11	Aguamarina claro
4	Rojo	12	Rojo claro
5	Fucsia	13	Fucsia claro
6	Amarillo	14	Amarillo claro
7	Blanco	15	Blanco brillante

Dibujo de dos series de datos de distinta o igual longitud

Un caso que suele presentarse cuando se manejan datos de ingeniería, en específico de hidrología, es la representación de dos series de distinto tamaño en la misma gráfica; la opción MsChart de Visual Basic 6® deja de ser viable en estos casos, debido a que demanda series de igual tamaño. Este problema se puede resolver con la ayuda de un PictureBox; como ejemplo se presenta el caso de dibujar datos de precipitación máximos históricos o medidos (vector de ordenadas) dibujados contra su correspondiente periodo de retorno, proporcionado en una escala logarítmica (abscisas) como puntos y una curva de ajuste a dichos datos, representada como una línea recta que tiene como ordenadas los valores estimados y como abscisas distintos valores del periodo de retorno en escala logarítmica; de nueva cuenta, en el código del botón dibujar se deben indicar las instrucciones para cada caso (Figura 7). A continuación se detalla el código que aparece en la Figura 7.

```
Private Sub cmddibujar7_Click()
‘escala gráfica para gráfica
TIE = 12
Q_MAX = yax(1)
Pic_Hid.ScaleWidth = TIE
Pic_Hid.ScaleTop = Q_MAX
Pic_Hid.ScaleHeight = -Q_MAX
Pic_Hid.ScaleLeft = -2
Pic_Hid.Refresh

‘dibuja puntos
DAT_REG = (ntes)
Pic_Hid.DrawWidth = 4
For i = 13 To DAT_REG
  Pic_Hid.PSet (trxaxgum(i), zax(i)), QBColor(4)
Next
```

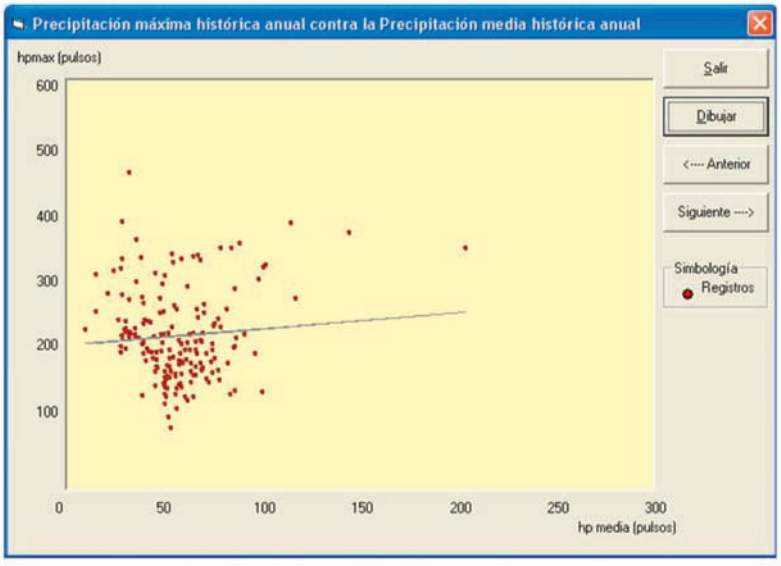

Figura 6. Dibujo de una serie por medio de puntos.

```
‘dibuja líneas
Pic_Hid.DrawWidth = 1
DAT_AJU = ntes + 12

For i = 2 To DAT_AJU
   TR_INI = trxaxgum(i - 1): yaxg_ini = yax(i - 1)
   TR_FIN = trxaxgum(i): yaxg_fin = yax(i)
    Pic_Hid.Line (TR_INI, yaxg_ini)-(TR_FIN,
yaxg_fin), QBColor(8)
Next

MsgBox (“DIBUJÉ DOS SERIES DE DISTINTO
TAMAÑO”)

End Sub
```

En este caso se da un valor TIE=12 para la escala el ancho ScaleWidth del PictureBox, y para la escala de la altura ScaleTop a la variable Q_Max se le asignó el primer valor del vector de la serie de ajuste que corresponde al primer dato del vector yax, es decir yax(1). Nuevamente se le asigna a la propiedad ScaleHeight el negativo del ScaleTop, es decir, –Q_Max. En este caso se le dio al ScaleLeft un valor igual a -2, tomando en cuenta la variación en la horizontal de las variables y también se indica la propiedad Refresh.

