Epanet2 user's manual

M. Cingi

November 27, 2015

Contents

List of Tables

Chapter 1

General Instructions

EPANET can also be run as a console application from the command line within a DOS window. In this case network input data are placed into a text file and results are written to a text file. The command line for running EPANET in this fashion is: epanet2d inpfile rptfile outfile Here inpfile is the name of the input file, rptfile is the name of the output report file, and outfile is the name of an optional binary output file that stores results in a special binary format. If the latter file is not needed then just the input and report file names should be supplied. As written, the above command assumes that you are working in the directory in which EPANET was installed or that this directory has been added to the PATH statement in your AUTOEXEC.BAT file. Otherwise full pathnames for the executable epanet2d.exe and the files on the command line must be used. The error messages for command line EPANET are listed in Appendix B.

Chapter 2

Epanet toolkit

EPANET is a program that analyzes the hydraulic and water quality behavior of water distribution systems. The EPANET Programmer's Toolkit is a dynamic link library (DLL) of functions that allows developers to customize EPANET's computational engine for their own specific needs. The functions can be incorporated into 32-bit Windows applications written in C/C++, Delphi Pascal, Visual Basic, or any other language that can call functions within a Windows DLL. The Toolkit DLL file is named EPANET2.DLL and is distributed with EPANET. The Toolkit comes with several different header files, function definition files, and .lib files that simplify the task of interfacing it with C/C++, Delphi, and Visual Basic code.

TASK	FUNCTION
Running a complete simulation	ENepanet
Opening and closing the EPANET Toolkit system	ENopen ENclose
Retrieving information about network nodes	ENgetnodeindex ENgetnodeid ENgetnodetype ENgetnodevalue
Retrieving information about network links	ENgetlinkindex ENgetlinkid ENgetlinktype ENgetlinknodes ENgetlinkvalue
Retrieving information about time patterns	ENgetpatternid ENgetpatternindex ENgetpatternlen ENgetpatternvalue
Retrieving other network information	ENgetcontrol ENgetqualtype

	ENgetoption
Setting new values for network parameters	ENsetcontrol ENsetnodevalue ENsetlinkvalue ENaddpattern ENsetpattern ENsetpatternvalue ENsetqualtype ENsettimeparam ENsetoption
Saving and using hydraulic analysis results files	ENsavehydfile ENusehydfile
Running a hydraulic analysis	ENsolveH ENopenH ENinitH ENrunH ENnextH ENcloseH
Running a water quality analysis	ENsolveQ ENopenQ ENinitQ ENrunQ ENnextQ ENstepQ ENcloseQ
Generating an output report	ENsaveH ENsaveinpfile ENreport ENresetreport ENsetreport ENsetstatusreport ENgeterror ENwriteline

2.1 Functions

2.1.1 ENepanet

Declaration

```
int ENepanet( char* f1, char* f2, char* f3, void (*) (vfunc) )
```

Description

Runs a complete EPANET simulation.

Arguments

f1	name of the input file
f2	name of an output report file
f3	name of an optional binary output file
vfunc	pointer to a user-supplied function which accepts a character string as its argument.

Returns

Returns an error code.

Notes

ENepanet is a stand-alone function and does not interact with any of the other functions in the toolkit. If there is no need to save EPANET's binary output file then f3 can be an empty string (""). The vfunc function pointer allows the calling program to display a progress message generated by EPANET during its computations. A typical function for a console application might look as follows:

```
void  writecon(char *s)
 {
    puts(s);
 }
```

and somewhere in the calling program the following declarations would appear:

```
 void (* vfunc) (char *);
 vfunc = writecon;
 ENepanet(f1,f2,f3,vfunc);
```

If such a function is not desired then this argument should be NULL (NIL for Delphi/Pascal, VBNULLSTRING for Visual Basic). ENepanet is used mainly to link the EPANET engine to third-party user interfaces that build network input files and display the results of a network analysis.

