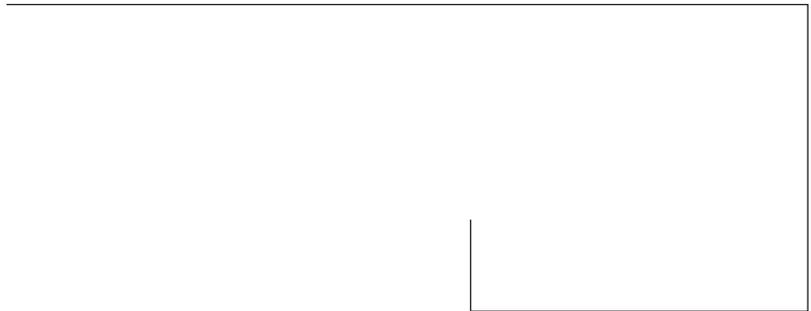






# **INICIACIÓN A LA PROGRAMACIÓN CON LA TOOLKIT DE EPANET v2.00.12 EN UN ENTORNO WINDOWS®**



Oscar Tomas Vegas Niño  
Fernando Martínez Alzamora  
Joan Carles Alonso Campos  
Velitchko G. Tzatchkov

2017

---

627.0113 Vegas Niño, Oscar Tomas  
V15 *Iniciación a la programación con la Toolkit de Epanet v2.00.12 en un entorno  
Windows® / Oscar Tomas Vegas Niño et al.-- Jiutepec, Mor. : Instituto Mexicano  
de Tecnología del Agua, 2017.*  
113 p.

ISBN 978-607-9368-72-2 (obra digital)

1. Programas de computación 2. Simulación hidráulica

---

Autores:

Oscar Tomas Vegas Niño,  
Fernando Martínez Alzamora  
Joan Carles Alonso Campos  
Velitchko G. Tzatchkov

Coordinación editorial:

José Manuel Rodríguez Varela  
Gema Alín Martínez Ocampo  
Coordinación de Hidráulica  
Instituto Mexicano de Tecnología del Agua

Primera edición: 2017

Instituto Mexicano de Tecnología del Agua  
Paseo Cuauhnáhuac 8532  
62550 Progreso, Jiutepec, Morelos  
México  
[www.imta.gob.mx](http://www.imta.gob.mx)

ISBN 978-607-9368-72-2 (obra digital)

Las opiniones, datos y citas presentados en esta obra son responsabilidad exclusiva de los autores y no reflejan, necesariamente, los puntos de vista de la institución que edita esta publicación.

Prohibida su reproducción parcial o total, por cualquier medio, mecánico, electrónico, de fotocopias, térmico u otros, sin permiso de los editores.

Hecho en México

# C o n t e n i d o

PRESENTACIÓN	9
PRÓLOGO	11
1. El Software Epanet	15
1.1. ¿Qué es Epanet?	15
1.2. Componentes físicos y no físicos	17
1.3. ¿Qué se puede hacer con Epanet?	17
1.4. Limitaciones de la aplicación	18
1.5. Instalando Epanet en Windows de 32/64 bits	19
1.6. Lista de usuarios	20
2. Estructura de datos de Epanet	23
2.1. Secciones que componen el fichero de entrada	23
2.2. Secciones principales para el dibujado de una red	24
3. La Toolkit de Epanet	27
3.1. La API de Epanet	27
3.2. Funciones de la API	30
4. Conectar la API de Epanet a los Entornos de Programación	35
4.1. Visual Basic 6.0 (Basic 6.0)	36
4.2. Visual Studio 2015 (Visual Basic .NET)	44
4.3. Matlab (Lenguaje de cálculo técnico)	51
4.4. Visual Studio 2015 (C#)	61
4.5. Python Shell (Python)	70
4.6. Dev-C++ (C++)	82
5. Ejercicios prácticos	91
DE LOS AUTORES	110

# T a b l a s

Tabla 2.1.	Palabras claves de las distintas secciones del fichero de entrada	23
Tabla 3.1.	Descripción de los ficheros que componen la Toolkit de Epanet	28
Tabla 3.2.	Funciones de la Toolkit de Epanet agrupadas por tareas	30
Tabla 4.1.	Controles añadidos al formulario principal (Visual Basic)	41
Tabla 4.2.	Controles añadidos al formulario principal (Visual .NET)	50
Tabla 4.3.	Ficheros y carpetas contenidos en el fichero comprimido EPANET-Matlab-Toolkit-master.zip	55
Tabla 4.4.	Controles añadidos al formulario principal	68
Tabla 5.1.	Palabras claves utilizadas para recuperar valores después de una simulación	104

## F i g u r a s

Figura 1.1.	Lista de ficheros Epanet desde la web de la EPA	16
Figura 3.1	Ficheros que componen la Toolkit de Epanet	28
Figura 3.2.	Ayuda en línea para programadores con la Toolkit de Epanet	33
Figura 4.1.	Cargar fichero epanet2.bas al proyecto	38
Figura 4.2.	Cambios de nombre al proyecto y formulario	39
Figura 4.3.	Ficheros guardados en la carpeta de proyectos VB6.0	39
Figura 4.4.	Control CommonDialog añadido al cuadro de herramientas de VB6.0	40
Figura 4.5.	Formulario en modo diseño y ejecución de nuestra aplicación	41
Figura 4.6.	Código fuente asociado al botón CmdAceptar	42
Figura 4.7.	Resultado de consultar el número de líneas y nudos de la red Net3	43
Figura 4.8.	Número de nodos y líneas de la Red Net3 desde Epanet	43
Figura 4.9.	Crear proyecto nuevo en Visual Studio 2015	45
Figura 4.10.	Declaración de los tipos de datos de variables y funciones en VB6.0 y VB.NET	46
Figura 4.11.	Módulo Epanet2.vb añadido al proyecto	46
Figura 4.12.	Visualización de las carpetas bin y obj desde el explorador de soluciones VB.NET	47
Figura 4.13.	Copiado del fichero epanet2.dll en la carpeta Debug	48
Figura 4.14.	Cuadro de herramientas dentro de Visual Basic 2015	49
Figura 4.15.	Parte del código fuente y el formulario principal en modo ejecución	51
Figura 4.16.	Descarga de la Toolkit de Epanet para Matlab	54
Figura 4.17.	Carpetas y ficheros que contendrá nuestro proyecto	54
Figura 4.18.	Ficheros indispensables para utilizar la Toolkit de Epanet con Matlab	56
Figura 4.19.	Pasos para seleccionar la carpeta de trabajo actual desde Matlab	57
Figura 4.20.	Carpeta de proyecto lista para trabajar con la Toolkit de Epanet	58
Figura 4.21.	Mensaje de error por no encontrar la carpeta 64bit en la carpeta de proyecto	59

Figura 4.22. Crear un nuevo script desde Matlab	60
Figura 4.23. Resultados mostrados después de lanzar el script	61
Figura 4.24. Crear nuevo proyecto y solución desde Visual Studio 2015	63
Figura 4.25. Copiado de la clase EpanetCSSharpLibrary a la carpeta de proyecto	65
Figura 4.26. Cargar la clase EpanetCSSharpLibrary desde Visual Studio 2015	65
Figura 4.27. Visualización de las carpetas bin y obj desde el explorador de soluciones	66
Figura 4.28. Copiado del fichero epanet2.dll en la carpeta Debug	67
Figura 4.29. Cuadro de herramientas dentro de Visual Studio 2015 (C#)	68
Figura 4.30. Ejecución y resultados desde Visual Studio 2015 (C#)	70
Figura 4.31. Enlace para descargar Python 3.3 desde la página oficial	72
Figura 4.32. Instalación de Python 3.3	73
Figura 4.33. Herramienta Python Shell dentro de Python 3.3	73
Figura 4.34. Búsqueda del instalador Epanet2 0.4.0.1dev	74
Figura 4.35. Resultado de la búsqueda del instalador Epanet2 0.4.0.1dev	75
Figura 4.36. Descarga del instalador “Epanet2 0.4.0.1dev”	75
Figura 4.37. Comprobación del directorio Python 3.3 en el proceso de instalación	76
Figura 4.38. Instalación del paquete epanet2, EPANET2-0.4.0.1dev, y _epanet2	76
Figura 4.39. Paquete “epanet2” y módulo “epanet2”	77
Figura 4.40. Importación del módulo epanet2.py	78
Figura 4.41. Resultado de la consulta al fichero Net3.inp desde Python Shell	78
Figura 4.42. Código fuente (Lenguaje Python) escrito desde el NotePad++	79
Figura 4.43. Copiar fichero ScriptEpanetToolkit.py en la carpeta site-packages	80
Figura 4.44. Importar fichero ScriptEpanetToolkit y mostrar resultados	80
Figura 4.45. Ejecutar el script desde Python Shell	81
Figura 4.46. Resultado de Ejecutar el script	81
Figura 4.47. Copiado de los ficheros epanet2.h y epanet2.dll dentro de la carpeta Proyecto1	84
Figura 4.48. Crear un nuevo proyecto desde Dev-C++	85

Figura 4.49. Guardar fichero ConectarAPIEpanet.cpp dentro de la carpeta Proyecto1	85
Figura 4.50. Ficheros generados en la carpeta Proyecto1	86
Figura 4.51. Ventana “Opciones de Proyecto” desde Dev-C++	86
Figura 4.52. Selección del compilador a utilizar	87
Figura 4.53. Búsqueda y selección de los ficheros epanet2.dll y epanet2.h	88
Figura 4.54. Añadir el directorio donde se almacena los ficheros epanet2.h y epanet2.dll	88
Figura 4.55. Configuración de la pestaña Construcción	89
Figura 4.56. Resultado de compilar el código fuente	90
Figura 5.1. Funciones utilizadas para recuperar datos de los nudos de caudal	93
Figura 5.2. Funciones utilizadas para recuperar datos de las tuberías	93
Figura 5.3. Inspección de los datos almacenados en los vectores NudoCaudal y Tuberia	94
Figura 5.4. Consulta del número de tuberías con diámetro igual a 12” desde Epanet	95
Figura 5.5. Modificar diámetros de tubería con la herramienta Group Edit desde Epanet	95
Figura 5.6. Modificar diámetros de las tuberías con las funciones de la librería de Epanet	96
Figura 5.7. Uso de la función ENSolveH y ENSaveH de la librería de Epanet	98
Figura 5.8. Ficheros Net3.rpt y Net3.out creados al finalizar el proceso	99
Figura 5.9. Simulación hidráulica paso a paso con la librería de Epanet	100
Figura 5.10. Parámetros de calidad a introducir en Epanet	102
Figura 5.11. Uso de la función ENSolveQ de la librería de Epanet	102
Figura 5.12. Simulación de la calidad del agua paso a paso con la librería de Epanet	103
Figura 5.13. Funciones que ayudan a generar un informe de resultados	105
Figura 5.14. Funciones utilizadas para el cálculo de presiones en el nudo 23	107
Figura 5.15. Herramienta para invertir las líneas mal orientadas con ayuda de la librería de Epanet	109



## P R E S E N T A C I Ó N

**L**as compañías o municipios que gestionan los abastecimientos de agua potable, están apostando fuertemente por el uso de aplicaciones informáticas como Dibujo Asistido por Computadora (CAD), Sistemas de Información Geográfica (SIG), herramientas de cálculo y software especializado para el modelado de sistemas de suministro de agua, sean éstas de uso libre o adquiriendo una licencia.

Muchas de las aplicaciones informáticas vinculadas al uso y explotación de los modelos necesitan disponer de un simulador hidráulico que proporcione la respuesta de la red ante cualquier escenario, tanto en lo que se refiere a las variables hidráulicas como de calidad del agua. EPANET, además de responder a esta necesidad funcionando como una aplicación independiente, con su propia interfaz gráfica para construir el modelo de la red y analizar los resultados, permite también conectar su librería de cálculo, o Toolkit como usualmente se la conoce, a cualquier otra aplicación o interface gráfica, abriendo así un campo de posibilidades enorme para cualquier técnico o investigador que quiera ir más allá de lo que actualmente dicha aplicación ofrece.

La Toolkit de Epanet puede integrarse en cualquier entorno de programación nativo o en las propias herramientas de programación que ofrecen muchos paquetes de software propietario, ya sean de pago como Excel, AutoCad, ArcGIS, etc o bien libres como OpenOffice, gvSIG, QSIG, etc. De hecho, es lo que han hecho algunos programas comerciales desarrollados por empresas o universidades, orientados al diseño y análisis de redes de distribución de agua.

La librería de cálculo de Epanet cuenta con un elevado número de funciones que nos ayudan a recuperar y modificar determinados parámetros del modelo de red, antes y después de realizar una simulación hidráulica, permitiendo además controlar el proceso de simulación paso a paso. Sin embargo, no existe en la literatura mucha información sobre el manejo de todas estas funciones o bien la existente se encuentra en otros idiomas. Por otra parte, se echa en falta más información sobre cómo integrar la Toolkit en los diversos entornos de programación más usuales, y cómo resolver los problemas más frecuentes, lo que constituye un primer paso siempre difícil de dar.

El presente libro está pensando para ayudar a alumnos de master, personal técnico e investigadores a iniciarse en la programación con la librería de Epanet, con distintos lenguajes y entornos. Asimismo, se enseña a utilizar la mayoría de las funciones contenidas en la librería desarrollando ejercicios prácticos, desde el entorno de programación Visual Studio 2015, para que el usuario pueda entenderlos y posteriormente consiga crear sus propias herramientas.

## P R Ó L O G O

**E**l software Epanet, que permite realizar simulaciones hidráulicas y de calidad del agua de una red de suministro, se ha convertido en la herramienta más popular y utilizada mundialmente por las compañías o municipios que operan y gestionan las redes de abastecimiento. Esto se debe a que dicha aplicación es de uso libre y de código abierto. Su popularidad se ha incrementado debido a que centros de investigaciones, entidades públicas y empresas privadas de todo el mundo, han colaborado en la traducción del programa así como de su manual en más de cinco idiomas distintos de su versión original (inglés).

Los requerimientos técnicos y operativos en la gestión de las redes de suministro de agua han aumentado considerablemente, y con ellos la utilización de herramientas específicas a cada problema particular. Para el desarrollo de trabajos a medida, la Agencia de Protección Ambiental de los Estados Unidos de Norteamérica (Environmental Protection Agency, EPA) ha proporcionado una caja de herramientas (Toolkit de Epanet) que permite conectar la librería dinámica de cálculo con algunos entornos de programación y resolver cierta problemática que en Epanet no se encuentra.

La ayuda en línea que se ofrece en la caja de herramientas del programador no es lo suficientemente entendible para un usuario con escaso conocimiento en programación. Además, los ficheros de cabecera, donde se declaran las variables y funciones, están preparados para ser utilizados sobre un Sistema Operativo (SO) Windows de 32 bits y por entornos de programación que hoy en día tienen poco uso.

En la actualidad existe poca literatura disponible al respecto. No es fácil encontrar una guía de ayuda, y si la hay, quizá sea con el lenguaje de programación que menos estemos familiarizados, o bien se encuentra en un idioma extranjero distinto a nuestra lengua materna. Los repositorios de código fuente, webs o blogs de investigadores, son una buena fuente de información, pero sin el conocimiento de algún lenguaje de programación puede retrasarse nuestro proceso de aprendizaje.

El libro consta de cinco capítulos. El capítulo uno se ha reservado para dar a conocer lo que es el software Epanet, sin ahondar mucho, que para eso está el manual de usuario. En el capítulo dos nos adentramos a lo que es la estructura de datos del fichero de entrada, ya que es importante conocer cómo está estructurado la información de una red de abastecimiento. En el capítulo tres se detallan las funciones que existen en la librería de Epanet y la manera de cómo utilizar cada una de ellas. En el capítulo cuatro, el más importante, se enseña a conectar la librería con los entornos de programación más utilizados en el mercado. Y por último, en el capítulo cinco se desarrollan unos ejemplos que serán de gran ayuda para el usuario, desde el entorno de programación de Microsoft Visual Studio 2015 (Visual Basic .NET)

En este libro no se pretende enseñar lenguajes de programación, sino cómo conectar la librería de Epanet desde los entornos de programación más utilizados hoy en día. Se recomienda, que el usuario tenga nociones básicas de al menos un lenguaje.

El libro está escrito en el lenguaje técnico y terminología que se utiliza en España. Algunos pocos términos, palabras o frases específicas pueden ser diferentes, o interpretados de manera distinta, en otros países de habla hispana. Para ayudar a los lectores mexicanos, y quizás de otros países de América Latina, la siguiente tabla presenta equivalencias de tales palabras o frases que se han identificado. Dado que varios de ellos provienen de traducción de inglés, se presenta también la correspondiente palabra o frase en inglés.

## P R Ó L O G O

---

Término que se usa en España	Término que se usa en México	Término en Inglés
fichero	archivo	file
curva de modulación	patrón, curva de variación	pattern
abonado	usuario del servicio	user
coste	costo	cost
parada de una bomba	paro de una bomba	pump shutdown
ordenador	computadora	computer
prestaciones	facilidades, capacidades	capabilities, features
pinchar	hacer (dar) clic	click
clicar	hacer (dar) clic	click
volcar resultados en ficheros	vaciar resultados en archivos	dump results in files
nudo	nodo	node
fiabilidad	confiabilidad	reliability



# 1

## E L S O F T W A R E E P A N E T

### 1.1. ¿QUÉ ES EPANET?

**E**panet es un programa de ordenador que permite realizar simulaciones en periodos prolongados (uno o varios días) del comportamiento hidráulico y de la evolución de la calidad del agua en redes de suministro a presión. Una red puede estar constituida por tuberías, nudos (uniones de tuberías), bombas, válvulas, depósitos de almacenamiento y embalses. Epanet efectúa un seguimiento de la evolución de los caudales en las tuberías, las presiones en los nudos, los niveles en los depósitos, y la concentración de las especies químicas presentes en el agua a lo largo del periodo de simulación discretizado en múltiples intervalos de tiempo. Además de la concentración de las distintas especies, puede también simular el tiempo de permanencia del agua en la red y su procedencia desde las diversas fuentes de suministro<sup>1</sup>.

Epanet ha sido diseñado y desarrollado por la Agencia de Protección Ambiental de los Estados Unidos (United States Environmental Protection Agency, USEPA). Como tal, es un software de dominio público que puede bajarse y distribuirse libremente desde la página oficial: <https://www.epa.gov/water-research/epanet>. La versión original del programa fue desarrollada en inglés y ha sido traducida al español por varias instituciones. En España y Latinoamérica se utiliza mucho la versión traducida por el Instituto de Ingeniería del Agua y Medio Ambiente (IIAMA) de la

---

<sup>1</sup> Martínez Alzamora, F (2002, 2017). Epanet 2.0 en Español. <http://www.iiama.upv.es/iiama/index.php/transferecia/software/epanet-esp>.

Universitat Politècnica de Valencia. La aplicación también ha sido traducida a otros idiomas como: francés, italiano, checo, alemán, portugués, etc.

Desde la página web oficial de la EPA, además de poder descargar el programa (v.2.00.12), se pueden descargar otros ficheros como los que se muestran en la Figura 1.1.

Date	File Description
3/5/08	<a href="#">Self-extracting installation program for EPANET 2.00.12 (EXE)</a> (2 MB) (1.5 MB)
9/11/00	<a href="#">EPANET 2 Users Manual (PDF)</a> (200 pp, 1 MB, See <a href="#">EPA's About PDF</a> )
3/20/08	<a href="#">EPANET 2 Programmer's Toolkit files (ZIP)</a> (253 K) (247 KB)
5/27/08	<a href="#">EPANET 2 source code files (ZIP)</a> (566 K) (553 K)
2/25/08	<a href="#">EPANET 2 Updates</a> (22 K) (22 K)
5/7/08	<a href="#">Multi-Species Extension</a>

Figura 1.1. Lista de ficheros Epanet desde la web de la EPA

Los ficheros que se listan en la web de la EPA son los siguientes:

- El manual de usuario
- La caja de herramientas del programador
- El código fuente del programa Epanet
- Un fichero de texto donde se describen las actualizaciones realizadas del programa
- La extensión multi-especies

Para aprender a manejar la herramienta, se puede consultar el manual de usuario que está disponible en la web de la EPA (inglés) o si se prefiere en español u otro idioma, buscar en los enlaces que se citan a continuación:

- <http://www.iiama.upv.es/iiama/index.php/transferecia/software>
- <http://www.arnalich.com/epanetes/descargas.html>
- <http://epanet.de/>

## 1.2. COMPONENTES FÍSICOS Y NO FÍSICOS

Epanet modela una red de suministro de agua como un conjunto de líneas conectadas por nudos extremos. Las líneas representan tuberías, bombas, o válvulas de control. Los nudos representan puntos de conexión entre tuberías o extremos de las mismas, con o sin demandas (nudos de caudal), y también depósitos o embalses. Epanet reconoce los seis objetos antes descritos (nudos de caudal, embalses, depósitos, tuberías, bombas y válvulas) para llevar a cabo una simulación de una red de abastecimiento de agua<sup>2</sup>.

Además de los elementos mencionados anteriormente, existen otros componentes complementarios que ayudan a describir el comportamiento y modo de operación del sistema. Estos componentes no físicos son: curvas de comportamiento, curvas de modulación y leyes de control. Para más información se recomienda consultar el manual de usuario de Epanet.

## 1.3. ¿QUÉ SE PUEDE HACER CON EPANET?

Epanet se ha concebido como una herramienta de investigación para mejorar nuestro conocimiento sobre el avance y destino final de las diversas sustancias transportadas por el agua, mientras ésta discurre por la red de distribución. Entre sus diferentes aplicaciones pueden citarse el diseño de programas de muestreo, la calibración de un modelo hidráulico, el análisis del cloro residual, o la evaluación de las dosis totales suministradas a un abonado. Epanet puede resultar también de ayuda para evaluar diferentes estrategias de gestión dirigidas a mejorar la calidad del agua a lo largo del sistema, como pueden ser:

- Alternar la toma de agua desde diversas fuentes de suministro
- Modificar el régimen de bombeo, o de llenado y vaciado de los depósitos

---

<sup>2</sup> <https://es.scribd.com/doc/8470575/Epanet-y-Cooperacion-Introduccion-al-Calculo-de-Redes-de-Agua-por-Ordenador>

- Implantar estaciones de tratamiento secundarias, tales como estaciones de recloración o depósitos intermedios
- Establecer planes de limpieza y reposición de tuberías, etc.

Epanet puede también utilizarse para mejorar las características de la red hidráulica. Entre los posibles usos tenemos el trazado y selección de nuevos elementos, reducción de los costes de operación, realización de estudios de vulnerabilidad, detección de los ‘cuellos de botella’, evaluación de la calidad y tiempo de permanencia del agua en las tuberías, regulación de las presiones, regulación del uso de los depósitos para reducir los tiempos de retención del agua, previsión de la respuesta de la red ante la clausura de un punto de alimentación o la incorporación de nuevas urbanizaciones, planificación de actuaciones en casos de emergencia, como la entrada de un contaminante no controlado, sectorización de la red para el control de fugas, etc. En el futuro los modelos deben constituir un soporte de uso continuado para la toma de decisiones de carácter técnico en todas las empresas encargadas de la distribución del agua potable.

#### 1.4. LIMITACIONES DE LA APLICACIÓN

Santiago Arnalich, en su libro “Epanet y Cooperación. Introducción al cálculo de redes de agua por ordenador” explica muy bien estas limitaciones. Epanet, al ser un modelo hidráulico cuasi-estático (modelo no inercial), es decir, que no contempla los cambios bruscos en la red, no tiene capacidad para modelar estos fenómenos reales y rápidos como: rotura en una tubería, golpe de ariete causado por el cierre de una válvula, el arranque o parada de una bomba, cierre repentino de una válvula de no retorno, etc.

En la parte de modelación de calidad del agua, Epanet no considera el término de dispersión en la ecuación diferencial del transporte de la sustancia modelada. Este término, y con esto la influencia de la dispersión, es importante en tuberías de la red con velocidades muy bajas. Tzatchkov et al. (2002)<sup>3</sup> publicaron y aplicaron un

---

<sup>3</sup> Velitchko G. Tzatchkov, Alvaro A. Aldama, Felipe I. Arreguin, “Advection-dispersion-reaction modeling in water distribution networks”, *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, vol. 128(5), p. 334-342.

algoritmo que permite considerar este término, usando el archivo de resultados de cálculo hidráulico del Epanet.

### 1.5. INSTALANDO EPANET EN WINDOWS DE 32/64 BITS

La última versión de Epanet, así como la librería dinámica (dll), han sido compiladas para funcionar bajo un SO Windows de 32 bits. Cuando aparecieron los nuevos ordenadores con arquitectura y SO Windows de 64 bits, los usuarios de Epanet se preguntaban si la aplicación iba a dejar de funcionar. Pues en realidad nunca hubo problema alguno. Actualmente se viene utilizando sobre Windows 10 con total normalidad<sup>4</sup>. Las versiones de Epanet, desde la 1.0 a la 2.0, emitidas por la USEPA, han sido diseñadas y desarrolladas para ser instaladas y ejecutadas desde el entorno Windows®. Algunos investigadores han reescrito el código fuente original en otros lenguajes para poder ser instalado en otros entornos como Linux y MAC.