Para el dibujo de la primera serie con puntos se le asigna a la variable DAT_REG el número de datos de la serie 1, en este caso de los datos de precipitación máxima medidos, se le proporcionó un ancho del dibujo DrawWidth igual a 4. Se utiliza un ciclo para el dibujo de los valores, en este caso particular los valores medidos aparecen a partir del dato 13 de los vectores utilizados, en este caso para las abscisas el vector traxgum(i) y para las ordenadas el vector zax(i), de nueva cuenta se utiliza la propiedad PSet y como argumentos los vectores traxgum y zax, además del color con la instrucción QBColor(j).

Para la segunda serie, dibujada con líneas, se indicó un valor de la propiedad DrawWidth igual a uno; en este ejemplo se indica que el total de datos de la segunda serie a dibujar lo contiene la variable DAT_AJU y corresponde al total de datos de la primera serie más doce, pero en cualquier otro caso se asigna el número que corresponda. La línea que corresponde a la segunda serie se dibuja con un ciclo, en este caso como se van dibujando de dos en dos segmentos hasta formar toda la línea, el ciclo se inicializa en i=2 hasta el número de datos del ajuste, se define una variable que indica el inicio de las abscisas de la línea TR_INI que se iguala al dato anterior del vector traxgum que contiene a la

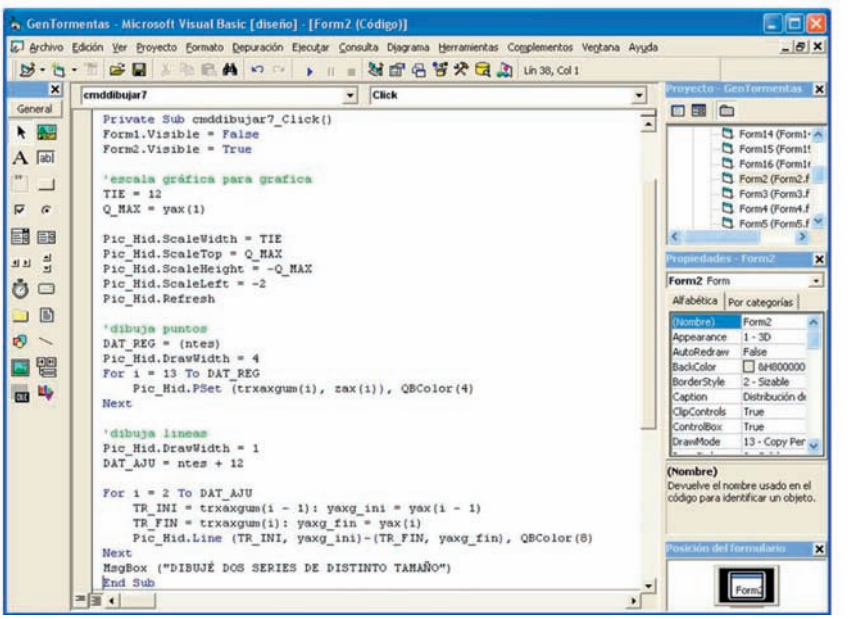

Figura 7. Ejemplo de código para dibujar dos series de igual o distinta longitud, una con puntos y otra con rectas.

