

Software Engineering Issues in the Design of an Upwardly-Complex Water Network Analysis Program

Lyes Khezzar, Saad Harous and Mohamed Benayoune.

Building a computer program for the steady state simulation of water distribution networks using state-of-the-art techniques and graphical user interfaces (GUI) involves the interaction of several disciplines and mobilization of appropriate resources. This chapter reports on the experience of building such software. The mathematical model is described and the use of graph theory tools in the description of networks together with an algorithm for the treatment of Pressure Reducing Valves have been highlighted. The GUI organization and CASE tools used are described. During testing of the program, lack of benchmark data has been recognized together with the major sources of uncertainties and it is imperative that benchmarks should be developed. The factors that influence quality assurance during the crucial phase of software development have also been identified and discussed in the light of the present experience. Although similar commercial packages do exist, this software will be used as a platform for future extensions to include: extended period simulation; explicit determination of network parameters; and the capability to simulate unsteady conditions.

4.1 Introduction

Proper design of reliable water distribution networks calls for a number of steady state simulations under conceivably critical conditions. This is usually carried out using a network solver program that determines the pressure and flow distributions

Khezzar, L., S. Harous and M. Benayoune. 2000. "Software Engineering Issues in the Design of an Upwardly-Complex Water Network Analysis Program." *Journal of Water Management Modeling* R206-04. doi: 10.14796/JWMM.R206-04.

© CHI 2000 www.chijournal.org ISSN: 2292-6062 (Formerly in Applied Modeling of Urban Water Systems. ISBN: 0-9683681-3-1)

in the network. Computer programs for the simulation of water distribution networks have their origin in the 1960s when digital computers started to become available (Martin and Peters, 1963; McCormick and Bellamy, 1968; Shamir and Howard, 1968). Widespread industrial use and the appearance of commercial codes did not start until the mid 1970s. Since then and especially in the late 80s, when the next significant change to simulation tools came about, several packages with varying degrees of capability emerged such as KYPIPES, WATER, WADISCO, WATSIM, EPANET, SIMNET, WATERCAD, GINAS and FLOWMASTER. Most, if not all, evolved from calculation programs developed initially at universities. Commercial packages are usually used as black boxes where input information is fed in for an engine code to perform the calculations and return a solution. In this process, the user has obviously limited knowledge of the details of the calculation procedure and no intervention is possible. Although interpretation of results of codes in practice is rarely influenced or helped by detailed knowledge of how the algorithm and code were constructed, adjustment in procedures is often required but is seldom available with commercial products.

The Ministry of Electricity and Water in Oman has the constant policy to improve the water supply conditions and thus maintain water quality with cost effective operation, maintenance and extension of present networks. This can only be achieved if a good understanding of water distribution systems modeling and analysis is cultivated. This will subsequently lead to developing in-house expertise that will serve the long term needs of the Ministry of Electricity and Water. For these reasons, the Ministry decided to build, in collaboration with Sultan Qaboos University, its own software for steady state analysis of water distribution networks.

It is believed the objective of this endeavour would eventually induce know-how acquisition and enhance expertise not only in network analysis but also in design. It should also provide impetus for technology diffusion for the management of water resources, which are scarce and costly in this part of the world. More importantly, the software will form a basis for future development and research since the source code will be available. Areas that will serve as a basis for extension of the software capabilities are explicit calculation of network parameters (network design), extended period simulation, unsteady flow simulation and dispersion of contaminants and chlorine.

Building a computer program with state-of-the-art techniques for calculation and visualization involves the interaction of several related disciplines. Conservation laws and equations that link pressure drop in pipes to flow rates and pipe friction-factor are provided by fluid mechanics theory. The information on the geometry and topology of the network is usually constructed and stored in memory using graph theory. The final non-linear set of equations that is obtained is solved numerically by using an appropriate and efficient algorithm. Input and

output information is best dealt with by a graphical user interface and, accepting the fact that MS-Windows is becoming the de facto standard, the required GUI will involve programming under MS-Windows. It is therefore evident that the task of building such software commands the mobilization of appropriate resources and expertise (Fowler, 1985; Lansey and Mays, 1989).

The purpose of this chapter is to relate an experience of building such software. Thus some of the lessons learned during the development of the Oman Water Networks (OWN) package will be shared. One lesson teaches engineers involved in the development of such packages how to take advantage of today's modern computer aided software engineering tools. The general strategy adopted is therefore presented and the difficulties encountered are also mentioned so that false routes will be avoided by future developers.

In developing a viable fully integrated network solver package, quality assurance of the model and software cannot be overlooked and must therefore be given serious consideration. The term quality is used here to designate accuracy or, since numerical algorithms are inherently subjected to errors, the level of achievable accuracy for any given application. The sources of uncertainty that can influence the quality of a software package can be mainly traced to the mathematical model, software development/assembly and user input. These issues will be addressed and discussed and their implications highlighted.