Lo que si hubo es un pequeño problema, que no se podía abrir el fichero de ayuda en línea cuando apareció el SO Windows Vista y posteriores versiones. Esto se debió a que Microsoft ya no incluía el programa de ayuda WinHlp32.exe en la distribución. Para poder seguir visualizando el fichero de ayuda en línea de Epanet hay que descargar e instalar el fichero WinHlp32.exe desde la web del soporte técnico que brinda la Microsoft para las distintas versiones de Windows <https://support.microsoft.com/es-es/kb/917607>.

Asimismo, Elad Salomons en su página [web: http://www.water-simulation.com/wsp/2015/10/01/how-to-open-epanets-help-file-in-windows-10/](http://www.water-simulation.com/wsp/2015/10/01/how-to-open-epanets-help-file-in-windows-10/), describe los pasos que hay que seguir para poder utilizar la ayuda en línea de Epanet.

---

<sup>4</sup> <http://www.water-simulation.com/wsp/>

## 1.6. LISTA DE USUARIOS

Epanet ha creado un antes y un después en el mundo del software orientado a la simulación del comportamiento de las redes hidráulicas. Su carácter gratuito, el respaldo de la EPA y el buen hacer de su autor, L. Rossman, han contribuido a ello definitivamente.

Desde la aparición de la primera versión de Epanet, el Prof. William James de la Universidad de Guelph (Canadá) tuvo la iniciativa de crear una lista de usuarios de Epanet, cuya suscripción es por ahora gratuita. A través de la misma, los usuarios de Epanet pueden conocer las últimas novedades en torno al programa, compartir experiencias, formular preguntas y conocer las respuestas emitidas por otros usuarios más experimentados.

Como es usual en las listas de usuarios de tipo abierto, existe una dirección reservada para la comunicación con el servidor a través de comandos: [listserv@listserv.uoguelph.ca](mailto:listserv@listserv.uoguelph.ca), y otra para el acceso a los mensajes propiamente dichos: [EPANET-USERS@LISTSERV.UOGUELPH.CA](mailto:EPANET-USERS@LISTSERV.UOGUELPH.CA). Para suscribirse a la lista de correo es muy sencillo, tan sólo hay que enviar un correo electrónico a la dirección [listserv@listserv.uoguelph.ca](mailto:listserv@listserv.uoguelph.ca) que contenga como texto del mensaje: SUBSCRIBE EPANET-USERS nombre de usuario. Ejemplo: SUBSCRIBE EPANET-USERS Oscar Vegas Niño.

Es importante no rellenar el campo Asunto, y procurar que el mensaje no contenga ningún otro texto o carácter extraño (como punto final, etc.). Al cabo de unos minutos, el sistema responderá con otro correo aceptando el alta o comunicando cualquier problema con el mensaje emitido.

Para darse de baja de la lista de correo, es necesario enviar un correo electrónico a la dirección [listserv@listserv.uoguelph.ca](mailto:listserv@listserv.uoguelph.ca) que contenga como texto del mensaje: UNSUBSCRIBE EPANET-USERS nombre de usuario. Para plantear cuestiones, realizar consultas o dar respuestas se deberá enviar un correo a la siguiente dirección: [EPANET-USERS@LISTSERV.UOGUELPH.CA](mailto:EPANET-USERS@LISTSERV.UOGUELPH.CA). Este mensaje será recibido por todos los usuarios de la lista.

También existe un foro para usuarios hispanoamericanos habilitado por Santiago Arnalich. Desde su página web <http://www.epanet.es/> puede darse uno de alta introduciendo un correo electrónico. Asimismo, existe una comunidad de desarrolladores (investigadores) que vienen trabajando en añadir nuevas prestaciones al programa Epanet, así como para resolver algunas dudas de otros usuarios. Para ello, hay que darse de alta a través del siguiente enlace: <http://community.wateranalytics.org/>.



# 2

## ESTRUCTURA DE DATOS DE EPANET

### 2.1. SECCIONES QUE COMPONEN EL FICHERO DE ENTRADA

Un fichero de entrada que Epanet pueda leer deberá tener la extensión \*.INP (fichero de texto ASCII) o \*.NET (fichero binario) y una estructura de datos definida donde se almacene toda la información necesaria para visualizar y realizar una simulación válida de nuestro modelo de red que estemos trabajando.

Los datos del fichero de entrada se encuentran agrupados en distintas categorías, cada una precedida por una etiqueta (palabra clave encerrada entre corchetes) que permite identificarla claramente. Las palabras claves, que en total son 29, se listan en la Tabla 2.1.

Tabla 2.1. Palabras claves de las distintas secciones del fichero de entrada

Componentes de la red	Operación del Sistema	Calidad del Agua	Opciones e Informes	Esquemas de la red/ Rótulos
[TITLE]	[CURVES]	[QUALITY]	[OPTIONS]	[COORDINATES]
[JUNCTIONS]	[PATTERNS]	[REACTIONS]	[TIMES]	[VERTICES]
[RESERVOIRS]	[ENERGY]	[SOURCES]	[REPORT]	[LABELS]
[TANKS]	[STATUS]	[MIXING]	[REACTIONS]	[TAGS]
[PIPES]	[CONTROLS]		[BACKDROP]	
[PUMPS]	[RULES]		[END]	
[VALVES]	[DEMANDS]			
[EMITTERS]				

El orden en que se escriban las distintas secciones en el fichero es irrelevante. La única consideración a tener en cuenta es que cuando en una sección se hace referencia a un nudo o línea, éste debe haber sido declarado anteriormente en alguna de las siguientes secciones: [JUNCTIONS], [RESERVOIRS], [TANKS], [PIPES], [PUMPS], o [VALVES]. Por ello, se recomienda ubicar estas secciones al comienzo del fichero, justo después de la sección [TITLE]. Las secciones correspondientes al “Esquema de la red y los rótulos” no son utilizadas durante una simulación de Epanet, de modo que pueden o no estar en el fichero.

Cada sección puede contener uno o más líneas de datos, en función del número de elementos que se vayan añadiendo a nuestro modelo de red. Es posible insertar líneas en blanco en cualquier lugar del fichero, y además se puede utilizar un punto y coma (;) para indicar que lo que viene a continuación es un comentario y no debe interpretarse como dato. El número máximo de caracteres que puede tener una línea de texto es de 255 y los identificativos “ID” utilizados para identificar los componentes visibles, curvas de comportamiento, y curvas de modulación pueden tener cualquier combinación de números y letras, hasta un máximo de 31 caracteres.

## **2.2. SECCIONES PRINCIPALES PARA EL DIBUJADO DE UNA RED**

Cuando dibujamos una red de abastecimiento de agua potable desde la propia interfaz gráfica de Epanet y procedemos a guardarla, la aplicación automáticamente genera todas las etiquetas (palabras claves entre corchetes) que conforman el fichero de entrada, que en total puede contener hasta 29 secciones.

Si la red de abastecimiento se encuentra dibujada en AutoCAD, GIS, o cualquier otra aplicación externa a Epanet, se tendría que buscar alguna herramienta que ayude a pasar la información almacenada en dichos formatos, y en particular el trazado de la red. Otra opción es programar nuestras propias herramientas que ayuden a dibujar las tuberías y los nudos de caudal, que son los elementos más numerosos en una red, y para ello tan sólo bastaría que el fichero de entrada tenga las cuatro secciones

principales: [JUNCTIONS], [PIPES], [COORDINATES], [VERTICES]. En caso que las tuberías no cuenten con vértices, la sección [VERTICES] no es necesario que esté presente. El fichero de entrada quedaría de la siguiente forma:

[JUNCTIONS]

;ID Nudo Cota Demanda Curva de modulación

[PIPES]

;ID Línea Nudo1 Nudo2 Longitud Diámetro Rugosidad PérdMen Estado

[COORDINATES]

;ID Nudo Coord X Coord Y

[VERTICES]

;ID Línea Coord X Coord Y

El resto de elementos faltantes como bombas, válvulas, depósitos y embalses, si lo hubiera, pueden ser añadidos manualmente desde el mismo entorno de Epanet, y completar la información faltante rellenando cada una de sus propiedades.



# 3

## LA TOOLKIT DE EPANET

### 3.1. LA API DE EPANET

La Toolkit de Epanet, que puede descargarse desde la página oficial de la EPA, es un conjunto de archivos conformados por el fichero de ayuda en línea, un fichero de texto, la librería dinámica (dll), y cuatro ficheros de cabecera que permiten enlazar las funciones escritas en la librería dinámica, según el entorno de programación que vayamos a utilizar. Con estas funciones podemos acceder a la información contenida en el fichero de entrada de Epanet, en formato INP, y a los resultados generados después de realizar una simulación.

La API de Epanet es una librería dinámica, un fichero con extensión .dll, que se distribuye conjuntamente con los demás ficheros que conforman la caja de herramientas para programadores (Toolkit). Dicha librería está constituida por una serie de funciones que permiten a los desarrolladores personalizar el módulo de cálculo de Epanet según sus necesidades particulares.

En el fichero de ayuda en línea, incluido en la caja de herramientas, se puede encontrar una pequeña ayuda donde se describe la secuencia de pasos de cómo utilizar las funciones, pero para un usuario con pocos conocimientos en programación le resultaría complicado poder empezar a utilizar dicha librería, ya que no cuenta con todas las líneas de código que deben estar escritas para su correcto funcionamiento.

La Toolkit de Epanet o caja de herramientas de la última revisión (v2.00.12), está conformada por siete ficheros que se muestran en la Figura 3.1 y se describen en la Tabla 3.1.

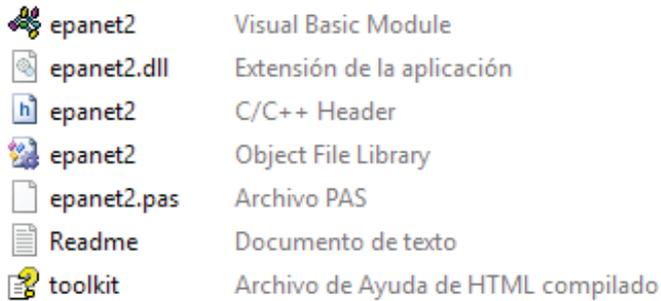


Figura 3.1. Ficheros que componen la Toolkit de Epanet

Tabla 3.1. Descripción de los ficheros que componen la Toolkit de Epanet

epanet2.bas	Módulo de declaraciones para utilizarlo con Visual Basic 6.0
epanet2.dll	Librería dinámica
epanet2.h	Fichero de cabecera para utilizar las funciones con C/C++ 5.0
epanet2.lib	Fichero LIB para utilizar las funciones con Borland C/++ 5.0 o Microsoft Visual C/C++ 6.0
epanet.pas	Unidad a importar para utilizar las funciones con Delphi 5.0 (Object Pascal)
Readme	Describe el contenido del módulo de herramientas
Toolkit.hlp	Fichero de ayuda para Windows que describe cómo utilizar el módulo de herramientas.

La declaración de variables y funciones escritas en cada uno de estos ficheros de cabecera están preparadas para ser utilizadas en entornos de programación que se ejecutan sobre el SO Windows de 32 bits. Si queremos utilizar la caja de herramientas en entornos de programación con versiones más actualizadas tendremos que modificar el tipo de dato de ciertas variables y funciones, y la ubicación de la librería de Epanet en el entorno de programación que estemos utilizando.

Los ficheros de cabecera, que contienen variables y funciones, pueden incorporarse en los entornos de programación que funcionan sobre el SO Windows de 32 bits como C/C++, Delphi Pascal, Visual Basic, o cualquier otro lenguaje que permita la

llamada a funciones incorporadas a una DLL de Windows. El fichero, con extensión .dll, incluido en la caja de herramientas, se llama epanet2.dll y se distribuye conjuntamente con la aplicación Epanet, el cual contiene 55 funciones que permiten abrir un fichero de datos, leer y modificar distintos parámetros de diseño y operación de la red, ejecutar simulaciones en periodo extendido paso a paso, acceder a los resultados a medida que son generados, guardar éstos en un fichero, o escribir determinados resultados en un fichero de texto bajo un formato determinado.

En el fichero de ayuda, incluido también en la caja de herramientas, se describe brevemente la manera de cómo utilizar algunas funciones en diversos entornos de programación sobre algunos ejemplos básicos. La caja de herramientas de la versión española, Epanet 2.00.10, traducida por el Grupo de Investigación REDHISP del IIAMA de la Universitat Politècnica de València, incluye el código fuente de tres aplicaciones completas, basadas en los ejemplos ofrecidos en el fichero de ayuda del mismo, y escritas en los lenguajes C++ Builder 5.0, Visual C++ 6.0, Delphi 5.0 y Visual Basic 6.0. Dichas instrucciones ayudan a iniciarse mínimamente en el uso de algunas funciones.

La caja de herramientas resulta útil para el desarrollo de aplicaciones a medida, tales como la optimización o la calibración automática de modelos, en las que se requiere efectuar múltiples análisis en función de los valores que adopten determinados parámetros de entrada dentro de un proceso iterativo. Asimismo, la librería dinámica (epanet2.dll) permite incorporar prestaciones de cálculo adicionales a entornos integrados de simulación basados en CAD, GIS o Bases de Datos.

### 3.2. FUNCIONES DE LA API

La librería dinámica de Epanet (v2.00.12), también llamada API (Application Programming Interface en inglés), está escrita en lenguaje ANSI C y posee módulos de código separados para el procesamiento de los datos de entrada, el análisis hidráulico, el análisis de calidad del agua, la resolución del sistema de ecuaciones lineales con matrices dispersas y la generación de informes. Consta de 55 funciones y 104 variables que se utilizan con las funciones como parámetros de entrada. En la Tabla 3.2, se describen las funciones organizadas por tareas:

Tabla 3.2. Funciones de la Toolkit de Epanet agrupadas por tareas

Tarea	Funciones	
Ejecutar una simulación completa en una sola línea de comandos	ENepanet	
Abrir y cerrar la librería dinámica de Epanet	ENopen	ENclose
Recuperar información acerca de los nudos de la red	ENgetnodeindex ENgetnodeid	ENgetnodetype ENgetnodevalue
Recuperar información acerca de las líneas de la red	ENgetlinkindex ENgetlinkvalue ENgetlinktype	ENgetlinknodes ENgetlinkid
Recuperar información de las curvas de modulación	ENgetpatternid ENgetpatternindex	ENgetpatternlen ENgetpatternvalue
Obtener otra información de la red	ENgetcontrol ENgetqualtype ENgetoption ENgetversion	ENgetcount ENgetflowunits ENgettimeparam
Establecer nuevos valores para los parámetros de red	ENsetcontrol ENsetnodevalue ENsetlinkvalue ENaddpattern ENsetpattern	ENserpatternvalue ENsetqualtype ENsettimeparam ENsetoption
Guardar y utilizar ficheros de resultados de análisis hidráulico	ENsavehydfile	ENusehyfile
Ejecutar un análisis hidráulico	ENSolveH ENopenH ENinitH	ENrunH ENnextH ENcloseH

Tabla 3.2 Funciones de la Toolkit de Epanet agrupadas por tareas (Continuación)

Tarea	Funciones	
Ejecutar un análisis de calidad del agua	ENSolveQ ENOpenQ ENinitQ ENrunQ	ENnextQ ENstepQ ENcloseQ
Generar un informe de salida	ENsaveH ENsaveinpfle ENreport ENresetreport	ENsetreport ENsetstatusreport ENgeterror ENwriteline

La forma de trabajar con las funciones es la siguiente:

1. Utilizar la función **ENOpen** para abrir la librería de Epanet, y poder utilizar el resto de las funciones comprendidas en dicho fichero (Epanet.dll), excepto la función ENepanet.
2. Utilizar las funciones **ENsetxxx** para cambiar las propiedades de los elementos o características de la red a analizar o las funciones **ENgetxxx** para recuperar datos de la red almacenada en el fichero de entrada (fichero con extensión INP).
3. Para una simulación completa utilizar la función **ENSolveH** (esta función guarda automáticamente los resultados en un fichero de resultados hidráulicos), o si se desea realizar una simulación paso a paso emplear la siguiente secuencia de funciones: **ENOpenH**, **ENinitH**, **ENrunH**, **ENnextH**, **ENcloseH**, y acceder a los resultados a través de la serie de funciones ENgetxxx.
4. La función **ENSolveQ** ejecuta una simulación completa del modelo de calidad (esta función guarda automáticamente los resultados hidráulicos y de calidad en un fichero de salida), o si se desea ejecutar paso a paso utilizar la siguiente secuencia de funciones: **ENOpenQ**, **ENinitQ**, **ENrunQ**, **ENnextQ**, **ENcloseQ**, y acceder a los resultados a través de la serie de funciones ENgetxxx.
5. Si se desea ejecutar nuevos análisis volver al paso 2, o bien utilizar la función **ENreport** para guardar un informe de resultados formateado en el fichero informe de resultados.

6. Llamar a la función **ENclose** para cerrar el fichero de Epanet abierto inicialmente y liberar la memoria ocupada del computador.

En resumen, con la API de Epanet podemos realizar las siguientes tareas:

1. Abrir y cerrar el módulo de herramientas
2. Recuperar y fijar los parámetros de la red
3. Ejecutar una simulación hidráulica
4. Ejecutar una simulación de la calidad del agua
5. Recuperar los resultados del cálculo
6. Generar un informe de resultados

Para conocer lo que hace cada una de las funciones, se recomienda abrir el fichero de ayuda que viene también en la caja de herramientas (Toolkit) y hacer clic en el apartado “Toolkit Functions by Name”, donde se desplegarán todas las funciones escritas en la librería de Epanet. En la Figura 3.2 se muestra el fichero de ayuda con las funciones de Epanet.

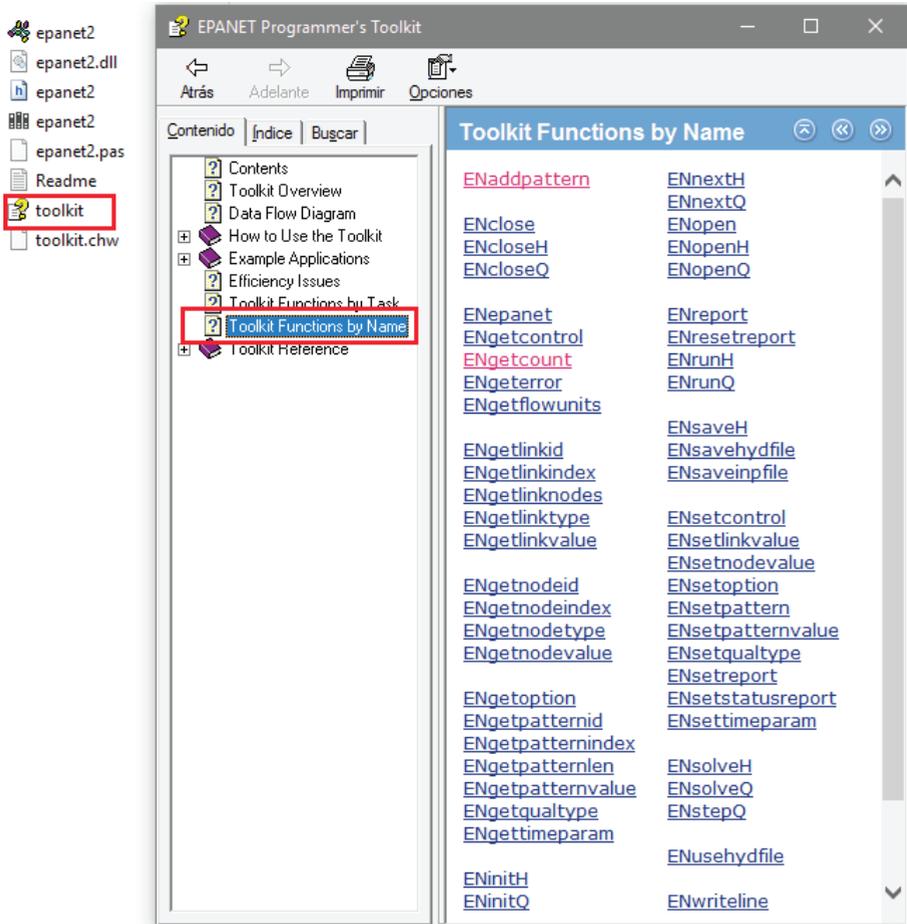


Figura 3.2. Ayuda en línea para programadores con la Toolkit de Epanet



# 4

## CONECTAR LA API DE EPANET A LOS ENTORNOS DE PROGRAMACIÓN

La librería de enlace dinámico (epanet2.dll), en su última versión 2.00.12 y fecha de lanzamiento marzo del 2008, ha sido escrita en el lenguaje de programación C y compilada para ser incorporada en entornos de programación para SO Windows de 32 bits como Borland C/C++ 5.0, Delphi Pascal 5.0, Visual Basic 6.0, o cualquier otro entorno que permita la llamada a funciones incorporadas a una DLL de Windows.

La caja de herramientas viene acompañada de algunos ficheros de cabecera, donde se declaran las variables y funciones que nos permitirán interactuar con la librería de Epanet para obtener o modificar información del modelo de red en estudio.

Si se quiere utilizar la librería dinámica de Epanet (epanet2.dll), sobre los nuevos entornos de programación de 64 bits, habrá que generar un nuevo fichero de cabecera con las modificaciones a los tipos de datos con que se declaran las variables, parámetros de entrada y funciones según el lenguaje de programación a utilizar.

En este capítulo se enseñará a utilizar la librería dinámica de Epanet con las herramientas de programación más utilizadas y una variedad de lenguajes que permitirá al usuario poder elegir, según el grado de conocimiento y comodidad a la hora de programar. Los lenguajes de programación más utilizados para trabajar con la Toolkit de Epanet son Basic 6.0, Basic .NET, Lenguaje de cálculo técnico (Matlab), C#, Python, C++. Los entornos de programación que soportan los lenguajes anteriormente mencionados serán instalados sobre un SO Windows 10 de 64 bits.

En el SO Windows 10 de 64 bits se pueden instalar y ejecutar sin problema alguno aplicaciones de 32 bits. Por cada entorno de programación se describirán todos los pasos que hay que realizar para conectar la API de Epanet, y un ejercicio sencillo para comprobar que podemos acceder a la información almacenada en un fichero INP de Epanet a través de la librería dinámica (epanet2.dll). Se recomienda, antes de empezar a utilizar la librería de Epanet, tener instalada la aplicación Epanet 2.00.12, la cual se puede descargar desde el siguiente enlace: [https://www.epa.gov/sites/production/files/2014-06/en2setup\\_0.exe](https://www.epa.gov/sites/production/files/2014-06/en2setup_0.exe).

El modelo de red sobre el cual se trabajará es la red de abastecimiento Net3.inp que viene conjuntamente con el instalador de la aplicación Epanet. El ejercicio consistirá en recuperar el número de nudos (suma de número de depósitos, embalses, y nudos de caudal), número de líneas (suma de tuberías, válvulas, y bombas), y la unidad de caudal. En el Capítulo 5 se desarrollarán ejemplos más complejos donde se utilizará la mayoría de las funciones que contiene la librería de Epanet.

#### **4.1. VISUAL BASIC 6.0 (BASIC 6.0)**

Es un lenguaje de programación y entorno de desarrollo integrado (IDE). Se deriva del lenguaje BASIC más antiguo, y por lo tanto se considera un lenguaje útil y relativamente fácil de aprender para principiantes. Visual Basic 6.0 es la última edición de Visual Basic. Es un lenguaje de programación dirigido por eventos. Esta última versión liberada en 1998, ya no cuenta con soporte técnico de Microsoft desde marzo del 2008, pero a pesar de ello las aplicaciones que genera (de 32 bits) son compatibles con las plataformas más modernas como Windows Vista, Windows Server 2008, Windows 7, Windows 8 y Windows 10. Visual Basic contiene un entorno de desarrollo integrado que tiene un editor de textos para la edición de código fuente, un depurador, un compilador, y un editor de interfaces gráficas.

Aunque Basic 6.0, como lenguaje nativo, ha dejado de estar soportado por Microsoft, su sintaxis sigue vigente para el lenguaje VBA (Visual Basic for Applications) propietario de algunas aplicaciones de Microsoft como Excel, hasta la última

versión de Office 2016. Los procedimientos indicados en este apartado para instalar EPANET en Visual Basic 6.0 son igualmente válidos para el lenguaje VBA.

Cuando se trate de instalar Visual Basic 6.0, en cualquiera de sus versiones, sobre un SO Windows de 64 bits se mostrará un mensaje de error de instalación imposibilitando continuar con el proceso. Esto se debe a que el instalador setup.exe de Visual Basic 6.0 esta compilado en 16 bits. En los SO Windows de 64 bits corren sin problemas aplicaciones de 32 bits, pero no logran ejecutar el añejo de 16 bits<sup>5</sup>. En Internet podemos encontrar muchas ayudas como blogs y videos tutoriales, que nos dicen cómo podemos seguir utilizando esta aplicación sobre un SO Windows de 64 bits.