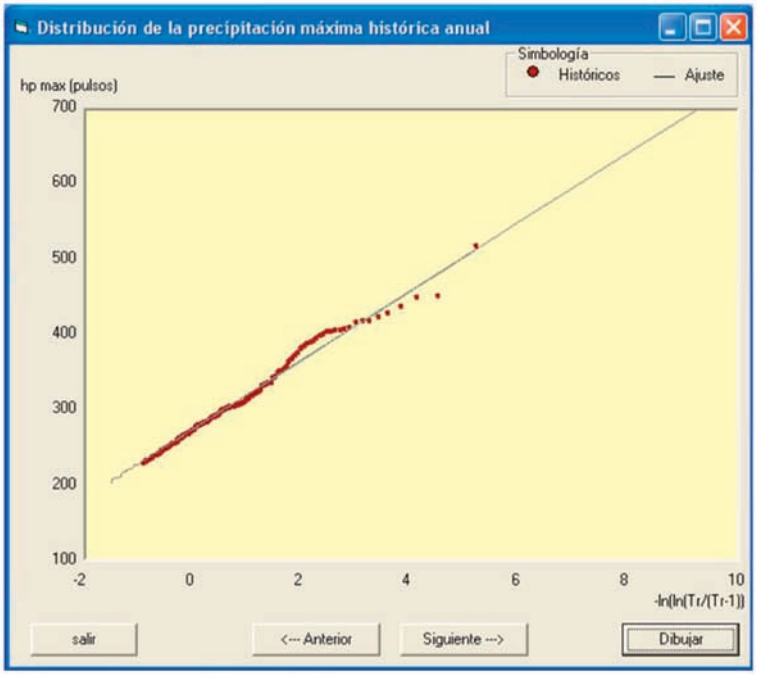

Figura 8. Dibujo de dos series de distinta longitud usando picturebox.

segunda serie, se define la variable que indica el inicio de las ordenadas de la línea yaxg_ini que es igual al dato anterior del vector de ordenadas de la segunda serie, en este caso particular lo contiene el vector yax. En forma similar se definen los valores de las abscisas y ordenadas del final del segmento, las variables usadas en el ejemplo se llaman TR_fin y yaxg_fin y se les asigna el valor de los vectores de la serie dos en cada caso, para el valor de i. En el ejemplo mostrado, estos datos los contienen los vectores trxaxgum(i) y yax(i). La línea se dibuja con la instrucción Line indicando la diferencia del punto inicial y del punto final y a continuación el color que tendrá la línea con el instrucción QBColor(j).

Al dar click en el botón de comandos dibujar, que tiene estas instrucciones, se deben desplegar las dos series deseadas (Figura 8).

APLICACIÓN

Un problema recurrente de ingeniería hidráulica-hidrológica es la generación sintética de registros de tormentas más largos que los históricos; una vez generados dichos registros es de interés determinar si se lograron reproducir los estadísticos: media, desviación estándar, coeficiente de asimetría de la muestra analizada.

En el caso presentado se generaron 500 tormentas sintéticas para 49 estaciones pluviográficas; se estimaron los estadísticos de los datos generados estación por estación y se compararon con los históricos (en este caso se tenían un total de 177 tormentas históricas registradas en 49 estaciones). Para ello se requirió dibujar tres gráficas: 1) una gráfica que presenta los estadísticos de la serie histórica, de la serie sintética por el primer procedimiento de generación y de la serie sintética por el segundo procedimiento, estación por estación. 2) una gráfica en la que se dibuja una función identidad y los puntos correspondientes a la pareja de valores históricos y sintéticos del primer procedimiento, estación por estación y 3) una gráfica que contiene una función identidad y los puntos correspondientes a la pareja de valores históricos y sintéticos del segundo procedimiento, estación por estación. Lo anterior se resolvió usando tres PictureBox en una forma.

La forma que contiene las gráficas requeridas para el caso del estadístico media de las precipitaciones analizadas se presenta en la Figura 9.

La simbología de cada serie se puede hacer con un frame, insertando un label con el nombre de cada serie, con shapes y con lines se indica el símbolo de cada serie.

El Código de cada PictureBox se indica en las Figuras 10 y 11, en las que siguió un procedimiento similar al descrito en los ejemplos de la metodología. A continuación se detalla dicho código.

```
Private Sub cmddibujar12_Click()
Form14.Visible = True
Form13.Visible = False

'escala gráfica para gráfica
TIE = 50
Q_MAX = 90

Pic_Hid.ScaleWidth = TIE
Pic_Hid.ScaleTop = Q_MAX
Pic_Hid.ScaleHeight = -Q_MAX
Pic_Hid.ScaleLeft = 0

Pic_Hid.Refresh

'Picture box históricos sintéticos 1 y 2 procedimientos
'dibuja puntos
DAT_REG = nest
Pic_Hid.DrawWidth = 4
For i = 1 To DAT_REG
Pic_Hid.PSet ((i), medesth(i)), QBColor(4)
Next

'dibuja líneas sintético 1
Pic_Hid.DrawWidth = 1
DAT_AJU = nest

For i = 2 To DAT_AJU
   TR_INI = (i - 1): medests1_ini = medests1(i - 1)
    TR_FIN = (i): medests1_fin = medests1(i)
   Pic_Hid.Line (TR_INI, medests1_ini)-(TR_FIN,
medests1_fin), QBColor(8)

Next

'dibuja líneas sintético 2
Pic_Hid.DrawWidth = 1
DAT_AJU = nest
For i = 2 To DAT_AJU
   TR_INI = (i - 1): medests2_ini = medests2(i - 1)
    TR_FIN = (i): medests2_fin = medests2(i)
   Pic_Hid.Line (TR_INI, medests2_ini)-(TR_FIN,
medests2_fin), QBColor(10)
```