The next section reviews the theoretical background in modeling networks, which includes the graph theoretic concepts, hydraulic correlations and numerical analysis algorithms. Section 4.3 discusses the graphical user interface and computer aided software engineering (CASE) tools used for software development. Results of benchmarking are given in sections 4.4. Section 4.5 summarizes the conclusions.

4.2 Mathematical Model and Analysis

Steady state analysis of water distribution networks is based on the use of two conservation principles: flow continuity at junction nodes and energy conservation around loops. These laws can be applied in different ways (Ozsiadcz, 1987; Lansey and Mays, 1989), to provide loop or nodal based formulations. With respect to quality assurance, great advantage will be derived if emphasis is placed on the choice of reliable and robust models and fast numerical solvers coupled with efficient programming. In this regard, the most reliable methods (Wood and Funk, 1992; Nielsen, 1989; Wood and Rayes, 1981) are the linear theory method of Wood and Charles (1972) and the Newton loop method. The Newton loop method involves the solution of a smaller number of equations, but the discrepancy reduces as the network increases in size. The other methods based on nodal and Hardy-Cross formulations have been shown to exhibit significant

convergence problems in severe, albeit, real situations that include short and zero flow pipes and the frequency of problems increases as larger size problems are considered.

A water distribution network is made up of a collection of pipes (branches) joined together at nodes. We can distinguish between fixed grade or reservoir/source nodes where a constant energy grade is maintained and demand nodes where a fixed consumption is specified. The following useful general relationship holds between the number of fixed grade nodes n_{fg} , fixed demand nodes n_d , fundamental loops n_L , and number of pipes n_p :

$$n_p = n_d + n_L + n_{fg} - 1 \quad (4.1)$$

Water distribution networks are best represented by a directed graph, which in a computer can be described by an adjacency list using two vectors as shown in Ozsiadacz (1987). This representation is efficient (Christofides, 1975; Thulasiraman and Swamy, 1992), and the two vectors contain all the information on the connectivity of the graph.

Two basic matrices are needed to describe structural properties of graphs and in the formulation of equations. The fundamental loop incidence matrix, $[B^L]$ and the node-branch incidence matrix, $[A]$. The former is constructed automatically by using a fundamental loop finding algorithm such as that of Travers (1967) which generates the least number of interconnections thus generating a sparse matrix and works even if the graph comprises several disconnected graphs. The second part of this matrix, $[B^{PL}]$, is built using the shortest path algorithm of Dijkstra (1959) to find the path between fixed grade nodes. Matrix $[A]$ can be constructed from the adjacency lists. Prim's (1957) algorithm constructs the minimum resistance-spanning tree. This tree is used to generate an initial balanced flow distribution, which is needed in the Newton loop algorithm as applied by Epp and Fowler (1970) and can also be used to advantage in the linear theory algorithm. However, to find component trees of a disconnected graph as will be necessary in the treatment of pressure reducing valves, Kruskal's (1956) algorithm is used. All of the above searching algorithms have been programmed to reduce to a minimum the effort required for the preparation of model data and initial conditions required for starting the iterative solution process.

The n_d linear equations expressing flow continuity at each node can be written:

$$[A] \vec{q}_p = \vec{q}_l \quad (4.2)$$

where:

$$\begin{aligned} \vec{q}_l &= \text{the demand flow vector, and} \\ \vec{q}_p &= \text{the flow vector in all pipes.} \end{aligned}$$

Energy conservation around real and pseudo loops is expressed in the following manner by $n_L + n_{fg} - 1$ non linear equations:

$$\begin{bmatrix} B^L \\ B^{PL} \end{bmatrix} \overrightarrow{\nabla p} = \begin{bmatrix} 0 \\ \overrightarrow{\nabla p}_{fg} \end{bmatrix} \quad (4.3)$$

Each pipe has also a component equation that relates pressure drop to flow rate; thus:

$$\nabla p = [R(q_p)]q_p \quad (4.4)$$

The particular form of $[R(q_p)]$ will depend upon the chosen pressure drop relation, i.e. Hazen-Williams:

$$\Delta P_p = \frac{104.72 \times 10^3 L_p}{C_{HW}^{1.85185} D_p^{4.87}} q_p^{1.85185} \quad (4.5)$$

or Darcy:

$$\Delta P_p = 4.242 C_{fp} \rho \frac{L_p}{D_p^5} q_p^2 \quad (4.6)$$

where:

L_p	=	pipe length
D_p	=	pipe diameter
C_{HW}	=	Hazen-Williams
C_{fp}	=	pipe friction coefficient, and
ρ	=	fluid density.

If the Darcy relation is used, implicit (Colebrook, 1939) or explicit relations (Swamee and Jain, 1976; or Zigrang and Sylvester, 1982) for the friction factor can be used. Tests on networks have shown that explicit formulations lead to faster execution times.