Una vez instalada nuestra aplicación en Windows de 32/64 bits tenemos que seguir una serie de pasos previos para poder empezar a utilizar las funciones escritas dentro de la librería de Epanet (epanet2.dll). Estos pasos son los siguientes:

A. Crear una carpeta de proyecto.

Se recomienda crearla en una unidad que no sea la unidad raíz (C:), esto evitaría problemas futuros de lectura y escritura. En dicha carpeta se guardarán los ficheros epanet2.dll y epanet2.bas. Ambos ficheros se encuentran en la carpeta comprimida que se puede descargar desde el siguiente enlace: <https://www.epa.gov/water-research/epanet>. Una ruta posible sería: D:\ConectarAPIEpanet\VB6.0\, donde en la carpeta VB6.0 se guardarán los dos ficheros anteriormente mencionados

B. Añadir el fichero epanet2.bas al nuevo proyecto

El siguiente paso es abrir la aplicación Visual Basic 6.0 y cargar el fichero epanet2.bas. Para ello debemos ir al menú proyecto y seleccionar la opción agregar módulo. Se nos muestra un formulario donde debemos pinchar sobre la

---

<sup>5</sup> <http://www.jorgeoyhenard.com/instalar-visual-basic-6-en-windows-7/2241/>

pestaña “existente” y procedemos a navegar por los directorios hasta encontrar el fichero epanet2.bas guardado en la carpeta VB6.0. Cambiamos el nombre del módulo “Module1” por “Epanet2\_bas” tal como aparece en la Figura 4.1.

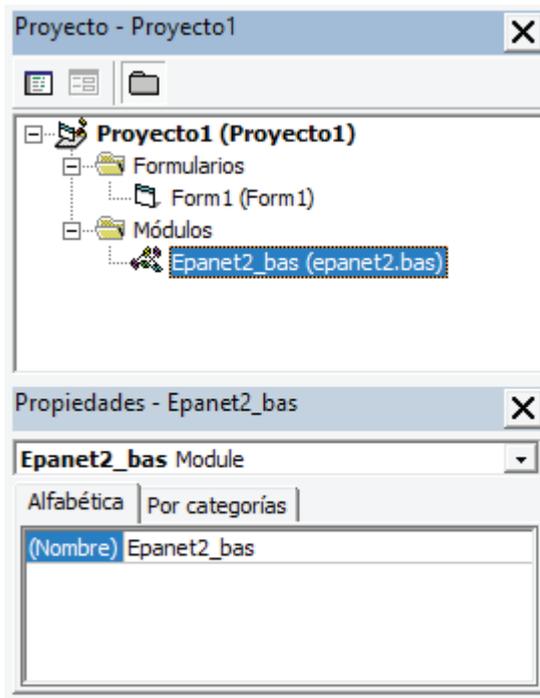


Figura 4.1. Cargar fichero epanet2.bas al proyecto

Asimismo, se cambiará el nombre de nuestro proyecto y formulario. Por defecto, el nombre del proyecto es “Proyecto1” y el del formulario “Form1”. El nuevo nombre del proyecto será “ToolkitEpanet” y el del formulario “FormConectarAPI”. A cada formulario que se cree se le debe dar un nombre y un título (Caption), que en nuestro caso va a ser “ConectarAPI”. Los nuevos cambios pueden verse en la Figura 4.2.

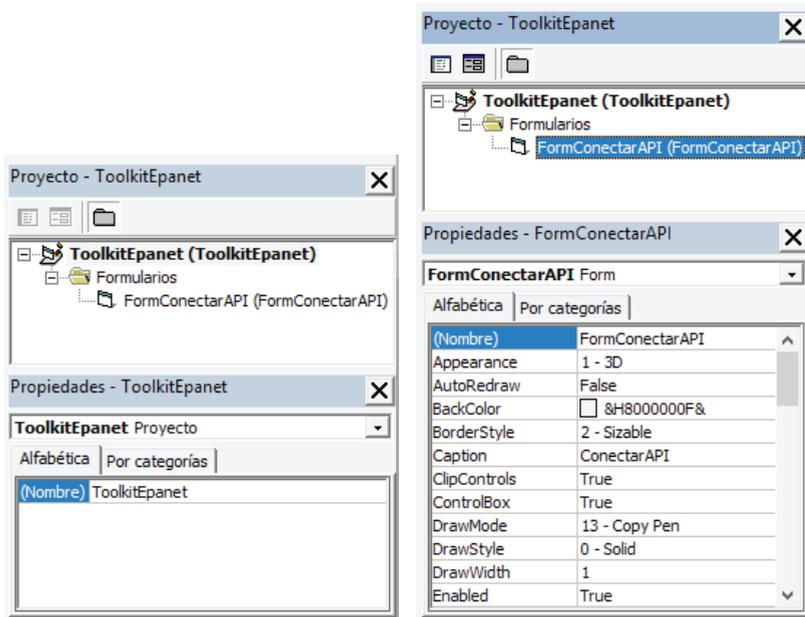


Figura 4.2. Cambios de nombre al proyecto y formulario

Con estos cambios realizados, ya podemos guardar nuestro proyecto y el formulario en la carpeta del proyecto “VB6.0”. En dicha carpeta deberá haber cuatro ficheros como los mostrados en la Figura 4.3.

Nombre	Tipo
epanet2	Visual Basic Module
epanet2.dll	Extensión de la aplicación
FormConectarAPI	Visual Basic Form File
ToolkitEpanet	Visual Basic Project

Figura 4.3. Ficheros guardados en la carpeta de proyectos VB6.0

Ahora sí, ya estamos preparados para empezar a utilizar las funciones contenidas en la librería dinámica de Epanet. Lo siguiente será preparar un formulario donde se pueda visualizar la ruta de nuestro fichero INP de Epanet, así como el número de líneas, nudos y la unidad de caudal.

### C. Preparando el formulario

Nuestro formulario tendrá los siguientes controles: 01 Frame, 04 Label, 02 CommandButton, 04 TextBox, 01 CommonDialog. Este último control no aparece en el cuadro de herramientas cuando se empieza con un proyecto nuevo. Para poder añadirlo al cuadro de herramientas hay que ir al menú Proyecto, luego la opción Componentes y hacemos clic en la pestaña controles, buscamos Microsoft Common Dialog Control 6.0 (SP6) y lo marcamos, y finalmente damos clic al botón Aceptar. Este control no es visible en modo ejecución, tan sólo en modo diseño. El control CommonDialog añadido se puede ver en la Figura 4.4.

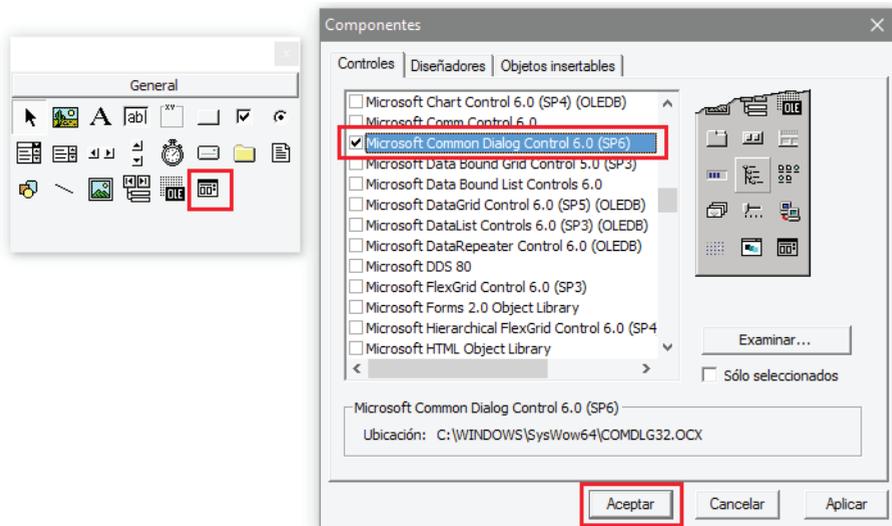


Figura 4.4. Control CommonDialog añadido al cuadro de herramientas de VB6.0

En la Tabla 4.1, se muestran los nombres originales de los controles añadidos al formulario y los nombres cambiados, así como el contenido que tendrán éstos en la propiedad Caption y Text.

Tabla 4.1. Controles añadidos al formulario principal (Visual Basic)

Propiedad: Nombre [Original]	Propiedad: Nombre [Cambiado]	Propiedad:
Frame1	FrameContenedor	Caption: vacío
Label1	LblINP	Caption: Seleccione el fichero Inp de Epanet
Label2	LblNumLineas	Caption: # de Líneas
Label3	LblNumNudos	Caption: # de Nudos
Label4	LblUndCaudal	Caption: Unidad Caudal
TextBox1	TxtINP	Text: vacío
TextBox2	TxtLineas	Text: vacío
TextBox3	TxtNudos	Text: vacío
TextBox4	TxtUndCaudal	Text: vacío
CommandButton1	CmdAbrir	Caption: ...
CommandButton2	CmdAceptar	Caption: Aceptar
CommonDialog	CDialog	----

En la Figura 4.5, podemos ver el diseño final que tendrá nuestro formulario, tanto en modo diseño como en modo ejecución.

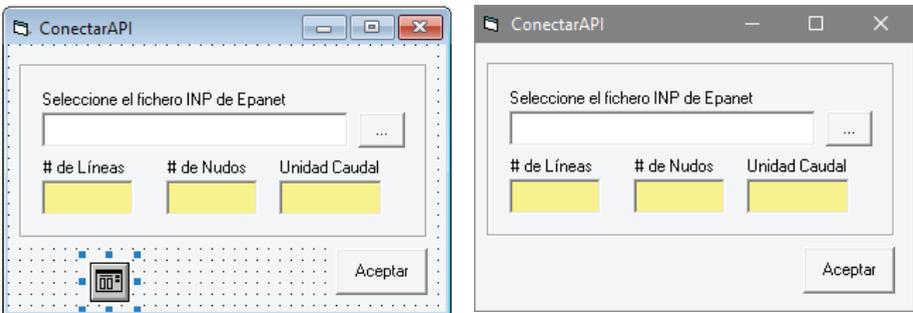


Figura 4.5. Formulario en modo diseño y ejecución de nuestra aplicación

#### D. Trabajando con la Toolkit de Epanet

Para poder ejecutar una secuencia de líneas de código, éstas deberán estar asociadas a algún evento de los objetos insertados en el formulario. En nuestro caso, tendremos que asociar código a los objetos CmdAbrir y CmdAceptar, y para ello tan sólo hay que dar un doble clic a cada objeto y se nos mostrará el editor de texto donde escribiremos nuestras sentencias de código. En la Figura 4.6 se muestra parte de las sentencias de código asociado al botón CmdAceptar. Para descargarse el código completo hacer clic en el siguiente enlace:<https://www.imta.gob.mx/biblioteca/download/?key=962782>.

```
'Especificar la ruta donde se crearán
'los ficheros .rpt y .out
DirInp = TxtINP.Text
i = InStr(1, TxtINP, ".inp")
DirRpt = Left(TxtINP, i - 1) & ".rpt"
DirOut = Left(TxtINP, i - 1) & ".out"

'Abrir la toolkit
error = ENopen(DirInp, "", DirOut)
'Recuperar el número de líneas
error = ENgetcount(EN_LINKCOUNT, numLineas)
'Recuperar el número de nodos
error = ENgetcount(EN_NODECOUNT, numNodos)
'Mostrar valores en las cajas de textos
TxtNodos.Text = CStr(numNodos)
TxtLineas.Text = CStr(numLineas)
'Recuperar la unidad de caudal
error = ENgetflowunits(UndCaudal)
Select Case UndCaudal
    Case EN_CFS: TxtUndCaudal.Text = "CFS"
    Case EN_GPM: TxtUndCaudal.Text = "GPM"
    Case EN_MGD: TxtUndCaudal.Text = "MGD"
    Case EN_IMGD: TxtUndCaudal.Text = "IMGD"
    Case EN_AFD: TxtUndCaudal.Text = "AFD"
    Case EN_LPS: TxtUndCaudal.Text = "LPS"
    Case EN_LPM: TxtUndCaudal.Text = "LPM"
    Case EN_MLD: TxtUndCaudal.Text = "MLD"
    Case EN_CMH: TxtUndCaudal.Text = "CMH"
    Case EN_CMD: TxtUndCaudal.Text = "CMD"
End Select

'Cerrar la toolkit
error = ENclose()

MsgBox "Proceso terminado"
```

Figura 4.6. Código fuente asociado al botón CmdAceptar

En la aplicación se ejecutan dos eventos. En el primero de ellos se tiene que hacer clic en el botón CmdAbrir para seleccionar el fichero Inp de Epanet, Net3.inp, y mostrar la ruta completa en la caja de texto junto al botón. El segundo evento es darle clic al botón CmdAceptar para que nos muestre el número total de nodos (suma de nodos de caudal, embalses, y depósitos) y el número total de líneas (suma de tuberías, válvulas, y bombas), y la unidad de caudal. La Figura 4.7 muestra el resultado de dicha consulta y en la Figura 4.8 se puede ver el resumen del número de elementos desde Epanet.

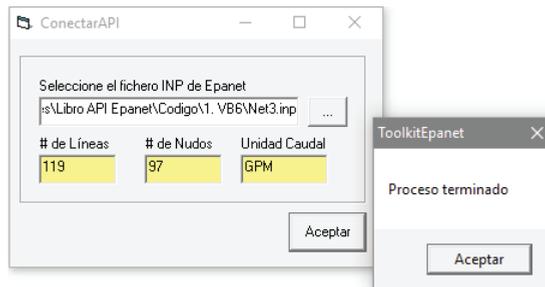


Figura 4.7. Resultado de consultar el número de líneas y nodos de la red Net3

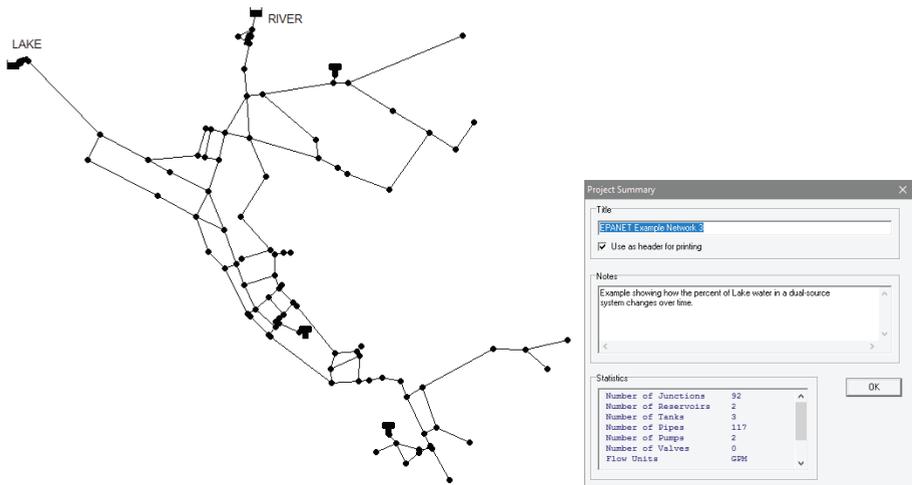


Figura 4.8. Número de nodos y líneas de la Red Net3 desde Epanet

Las funciones que se han utilizado para poder consultar la información requerida a través de la Toolkit de Epanet son: ENopen, ENgetcount, ENgetflowunits, y ENclose. Si queremos consultar la fórmula de pérdida de carga que utiliza la red, no podremos hacerlo con la Toolkit de Epanet porque no hay una función programada en dicha librería. Para obtenerlo, habría que utilizar las funciones propias de Visual Basic y recorrer el fichero INP hasta poder encontrar dicho parámetro dentro de la sección [OPTIONS].

#### 4.2. VISUAL STUDIO 2015 (VISUAL BASIC .NET)

En 2001 Microsoft propuso abandonar el desarrollo basado en la API Win32 y pasar a un framework o marco común de librerías, conocidas como .NET Framework, independientes de la versión del SO, para dar soporte a diversos lenguajes de programación como Basic .NET, C#, etc, facilitando así la transición de código entre ellos; fue el sucesor de Visual Basic 6.

El lenguaje Basic evolucionó para integrar la plataforma .NET; allí perdió su propia identidad como lenguaje único adquirible, pasando a integrar un paquete de productos, llamado precisamente Microsoft .NET; dentro de ese paquete o framework se encuentra el nuevo y llamado Visual Basic .NET, que trabaja sobre el entorno de Microsoft Visual Studio. Esta nueva versión del lenguaje posee profundas diferencias en la forma de programar respecto de Visual Basic 6, pero gran semejanza en su sintaxis básica.

Para este lenguaje de programación se enseñará cómo conectar la API de Epanet (versión 2.00.12 de 32 bits) dentro de Visual Studio 2015 y cómo transformar el fichero epanet2.bas a un nuevo fichero de módulo (Epanet2.vb) entendible por el nuevo entorno de programación (cambios en la declaración de los tipos de datos de las variables, parámetros de entrada y funciones).

Una vez instalado Visual Studio 2015, prepararemos nuestro entorno de trabajo para poder utilizar las funciones del fichero epanet2.dll. Se trabajará con el mismo modelo

de red (Net3.inp) y se recuperará el número total de nodos, líneas y la unidad de caudal utilizada. La secuencia de pasos es la siguiente:

A. Crear un proyecto nuevo

Abrimos Visual Studio 2015 y hacemos un clic en Nuevo proyecto..., luego nos aparecerá una ventana donde elegimos el lenguaje de programación a utilizar (Visual Basic) y seleccionamos la opción de crear una aplicación con una interfaz de usuario de Windows (Aplicación de Windows Forms). Seguidamente escribimos un nombre a nuestro proyecto y a la solución (ConectarApiEpanet), lo guardamos en el directorio elegido, y finalizamos dando clic al botón Aceptar. La Figura 4.9 muestra de manera resumida los pasos antes descritos.

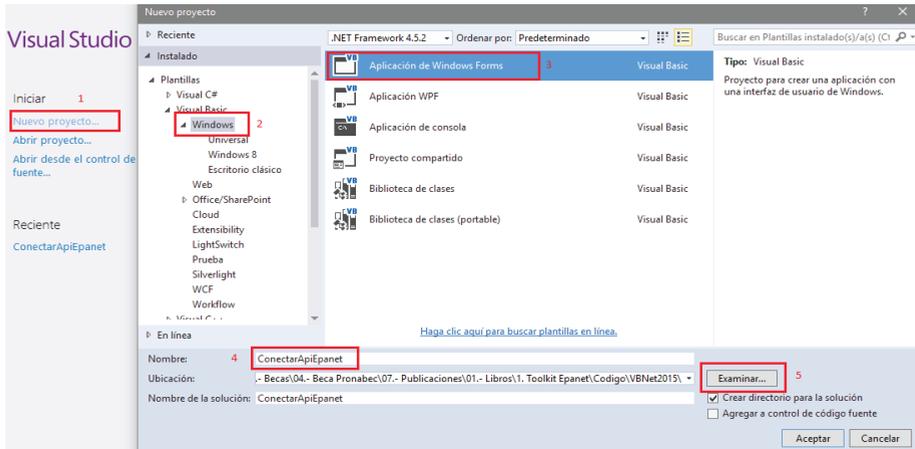


Figura 4.9. Crear proyecto nuevo en Visual Studio 2015

Luego de crear el proyecto se mostrará un formulario vacío donde podemos añadir los objetos (Controles) necesarios para diseñar nuestras propias herramientas. Pero esto lo haremos más adelante, ya que primero prepararemos nuestro nuevo módulo base (Epanet2.vb) donde se declararán las variables globales y funciones que interactuarán con la librería de Epanet (epanet2.dll).

B. Añadir el fichero Epanet2.vb al nuevo proyecto

El contenido del fichero epanet2.bas, que se utiliza en Visual Basic 6.0, debe de copiarse a un nuevo fichero de módulo base de Visual Studio 2015 (Epanet2.vb) y modificar las declaraciones de los tipos de datos de las variables globales, funciones, y parámetros de las funciones. En la Figura 4.10 se muestra una parte de los cambios efectuados sobre el módulo epanet2.bas convertido en Epanet2.vb para su comparación. En el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962783> se puede descargar el módulo Epanet2.vb para ser añadido directamente a nuestro proyecto, tal como se muestra en la Figura 4.11.

```

Global Const EN_ELEVATION = 0
Global Const EN_BASEDEMAND = 1
Global Const EN_PATTERN = 2

Global Const EN_NODECOUNT = 0
Global Const EN_TANKCOUNT = 1
Global Const EN_LINKCOUNT = 2

Declare Function Epanet Lib "epanet2.dll" ( _
ByVal F1 As String, ByVal F2 As String, _
ByVal F3 As String, ByVal F4 As Any) As Long

Declare Function ENopen Lib "epanet2.dll" ( _
ByVal F1 As String, ByVal F2 As String, _
ByVal F3 As String) As Long

Declare Function ENclose Lib "epanet2.dll" () As Long
Declare Function ENrun Lib "epanet2.dll" (T As Long) As Long

Declare Function ENgetcount Lib "epanet2.dll" ( _
ByVal Code As Long, Value As Long) As Long

Declare Function ENgetoption Lib "epanet2.dll" ( _
ByVal Code As Long, Value As Single) As Long

Declare Function ENgetnodevalue Lib "epanet2.dll" ( _
ByVal Index As Long, ByVal Code As Long, Value As Single) As Integer

Declare Function ENsetpattern Lib "epanet2.dll" ( _
ByVal Index As Long, ByRef F As Single, ByVal N As Integer) As Integer

Declare Function ENclose Lib "epanet2.dll" () As Integer
Declare Function ENrun Lib "epanet2.dll" (ByRef T As Integer) As Integer
Declare Function ENgetcount Lib "epanet2.dll" (ByVal Code As Integer, ByRef Value As Integer) As Integer
Declare Function ENgetoption Lib "epanet2.dll" (ByVal Code As Integer, ByRef Value As Integer) As Integer
Declare Function ENsetstatusreport Lib "epanet2.dll" (ByVal Code As Integer) As Integer

End Module
    
```

Figura 4.10. Declaración de los tipos de datos de variables y funciones en VB6.0 y VB.NET

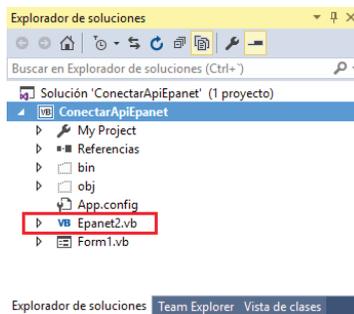


Figura 4.11. Módulo Epanet2.vb añadido al proyecto

C. Copiado del fichero epanet2.dll en la carpeta Debug

Después de añadir el módulo base, necesitamos copiar la librería de Epanet (epanet2.dll) dentro de la carpeta “Debug” que se encuentra alojada en la carpeta “bin”. Para ello tenemos que hacer clic en el botón Mostrar todos los archivos desde el explorador de soluciones para visualizar las carpetas “bin” y “obj” tal como se muestra en la Figura 4.12.

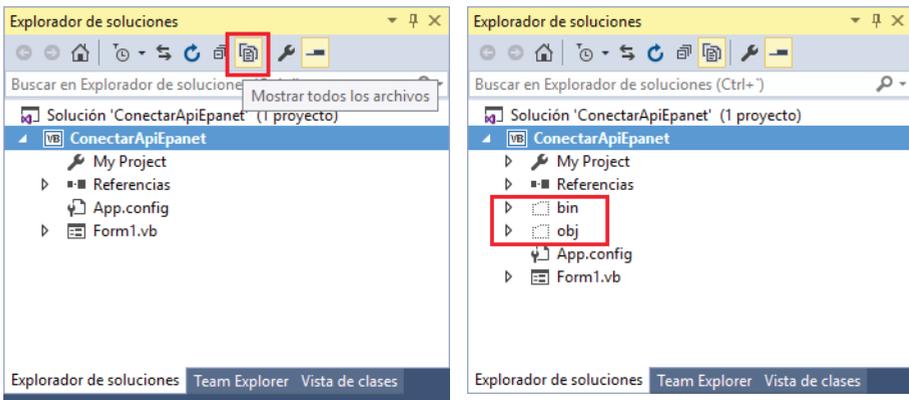


Figura 4.12. Visualización de las carpetas bin y obj desde el explorador de soluciones VB.NET

Al expandir la carpeta “Debug” se puede comprobar, tal como se muestra en la Figura 4.13, que la librería ha sido copiada correctamente. Finalmente procedemos a guardar todos los cambios realizados.

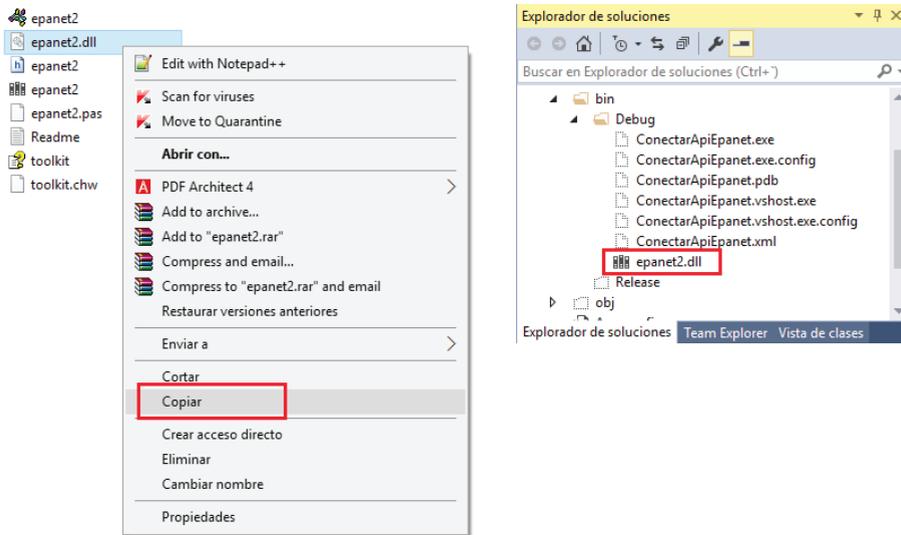


Figura 4.13. Copiado del fichero epanet2.dll en la carpeta Debug

Después de estos pasos previos, ya podemos añadir los controles a nuestro formulario. Se insertarán los mismos que añadieron para Visual Basic 6.0.