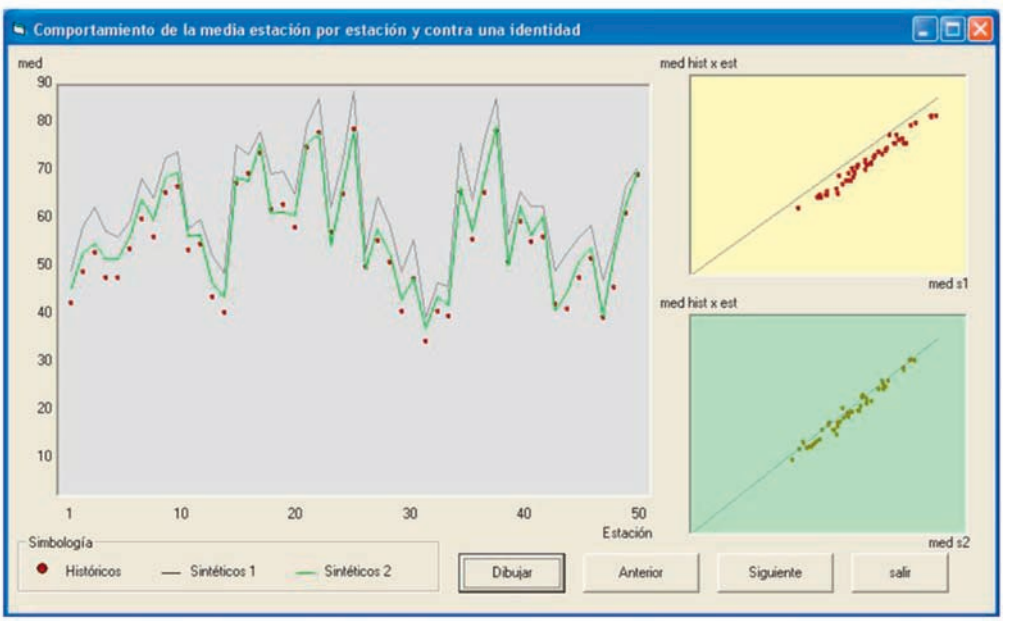

Figura 9. Dibujo de dos o más gráficos que contienen más de una serie en una misma forma.

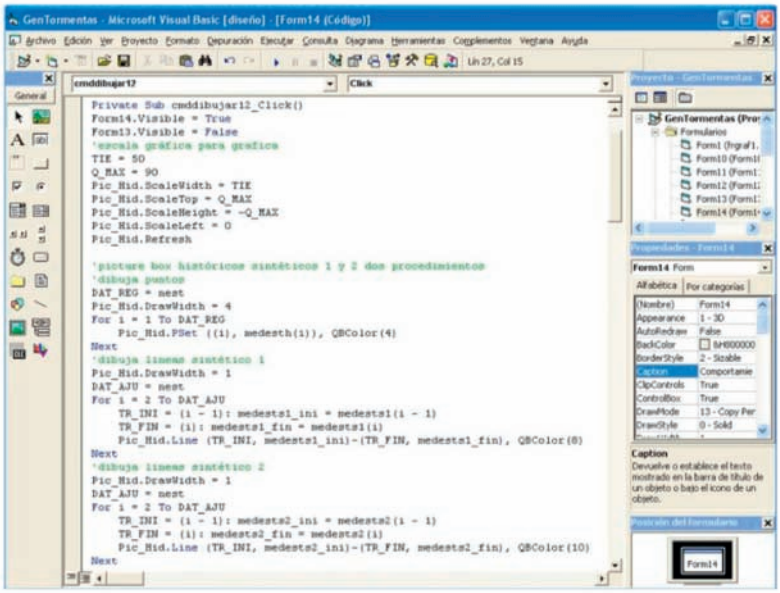

Figura 10. Código para dibujar tres series: media histórica, media sintética primer procedimiento y media sintética segundo procedimiento en 49 estaciones del Valle de México.