2.1.2 ENopen

Declaration

```
int ENopen( char* f1, char* f2, char* f3)
```

Description

Opens the Toolkit to analyze a particular distribution system.

Arguments

f1 name of the input file

f2 name of an output report file

f3 name of an optional binary output file

Returns

Returns an error code.

Notes

If there is no need to save EPANET's binary Output file then f3 can be an empty string ("").

ENopen must be called before any of the other toolkit functions (except ENepanet) are used.

See also

ENclose

2.1.3 ENclose

Declaration

```
int ENclose( void )
```

Description

Closes down the Toolkit system (including all files being processed).

Returns

Returns an error code.

Notes

ENclose must be called when all processing has been completed, even if an error condition was encountered.

See also

ENopen

2.1.4 ENgetnodeindex

Declaration

```
int ENgetnodeindex( char* id, int* index )
```

Description

Retrieves the index of a node with a specified ID.

Arguments

id	node ID label
index	node index

Returns

Returns an error code.

Notes

Node indexes are consecutive integers starting from 1.

See also

ENgetnodeid

Chapter 3

Input File Format

The input file for EPANET has the same format as the text file that Windows EPANET generates from its File Export Network command.

It is organized in sections, where each section begins with a keyword enclosed in brackets. The various keywords are listed in table 3.1.

The order of sections is not important. However, whenever a node or link is referred to in a section it must have already been defined in the [JUNCTIONS], [RESERVOIRS], [TANKS], [PIPES], [PUMPS], or [VALVES] sections. Thus it is recommended that these sections be placed first, right after the [TITLE] section. The network map and tags sections are not used by EPANET and can be eliminated from the file.Each section can contain one or more lines of data. Blank lines can appear anywhere in the file and the semicolon (;) can be used to indicate that what follows on the line is a comment, not data. A maximum of 255 characters can appear on a line. The ID labels used to identify nodes, links, curves and patterns can be any combination of up to 31 characters and numbers.

3.1 Network Components

3.1.1 [TITLE]

Purpose

Attaches a descriptive title to the network being analyzed.

Format

Any number of lines of text.

Remarks

The [TITLE] section is optional.

used keywords	
Network Components	[TITLE] [JUNCTIONS] [RESERVOIRS] [TANKS] [PIPES] [PUMPS] [VALVES] [EMITTERS]
System Operation	[CURVES] [PATTERNS] [ENERGY] [STATUS] [CONTROLS] [RULES] [DEMANDS]
Water Quality	[QUALITY] [REACTIONS] [SOURCES] [MIXING]
Options and Reporting	[OPTIONS] [TIMES] [REPORT]
unused keywords	
Network Map/Tags	[COORDINATES] [VERTICES] [LABELS] [BACKDROP] [TAGS]

Table 3.1: keywords used in inpfile

3.1.2 [JUNCTIONS]

Purpose

Defines junction nodes contained in the network.

Format

One line for each junction containing:

- ID label
- Elevation, ft (m)
- Base demand flow (flow units) (optional)
- Demand pattern ID (optional)

Remarks

1. A [JUNCTIONS] section with at least one junction is required.
2. If no demand pattern is supplied then the junction demand follows the Default Demand Pattern specified in the [OPTIONS] section or Pattern 1 if no default pattern is specified. If the default pattern (or Pattern 1) does not exist, then the demand remains constant.
3. Demands can also be entered in the [DEMANDS] section and include multiple demand categories per junction.

Example

```
[JUNCTIONS]
;ID Elev. Demand Pattern
;------------------------------
J1 100 50 Pat1
J2 120 10 ;Uses default demand pattern
J3 115 ;No demand at this junction
```

3.1.3 [RESERVOIRS]

Purpose

Defines all reservoir nodes contained in the network.