#### D. Preparando el formulario

Los controles que serán añadidos al formulario son los siguientes: 01 GroupBox, 04 Label, 02 Button, 04 Textbox, 01 OpenFileDialog. Este último control sólo es visible en modo diseño, más no en modo ejecución. Todos estos controles se encuentran en el cuadro de herramientas como se muestra en la Figura 4.14. Si no estuviera visible el cuadro de herramientas podemos hacerlo aparecer ya sea con la combinación de las teclas Ctrl+Alt+X o haciendo clic en el menú Ver y en el desplegable otro clic sobre el comando Cuadro de herramientas.

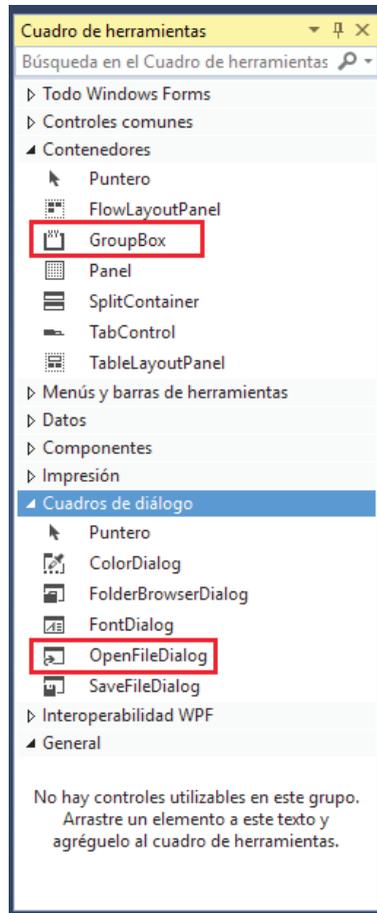


Figura 4.14. Cuadro de herramientas dentro de Visual Basic 2015

En la Tabla 4.2, se muestran los nombres originales de los controles añadidos al formulario y los nombres cambiados, así como el contenido que tendrán éstos en la propiedad Caption y Text.

Tabla 4.2. Controles añadidos al formulario principal (Visual .NET)

Propiedad: (Name) [Original]	Propiedad: (Name) [Cambiado]	Propiedad:
GroupBox1	GBContenedor	Text: vacío
Label1	LblINP	Text: Seleccione el fichero Inp de Epanet
Label2	LblNumLineas	Text: # de Líneas
Label3	LblNumNudos	Text: # de Nudos
Label4	LblUndCaudal	Text: Unidad Caudal
TextBox1	TxtINP	Text: vacío
TextBox2	TxtLineas	Text: vacío
TextBox3	TxtNudos	Text: vacío
TextBox4	TxtUndCaudal	Text: vacío
Button1	BtnAbrir	Text: ...
Button2	BtnAceptar	Text: Aceptar
OpenFileDialog	OpenFD	FileName: vacío

#### E. Trabajando con la Toolkit de Epanet

Tal como se ha hecho con la aplicación Visual Basic 6.0, hay que asociar un conjunto de sentencias a unos controles. En este caso vienen a ser los mismos botones, BtnAbrir y BtnAceptar, a los que se asocian las líneas de código a ejecutarse. En la Figura 4.15 podemos ver parte del código fuente utilizado para cumplir con nuestro objetivo y el formulario final en modo ejecución. Para ver todo el código fuente puede descargarse desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962784>.

Las funciones utilizadas son: ENopen, ENgetcount, ENgetflowunits, y ENclose. Si queremos consultar la fórmula de pérdida de carga que utiliza la red, no podremos hacerlo con la Toolkit de Epanet porque no hay una función programada en dicha librería. Para obtenerlo, habría que utilizar las funciones propias de Visual Basic.Net y recorrer el fichero INP hasta poder encontrar dicho parámetro dentro de la sección [OPTIONS].

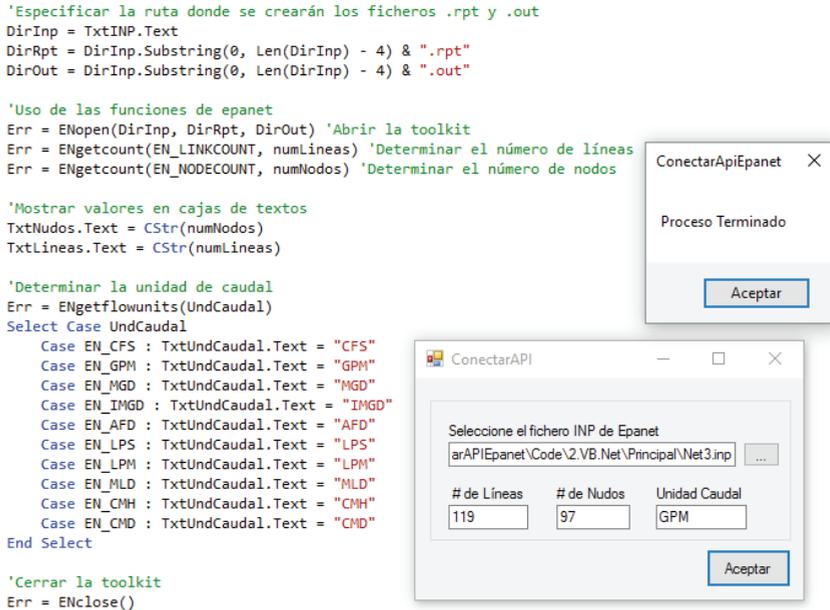


Figura 4.15. Parte del código fuente y el formulario principal en modo ejecución

### 4.3. MATLAB (LENGUAJE DE CÁLCULO TÉCNICO)

La primera versión de MATLAB data de los años 70, y fue diseñada como herramienta de apoyo para los cursos de *Teoría de Matrices*, *Álgebra Lineal* y *Análisis Numérico* por el matemático Cleve Moler. El nombre es un acrónimo: “MATrix LABoratory” (Laboratorio de Matrices). Es un software matemático que ofrece un entorno de desarrollo integrado con un lenguaje de programación propio (lenguaje m) y permite realizar operaciones con vectores y matrices, funciones, cálculo lambda, y programación orientada a objetos, así como la comunicación con programas escritos en otros lenguajes y con otros dispositivos hardware. Además, cuenta con “Simulink” y “Guide”, dos herramientas adicionales que aumentan aún más el número de prestaciones de MATLAB<sup>6</sup>. Originalmente fue

<sup>6</sup> <https://es.mathworks.com/products/matlab.html>

escrito en Fortran durante varios años, pero actualmente está escrito en C por The Mathworks.

MATLAB es un sistema de programación y cálculo basado en la manipulación de matrices, ya que de esta forma se puede usar el álgebra matricial y otras propiedades para ahorrar tiempo de cómputo<sup>7</sup>. Integra el análisis numérico, cálculo matricial, procesamiento de señales, gráficos, etc., en un entorno fácil de usar, donde los problemas y las soluciones son expresados tal como se escriben matemáticamente, sin necesidad de recurrir a la programación tradicional, de un nivel más bajo.

MATLAB es muy rápido para ciertas operaciones, cuando puede ejecutar sus funciones en código nativo, con los tamaños adecuados para aprovechar las capacidades de vectorización. En otras aplicaciones resulta bastante más lento que el código equivalente desarrollado en C/C++ o Visual .NET. Sin embargo, es una herramienta muy útil y de alto nivel para desarrollar aplicaciones técnicas, fácil de utilizar y puede ayudar a aumentar significativamente la productividad de los programadores respecto a otros entornos de desarrollo.

La programación en MATLAB se lleva a cabo mediante un lenguaje que es muy parecido a los lenguajes de alto nivel como BASIC o C. Esto permite que el usuario pueda agrupar sentencias que utiliza frecuentemente dentro del programa, el cual puede ser invocado posteriormente. De este modo se ahorra tiempo y esfuerzo en sucesivas sesiones, pues no es necesario escribir todas las sentencias de nuevo.

MATLAB dispone de un código básico y de varias librerías especializadas (toolboxes). Además, incluye una gran cantidad de funciones predefinidas, que ayudan a realizar cálculos de tipo, así como a visualizar datos y resultados. Puede ejecutar líneas de código escrito en un fichero ASCII (con extensión \*.m), si dicho fichero se encuentra almacenado en alguno de los subdirectorios indicados en el PATH o bien en el directorio de trabajo actual.

---

<sup>7</sup> <http://www.geociencias.unam.mx/~bole/eboletin/Matlabintro0408.pdf>

El usuario de MATLAB suele ser una persona que necesita algo más que una calculadora pero que no quiere involucrarse con un lenguaje de programación. Por eso, el entorno de trabajo es sencillo de manejar, casi tan fácil como una calculadora<sup>8</sup>.

MATLAB es un programa muy potente con un entorno de desarrollo agradable que incluye herramientas de desarrollo de cálculo científico y técnico y de visualización gráfica, así como un lenguaje de programación de alto nivel. La última versión actualizada de MATLAB es la R2016b y se puede descargar desde la página web oficial <https://www.mathworks.com>, según la versión del Sistema Operativo Windows que se tenga instalado en el ordenador (32 y 64 bits). La versión descargada para el desarrollo de este apartado es la R2016b de 64 bits y se seguirá trabajando con la red Net3.inp.

Marios Kyriakou y Demetrios Eliades han desarrollado una Toolkit de Matlab para conectar EPANET con esta aplicación, la cual es de dominio público y está accesible en un repositorio gestionado por la asociación OpenWaterAnalytics desde el portal de software libre github, la cual utilizaremos en el presente manual.

#### A. Crear una carpeta de proyecto

Como se ha venido explicando en los otros entornos de programación, se recomienda que la carpeta de proyecto se cree en una unidad que no tenga problemas de lectura y escritura de datos. Por ejemplo, si tenemos dos unidades compartidas C: para instalar los programas y D: para guardar nuestra información, un directorio posible sería D:\Epanet\_Matlab\Proyecto1. En la carpeta 'Proyecto1' se almacenará un fichero comprimido que se descargará desde el siguiente enlace, tal como se muestra en la Figura 4.16, en la que se enumeran los pasos que hay que realizar para ejecutar la descarga: <https://github.com/OpenWaterAnalytics/EPANET-Matlab-Toolkit>

---

<sup>8</sup> <http://www.esi2.us.es/~jaar/Datos/FIA/T9.pdf>

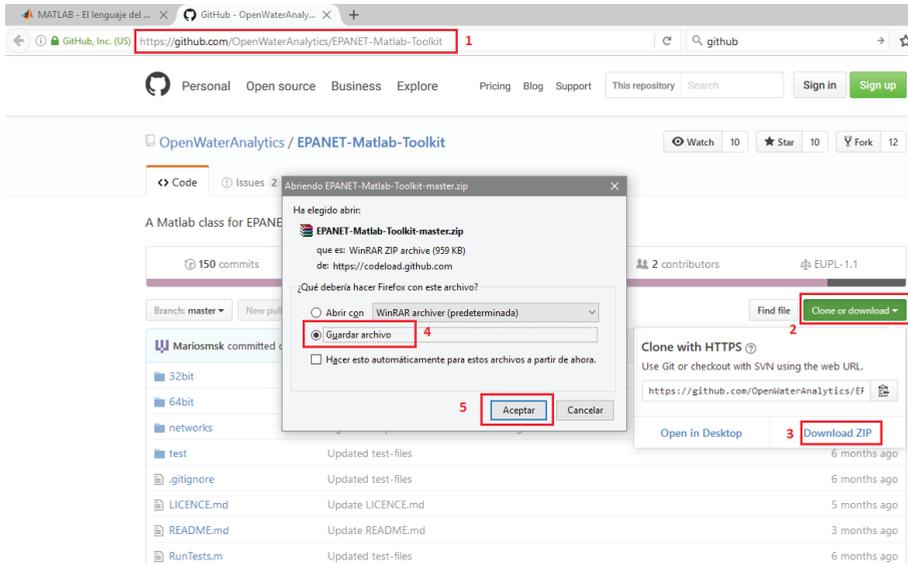


Figura 4.16. Descarga de la Toolkit de Epanet para Matlab

Luego procedemos a descomprimir el fichero y su contenido se copiará en la carpeta 'Proyecto1' tal como se muestra en la Figura 4.17. Hecho esto, ya podemos eliminar el fichero descargado inicialmente.

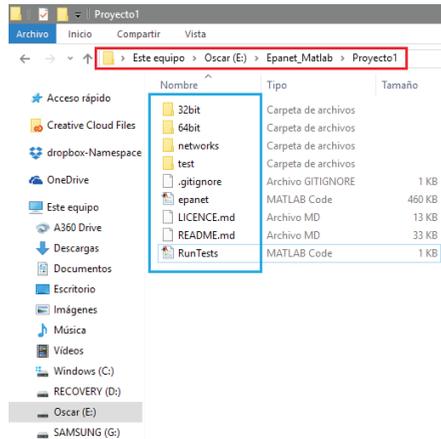


Figura 4.17. Carpetas y ficheros que contendrá nuestro proyecto

A continuación en la Tabla 4.3 se describe cada uno de los ficheros y el contenido que existe en cada carpeta.

Tabla 4.3. Ficheros y carpetas contenidos en el fichero comprimido  
EPANET-Matlab-Toolkit-master.zip

Carpeta/Fichero	Descripción
32bit	Carpeta que contiene los siguientes ficheros para SO Windows de 32 bits: epanet2.dll, epanet2.h, epanet2d.exe, epanetmsx.dll, epanetmsx.exe, epanetmsx.h
64bit	Carpeta que contiene los siguientes ficheros para SO Windows de 64 bits: epanet2.dll, epanet2.h, epanet2d.exe, epanetmsx.dll, epanetmsx.exe, epanetmsx.h
networks	Carpeta que contiene cinco ficheros *.inp de Epanet y dos ficheros con extensión *.msx
test	Carpeta con seis ficheros de MATLAB con extensión *.m, donde se puede realizar un test de las funciones de Epanet para recuperar o modificar los datos de los elementos de red
epanet	Fichero con extensión *.m, donde se crea una clase que sirve como una interfaz entre MATLAB y Epanet
LICENCE	Fichero con extensión *.md, donde se describe todo acerca de la licencia de uso de la herramienta
README	Fichero con extensión *.md, contiene información sobre los requisitos de software, el uso de la caja de herramientas, y un listado de las funciones que se utilizarán desde MATLAB, así como las nuevas funciones que soporta la nueva versión de la librería dinámica de Epanet 2.1
Run Tests	Fichero con extensión *.m, que contiene los nombres de funciones para ser llamadas después de hacer clic en el botón 'Run'

De las carpetas y ficheros contenidos en nuestra carpeta de proyecto ‘Proyecto1’, tan sólo es necesario que estén las carpetas 32bit y 64bit, así como el fichero epanet.m, tal como se ve en la Figura 4.18. El resto de carpetas y ficheros pueden guardarse en otro directorio.

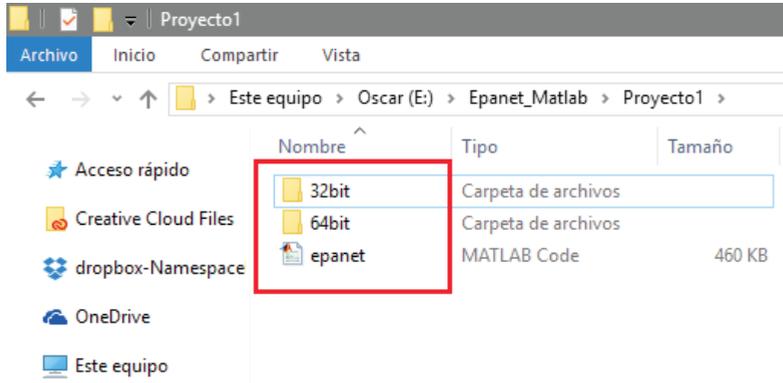


Figura 4.18. Ficheros indispensables para utilizar la Toolkit de Epanet con Matlab

## B. Preparando la carpeta de proyecto actual desde Matlab

Después de preparar nuestra carpeta de proyecto, donde se guardarán nuestros futuros ficheros Matlab, procedemos a abrir la aplicación, y primero tendremos que elegir nuestro directorio de trabajo, es decir, la carpeta ‘Proyecto1’. Para ello seguiremos los pasos mostrados en la Figura 4.19.

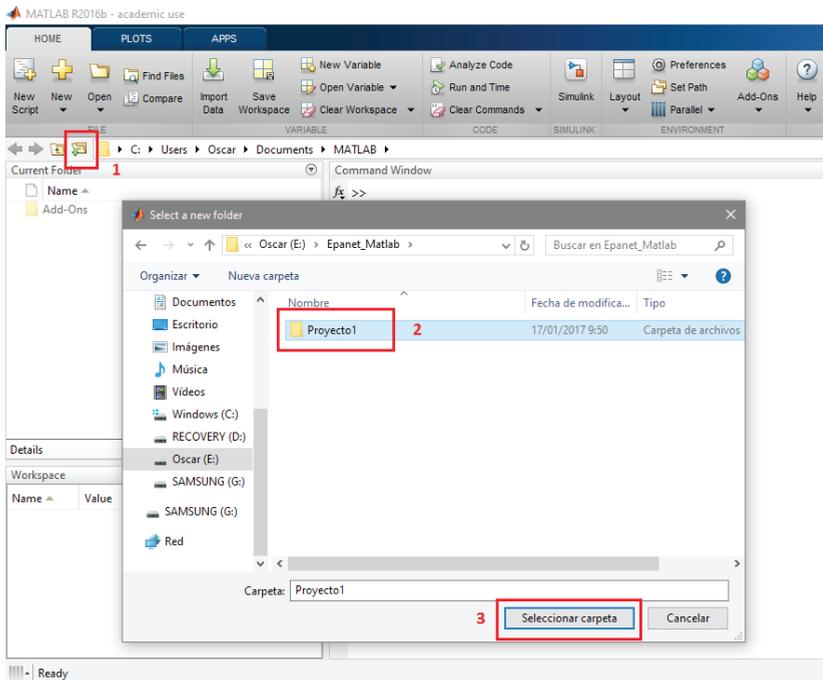


Figura 4.19. Pasos para seleccionar la carpeta de trabajo actual desde Matlab

Primero, haremos clic en el botón ‘browser for folder’ (buscar por carpetas) y se nos mostrará un cuadro de diálogo. Segundo, navegaremos por las unidades de disco hasta encontrar nuestra carpeta de proyecto ‘Proyecto1’ y la seleccionamos. Tercero, hacemos clic en el botón seleccionar carpeta. Luego se mostrará en la ventana ‘Current Folder’ las carpetas 32bit, 64bit, y el fichero epanet.m tal como se ve en la Figura 4.20.

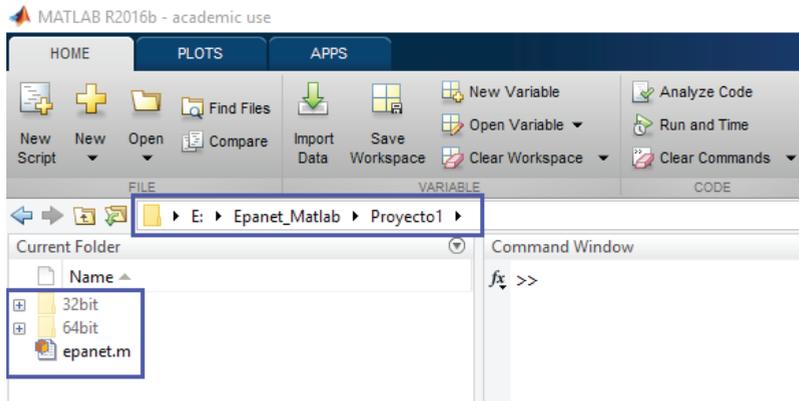


Figura 4.20. Carpeta de proyecto lista para trabajar con la Toolkit de Epanet

En este punto cabe hacer algunos comentarios acerca de la forma de cómo se trabaja con las carpetas 32bit y 64bit, así como del envoltorio creado (epanet2.m) para acceder de manera fácil a las funciones escritas en las librerías de epanet2.dll de 32 y 64 bits.

Estas librerías han sido modificadas y compiladas con las nuevas funciones que se han venido añadiendo gracias al trabajo de un grupo de investigadores que aportan sus conocimientos en mejorar la Toolkit de Epanet. Actualmente, la última versión disponible es la 2.1 con fecha de 18 Octubre 2016.

### 1. Carpetas 32bit y 64bit

Estas carpetas pueden guardarse en cualquier carpeta de proyecto pero sólo una será llamada desde el fichero epanet2.m. Esto es así, porque el código escrito en el fichero epanet2.m analiza primero si nuestro computador es de 32 ó 64 bits. El SO del computador desde donde se están desarrollando los ejercicios es un Windows 10 de 64 bits. En este caso si eliminamos la carpeta de nombre 64bit y ejecutamos el script Test1.m se mostrará un mensaje de error, Figura 4.21, porque no encuentra dicha carpeta. Para evitar cualquier problema futuro, se

recomienda que estas dos carpetas vayan juntas siempre en cualquier proyecto nuevo.

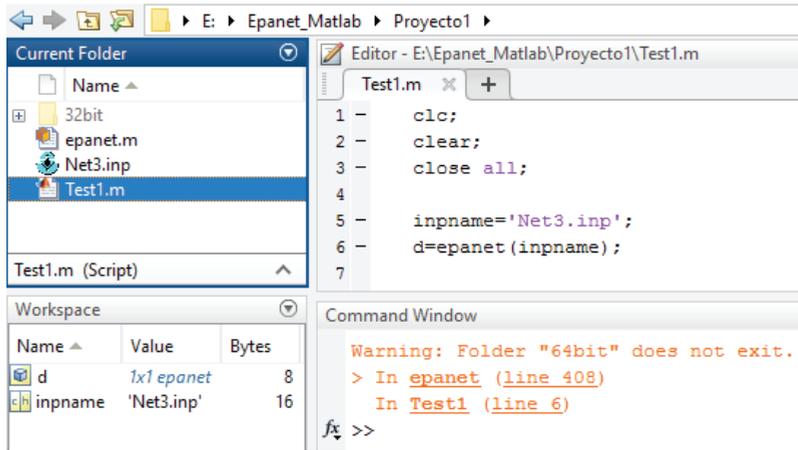


Figura 4.21. Mensaje de error por no encontrar la carpeta 64bit en la carpeta de proyecto

## 2. Fichero epanet2.m

En este fichero vamos a encontrar una clase definida con el nombre Epanet, dentro del cual se definen una serie de propiedades y métodos que permitirán utilizar las funciones escritas en la librería de Epanet.

Todas las funciones desarrolladas hasta la última versión de Epanet (2.00.12), desarrollada por la EPA, aún se mantienen. En la última versión (2.1), desarrollada por un grupo de investigadores, se han añadido nuevas funciones, como obtener las coordenadas de los nudos y vértices de cada línea, así como funciones para obtener los valores de las propiedades de un elemento, por ejemplo el diámetro de las tuberías o su longitud.

C. Creando nuestro primer script

Para crear un script haremos clic en el botón New Script de la barra de menú HOME y se nos creará una nueva pestaña de nombre Untitled en la ventana de editor, tal como se muestra en la Figura 4.22. Asimismo, copiaremos la red Net3.inp en nuestro directorio de proyecto.

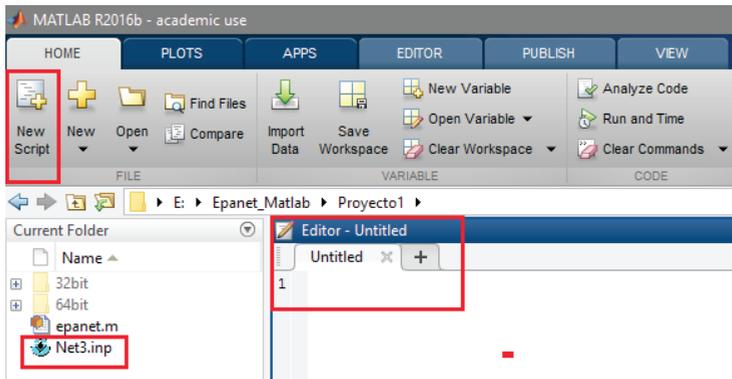


Figura 4.22. Crear un nuevo script desde Matlab

Luego escribimos unas líneas de código como se muestra en la Figura 4.23 y guardamos el fichero con el nombre Test1. Este script permitirá recuperar el número total de nodos y líneas, así como la unidad de caudal utilizado. En dicho código se inicializa la clase Epanet con un parámetro de entrada, el nombre de la red a analizar. Posteriormente, se pasa a utilizar los nuevos métodos (funciones) declarados dentro de la clase Epanet, y se imprimen los resultados en pantalla que se mostrarán en la ventana de comandos. Por último, desconectamos la librería de Epanet con el fichero de entrada, y se libera la memoria utilizada. Para visualizar el resultado en la ventana de comandos, presionamos F5 desde el teclado o hacemos clic en el botón Run desde el menú EDITOR. El fichero Test1.m se puede descargar desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962799>.