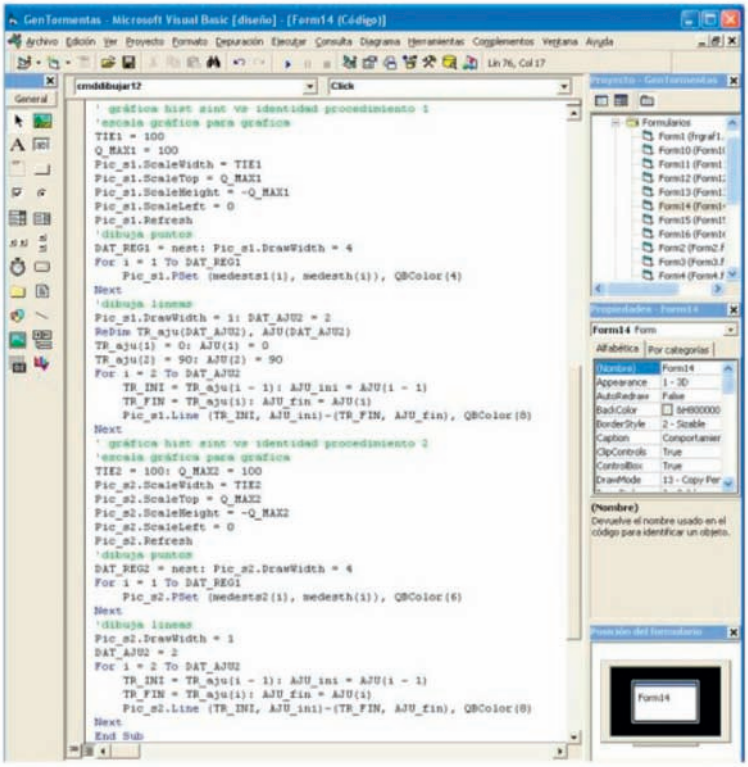

Figura 11. Código para dibujar dos series: valores históricos y sintéticos primer procedimiento con puntos y función identidad con líneas.

```
Next

‘gráfica hist sint vs identidad procedimiento 1

‘escala gráfica para gráfica
TIE1 = 100
Q_MAX1 = 100

Pic_s1.ScaleWidth = TIE1
Pic_s1.ScaleTop = Q_MAX1
Pic_s1.ScaleHeight = -Q_MAX1
Pic_s1.ScaleLeft = 0
Pic_s1.Refresh

‘dibuja puntos
DAT_REG1 = nest
Pic_s1.DrawWidth = 4
For i = 1 To DAT_REG1
  Pic_s1.PSet (medests1(i), medesth(i)), QBColor(4)
Next

‘dibuja líneas
Pic_s1.DrawWidth = 1
DAT_AJU2 = 2
ReDim TR_aju(DAT_AJU2), AJU(DAT_AJU2)
TR_aju(1) = 0: AJU(1) = 0
TR_aju(2) = 90: AJU(2) = 90

For i = 2 To DAT_AJU2
   TR_INI = TR_aju(i - 1): AJU_ini = AJU(i - 1)
   TR_FIN = TR_aju(i): AJU_fin = AJU(i)
    Pic_s1.Line (TR_INI, AJU_ini)-(TR_FIN,
AJU_fin), QBColor(8)

Next

‘gráfica hist sint vs identidad procedimiento 2

‘escala gráfica para gráfica
TIE2 = 100
Q_MAX2 = 100

Pic_s2.ScaleWidth = TIE2
Pic_s2.ScaleTop = Q_MAX2
Pic_s2.ScaleHeight = -Q_MAX2
Pic_s2.ScaleLeft = 0
Pic_s2.Refresh

‘dibuja puntos
DAT_REG2 = nest
Pic_s2.DrawWidth = 4
For i = 1 To DAT_REG1
Pic_s2.PSet (medests2(i), medesth(i)), QBColor(6)
Next

‘dibuja líneas
Pic_s2.DrawWidth = 1
DAT_AJU2 = 2

For i = 2 To DAT_AJU2
   TR_INI = TR_aju(i - 1): AJU_ini = AJU(i - 1)
   TR_FIN = TR_aju(i): AJU_fin = AJU(i)
    Pic_s2.Line (TR_INI, AJU_ini)-(TR_FIN,
AJU_fin), QBColor(8)
Next
End Sub
```