Format

One line for each junction containing:

- ID label
- Head, ft (m)
- Head pattern ID (optional)

Remarks

1. Head is the hydraulic head (elevation + pressure head) of water in the reservoir.
2. A head pattern can be used to make the reservoir head vary with time.
3. At least one reservoir or tank must be contained in the network.

Example

```
[RESERVOIRS]
;ID Head Pattern
;---------------------
R1 512 ;Head stays constant
R2 120 Pat1 ;Head varies with time
```

3.1.4 [TANKS]

Purpose

Defines all tank nodes contained in the network

Format

One line for each junction containing:

- ID label
- Bottom elevation, ft (m)
- Initial water level, ft (m)
- Minimum water level, ft (m)
- Maximum water level, ft (m)
- Nominal diameter, ft (m)
- Minimum volume, cubic ft (cubic meters)
- Volume curve ID (optional)

Remarks

1. Water surface elevation equals bottom elevation plus water level.
2. Non-cylindrical tanks can be modeled by specifying a curve of volume versus water depth in the [CURVES] section.
3. If a volume curve is supplied the diameter value can be any non-zero number
4. Minimum volume (tank volume at minimum water level) can be zero for a cylindrical tank or if a volume curve is supplied.
5. A network must contain at least one tank or reservoir.

Example

```
[TANKS]
;ID Elev. InitLvl MinLvl MaxLvl Diam MinVol VolCurve
;-----------------------------------------------------------
;Cylindrical tank
T1 100 15 5 25 120 0
;Non-cylindrical tank with arbitrary diameter
T2 100 15 5 25 1 0 VC1
```

3.1.5 [PIPES]

Purpose

Defines all pipe links contained in the network.

Format

One line for each junction containing:

- ID label of pipe
- ID of start node
- ID of end node
- Length, ft (m)
- Diameter, inches (mm)
- Roughness coefficient
- Minor loss coefficient
- Status (OPEN, CLOSED, or CV)

Remarks

1. Roughness coefficient is unitless for the Hazen-Williams and Chezy-Manning head loss formulas and has units of millifeet (mm) for the Darcy-Weisbach formula. Choice of head loss formula is supplied in the [OPTIONS] section.

2. Setting status to CV means that the pipe contains a check valve restricting flow to one direction.

3. If minor loss coefficient is 0 and pipe is OPEN then these two items can be dropped form the input line.

Example

```
[PIPES]
;ID Node1 Node2 Length Diam. Roughness Mloss Status
;-------------------------------------------------------------
P1  J1    J2    1200   12    120       0.2   OPEN
P2  J3    J2     600    6    110       0     CV
P3  J1    J10   1000   12    120
```

3.1.6 [PUMPS]

Purpose

Defines all pump links contained in the network.

Format

One line for each junction containing:

- ID label of pump
- ID of start node
- ID of end node
- Keyword and Value (can be repeated)

Remarks

1. Keywords consists of:

   - POWER power value for constant energy pump, hp (kW)
   - HEAD ID of curve that describes head versus flow for the pump
   - SPEED relative speed setting (normal speed is 1.0, 0 means pump is off)

Valve	Type	Setting
PRV	pressure reducing valve	Pressure, psi (m)
PSV	pressure sustaining valve	Pressure, psi (m)
PBV	pressure breaker valve	Pressure, psi (m)
FCV	flow control valve	Flow (flow units)
TCV	throttle control valve	Loss Coefficient
GPV	general purpose valve	ID of head loss curve

Table 3.2: Valve types and settings

  - PATTERN ID of time pattern that describes how speed setting varies with time

2. Either POWER or HEAD must be supplied for each pump. The other keywords are optional.

Example

```
[PUMPS]
;ID   Node1 Node2 Properties
;---------------------------------------------
Pump1 N12   N32   HEAD Curve1
Pump2 N121  N55   HEAD Curve1   SPEED 1.2
Pump3 N22   N23   POWER 100
```

3.1.7 [VALVES]

Purpose

Defines all control valve links contained in the network.