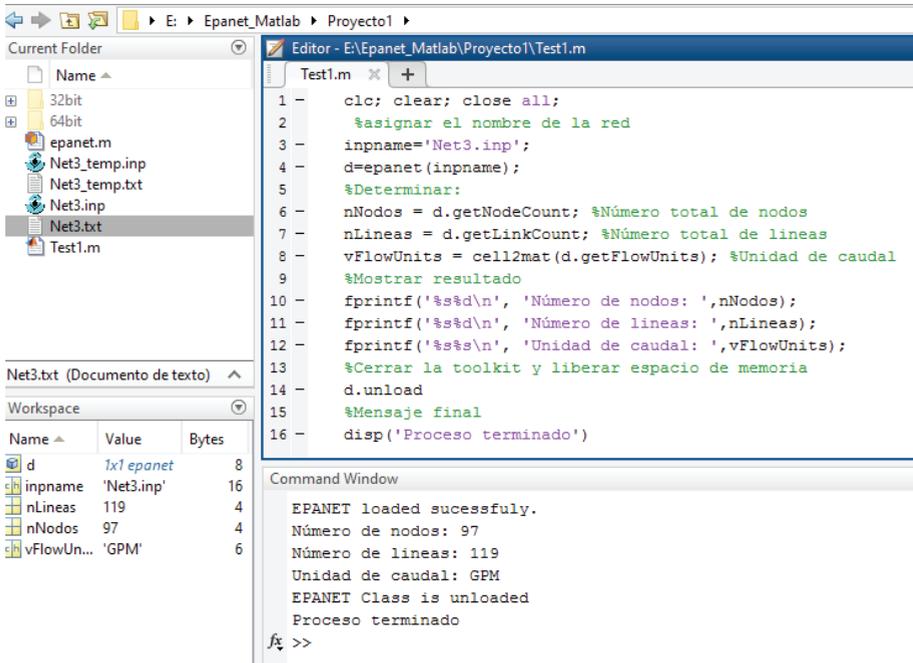


Figura 4.23. Resultados mostrados después de lanzar el script

#### 4.4. VISUAL STUDIO 2015 (C#)

C# (pronunciado en inglés “si sharp”) es un lenguaje de programación orientado a objetos, desarrollado y estandarizado por Microsoft® como parte de su plataforma .NET, que después fue aprobado como un estándar por la European Computer Manufactures Association (ECMA) y la International Organization for Standardization (ISO). Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes. La creación del nombre del lenguaje, C#, proviene de dibujar dos signos positivos encima de los dos signos de “C++”, queriendo dar una imagen de salto evolutivo del mismo que ocurrió con el paso de C a C++<sup>9</sup>.

<sup>9</sup> [https://www.ecured.cu/Lenguaje\\_de\\_Programaci%C3%B3n\\_C\\_Sharp](https://www.ecured.cu/Lenguaje_de_Programaci%C3%B3n_C_Sharp)

C# o C Sharp es un lenguaje moderno, simple y enteramente orientado a objetos. Simplifica y moderniza a C++ en las áreas de clases, namespaces, sobrecarga de métodos y manejo de excepciones. Se eliminó la complejidad de C++ para hacerlo más fácil de utilizar y menos propenso a errores. Se puede utilizar C# para crear aplicaciones cliente de Windows, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos y diseñadores de interfaz de usuario. Incluye un depurador integrado y numerosas herramientas más. El lenguaje C# admite valores NULL, enumeraciones, delegados, expresiones lambda y acceso directo a memoria, que no se encuentran en Java. También admite métodos y tipos genéricos que proporcionan mayor rendimiento y seguridad de tipos.

Como lenguaje orientado a objetos, C# admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluido el método Main, que es el punto de entrada de la aplicación, se encapsulan dentro de definiciones de clase. El proceso de compilación de C# es simple en comparación con C/C++, y es más flexible que en Java. No hay archivos de encabezado independientes, ni se requiere que los métodos y los tipos se declaren en un orden determinado. Un archivo de código fuente de C# puede definir cualquier número de clases, structs, interfaces y eventos. Se ahorra tiempo en la programación, ya que tiene una librería de clases muy completa y bien diseñada<sup>10</sup>.

Microsoft Visual Studio 2015 ofrece compatibilidad con Visual C# con un completo editor de código, un compilador, plantillas de proyecto, diseñadores, asistentes para código, un depurador eficaz y de fácil uso y otras herramientas. La biblioteca de clases de .NET Framework ofrece acceso a numerosos servicios del sistema operativo y a otras clases útiles y adecuadamente diseñadas que aceleran el ciclo de desarrollo de manera significativa<sup>11</sup>.

---

<sup>10</sup> <http://www.larevistainformatica.com/C1.htm>

<sup>11</sup> <https://msdn.microsoft.com/es-es/library/kx37x362.aspx>

Con este lenguaje de programación, se enseñará a como conectar la API de Epanet (versión 2.00.12 de 32 bits) dentro del entorno de Visual Studio 2015 y poder utilizar sus funciones. Se tendrá que crear una clase que contenga todas las propiedades y métodos que permitan llamar a las funciones contenidas en la librería de Epanet, lo que en términos informáticos se le llamaría un envoltorio.

Una vez instalado Visual Studio 2015, incluyendo el lenguaje C#, prepararemos nuestro entorno de trabajo para empezar a utilizar la librería de Epanet. Se trabajará con el mismo modelo de red (Net3.inp) y se recuperará el número total de nodos, líneas y la unidad de caudal utilizada. A continuación, se describen los pasos a seguir.

#### A. Crear un proyecto nuevo

Abrimos Visual Studio 2015 y hacemos clic en Nuevo proyecto..., luego nos aparece una ventana donde elegimos el lenguaje de programación a utilizar (C#) y seleccionamos la opción de crear una aplicación con una interfaz de usuario de Windows (Aplicación de Windows Forms). Seguidamente escribimos un nombre a nuestro proyecto y a la solución (ConectarAPI) y lo guardamos en un directorio que queramos, y finalizamos dando clic al botón Aceptar. La Figura 4.24 resume los pasos antes descritos.

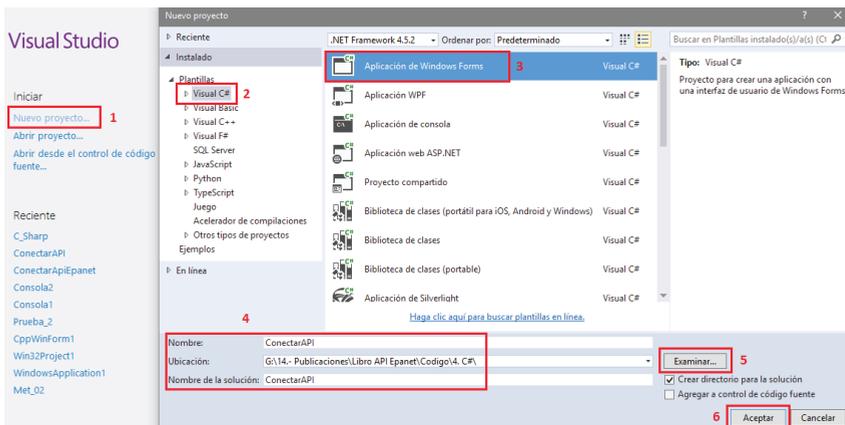


Figura 4.24. Crear nuevo proyecto y solución desde Visual Studio 2015

Luego de crear nuestro proyecto, se mostrará un formulario vacío donde podemos añadir nuestros objetos (Controles) necesarios para diseñar nuestras propias herramientas. Pero esto lo haremos más adelante, ya que primero debemos preparar un archivo de clase (EpanetCSharpLibrary.cs) donde se declararán un grupo de variables globales y funciones que permitirán interactuar con la librería de Epanet (epanet2.dll).

B. Añadir el fichero EpanetCSharpLibrary.cs al nuevo proyecto

Para poder utilizar las funciones escritas en la librería de Epanet, se tendrá que crear una clase que contenga las variables globales y funciones que permitirán interactuar con la librería de Epanet.

Elad Salomons, a través de su página web <http://www.water-simulation.com/wsp/> publicó un artículo, con fecha 21 de abril del 2013, titulado 'Using EPANet Toolkit in C#', cuyo enlace es el siguiente: <http://www.water-simulation.com/wsp/2013/04/21/using-epanet-Toolkit-in-csharp/>, donde se puede ver el contenido de la nueva clase creada (Epanet) dentro del espacio de trabajo EpanetCSharpLibrary, y que permitirá interactuar con las funciones escritas en la librería de Epanet (epanet2.dll). Dicha clase se puede descargar desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962785>, aunque se recomienda revisar periódicamente el enlace del artículo, por posibles cambios futuros. La última actualización del fichero de clase, creado para conectar epanet2.dll desde C#, es del 24 febrero del 2014.

Desde el siguiente enlace <http://www.water-simulation.com/wsp/2014/02/25/epanet-class-for-c-sharp/>, Elad publicó otro artículo relacionado a Epanet y C#, esta vez desarrollado por Vyacheslav Shevelyov.

Para añadir el fichero de clase EpanetCSharpLibrary, primeramente lo descargamos desde el enlace <https://www.imta.gob.mx/biblioteca/download/?key=962785> o desde la web de Elad Salomons, y lo copiamos

dentro de la carpeta ConectarAPI, tal como se ve en la Figura 4.25. Luego desde Visual Studio 2015 cargamos dicho fichero de clase, como se muestra en la Figura 4.26. Se recomienda ir guardando los cambios que se vayan dando en el proyecto.

SAMSUNG (G:) > 14.- Publicaciones > Libro API Epanet >Codigo > 4. C# > ConectarAPI > ConectarAPI

Nombre	Fecha de modificación	Tipo	Tamaño
bin	20/01/2017 11:29	Carpeta de archivos	
obj	20/01/2017 11:29	Carpeta de archivos	
Properties	20/01/2017 11:29	Carpeta de archivos	
App	20/01/2017 11:29	XML Configuration File	1 KB
ConectarAPI	20/01/2017 11:29	Visual C# Project file	4 KB
EpanetCSharpLibrary.cs	19/01/2017 18:00	Visual C# Source file	29 KB
Form1.cs	20/01/2017 11:29	Visual C# Source file	1 KB
Form1.Designer.cs	20/01/2017 11:29	Visual C# Source file	2 KB
Program.cs	20/01/2017 11:29	Visual C# Source file	1 KB

Figura 4.25. Copiado de la clase EpanetCSharpLibrary a la carpeta de proyecto

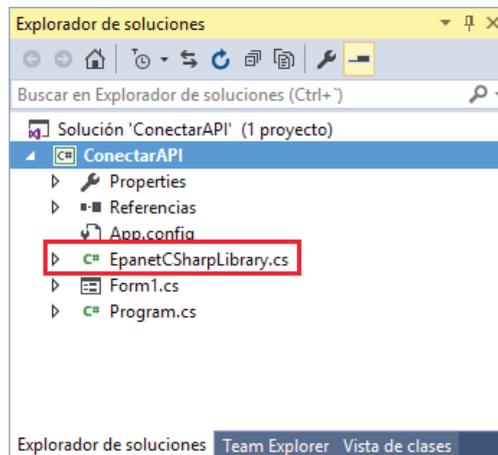


Figura 4.26. Cargar la clase EpanetCSharpLibrary desde Visual Studio 2015

C. Copiado del fichero epanet2.dll en la carpeta Debug

Después de añadir la clase EpanetCSSharpLibrary al proyecto, necesitamos copiar la librería de Epanet (epanet2.dll versión 2.00.12) dentro de la carpeta “Debug” que se encuentra alojada en la carpeta “bin”. Para ello, tenemos que hacer clic en el botón mostrar todos los archivos desde el explorador de soluciones para visualizar las carpetas “bin” y “obj”, tal como se muestra en la Figura 4.27.

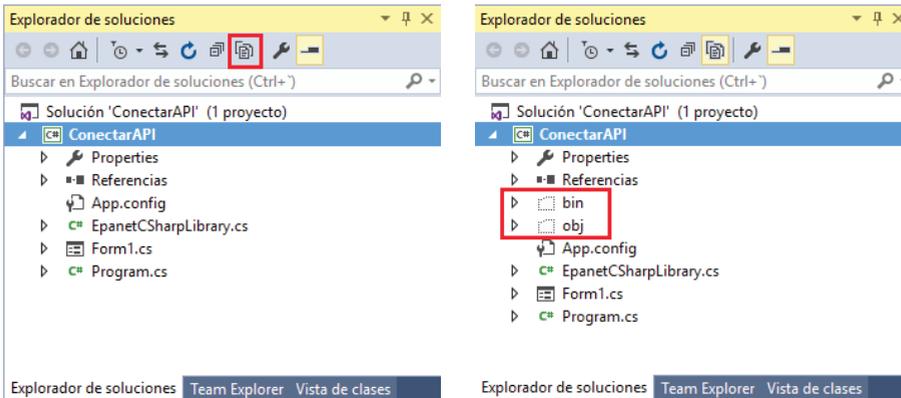


Figura 4.27. Visualización de las carpetas bin y obj desde el explorador de soluciones

Si no contamos con la librería de Epanet (v. 2.00.12), la podemos descargar desde la página oficial de la EPA <https://www.epa.gov/sites/production/files/2014-06/en2Toolkit.zip>.

Al expandir la carpeta “Debug” (Figura 4.28), se puede comprobar que la librería de Epanet ha sido copiada correctamente.

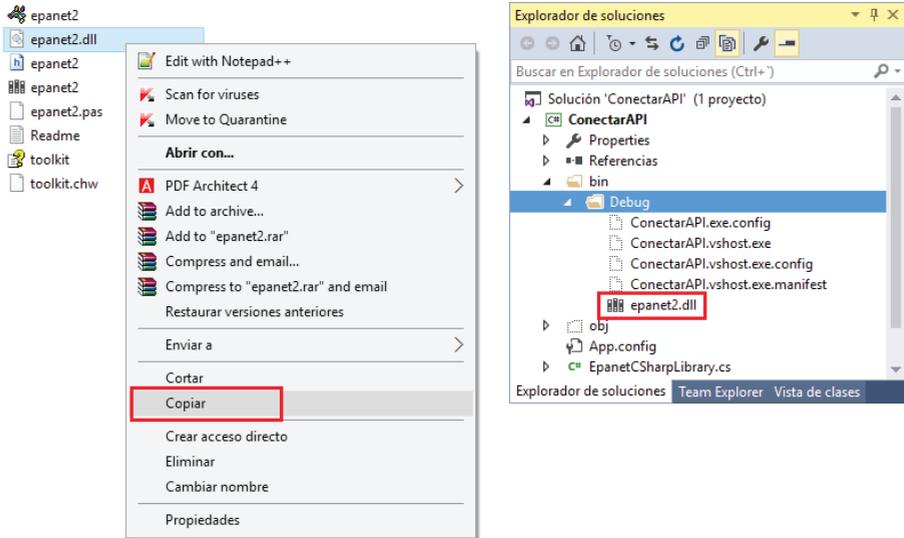


Figura 4.28. Copiado del fichero epanet2.dll en la carpeta Debug

Después de estos pasos previos, ya podemos añadir los controles a nuestro formulario. Se insertarán los mismos que se añadieron tanto en Visual Basic 6.0 como en Visual Basic 2015.

#### D. Preparando nuestro formulario

Los controles que serán añadidos al formulario son los siguientes: 01 GroupBox, 04 Label, 02 Button, 04 Textbox, 01 OpenFileDialog. Este último control sólo es visible en modo diseño más no en modo ejecución. Todos estos controles se encuentran en la ventana del cuadro de herramientas (Figura 4.29). Si no estuviera visible, podemos activarla mediante la combinación de las teclas Ctrl+Alt+X o haciendo clic en el comando Cuadro de herramientas dentro del menú Ver.

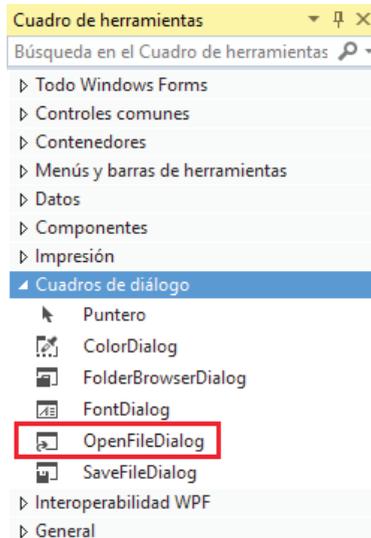


Figura 4.29. Cuadro de herramientas dentro de Visual Studio 2015 (C#)

En la Tabla 4.4, se muestran los nombres originales de los controles añadidos al formulario y los nombres cambiados, así como el contenido que tendrán éstos en la propiedad Caption y Text.

Tabla 4.4. Controles añadidos al formulario principal

Propiedad: (Name) [Original]	Propiedad: (Name) [Cambiado]	Propiedad:
GroupBox1	GBContenedor	Text: vacío
Label1	LblINP	Text: Seleccione el fichero Inp de Epanet
Label2	LblNumLineas	Text: # de Líneas
Label3	LblNumNudos	Text: # de Nudos
Label4	LblUndCaudal	Text: Unidad Caudal
TextBox1	TxtINP	Text: vacío
TextBox2	TxtLineas	Text: vacío
TextBox3	TxtNudos	Text: vacío
TextBox4	TxtUndCaudal	Text: vacío
Button1	BtnAbrir	Text: ...
Button2	BtnAceptar	Text: Aceptar
OpenFileDialog	OpenFileDialog	FileName: vacío

## E. Trabajando con la Toolkit de Epanet

A los botones BtnAbrir y BtnAceptar se les asociará un conjunto de sentencias que permitan elegir el fichero Inp de Epanet y recuperar el número total de nodos, líneas y la unidad de caudal utilizado en nuestro modelo de red. Antes de utilizar los métodos o funciones ENopen, ENgetcount, ENgetflowunits, y ENclose, se tiene que importar el espacio de nombres (namespaces) EpanetCSSharpLibrary, que contendrá la clase 'Epanet' y dentro de ella los métodos antes mencionados.

Si queremos consultar la fórmula de pérdida de carga que utiliza la red, no podremos hacerlo con la Toolkit de Epanet porque no existe una función en dicha librería. Para obtenerlo, habría que utilizar las funciones propias de Visual C# y recorrer el fichero INP hasta poder encontrar dicho parámetro dentro de la sección [OPTIONS]. En la Figura 4.30, se puede ver parte del código fuente utilizado conjuntamente con el formulario en modo ejecución y los resultados obtenidos. Para descargar el código fuente al completo hacer clic en el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962786>.

```

1 referencia
private void BtnAceptar_Click(object sender, EventArgs e)
{
    int Err = 0, UndCaudal = 0;
    int numLineas = 0, numNodos = 0;

    //Especificar la ruta donde se crearán los ficheros .rpt y .out
    DirInp = TxtINP.Text;
    DirRpt = DirInp.Substring(0, DirInp.Length - 4) + ".rpt";
    DirOut = DirInp.Substring(0, DirInp.Length - 4) + ".out";

    //Uso de las funciones de epanet
    Err = Epanet.ENopen(DirInp, DirRpt, DirOut); //Abrir la toolkit
    Err = Epanet.ENgetcount(Epanet.EN_LINKCOUNT, ref numLineas); //Determinar el número de líneas
    Err = Epanet.ENgetcount(Epanet.EN_NODECOUNT, ref numNodos); //Determinar el número de nodos

    //Mostrar valores en cajas de textos
    TxtNodos.Text = numNodos.ToString();
    TxtLineas.Text = numLineas.ToString();

    //Determinar la unidad de caudal
    Err = Epanet.ENgetflowunits(ref UndCaudal);
    switch (UndCaudal)
    {
        case 0: TxtUndCaudal.Text = "CFS"; break;
        case 1: TxtUndCaudal.Text = "GPM"; break;
        case 2: TxtUndCaudal.Text = "MGD"; break;
        case 3: TxtUndCaudal.Text = "IMGD"; break;
        case 4: TxtUndCaudal.Text = "AFD"; break;
        case 5: TxtUndCaudal.Text = "LPS"; break;
        case 6: TxtUndCaudal.Text = "LPM"; break;
        case 7: TxtUndCaudal.Text = "MLD"; break;
    }
}

```

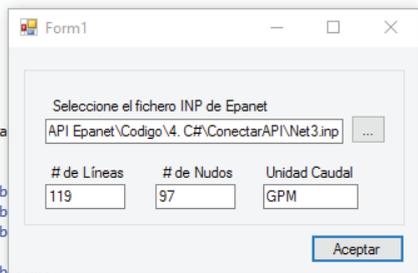
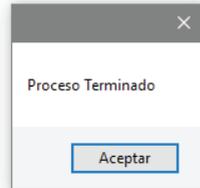


Figura 4.30. Ejecución y resultados desde Visual Studio 2015 (C#)

## 4.5. PYTHON SHELL (PYTHON)

Python es un lenguaje de programación de propósito general y además gratuito, incluso para propósitos empresariales. Apuesta por la simplicidad, versatilidad y rapidez de desarrollo. Es un lenguaje de scripting independiente de la plataforma (Windows, Mac, Linux, etc.) y orientado a objetos, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas web.

Es un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional<sup>12</sup>.

<sup>12</sup> <https://es.wikipedia.org/wiki/Python>

Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad<sup>13</sup>.

Python se ha hecho muy popular gracias a la cantidad de librerías que contiene, y a las funciones incorporadas en el propio lenguaje, que ayudan a realizar muchas tareas habituales sin necesidad de tener que programarlas desde cero. Hay que destacar que Python tiene una sintaxis muy visual, gracias a su notación indentada (con márgenes) de obligado cumplimiento.

El intérprete de Python y la extensa biblioteca estándar están a libre disposición en forma binaria y en código fuente para las principales plataformas desde el sitio web de Python, <http://www.python.org/>, y pueden distribuirse libremente. El mismo sitio contiene también distribuciones y enlaces a muchos módulos libres de Python de terceros, programas y herramientas, y documentación adicional.

Python es Open Source, cualquiera puede contribuir a su desarrollo y divulgación. Además, no es necesario pagar una licencia para distribuir software desarrollado con éste lenguaje. Hasta su intérprete se distribuye de forma gratuita para diferentes plataformas. La última versión de Python recibe varios nombres, entre ellos, Python 3000 o Py3K, aunque habitualmente se le denomina Python 3.

Utilizando este lenguaje de programación, se enseñará cómo importar el módulo de Epanet (una especie de envoltorio) para poder llamar a las funciones contenidas en la API de Epanet (versión 2.00.12) desde el entorno de desarrollo integrado de Python 3.3 (ya que dicho envoltorio sólo funciona con Python 3). Para poder utilizar las funciones de la librería de Epanet hay que instalar previamente la aplicación Epanet 2.00.12 desde la página oficial de la EPA, ya que el envoltorio (epanet2.py) busca a la librería (epanet2.dll) dentro de nuestro ordenador: [https://www.epa.gov/sites/production/files/2014-06/en2setup\\_0.exe](https://www.epa.gov/sites/production/files/2014-06/en2setup_0.exe).

---

<sup>13</sup> <http://www.desarrolloweb.com/articulos/1325.php>

Una vez instalado Python 3.3, abriremos la aplicación Python Shell y se enseñará paso a paso la utilización de las funciones de Epanet. Se trabajará con el mismo modelo de red (Net3.inp) y se recuperará el número total de nodos, líneas y la unidad de caudal utilizada.

### A. Instalando Python 3.3

Si tenemos instalado en nuestro ordenador Python 3.3, ya no es necesario hacerlo. En caso que no esté instalado podemos ir al siguiente enlace <https://www.python.org/downloads/windows/>, buscar en el apartado Python 3.3.0-2012-09-29 y descargar el ejecutable Windows x86-64 MSI installer, tal como se muestra en la Figura 4.31.

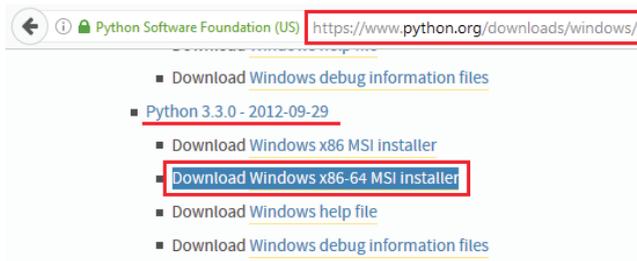


Figura 4.31. Enlace para descargar Python 3.3 desde la página oficial

Después de seguir el proceso de instalación por defecto, ya tendremos Python instalado en nuestro ordenador (Figura 4.32). Dentro de los ficheros instalados hay uno que se llama IDLE (Python GUI). IDLE significa Integrated DeveLopment Environment (entorno de desarrollo integrado para Python). Se trata de una herramienta de programación que permite escribir y editar código Python (Figura 4.33). Es un editor de texto multi ventana, tiene la función de autocompletar y cuenta con un depurador integrado con la posibilidad de ir paso a paso, con puntos de interrupción y visión de la pila de llamadas. Acerca de IDLE, se puede consultar el siguiente enlace:

[http://www.mclibre.org/consultar/python/otros/in\\_idle.html](http://www.mclibre.org/consultar/python/otros/in_idle.html).