CONCLUSIONES

El dibujo de gráficos para el manejo e interpretación de datos constituye una herramienta fundamental en numerosos problemas de ingeniería práctica; en el campo de la ingeniería hidráulica e hidrológica se han utilizado PictureBox en problemas de redes de agua potable, análisis de flujo permanente y no permanente en tuberías, ubicación de válvulas de control en redes; también se han utilizado con éxito en problemas de flujo no permanente unidimensional y bidimensional como es el análisis de ondas de avenidas en cauces y en cuencas urbanas. No es frecuente encontrar publicaciones que paso a paso orienten a un ingeniero para la realización de estos dibujos. El lenguaje de programación Visual Basic es una opción viable para llevar a cabo estas tareas con ayuda del objeto PictureBox debido a que tiene la ventaja de que con éste se pueden representar series de igual o de distinta longitud (situación que el objeto MsChart no logra resolver al considerar arreglos del mismo tamaño).

En este trabajo se vio la conveniencia de utilizar PictureBox, en ambiente Visual Basic, como herramienta auxiliar para comparar los estadísticos de una serie histórica de precipitaciones con aquellos obtenidos por dos procedimientos de generación sintética, para 49 estaciones pluviográficas. Los resultados se consideran alentadores y se sugiere el empleo de esta herramienta como una alternativa para la representación de gráficos. Cabe mencionar que el procedimiento aquí presentado puede codificarse

en distintos lenguajes de programación de uso frecuente en el ámbito de la ingeniería.

Estas subrutinas con algunos cambios pueden adaptarse para ser utilizadas en Visual Studio 2008 y posteriores; su versión ejecutable puede emplearse en el ambiente Windows 7.

AGRADECIMIENTOS

Al Instituto de Ingeniería de la UNAM, por el apoyo en equipo y materiales utilizados para realizar este documento.

El lector interesado puede contactar a los autores, vía electrónica, para que le proporcionen los datos para reproducir las gráficas presentadas en este trabajo.

REFERENCIAS

[1] S.C. Chapra y R.P. Canale. "Métodos Numéricos para Ingenieros". McGraw Hill, México D.F., México. 2000.

[2] C. Igathinathane, V.S.S. Prakashb, U. Padmab, G. Ravi Babub and A.R. Womaca. "Interactive computer software development for leaf area measurement". Computers and Electronics in Agriculture. Vol. 51, Issues 1-2. April, 1-16. 2006.

[3] D. Zhenhai, W. Yajing and C. Wengang. "Teaching Software Development of Digital Image Processing Based on VB". International Conference on Educational and Information Technology (ICEIT 2010). Chongqing, China. 2010.

[4] K. Mock. "A Visualization System for Tidal Basins". CS 470 – Sample Project Write-up, p. 25. April, 2010. Fecha de consulta: 8 de febrero de 2011. URL: http://www.math.uaa.alaska.edu/~afkjm/cs470/handouts/basinvis-writeup.pdf

[5] X.L. Song, C.-Y. Liu, Z.Y. Song and L.C. Peng. "Stepping Motor Graph Drawing System Design". Proceedings of 2007 IEEE International Conference on Grey Systems and Intelligent Services. Nanjing, China. 2007.

[6] H. Liu and M. Huang. "Open GL–based conceptual models and algorithms for visualization of three-dimensional fracture networks and flow paths". International Conference on Computational Intelligence and Software Engineering (CiSE), International Conference on. Wuhan, China. 2010.

[7] C.B. Eaton. "Exploring Microsoft Visual Basic Versión 6.0". Prentice Hall. New Jersey, USA. 1999.

[8] J. García de Jalón, J.I. Rodríguez y A. Brazález. "Aprenda Visual Basic 6.0 como si estuviera en primero". Escuela Superior de Ingenieros Industriales de San Sebastián. Universidad de Navarra. San Sebastián, España. Agosto 1999.