Format

One line for each junction containing:

- ID label of valve
- ID of start node
- ID of end node
- Diameter, inches (mm)
- Valve type
- Valve setting
- Minor loss coefficient

Remarks

1. Valve types and settings see Table 3.2
2. Shutoff valves and check valves are considered to be part of a pipe, not a separate control valve component (see [PIPES])

3.1.8 [EMITTERS]

Purpose

Format

One line for each junction containing:

-

3.2 System Operation

... todo

3.3 Water Quality

... todo

3.4 Options and Reporting

... todo

3.5 Network Map/Tags

... todo

Chapter 4

Binary Output File Format

If a third file name is supplied to the command line that runs EPANET then the results for all parameters for all nodes and links for all reporting time periods will be saved to this file in a special binary format. This file can be used for special postprocessing purposes. Data written to the file are 4-byte integers, 4-byte floats, or fixed-size strings whose size is a multiple of 4 bytes. This allows the file to be divided conveniently into 4-byte records. The file consists of four sections of the following sizes in bytes:

Section	Size in bytes
Prolog	$852 + 20 \cdot Nnodes + 36 \cdot Nlinks + 8 \cdot Ntanks$
Energy Use	$28 \cdot Npumps + 4$
Extended Period	$(16 \cdot Nnodes + 32 \cdot Nlinks) * Nperiods$
Epilog	28

$Nnodes$	number of nodes (junctions + reservoirs + tanks)
$Nlinks$	number of links (pipes + pumps + valves)
$Ntanks$	number of tanks and reservoirs
$Npumps$	number of pumps
$Nperiods$	number of reporting periods

All of these counts are themselves written to the file's Prolog or Epilog sections.

4.1 Prolog

Item	Type	Number of Bytes
Magic Number (= 516114521)	Integer	4
Version	Integer	4
Number of Nodes (Junctions + Reservoirs + Tanks)	Integer	4
Number of Reservoirs & Tanks	Integer	4
Number of Links (Pipes + Pumps + Valves)	Integer	4
Number of Pumps	Integer	4
Number of Valves	Integer	4
Water Quality Option	Integer	4
Index of Node for Source Tracing	Integer	4
Flow Units Option	Integer	4
Pressure Units Option	Integer	4
Statistics Flag	Integer	4
Reporting Start Time (seconds)	Integer	4
Reporting Time Step (seconds)	Integer	4
Simulation Duration (seconds)	Integer	4
Problem Title (1st line)	Char	80
Problem Title (2nd line)	Char	80
Problem Title (3rd line)	Char	80
Name of Input File	Char	260
Name of Report File	Char	260
Name of Chemical	Char	32
Chemical Concentration Units	Char	32
ID Label of Each Node	Char	$32 \cdot Nnodes$
ID Label of Each Link	Char	$32 \cdot Nlinks$
Index of Start Node of Each Link	Integer	$4 \cdot Nlinks$
Index of End Node of Each Link	Integer	$4 \cdot Nlinks$
Type Code of Each Link	Integer	$4 \cdot Nlinks$
Node Index of Each Tank	Integer	$4 \cdot Ntanks$
Cross-Sectional Area of Each Tank	Float	$4 \cdot Ntanks$
Elevation of Each Node	Float	$4 \cdot Nnodes$
Length of Each Link	Float	$4 \cdot Nlinks$
Diameter of Each Link	Float	$4 \cdot Nlinks$