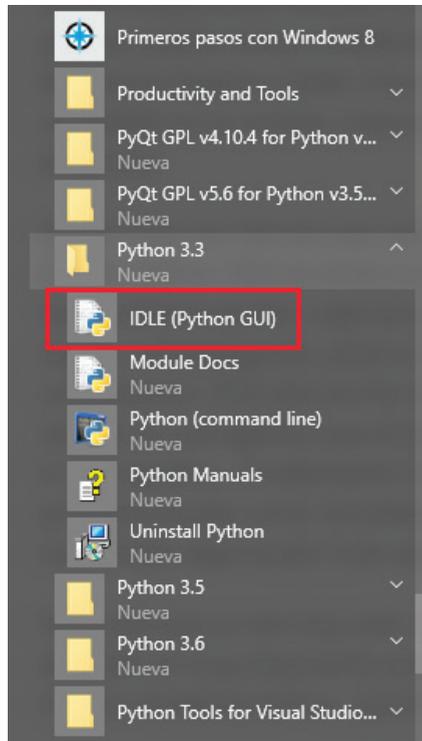


Figura 4.32. Instalación de Python 3.3

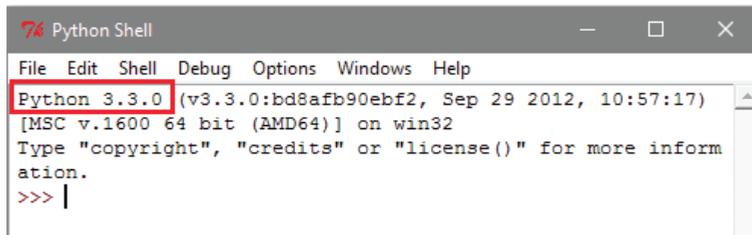


Figura 4.33. Herramienta Python Shell dentro de Python 3.3

B. Descarga e instalación del archivo Epanet2 0.4.0.1dev

Para poder conectar la librería de Epanet y utilizar sus funciones desde la herramienta Python Shell hay que instalar un paquete dentro del directorio de Python, concretamente en la carpeta: “/Python33/Lib/site-packages/”. Para ello ingresaremos al siguiente enlace <https://pypi.python.org/pypi>, que viene a ser un repositorio de paquetes y/o módulos de Python que la comunidad desarrolla y que se pueden descargar sin pago alguno bajo responsabilidad del usuario final.

Si queremos buscar el paquete o módulo, y no recordamos el enlace, podemos escribir en la caja de texto que está al lado derecho de la página la palabra Epanet (Figura 4.34), y al momento obtendremos un listado de lo que posiblemente estemos buscando, tal como se ve en la Figura 4.35.

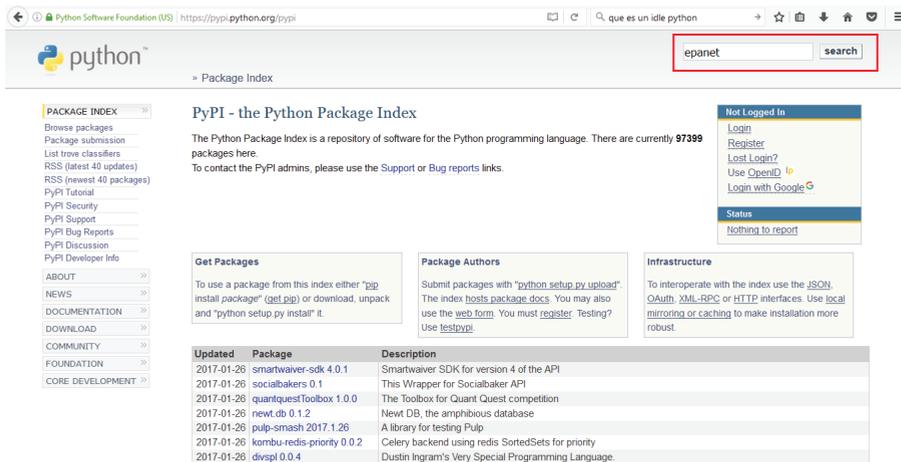


Figura 4.34. Búsqueda del instalador Epanet2 0.4.0.1dev

python™ > Package Index

Index of Packages Matching 'epanet'

Package	Weight*	Description
EPANET 0.4.0.1dev	11	EPANET 2.0 calls from python
<b>EPANET2 0.4.0.1dev</b>	7	EPANET 2.0 calls from python
EPANETTOOLS 0.9.2	7	Epanet 2.0 Python calling interface

\*. occurrence of search term weighted by field (name, summary, keywords, description, author, maintainer)

Figura 4.35. Resultado de la búsqueda del instalador Epanet2 0.4.0.1dev

Al clicar sobre el nombre del instalador “Epanet2 0.4.0.1dev” nos re-direcciona a la página principal del fichero que queremos descargar. Una vez dentro de ella, clicamos en el botón verde que está a la derecha de la página y nos lleva a la sección de ficheros para su descarga. Haremos un clic sobre “Epanet2-0.4.0.1dev.win-amd64-py3.3.exe (md5)”, tal como se muestra en la Figura 4.36.

File	Type	Py Version	Uploaded on	Size
<a href="#">EPANET2-0.4.0.1dev-cp33-none-win_amd64.whl (md5)</a>	Python Wheel	3.3	2014-09-13	122KB
<b><a href="#">EPANET2-0.4.0.1dev.win-amd64-py3.3.exe (md5)</a></b>	MS Windows installer	3.3	2014-09-13	341KB
<a href="#">EPANET2-0.4.0.1dev.zip (md5)</a>	Source		2014-09-13	154KB

Figura 4.36. Descarga del instalador “Epanet2 0.4.0.1dev”

Una vez descargado empezamos el proceso de instalación. Dentro de dicho proceso se comprobará que se tenga instalado previamente Python 3.3 tal como se muestra en la Figura 4.37.

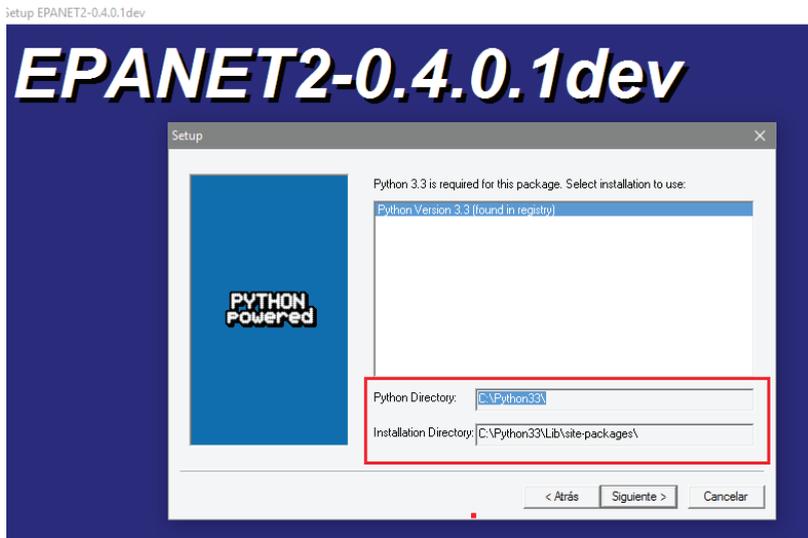


Figura 4.37. Comprobación del directorio Python 3.3 en el proceso de instalación

Ahora la carpeta “site-packages” deberá contener dos carpetas y un fichero Python, como se ve en la Figura 4.38. En dicha carpeta puede haber otros paquetes o módulos de otras instalaciones.

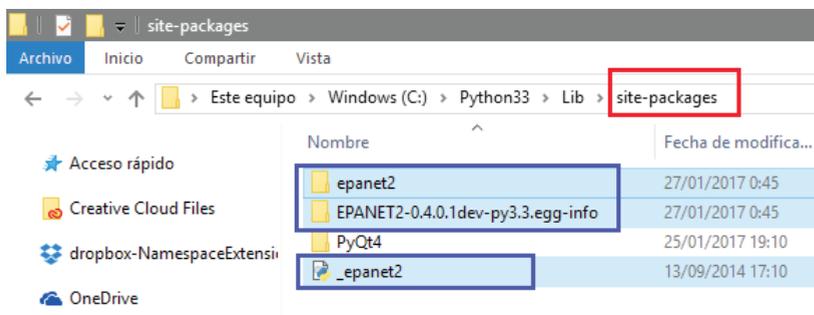


Figura 4.38. Instalación del paquete epanet2, EPANET2-0.4.0.Idev, y \_epanet2

### C. Trabajando con la Toolkit de Epanet

Ahora sí ya estamos casi listos para poder empezar a utilizar las funciones de Epanet, pero antes habrá que explicar algunas cosas. El paquete y el módulo que nos interesa se llaman epanet2, tal como se ve en la Figura 4.39. El módulo epanet2.py es el envoltorio donde se encuentran declaradas las variables globales y funciones que se utilizarán para recuperar o modificar valores de nuestro modelo de red a analizar.

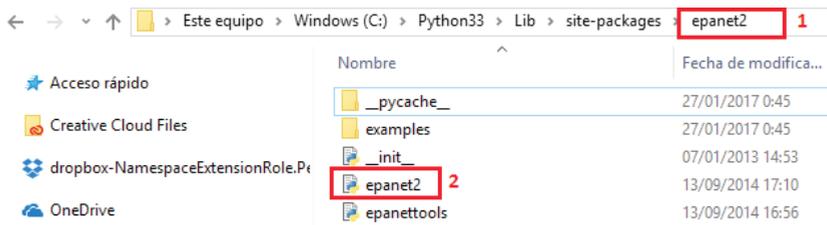


Figura 4.39. Paquete “epanet2” y módulo “epanet2”

Lo siguiente será importar el módulo “epanet2” que se encuentra dentro del paquete “epanet2”. Para ello abrimos la herramienta Python Shell y escribimos la línea de código que se ve en la Figura 4.40. Para acceder a las funciones de la librería de Epanet escritas en el fichero epanet2.py hay que importar el espacio de nombres “namespace” (epanet2.epanet2). Seguidamente, en la siguiente línea de código volvemos a escribir dicho espacio de nombres seguido de un punto y el nombre de la función a utilizar.

Es posible también abreviar los namespaces mediante un alias. Para ello, durante la importación, se asigna la palabra clave “as” seguido del alias con el cual nos referiremos en el futuro a ese namespace importado. En nuestro caso, escribir “epa” equivaldrá a escribir el namespace “epanet2.epanet2”.

Gran parte del código fuente se puede ver en la Figura 4.41, donde se llega a recuperar el número total de nudos y líneas de la red Net3.inp, así

```

Python Shell
File Edit Shell Debug Options Windows Help
Python 3.3.0 (v3.3.0:bd8afb90ebf2, Sep 29 2012, 10:57:17)
[MSC v.1600 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import epanet2.epanet2 as epa
>>>
    
```

Figura 4.40. Importación del módulo epanet2.py

como su unidad de caudal. El fichero ConectarAPIEpanet.py se puede descargar desde el siguiente enlace <https://www.imta.gob.mx/biblioteca/download/?key=962787>. En dicho fichero se guarda todo lo que aparece en el editor de texto de la herramienta Python Shell (código y resultado) cuando se trabaja en modo línea a línea (no copiar el fichero en la ventana de comandos). Se ha supuesto que el fichero Net3 se ubica en E:/; actualizar la ruta en otro caso. Los nombres de rutas no admiten blancos ni caracteres especiales.

```

File Edit Shell Debug Options Windows Help
>>> #Asignar la ruta del fichero inp analizar
>>> path_Inp = r"E:/Net3.inp";
>>> path_Rpt = r"E:/Net3.rpt";
>>> path_Out = r"E:/Net3.out";
>>> #Abrir fichero inp con la función ENopen
>>> err = epa.ENopen(path_Inp, path_Rpt, path_Out);
>>> #Recuperar el número total de nudos y líneas de la red
>>> (err,numNudos) = epa.ENgetcount(epa.EN_NODECOUNT);
>>> (err,numLineas) = epa.ENgetcount(epa.EN_LINKCOUNT);
>>> #Recuperar la unidad de caudal
>>> (err,valor) = epa.ENgetflowunits();
>>> if valor == epa.EN_CFS:
    UndCaudal = "CFS";
elif valor == epa.EN_GPM:
    UndCaudal = "GPM";
elif valor == epa.EN_MGD:
    UndCaudal = "MGD";
elif valor == epa.EN_IMGD:
    UndCaudal = "IMGD";
elif valor == epa.EN_AFD:
    UndCaudal = "AFD";
elif valor == epa.EN_LPS:
    UndCaudal = "LPS";
elif valor == epa.EN_LPM:
    UndCaudal = "LPM";
elif valor == epa.EN_MLD:
    UndCaudal = "MLD";
elif valor == epa.EN_CMH:
    UndCaudal = "CMH";
elif valor == epa.EN_CMD:
    UndCaudal = "CMD";

>>> #Cerrar fichero inp y liberar memoria
>>> err = epa.ENclose();
>>> print("El número total de nudos es: ", str(numNudos), "\n\
El número total de líneas es: ", str(numLineas), "\n\
La unidad de caudal utilizada es: ", UndCaudal)
El número total de nudos es: 97
El número total de líneas es: 119
La unidad de caudal utilizada es: GPM
>>>
    
```

Figura 4.41. Resultado de la consulta al fichero Net3.inp desde Python Shell

Como se puede ver en la Figura 4.41, el resultado nos aparece inmediatamente después de presionar la tecla Enter de la última línea de comando. Si deseamos modificar algún dato de entrada del fichero de red Net3.inp, ya sea eliminando o añadiendo más elementos (nodos/líneas), tendremos que volver a escribir todo de nuevo. Una práctica muy utilizada por los programadores, es escribir el código fuente desde la aplicación NotePad++ y ejecutar el módulo creado desde Python Shell.

Ahora se enseñará a crear código fuente desde NotePad++. Si no tenemos instalado dicha herramienta, la podemos descargar desde el siguiente enlace: <https://notepad-plus-plus.org/>. El código fuente es casi exactamente el mismo que el escrito desde Python Shell. Abrimos la aplicación NotePad++, escribimos el código que aparece en la Figura 4.42 y guardamos el fichero con el nombre ScriptEpanetToolkit.py. Si se prefiere descargar el fichero ya codificado, clicar el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962788>.

```

ScriptEpanetToolkit.py [3]
1  #Importar módulo epanet2 dentro del paquete epanet2
2  import epanet2.epanet2 as epa
3
4  #Asignar la ruta del fichero inp analizar
5  path_Inp = r"E:/Net3.inp";
6  path_Rpt = r"E:/Net3.rpt";
7  path_Out = r"E:/Net3.out";
8
9  #Abrir fichero inp con la función ENopen
10 err = epa.ENopen(path_Inp, path_Rpt, path_Out);
11
12 #Recuperar el número total de nodos y líneas de la red
13 (err,numNodos) = epa.ENgetcount(epa.EN_NODECOUNT);
14 (err,numLineas) = epa.ENgetcount(epa.EN_LINKCOUNT);
15
16 #Recuperar la unidad de caudal
17 (err,valor) = epa.ENgetflowunits();
18 if valor == epa.EN_CFS: UndCaudal = "CFS";
19 if valor == epa.EN_GPM: UndCaudal = "GPM";
20 if valor == epa.EN_MGD: UndCaudal = "MGD";
21 if valor == epa.EN_IMGD: UndCaudal = "IMGD";
22 if valor == epa.EN_AFD: UndCaudal = "AFD";
23 if valor == epa.EN_LFS: UndCaudal = "LFS";
24 if valor == epa.EN_LPM: UndCaudal = "LPM";
25 if valor == epa.EN_MLD: UndCaudal = "MLD";
26 if valor == epa.EN_CMH: UndCaudal = "CMH";
27 if valor == epa.EN_CMD: UndCaudal = "CMD";
28
29 #Cerrar fichero inp y liberar memoria
30 err = epa.ENclose();
31
32 #Visualizar resultados
33 print("El total de nodos es: ", numNodos)
34 print("El total de líneas es: ", numLineas)
35 print("La unidad de caudal es: ", UndCaudal)

```

Figura 4.42. Código fuente (Lenguaje Python) escrito desde el NotePad++

Para ejecutar ficheros Python existen dos maneras de hacerlo: Importando el script o ejecutando el script desde Python Shell.

### 1. Importando el script desde Python Shell

Para poder ejecutar directamente un script desde Python Shell, éste deberá estar almacenado en la dirección siguiente /Python33/Lib/site-packages, tal cual como se ve en la Figura 4.43.

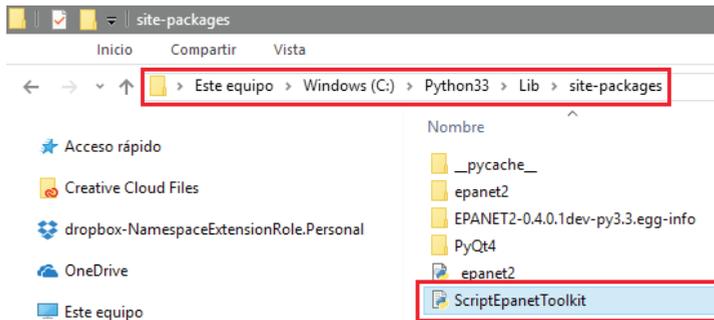


Figura 4.43. Copiar fichero ScriptEpanetToolkit.py en la carpeta site-packages

Luego de escribir “import ScriptEpanetToolkit” en el editor de Python Shell y presionar la tecla Enter se mostrará el resultado (Figura 4.44).

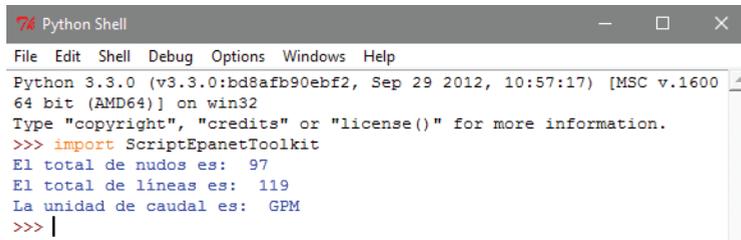


Figura 4.44. Importar fichero ScriptEpanetToolkit y mostrar resultados

## 2. Ejecutando el script desde Python Shell

En este caso no es necesario guardar nuestro script en la carpeta “site-packages”, puede estar en cualquier unidad de disco o extraíble. En primer lugar, abriremos nuestro Python Shell y desde la opción “open...” del menú File buscaremos nuestro script, el cual se mostrará en otra ventana donde ejecutaremos el Script. Para ello haremos clic en el Menú “Run” y luego clic en el comando “Run Module”, tal como se muestra en la Figura 4.45. El resultado aparecerá en la ventana principal de Python Shell, como se muestra en la Figura 4.46.

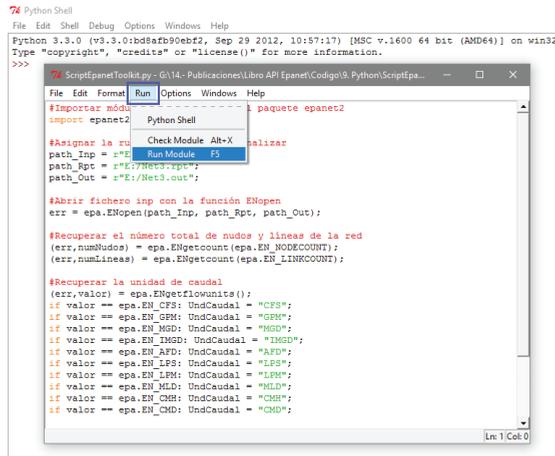


Figura 4.45. Ejecutar el script desde Python Shell

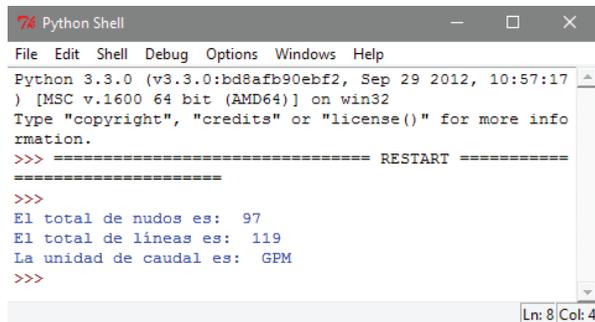


Figura 4.46. Resultado de Ejecutar el script

#### 4.6. DEV-C++ (C++)

C++ es un lenguaje de programación creado por Bjarne Stroustrup en los laboratorios de AT&T en 1983. Su creador tomó como base el lenguaje de programación más popular de aquella época, C. La intención de su creador fue el extender el lenguaje C con mecanismos que permitieran la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, C++ es un lenguaje híbrido<sup>14</sup>.

El nombre de C++ fue propuesto por Rick Mascitti en el año 1983, cuando el lenguaje fue utilizado por primera vez fuera de un laboratorio científico. Antes se había usado el nombre “C con clases”. En C++, la expresión “C++” significa “incremento de C” y se refiere a que C++ es una extensión de C.

Stroustrup vió la necesidad de que la programación en el lenguaje C fuera más fácil. Para ello rediseñó C, ampliando sus posibilidades pero manteniendo su mayor cualidad, la de permitir al programador en todo momento tener controlado lo que está haciendo, consiguiendo así una mayor rapidez, no permitida por otros lenguajes.

C++ llevó al lenguaje C a un nuevo paradigma de clases y objetos con los que se buscó una comprensión más humana basándose en la construcción de objetos, con características propias solo de ellos, agrupados en clases.

En este apartado se enseñará a importar el fichero de cabecera de Epanet (epanet2.h), y poder trabajar con las funciones escritas en la librería dinámica de Epanet (epanet2.dll), desde el entorno de programación de Dev-C++ (software libre). El fichero de cabecera que se utilizará no será el mismo que se distribuye desde la página web oficial de la EPA (Toolkit), ya que dicho fichero está preparado para ser utilizado desde la aplicación Microsoft Visual C 6.0 de 32 bits.

---

<sup>14</sup> <https://es.wikipedia.org/wiki/C%2B%2B>

Tal como hemos venido trabajando con los anteriores lenguajes, se utilizará el modelo de red Net3.inp para recuperar el número total de nodos y líneas, así como la unidad de caudal utilizada.

A. Instalación de la aplicación Dev-C++

Dev-C++ es un entorno de desarrollo integrado (IDE), y a la vez un compilador gratuito, para programar en lenguaje C/C++. El entorno está desarrollado en el lenguaje Delphi de Borland. Tiene una página de paquetes opcionales para instalar, con diferentes bibliotecas de código abierto.

Dentro de las características más destacables tenemos: depurador integrado, editor de programas fuente sensible a la sintaxis C/C++ configurable, editor multiventana con múltiples opciones de edición, se puede trabajar con ficheros independientes o con proyectos multificheros, posee un generador de paquetes de instalación de programas para entorno Windows, puede generar programas DOS (modo consola) - aplicaciones Windows – DLLs, ventanas independientes para el gestor de proyectos-editor-resultados de compilación, resultados de compilación-enlace (linker)-generación de recursos, permite integrar herramientas externas mediante el “tool manager”, etc.

Para descargar la última versión de Dev-C++ (Dev-Cpp 5.11 TDM-GCC 4.9.2 Setup.exe), lo podemos hacer desde el blog oficial <http://orwelldevcpp.blogspot.com.es/>. El proceso de instalación es sencillo, basta con seguir al asistente de ayuda.

B. Crear el directorio de proyecto

Antes de empezar a trabajar desde la aplicación Dev-C++ debemos crear una carpeta de proyecto para poder almacenar los ficheros epanet2.h y epanet2.dll, así como los ficheros que se vayan creando para el proyecto. En la Figura 4.47, se puede ver que se ha creado una carpeta de nombre Proyecto1, en el siguiente directorio.C++\, y dentro de la carpeta de proyecto se guardará el fichero de

cabecera (epanet2.h) y la librería dinámica de Epanet (epanet2.dll). Estos dos ficheros pueden descargarse desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962789>.

Cada usuario puede crear su carpeta de proyecto en cualquier unidad de almacenamiento, siempre y cuando no esté protegido contra lectura y escritura.

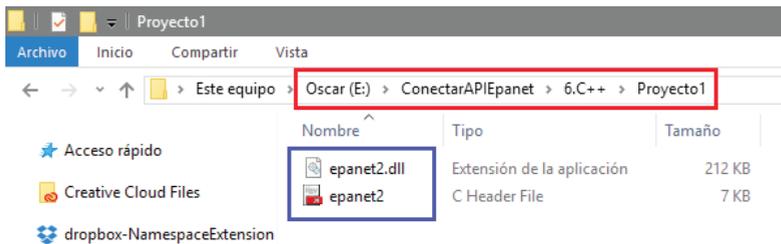


Figura 4.47. Copiado de los ficheros epanet2.h y epanet2.dll dentro de la carpeta Proyecto1

### C. Configurar las opciones de proyecto desde Dev-C++

Lo primero que haremos será abrir nuestra aplicación Dev-C++ 5.11 y crearemos un nuevo proyecto. Para ello seguiremos la siguiente secuencia de pasos: Archivo >> Nuevo >> Proyecto. Se nos abrirá una nueva ventana donde seleccionaremos “Empty Project” desde la pestaña Basic, y escribiremos un nombre para nuestro proyecto. Mantendremos el nombre sugerido “Proyecto1” y damos clic al botón Aceptar (Figura 4.48). Luego guardamos nuestro fichero dentro de la carpeta “Proyecto1”. Después de crear nuestro proyecto, hacemos clic en el botón Guardar y cambiamos el nombre de nuestro primer fichero, con extensión “.cpp”, por ConectarAPIEpanet (Figura 4.49). El aspecto final que tendrá nuestro proyecto, desde la aplicación Dev-C++, y los ficheros creados en la carpeta Proyecto1 se puede ver en la Figura 4.50.

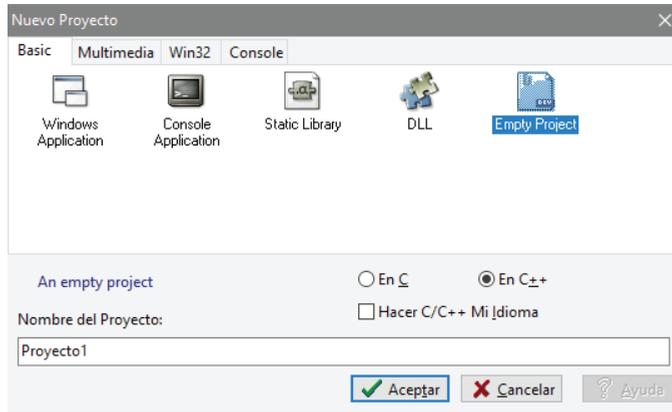


Figura 4.48. Crear un nuevo proyecto desde Dev-C++

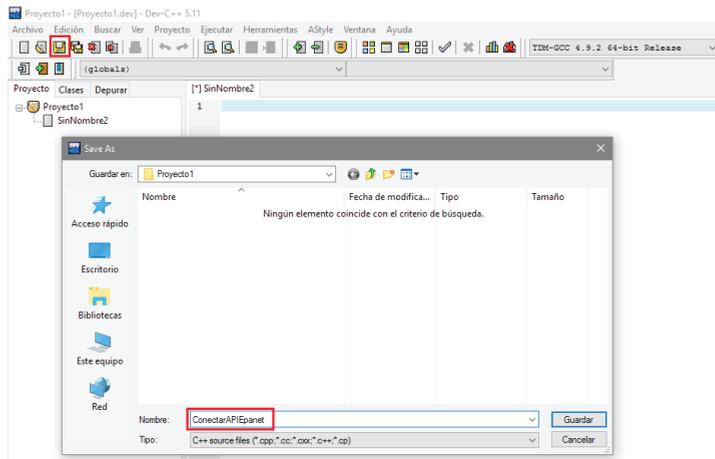


Figura 4.49. Guardar fichero ConectarAPIEpanet.cpp dentro de la carpeta Proyecto1

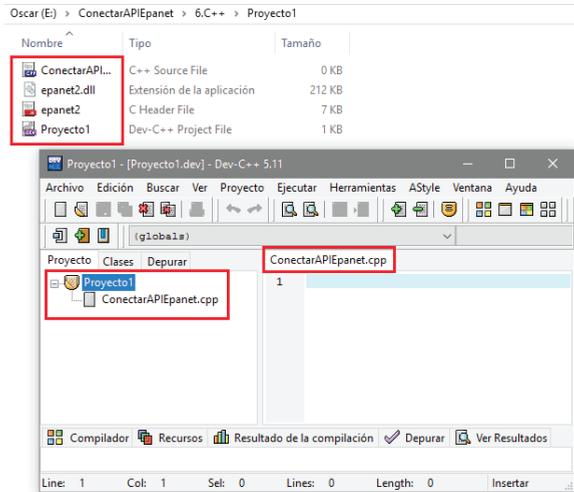


Figura 4.50. Ficheros generados en la carpeta Proyecto1

El siguiente paso será configurar algunas opciones de nuestro proyecto para poder trabajar con nuestra librería de Epanet. Para ello iremos al menú Proyecto >> Opciones de Proyecto y se nos mostrará una ventana de diálogo como se ve en la Figura 4.51. Esta ventana dispone de ocho pestañas donde iremos modificando los valores por defecto en algunas de ellas.

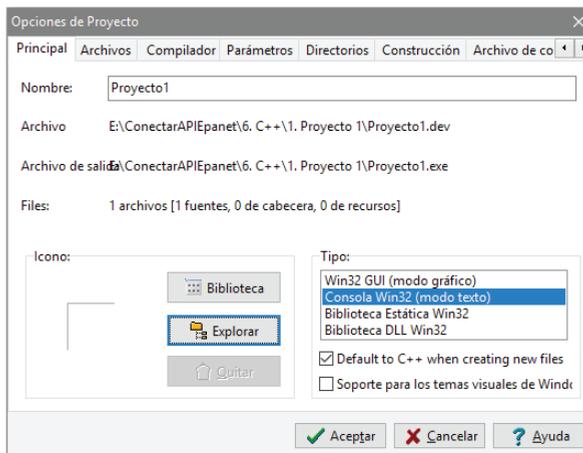


Figura 4.51. Ventana "Opciones de Proyecto" desde Dev-C++

En las pestañas “Principal” y “Archivos” no se realizará ninguna modificación a los parámetros por defecto. En la pestaña “Compilador” seleccionaremos la opción TDM-GCC 4.9.2 32-bit Release y confirmamos con el botón “Yes” el cambio realizado (Figura 4.52). Dentro de la pestaña “Compilador” existen otras seis pestañas más, donde sólo modificaremos el parámetro Generar Información de Debug de No a Yes desde la pestaña “Linker”.

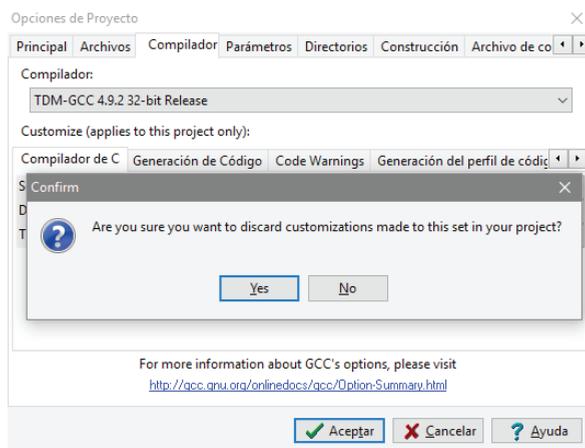


Figura 4.52. Selección del compilador a utilizar

Para enlazar las rutas de los ficheros epanet2.h y epanet2.dll, debemos ir a la pestaña “Parámetros”, situar el puntero del ratón en el apartado Linker y buscar ambos archivos en la carpeta del proyecto, haciendo clic en el botón Añadir Biblioteca o Archivo y añadirlos al cuadro de lista, tal como se ve en la Figura 4.53.

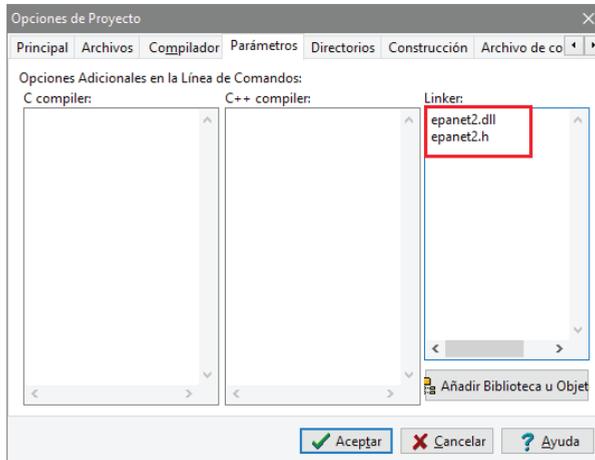


Figura 4.53. Búsqueda y selección de los ficheros epanet2.dll y epanet2.h

En la pestaña Dir. de Bibliotecas y Dir. de Include, ubicadas dentro de la pestaña “Directorios”, vamos añadir el directorio donde se encuentran almacenados los ficheros epanet2.dll y epanet2.h. Ambos ficheros se encuentran en el siguiente directorio: E:\ConectarAPIEpanet\6.C++\Proyecto1, tal como se muestra en la Figura 4.54.

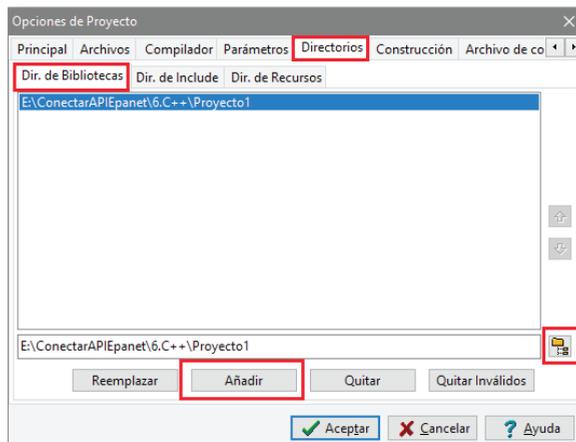


Figura 4.54. Añadir el directorio donde se almacena los ficheros epanet2.h y epanet2.dll

Finalmente, en la pestaña “Construcción” se buscará y se seleccionará la carpeta Proyecto1 para almacenar un fichero ejecutable y un fichero objeto, después de compilar el código escrito tal como se ve en la Figura 4.55. Por último forzaremos el nombre del fichero ejecutable como ConectarAPIEpanet.exe, y cerraremos la ventana de Opciones para Aceptar todos los cambios.

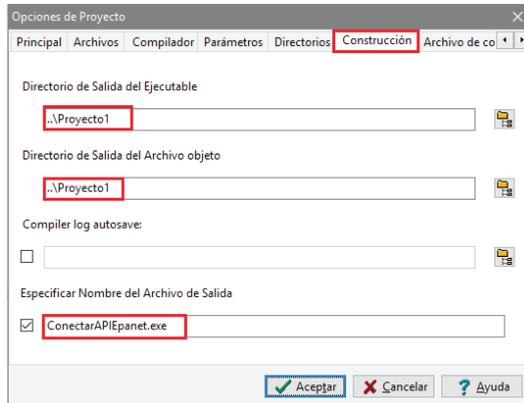


Figura 4.55. Configuración de la pestaña Construcción

#### D. Trabajando con la Toolkit de Epanet

Después de configurar las opciones del proyecto, lo siguiente es escribir sentencias de código en el lenguaje C++ que nos permitan recuperar el número de nudos y líneas totales de la red, así como la unidad de caudal utilizada. Para compilar y ejecutar nuestro código hacemos clic en el Menú Compilar (dos veces) y luego clic en Ejecutar (una vez). En la Figura 4.56 se muestra el resultado obtenido desde la consola del sistema.

El proyecto se puede descargar desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962790>. Si se quiere ejecutar directamente el código fuente administrado en el enlace anterior, hay que tener en cuenta el cambio de ubicación de la carpeta Proyecto1 y modificar la ruta de la ubicación de la red en estudio (Net3.inp), así como la modificación en las opciones del proyecto desde la aplicación Dev-C++.



# 5

## EJERCICIOS PRÁCTICOS

**A** continuación, se presentan cinco casos prácticos en donde se utilizará la mayoría de las funciones contenidas en la librería de Epanet v2.00.12 escritas desde el entorno de programación de Microsoft Visual Studio 2015 (Visual Basic .NET). En estos ejercicios no se pretende resolver problemas de mejoras en la operación de los sistemas de abastecimiento de agua como por ejemplo sectorización, simplificación, fiabilidad del sistema, dimensionado óptimo, optimización, etc., tan sólo se busca enseñar a utilizar el conjunto de funciones, de forma que combinándolas con las funciones propias de cada lenguaje de programación, se puedan diseñar herramientas más potentes en favor de la hidráulica urbana.

Para cada uno de los casos prácticos no se escribirá el código fuente, ya que podría ocupar un volumen de hojas importante, así que tan sólo se describirá la secuencia de pasos que hay que realizar, y se mencionarán las funciones que se vayan utilizando. El modelo de red que se empleará en todas las tareas propuestas será la red Net3.inp. Todo el código fuente que se genere podrá descargarse desde un enlace que se indicará al final de cada tarea y vendrá comentado para su mejor entendimiento.

Las tareas que se llevarán a cabo en este capítulo son: recuperar y modificar parámetros de la red, ejecutar una simulación hidráulica y de calidad del agua, recuperar los resultados del cálculo y generar un informe, calcular la presión en un nudo específico para un periodo extendido, y cambiar el trazado de líneas.

## A. Recuperar y modificar parámetros de la red

Si queremos recuperar por ejemplo el identificador de un nudo de caudal, su elevación y demanda base, se necesitarán las funciones ENgetnodetype, ENgetnodeid, y ENgetnodevalue. Para el caso de las tuberías, si se quiere recuperar su identificador, nudo inicial y final, la longitud y su estado, se usarán las funciones ENgetlinkid, ENgetlinknodes, y ENgetlinkvalue. Pero antes de poder utilizar dichas funciones, se deberá llamar a la función ENopen para abrir el fichero Inp de Epanet, y después de utilizarlas se deberá llamar a la función ENclose para cerrar el fichero de entrada y liberar la memoria utilizada del ordenador.

Para guardar la información solicitada, se creará un vector de tipo definido por el usuario, donde se almacenarán los datos del fichero Inp de Epanet. En la Figura 5.1 y Figura 5.2 se muestra parte del código fuente utilizado para recuperar los datos de los nudos y tuberías, el cual se ha añadido al código del ejemplo ConectarAPIEpanet comentado en el apartado anterior.

Es importante ir comprobando que la información que se vaya guardando en nuestro vector sea la correcta. Para ello, se insertará un punto interrupción casi al terminar el evento clic del botón Aceptar, concretamente en la línea de código `error = ENclose()`. En la Figura 5.3 se muestra el resultado de los datos almacenados en los vectores NudoCaudal y Tuberia añadidos a la ventana de inspección.

## EJERCICIOS PRÁCTICOS

---

```
'====='  
'   RECUPERAR DATOS DE NUDOS DE CAUDAL  
'====='  
k = 0  
For j = 1 To numNodos  
    'Recuperar el ID del nudo  
    Err = ENgetnodeid(j, ID)  
    'Añadir el ID del nudo a la colección  
    ColIDNudo.Add(ID.ToString, CStr(j))  
    'Recuperar el tipo de nudo  
    Err = ENgetnodetype(j, tipoNudo)  
    If tipoNudo = EN_JUNCTION Then  
        'Dimensionar el vector  
        k = k + 1  
        ReDim Preserve NudoCaudal(k)  
        'Asignar el ID del nudo  
        NudoCaudal(k).ID = ID.ToString  
        'Recuperar la elevación del nudo  
        Err = ENgetnodevalue(j, EN_ELEVATION, Valor)  
        NudoCaudal(k).Elevacion = Valor  
        'Recuperar la demanda base del nudo  
        Err = ENgetnodevalue(j, EN_BASEDEMAND, Valor)  
        NudoCaudal(k).DemandaBase = Valor  
    End If  
Next j
```

Figura 5.1. Funciones utilizadas para recuperar datos de los nudos de caudal

```
'====='  
'   RECUPERAR DATOS DE TUBERÍAS  
'====='  
k = 0  
For j = 1 To numLineas  
    'Recuperar el ID de la tubería  
    Err = ENgetlinkid(j, ID)  
    'Recuperar el tipo de línea  
    Err = ENgetlinktype(j, tipoLinea)  
    If tipolinea = EN_PIPE Then  
        'Dimensionar el vector  
        k = k + 1  
        ReDim Preserve Tuberia(k)  
        'Asignar el ID de la tubería  
        Tuberia(k).ID = ID.ToString  
        'Recuperar los nudos extremos de cada tubería  
        Err = ENgetlinknodes(j, IndN1, IndN2)  
        Tuberia(k).N1 = ColIDNudo(IndN1)  
        Tuberia(k).N2 = ColIDNudo(IndN2)  
        'Recuperar longitud de la tubería  
        Err = ENgetlinkvalue(j, EN_LENGTH, Valor)  
        Tuberia(k).Longitud = Valor  
        'Recuperar estado de la tubería  
        Err = ENgetlinkvalue(j, EN_INITSTATUS, Valor)  
        If Valor = 0 Then Tuberia(k).Estado = "Cerrada" '0 = Cerrada  
        If Valor = 1 Then Tuberia(k).Estado = "Abierta" '1 = Abierta  
        'Recuperar el diámetro de las tuberías  
        Err = ENgetlinkvalue(j, EN_DIAMETER, Valor)  
        Tuberia(k).Diametro = Valor  
    End If  
Next j
```

Figura 5.2. Funciones utilizadas para recuperar datos de las tuberías

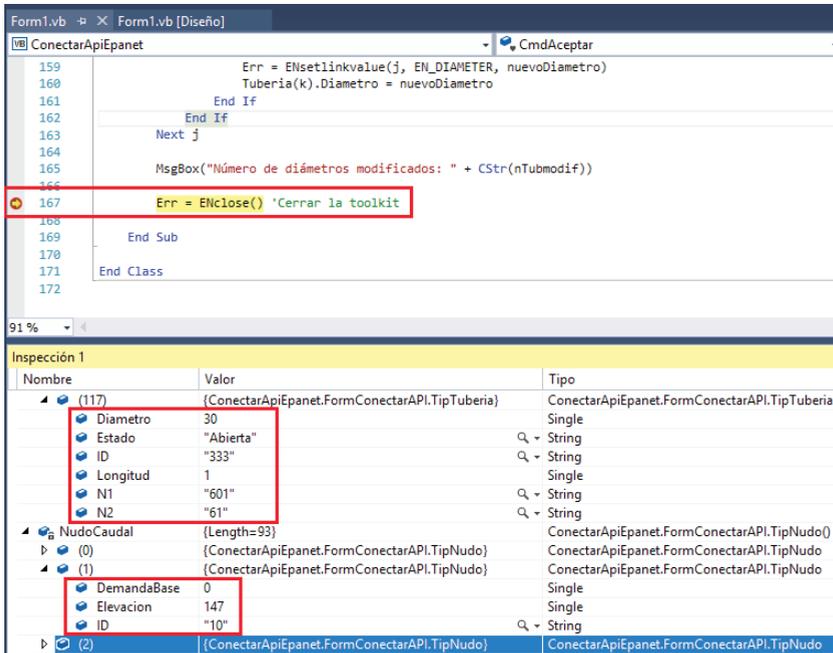


Figura 5.3. Inspección de los datos almacenados en los vectores NudoCaudal y Tuberia

Para realizar alguna modificación de los parámetros que definen las características o el modo de operación de nuestro modelo de red se deben utilizar las funciones que comienzan con el prefijo ENset (ej. ENsetnodevalue, ENsetlinkvalue, ENsetpattern, etc.). Como ejemplo, vamos a cambiar los tamaños de los diámetros de las tuberías del modelo de red Net3 que tengan valor de 12" a 16" (pulgadas). Además, se guardarán los cambios realizados en el vector Tuberia().Diametro por si se quiere tener una lista de los identificadores de las tuberías con los nuevos diámetros. Desde la propia aplicación Epanet, podemos realizar una consulta que nos permitirá visualizar y conocer el número total de tuberías a modificar. Esto lo podemos hacer a través desde la siguiente secuencia de comandos menú View >> Query (Figura 5.4).

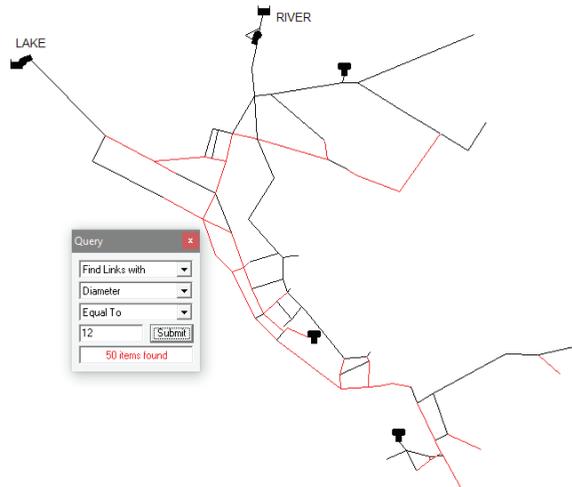


Figura 5.4. Consulta del número de tuberías con diámetro igual a 12" desde Epanet

Si queremos modificar el tamaño de los diámetros habrá que seleccionar primero toda la red desde el menú Edit >> Select Region y luego Edit >> Group Edit y configurar las opciones del cuadro de diálogo para efectuar los cambios (Figura 5.5).

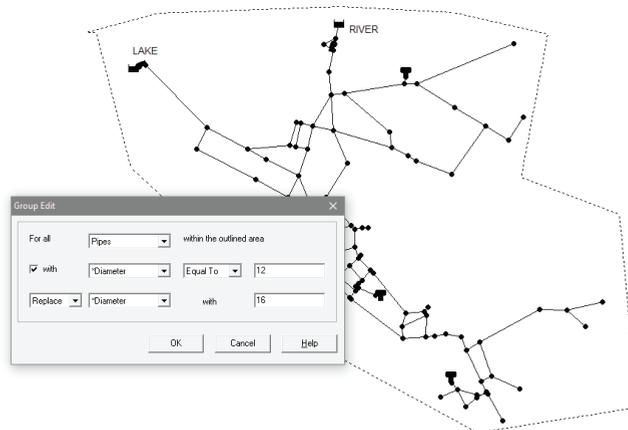


Figura 5.5. Modificar diámetros de tubería con la herramienta Group Edit desde Epanet

Utilizando las funciones de la librería de Epanet también obtenemos los mismos resultados, tal como se puede ver en la Figura 5.6.

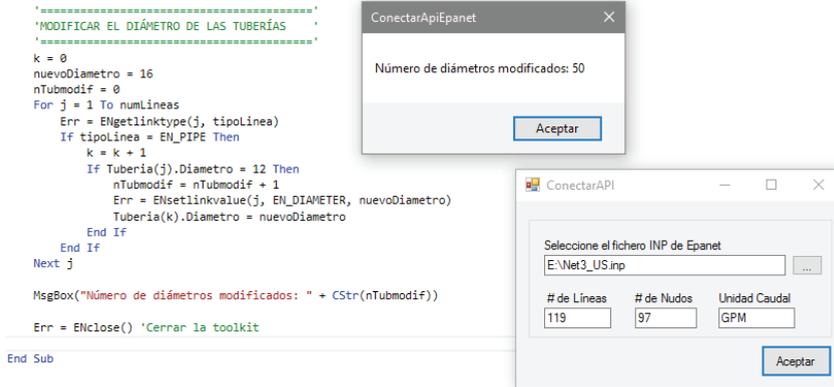


Figura 5.6. Modificar diámetros de las tuberías con las funciones de la librería de Epanet

El código fuente utilizado para llevar a cabo esta primera tarea se puede descargar desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962791>. En dicha carpeta comprimida está el fichero de módulo, la librería de Epanet (v2.00.12) y el modelo de red Net3.inp.

## B. Ejecutar una simulación hidráulica y de calidad del agua

La parte más importante de la librería de Epanet es la simulación hidráulica y de calidad del agua. Podemos obtener los resultados de una simulación hidráulica de dos maneras. Una de ellas es utilizando la función ENsolveH, que permite ejecutar un análisis completo en periodo extendido, sin tener acceso a los resultados intermedios, y la otra utilizando la secuencia de funciones ENopenH, ENinitH, ENrunH, ENnextH, ENcloseH, para ejecutar una simulación paso a paso, avanzando un intervalo hidráulico de cálculo cada vez.

El primer método es recomendado si se va a realizar posteriormente un análisis de calidad del agua. Con este método, los resultados del cálculo hidráulico para cada uno de los intervalos se guardan siempre en un fichero de resultados hidráulicos (fichero binario). El segundo método es más apropiado cuando se quiere tener acceso a los resultados intermedios entre los diferentes intervalos de cálculo, o si

se pretenden realizar muchas ejecuciones de una manera eficiente. En este caso, se realizará una única llamada a la función ENopenH para empezar el proceso, a continuación se harían sucesivas llamadas a ENinitH-ENrunH-ENnextH para ejecutar cada etapa del análisis, y finalmente se llamaría a ENcloseH para cerrar el módulo hidráulico. Si se va a realizar un análisis en régimen permanente no es necesario llamar a la función ENnextH.

Para el caso del análisis de la calidad del agua, se deben haber generado previamente los resultados hidráulicos, bien mediante la ejecución de una simulación hidráulica o bien importando un fichero de resultados hidráulicos previamente guardado. Al igual que para el análisis hidráulico, existen dos formas de llevar a cabo una simulación de la calidad del agua. La primera de ellas es utilizando la función ENSolveQ para ejecutar un análisis de calidad completo en periodo extendido, sin tener acceso a resultados intermedios, y la segunda utilizando la secuencia de funciones ENopenQ-ENinitQ-ENrunQ-ENnextQ-ENcloseQ para realizar una simulación paso a paso, avanzando un intervalo de cálculo hidráulico cada vez. (Reemplazando ENnextQ por ENstepQ se puede realizar la simulación avanzando un intervalo de cálculo de calidad cada vez).

Descrita brevemente la manera de cómo utilizar las funciones de la librería de Epanet para un cálculo hidráulico y de calidad del agua, pasaremos a escribir líneas de código con los métodos antes vistos.

### SIMULACIÓN HIDRÁULICA

#### 1) Método 1. Utilizando la función ENSolveH

Para ejecutar una simulación hidráulica completa, es necesario utilizar la siguiente secuencia de funciones: ENopen-ENSolveH-ENsaveH-ENclose. La función ENopen recibe tres parámetros de entrada. Si se quiere disponer del fichero de resultados hidráulicos, habrá que especificarlo en uno de esos parámetros y utilizar la función ENsaveH para guardarlo, caso contrario dicho fichero de extensión \*.out será borrado tras llamar a la función ENclose.

El Fichero de Resultados Hidráulicos (\*.out) es un fichero binario, utilizado para almacenar los resultados de un análisis hidráulico. En él, se almacenan los resultados para todos los intervalos de tiempo regulares prefijados en la cláusula REPORT TIMESTEP de la sección [TIMES] (por defecto 1 hora). Sin embargo, no se escriben los instantes intermedios en los cuales se producen cambios hidráulicos (ej. apertura o cierre de bombas o válvulas debido a alguna ley de control).