Appendix A

Example input file

```
[TITLE]
EPANET TUTORIAL
[JUNCTIONS]
;ID Elev Demand
;------------------
2     0    0
3   710  650
4   700  150
5   695  200
6   700  150
[RESERVOIRS]
;ID Head
;---------
1   700
[TANKS]
;ID Elev InitLvl MinLvl MaxLvl Diam Volume
;-----------------------------------------------
7   850  5       0      15     70   0
[PIPES]
;ID Node1 Node2 Length Diam Roughness
;-----------------------------------------
1   2     3     3000   12   100
2   3     6     5000   12   100
3   3     4     5000    8   100
4   4     5     5000    8   100
5   5     6     5000    8   100
6   6     7     7000   10   100
[PUMPS]
;ID Node1 Node2 Parameters
;---------------------------------
```

```
7   1     2     HEAD 1
[PATTERNS]
;ID Multipliers
;-----------------------
1 0.5 1.3 1 1.2
[CURVES]
;ID X-Value Y-Value
;--------------------
1   1000    200
[QUALITY]
;Node InitQual
;-------------
1 1
[REACTIONS]
Global Bulk -1
Global Wall 0
[TIMES]
Duration 24:00
Hydraulic Timestep 1:00
Quality Timestep 0:05
Pattern Timestep 6:00
[REPORT]
Page 55
Energy Yes
Nodes All
Links All
[OPTIONS]
Units GPM
Headloss H-W
Pattern 1
Quality Chlorine mg/L
Tolerance 0.01
[END]
```

Appendix B

Error Messages

101	An analysis was terminated due to insufficient memory available.
110	An analysis was terminated because the network hydraulic equations could not be solved. Check for portions of the network not having any physical links back to a tank or reservoir or for unreasonable values for network input data.
200	One or more errors were detected in the input data. The nature of the error will be described by the 200-series error messages listed below.
201	There is a syntax error in a line of the input file created from your network data. This is most likely to have occurred in .INP text created by a user outside of EPANET.
202	An illegal numeric value was assigned to a property.
203	An object refers to undefined node.
204	An object refers to an undefined link.
205	An object refers to an undefined time pattern.
206	An object refers to an undefined curve.
207	An attempt is made to control a check valve. Once a pipe is assigned a Check Valve status with the Property Editor, its status cannot be changed by either simple or rule-based controls.
208	Reference was made to an undefined node. This could occur in a control statement for example.
209	An illegal value was assigned to a node property.
210	Reference was made to an undefined link. This could occur in a control statement for example.
211	An illegal value was assigned to a link property.

212 A source tracing analysis refers to an undefined trace node.

213 An analysis option has an illegal value (an example would be a negative time step value).

214 There are too many characters in a line read from an input file. The lines in the .INP file are limited to 255 characters.

215 Two or more nodes or links share the same ID label.

216 Energy data were supplied for an undefined pump.

217 Invalid energy data were supplied for a pump.

219 A valve is illegally connected to a reservoir or tank. A PRV, PSV or FCV cannot be directly connected to a reservoir or tank. Use a length of pipe to separate the two.

220 A valve is illegally connected to another valve. PRVs cannot share the same downstream node or be linked in series, PSVs cannot share the same upstream node or be linked in series, and a PSV cannot be directly connected to the downstream node of a PRV.

221 A rule-based control contains a misplaced clause.

223 There are not enough nodes in the network to analyze. A valid network must contain at least one tank/reservoir and one junction node.

224 There is not at least one tank or reservoir in the network.

225 Invalid lower/upper levels were specified for a tank (e.g., the lower lever is higher than the upper level).

226 No pump curve or power rating was supplied for a pump. A pump must either be assigned a curve ID in its Pump Curve property or a power rating in its Power property. If both properties are assigned then the Pump Curve is used.

227 A pump has an invalid pump curve. A valid pump curve must have decreasing head with increasing flow.

230 A curve has non-increasing X-values.

233 A node is not connected to any links.

302 The system cannot open the temporary input file. Make sure that the EPANET Temporary Folder selected has write privileges assigned to it (see Section 4.9).

303 The system cannot open the status report file. See Error 302.

304 The system cannot open the binary output file. See Error 302.

308 Could not save results to file. This can occur if the disk becomes full.

309	Could not write results to report file. This can occur if the disk becomes full.