En la Figura 5.7 se pueden ver las funciones utilizadas para realizar una simulación completa. Después de ejecutar el código se habrán creado dos ficheros con el nombre de la red y de extensiones \*.rpt y \*.out en la misma carpeta donde está almacenado el fichero Inp de Epanet (Figura 5.8).

Si queremos almacenar los resultados relativos solo a los caudales, para todos los instantes de cálculo, incluidos los instantes intermedios, al objeto de utilizarlos después en un análisis de calidad, deberemos indicarlo introduciendo el comando HYDRAULICS USE *nombrefichero* en la sección [OPTIONS] del fichero de entrada, o realizando una llamada a la función ENusehydfile. El código fuente puede descargarse desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962792>.

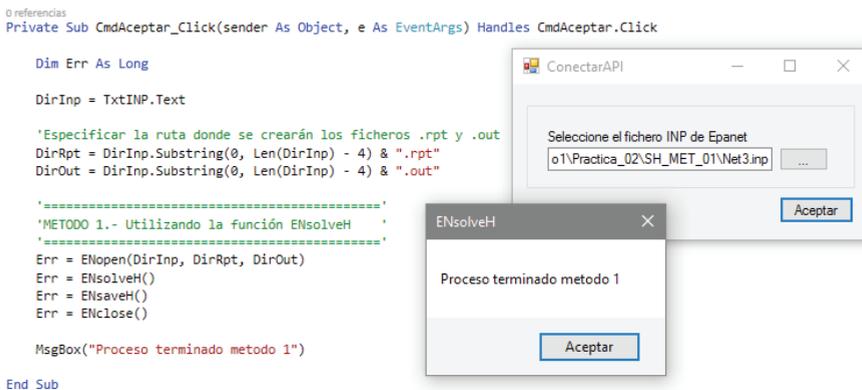


Figura 5.7. Uso de la función ENSolveH y ENsaveH de la librería de Epanet

Nombre ^	Tipo	Tamaño
.vs	Carpeta de archivos	
ENsolveH	Carpeta de archivos	
Met_01	Microsoft Visual Studio Solution	1 KB
Net3	Archivo INP	30 KB
Net3.out	Archivo OUT	142 KB
Net3.rpt	Archivo RPT	4 KB

Figura 5.8. Ficheros Net3.rpt y Net3.out creados al finalizar el proceso

2) Método 2. Utilizando las funciones ENopenH-ENinitH-ENrunH-ENnextH-ENcloseH

Para comprender mejor el uso del conjunto de funciones destinadas a controlar el proceso de simulación se propone el siguiente ejercicio. Se pide calcular para el instante de tiempo 14:00 horas el caudal, la velocidad y el estado de las tuberías, así como la demanda actual, la altura total y la presión en los nudos de caudal.

El modelo de red Net3, en el apartado opciones de tiempo, tiene la siguiente configuración: tiempo de duración 24 horas, intervalo de cálculo hidráulico 1 hora, intervalo de reporte 1 hora, inicio de reporte 0 horas.

Se pide obtener los valores de las propiedades antes mencionadas de los elementos de red nudos de caudal y tuberías para el instante de tiempo 14:00 horas; la variable `Tiempo = 14*3600.0#` determina el instante a capturar. En la Figura 5.9 se puede ver una parte importante del código, donde se hace uso del par de funciones ENrunH-ENnextH para llevar a cabo la simulación paso a paso, y capturar los resultados deseados a las 14:00. Para descargar el código fuente completo pinchar el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962793>.

```

Do
  'bucle ENrunH-ENnextH
  Err = ENrunH(tt) : If Err > 6 Then Stop
  If Tiempo = tt Then
    '-----Para las líneas
    k = 0
    For j = 1 To nLíneas
      Err = ENgetlinktype(j, tipoLínea)
      If tipoLínea = EN_PIPE Then
        'Dimensionar el vector
        k = k + 1
        ReDim Preserve Tubería(k)
        'Recuperar el Caudal
        Err = ENgetlinkvalue(j, EN_FLOW, Valor)
        Tubería(k).Caudal = Valor
        'Recuperar estado actual
        Err = ENgetlinkvalue(j, EN_STATUS, Valor)
        If Valor = 0 Then Tubería(k).Estado = "Cerrada" '0 = Cerrada
        If Valor = 1 Then Tubería(k).Estado = "Abierta" '1 = Abierta
        'Recuperar la velocidad
        Err = ENgetlinkvalue(j, EN_VELOCITY, Valor)
        Tubería(k).Velocidad = Valor
      End If
    Next j
    '-----Para los nudos
    k = 0
    For j = 1 To nNudos
      Err = ENgetnodetype(j, tipoNudo)
      If tipoNudo = EN_JUNCTION Then
        'Dimensionar el vector
        k = k + 1
        ReDim Preserve NudoCaudal(k)
        'Recuperar demanda actual
        Err = ENgetnodevalue(j, EN_DEMAND, Valor)
        NudoCaudal(k).DemActual = Valor
        'Recuperar altura total
        Err = ENgetnodevalue(j, EN_HEAD, Valor)
        NudoCaudal(k).AltTotal = Valor
        'Recuperar la presión
        Err = ENgetnodevalue(j, EN_PRESSURE, Valor)
        NudoCaudal(k).Presion = Valor
      End If
    Next j
  End If
  Err = ENnextH(ttt)
Loop Until ttt = 0

```

Figura 5.9. Simulación hidráulica paso a paso con la librería de Epanet

### SIMULACIÓN DE LA CALIDAD DEL AGUA

Epanet nace a raíz de querer conocer el avance y destino final de distintas sustancias químicas transportadas por el agua mientras ésta discurre por la red de distribución.

Además del transporte de sustancias químicas, Epanet permite estudiar otros fenómenos relacionados con la calidad del agua como la mezcla de agua procedente de diversas fuentes (procedencia), el envejecimiento del agua mientras discurre por la red (tiempo de permanencia), la pérdida de cloro residual, el crecimiento de los subproductos derivados de la cloración, el seguimiento del avance de un contaminante tras su intrusión en la red, etc.

Antes de realizar una simulación de la calidad del agua se deben haber generado los resultados hidráulicos, bien mediante la ejecución de una simulación hidráulica o bien importando un fichero de resultados hidráulicos previamente guardado.

Al igual que para el análisis hidráulico, existen dos formas de llevar a cabo una simulación de la calidad del agua. Podemos utilizar la función ENsolveQ o el conjunto de funciones ENopenQ-ENinitQ-ENrunQ-ENnextQ-ENcloseQ.

Para conocer la forma de utilizar ambos métodos se plantea el siguiente ejercicio. Se pide determinar la fracción de caudal que llega a cada nudo del modelo de red Net3, procedente del embalse Lake (análisis de procedencia). Para ello, hay que comprobar que los valores introducidos en el apartado opciones de calidad sea el correcto. En la Figura 5.10 podemos ver la información que hay que introducir para realizar el análisis de procedencias.

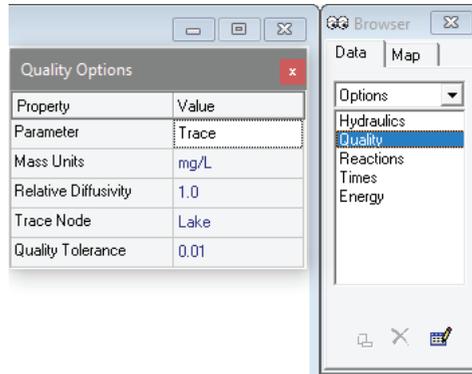


Figura 5.10. Parámetros de calidad a introducir en Epanet

### 1) Método 1. Utilizando la función ENSolveQ

La función ENSolveQ ejecuta una simulación completa de la calidad del agua, volcando los resultados a intervalos regulares de tiempo en el fichero binario de salida de Epanet (\*.out), sin tener acceso a resultados intermedios. En la Figura 5.11 se puede ver la secuencia de funciones a utilizar conjuntamente con la función ENSolveQ. Para descargar el código completo puede hacer clic en el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962794>.

```

Referencias
Private Sub CmdAceptar_Click(sender As Object, e As EventArgs) Handles CmdAceptar.Click

    Dim Err As Long

    DirInp = TxtINP.Text

    'Especificar la ruta donde se crearán los ficheros .rpt y .out
    DirRpt = DirInp.Substring(0, Len(DirInp) - 4) & ".rpt"
    DirOut = DirInp.Substring(0, Len(DirInp) - 4) & ".out"

    '=====
    'METODO 1.- Utilizando la función ENSolveQ
    '=====
    Err = ENopen(DirInp, DirRpt, DirOut)
    Err = ENSolveH()
    Err = ENSaveH()
    Err = ENSolveQ()
    Err = ENclose()

    MsgBox("Análisis de calidad terminado. Uso de ENSolveQ")

End Sub

```

Figura 5.11. Uso de la función ENSolveQ de la librería de Epanet

2) Método 2. Utilizando las funciones ENopenQ-ENinitQ-ENrunQ-ENnextQ-ENcloseQ

Para aplicar el conjunto de funciones destinadas al análisis de calidad de agua, se propone realizar el siguiente ejercicio. Se pide determinar el porcentaje de caudal (análisis de procedencias) que procede del embalse “Lake” para todos los nudos de la red en el instante de tiempo 14:00 horas. En la Figura 5.12 se puede ver el núcleo central del código correspondiente. Para descargar el código completo pinchar el enlace <https://www.imta.gob.mx/biblioteca/download/?key=962795>.

```

'=====
'METODO 2.- Utilizando las funciones:
'ENopenQ - ENinitQ - ENrunQ - ENnextQ - ENcloseQ'
'=====
Tiempo = 14 * 3600.0# 'recoger información a las 14:00 horas

Err = ENopen(DirInp, DirRpt, DirOut) 'Abrir la Toolkit
Err = ENgetcount(EN_NODECOUNT, nNudos)
Err = ENSolveH()
Err = ENopenQ : If Err <> 0 Then Stop
Err = ENinitQ(1) : If Err <> 0 Then Stop

Do
  'buclé ENrunQ-ENnextQ
  Err = ENrunQ(ttt) : If Err > 6 Then Stop
  If Tiempo = ttt Then
    k = 0
    For j = 1 To nNudos
      Err = ENgetnodetype(j, tipoNudo)
      If tipoNudo = EN_JUNCTION Then
        k = k + 1
        ReDim Preserve NudoCaudal(k)
        Err = ENgetnodevalue(j, EN_QUALITY, Valor)
        NudoCaudal(k).PorcProc = Valor
      End If
    Next j
  End If
  Err = ENnextQ(ttt)
Loop Until ttt = 0

Err = ENcloseQ
Err = ENclose()

MsgBox("Análisis de calidad terminado. Método 2")

```

Figura 5.12. Simulación de la calidad del agua paso a paso con la librería de Epanet

### C. Recuperar los resultados del cálculo y generar un informe

Con las funciones ENgetnodevalue y ENgetlinkvalue, que ya hemos venido utilizando en ejercicios anteriores, podemos recuperar los resultados del cálculo hidráulico y de la calidad del agua. En la Tabla 5.1 se muestran las palabras claves utilizadas para recuperar los valores de los elementos de una red de abastecimiento con Epanet.

Tabla 5.1. Palabras claves utilizadas para recuperar valores después de una simulación

Para Nudos	Para Líneas
EN_DEMAND (demanda actual)	EN_FLOW (caudal)
EN_HEAD (altura piezométrica)	EN_VELOCITY (velocidad del flujo)
EN_PRESSURE (presión)	EN_HEADLOSS (pérdida de carga)
EN_QUALITY (calidad del agua)	EN_STATUS (estado actual de la línea)
EN_SOURCEMASS (flujo másico de una fuente contaminante).	EN_SETTING (velocidad de giro impuesta a una bomba o consigna impuesta a una válvula)

La sintaxis para recuperar los valores de un elemento de la red después de aplicar una simulación es la siguiente: `error = ENgetnodevalue (i, palabra clave, j)`, donde `error` es un valor numérico que devuelve la función y que en caso de salir todo bien devolverá cero. El parámetro “`i`” es el índice, es decir la posición que guarda el nudo dentro del fichero INP de Epanet (el cual se obtiene con las funciones ENgetnodeindex ó ENgetlinkindex), el parámetro “palabra clave” es un código que se utiliza para especificar la magnitud a recuperar según el tipo de elemento (ver Tabla 5.1), y finalmente el parámetro “`j`” es la variable donde se devuelve el valor calculado.

La librería de Epanet lleva incorporadas algunas funciones que ayudan a generar un informe de los resultados obtenidos con un formato propio. Aunque siempre existe la posibilidad de escribir nuestros propios informes con ayuda de las funciones anteriores y otras propias del lenguaje de programación que estemos utilizando, se puede simplificar la escritura de un informe personalizado, por

ejemplo solo para las variables deseadas, mediante el uso de algunas funciones de la Toolkit. La función `ENsetreport` se utiliza para definir el formato de un informe, mientras que la función `ENreport` genera propiamente dicho informe. Esta última función deberá llamarse únicamente después de haber realizado un análisis hidráulico o de calidad.

En la Figura 5.13 se muestra el núcleo central del código escrito en el que se crea un informe con la lista de todos los nudos de la red cuya variación de presión a lo largo de la simulación supera los 50 psi. El código completo puede descargarse desde el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962796>.

Los resultados serán guardados en el fichero de resultados con extensión \*.rpt

Figura 5.13. Funciones que ayudan a generar un informe de resultados

```
Dim Err As Long

DirInp = TxtINP.Text

'Especificar la ruta donde se crearán los ficheros .rpt y .out
DirRpt = DirInp.Substring(0, Len(DirInp) - 4) & ".rpt"
DirOut = DirInp.Substring(0, Len(DirInp) - 4) & ".out"

'=====
'Utilizar funciones ENresetreport-ENsetreport-ENreport'
'=====

'Abrir la Toolkit
Err = ENopen(DirInp, DirRpt, DirOut)
'Realizar la simulación hidráulica
Err = ENSolveH()
'Fijar el tipo del postproceso de los resultados
Err = ENsettimeparam(EN_STATISTIC, EN_NONE)
'Transferir los resultados del fichero de
'resultados hidráulicos al fichero de salida
Err = ENSaveH()
'Definir las características del informe
Err = ENresetreport()
Err = ENsetstatusreport(2)
Err = ENsetreport("NODES ALL")
Err = ENsetreport("PRESSURE PRECISION 1")
Err = ENsetreport("PRESSURE ABOVE 50")
'Escribir el informe en el fichero
Err = ENreport()
'Cerrar fichero inp y liberar memoria
Err = ENclose()

'Mostrar mensaje final
MsgBox("Informe terminado")
```

La función ENsettimeparam permite fijar el valor de un parámetro de tiempo. Es este caso se ha utilizado para especificar que se guarden los resultados de todos los instantes de cálculo, a intervalos regulares. La función ENresetreport, elimina cualquier comando de formato previo que aparezca en la sección [REPORT] del fichero de entrada a Epanet, o que haya sido fijado mediante la función ENsetreport. La función ENsetreport, establece los comandos de formato para personalizar el informe de resultados. Los comandos de formato son los mismos que los utilizados en la sección [REPORT] del fichero de entrada de Epanet. La función ENreport genera un informe, en formato de texto, con los resultados de la simulación deseados y lo guarda en el fichero de informe de resultados (\*.rpt). Hay otras funciones dentro de la librería de Epanet como ENsaveinpfile, ENgeterror, ENwriteline, ENsavehydfile, ENusehyfile, que pueden utilizarse según sea el caso. La descripción de cada una de ellas y las que se han venido utilizando en los ejemplos anteriores se pueden consultar en el fichero de ayuda de la Toolkit de Epanet.

#### D. Calcular las presiones en un nudo de caudal según la variación de la demanda

Para resolver este caso se utilizará como caso de estudio el modelo de red Red1\_SI.inp. En este ejercicio se pide calcular las presiones resultantes en el nudo “23”, para todo el periodo de simulación (24 horas), y para tres valores diferentes de la demanda base en dicho nudo.

Una solución es utilizar la secuencia de funciones ENopenH-ENinitH-ENrunH-ENnextH-ENcloseH e ir guardando los valores calculados de las presiones en un vector. Las presiones resultantes se obtienen con la función ENgetnodevalue. La otra forma de obtener las presiones es mediante un fichero de informe de resultados, donde sólo se especifique el nudo a analizar. En este caso, se ha optado por utilizar la primera solución.

En la Figura 5.14 se puede ver el código central de la solución del ejercicio. Si se quiere descargar el código completo, así como el modelo de red, hacer clic en el enlace <https://www.imta.gob.mx/biblioteca/download/?key=962797>.

```

Dim DemBase(3) As Int32
Dim Err, indexNudo, k, tiempo As Integer
Dim tt As Long, ttt As Long
Dim Valor As Single
Dim IDNudo As String

'Fijar los valores de Demanda Base para el Nudo ID = 23
DemBase(1) = 10
DemBase(2) = 30
DemBase(3) = 60

'ID del Nudo a analizar
IDNudo = "23"

'Dimensionar vector
ReDim Demanda(3)

'Asignar ruta del Fichero Inp a la variable
'Especificar la ruta donde se crearán los ficheros .rpt y .out
DirInp = TxtINP.Text
DirRpt = DirInp.Substring(0, Len(DirInp) - 4) & ".rpt"
DirOut = DirInp.Substring(0, Len(DirInp) - 4) & ".out"

'Abrir la Toolkit
Err = ENopen(DirInp, DirRpt, DirOut)
'Obtener el índice del nudo de interés
Err = ENgetnodeindex(IDNudo, indexNudo)

'Abrir el módulo de cál. hidráulico e iterar todas las demandas a ensayar
Err = ENopenH : If Err <> 0 Then Stop
For k = 1 To 3
    tiempo = 0
    Err = ENsetnodevalue(indexNudo, EN_BASEDEMAND, DemBase(k))
    Err = ENinitH(1) : If Err <> 0 Then Stop
    Do
        Err = ENrunH(tt) : If Err > 6 Then Stop
        If tiempo = tt / 3600 Then
            'Recuperar el valor de la presion
            Err = ENgetnodevalue(indexNudo, EN_PRESSURE, Valor)
            ReDim Preserve Demanda(k).Presion(tiempo + 1)
            Demanda(k).Presion(tiempo) = Valor
            tiempo = tiempo + 1
        End If
        Err = ENnextH(ttt)
    Loop Until ttt = 0
Next

'Cerrar módulo de cálculo hidráulico, fichero inp y liberar memoria
Err = ENcloseH
Err = ENclose()

MsgBox("Proceso terminado")

```

Figura 5.14. Funciones utilizadas para el cálculo de presiones en el nudo 23

## E. Cambiar la orientación de las tuberías

Muchas veces nos hemos encontrado con modelos de red donde al realizar una simulación hidráulica el caudal que circula por las tuberías aparece con signo negativo. Ello no es ningún error, esto se debe a que cuando hemos importado

una red desde AutoCAD, GIS, Base de Datos, o cualquier otro medio de almacenamiento, el trazado de las líneas no coincide con la dirección del flujo en el instante actual.

Si el signo del flujo es predominantemente negativo, desde Epanet podemos solucionar este inconveniente utilizando la opción “Reverse” o “Invertir”, según la versión (Inglés/Español). Para ello, tenemos que situarnos sobre la línea, hacer clic con el botón derecho del ratón, y seleccionar la opción “Reverse/Invertir”. Esto no lleva mucho esfuerzo si son unas pocas líneas que se deba invertir su trazado, ¿pero qué pasa si son cientos o miles? Con ayuda de algunas funciones contenidas en la librería de Epanet, podemos solucionar este problema.

Este caso se resuelve realizando una simulación hidráulica paso a paso y obteniendo de las tuberías el caudal de circulación a lo largo de todo el periodo de simulación. Si el caudal es negativo en todos los instantes del periodo de simulación, es que la tubería ha sido digitalizada al contrario del sentido normal del flujo.

Como no existe una función específica en la librería de Epanet v2.00.12, utilizaremos las funciones propias de Visual Studio 2015 (Visual Basic .NET) para recuperar la información de las tuberías (Sección [PIPES]), y las coordenadas de los vértices de las mismas (Sección [VERTICES]). Trabajaremos con el modelo de red Red1\_SI.inp, sobre el cual se ha modificado el trazado de algunas de sus tuberías, para comprobar que el algoritmo funciona correctamente.

La herramienta nos pedirá que seleccionemos un fichero Inp de Epanet, y nos devolverá otro fichero Inp (corregido) en el mismo directorio del fichero original, así como un fichero de texto con los identificadores de las tuberías cuyo trazado se ha modificado. Para descargar el código completo se puede hacer clic en el siguiente enlace: <https://www.imta.gob.mx/biblioteca/download/?key=962798>. En la Figura 5.15 puede verse el funcionamiento de la herramienta.

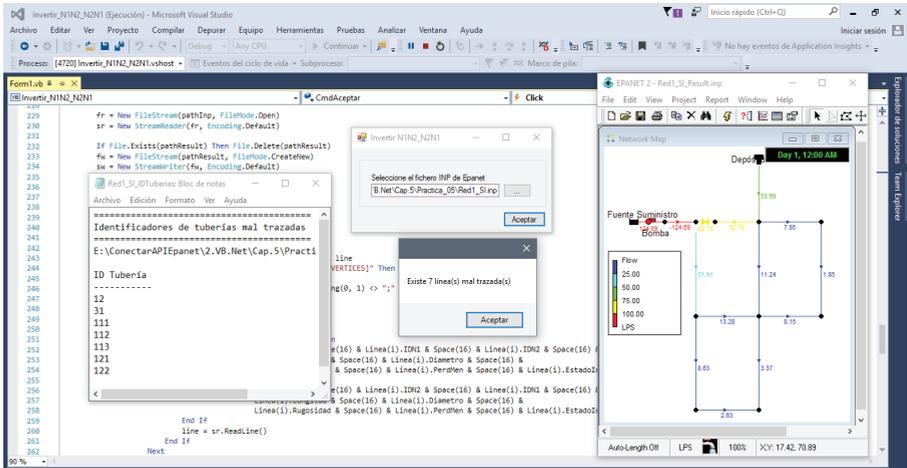


Figura 5.15. Herramienta para invertir las líneas mal orientadas con ayuda de la librería de Epanet

## DE LOS AUTORES



---

### **Oscar Tomas Vegas Niño**

---

Ingeniero Agrícola por la Universidad Nacional de Trujillo (Perú), cuenta con una Maestría en Ingeniería Hidráulica y Medio Ambiente por la Universitat Politècnica de València (España) y dos especialidades en el campo de la hidráulica urbana y los recursos hídricos. Participa activamente en congresos nacionales e internacionales, evaluador de artículos científicos y proyectos financiados por FYNcyT (Perú), asimismo, desarrolla herramientas informáticas utilizando la librería de EPANET y los Sistemas de Información Geográfica (SIG). A la fecha de publicación de este libro viene desarrollando su tesis doctoral en el área de la hidráulica urbana financiado por el Gobierno Peruano (PRONABEC)



---

### **Fernando Martínez Alzamora**

---

Dr. Ingeniero Industrial y Catedrático de Ingeniería Hidráulica en la Universitat Politècnica de València (España). Autor de la versión española de EPANET 2.0, ha publicado en los últimos 35 años numerosos artículos en revistas y congresos sobre el uso y mejora de esta herramienta. Asimismo es responsable de las aplicaciones GISRed y SCARed dirigidas a integrar EPANET en entornos GIS para facilitar la construcción de modelos y en sistemas SCADA para el control en tiempo real de las redes de agua. Lidera el grupo de investigación REDHISP del Instituto de Ingeniería del Agua y Medio Ambiente (IIAMA) de la UPV.

## DE LOS AUTORES



---

### **Joan Carles Alonso Campos**

---

Ingeniero Industrial por la Universitat Politècnica de València (España). A la fecha de publicación de este libro, cursa el doctorado en el Programa de Ingeniería del Agua y Medio Ambiental en la misma Universidad. El proyecto de Tesis, financiado mediante el programa Val I+D de la Generalitat Valenciana, se enfoca en el desarrollo e implementación de algoritmos para la optimización energética de redes hidráulicas a presión.



---

### **Velitchko G. Tzatchkov**

---

Ingeniero Civil por el Instituto Superior de Ingeniería Civil de Sofía (Bulgaria) y Doctor en Hidráulica por el Instituto de Hidrotecnia, Riego y Drenaje de la misma ciudad. Desde 1991 es investigador Especialista en Hidráulica y Tecnólogo del Agua en el Instituto Mexicano de Tecnología del Agua. Ha desarrollado algoritmos y herramientas informáticas para análisis, diseño y operación de redes de distribución de agua y riego a presión, incluyendo un algoritmo que amplía la modelación de calidad del agua que realiza Epanet con el término de dispersión, publicado y premiado en los Estados Unidos de Norteamérica. Ha publicado numerosos artículos en revistas y congresos, y varios libros.



El libro *Iniciación a la programación con la Toolkit de Epanet v2.00.12 en un entorno Windows®*, se publicó en el mes mayo de 2017, en versión electrónica. Instituto Mexicano de Tecnología del Agua.



**SEMARNAT**  
SECRETARÍA DE  
MEDIO AMBIENTE  
Y RECURSOS NATURALES