If valves, minor loss components and pumps are associated with the pipe, equation (4.4) is modified accordingly to take into account the additional elements.

Replacing equation (4.4) into equations (4.3) we obtain a set of equations in terms of unknown flow rates:

$$\begin{bmatrix} B^L \\ B^{PL} \end{bmatrix} [R(q_p)] \overrightarrow{q}_p = \begin{bmatrix} 0 \\ \overrightarrow{\nabla p}_{fg} \end{bmatrix} \quad (4.7)$$

The Newton-Loop algorithm (Osiadacz, 1987; Jeppson, 1976), seeks a flow solution vector to the system of equations (4.7) by solving iteratively using the Newton-Raphson algorithm for a loop flow correction vector. Convergence

is assumed if the sum of the pressure drops around the loop attains a specified tolerance.

If equation (4.2) is added to equations (4.7) we obtain a mixed system of linear and non-linear equations thus:

$$\begin{cases} [A]\vec{q}_p = \vec{q}_l \\ \begin{bmatrix} B^L \\ B^{PL} \end{bmatrix} [R(q_p)]\vec{q}_p = \begin{bmatrix} 0 \\ \nabla p^{fg} \end{bmatrix} \end{cases} \quad (4.8)$$

In the linear theory method, equations (4.8) are linearized and the whole system is solved by Crout's LU decomposition (Press et al., 1989) algorithm iteratively until convergence. To start Newton's iterative algorithm an initial balanced set of flows is required, however the linear theory method obviates this need.

Inclusion of pressure-reducing valves needs special treatment since they may assume three different modes of operation (closed, open, active). These components are complex to incorporate in a calculation routine. Their detailed treatment is outside the scope of the present chapter, however an algorithm that is relatively easy to implement in conjunction with the linear theory method is being developed and implemented and can be summarized as follows:

1. Start the calculation loop with the valve initially open. From the initial graph of the network, determine the incidence matrix (continuity equations), this will remain the same for all subsequent steps. Determine also the real and pseudo-loops for the initial network (energy equations).
2. After each iteration, check for status of valves and modify the graph structure accordingly. Then identify the tree components, their datum nodes and number of fixed grade nodes that result from the new connectivity of the graph.
3. If more than one tree exists, find the pseudo-loops between the fixed grade-nodes and source for each component tree.
4. Find the loops for the whole graph, modify the energy equations and solve.

The key issue is to recognize that if the graph becomes disconnected, when a PRV is active and may contain two or more sub-graphs, the continuity equations are unaffected throughout and it is then necessary to identify the sub-trees of the original graph with its fixed grade nodes.

Finally, when considering inputs to the above models, three factors may influence the accuracy of the solution and thus deserve particular mention. Pipe friction is usually estimated and can be a serious source of uncertainty in any simulation. It can however be reduced with proper calibration. This implies that programs need to include calibration procedures. It is also usually the case that

consumption or loads are at best estimated, with consequences for prediction accuracy of the network behaviour that are sometimes not fully appreciated. Finally, convergence tests are provided in programs, but users are usually tempted to reduce their stringency with the aim to save on computing time without full knowledge of the consequence on accuracy.

4.3 Graphical User Interface

4.3.1 The User Interface and OWN System

In the past, engineers, who have been involved in software development, have tended to concentrate more on the numerical code which dealt with the engineering calculations rather than the ease of use of the final package (Fowler, 1985; McCormick & Bellamy, 1968). The tools required to develop good user interfaces were not available in the 70s and 80s when many water network analysis programs originated. The X-Window tools that were made available in the late eighties and early nineties had steep learning curves. The dramatic increase in the power of the desktop personal computers and the steep decrease in their cost coupled with a wide acceptance of MS-Windows as a *de facto* GUI industry standard has brought the development of sophisticated user-friendly interfaces within the reach of most engineers involved in the development of computer-based design and analysis tools (Snell, 1995).

The question of upward migration has received relatively little attention, in developing design systems. Developers were mainly concerned with delivering bug-free software to their customers. However, given the rate of change of the functionality in hardware and software tools, upward migration has become a major issue that needs to be addressed.

The OWN system is a totally graphically based environment oriented towards the transfer of information along the design path rather than towards individual application packages. Although OWN is powerful, a major objective has been to keep the user-interface simple and easy to learn. An application that is too difficult to navigate or understand would require high support, maintenance and training costs. More specifically, the identification of primary loops, pseudo-loops and the generation of an initial flow vector for starting the solution can be a tedious and time consuming task. All of these are now handled automatically by the program, which also has the ability to generate a solution for partially independent sub-networks.

OWN is based on MS-Windows using Visual Basic as the programming language. Using the multitasking MS-Windows-based operating environment makes the interface familiar to new users, on the basis that anyone familiar with MS Windows can use any Windows application with minimum effort. There are

obvious benefits to this approach, both to the programmer and to the user, since little programming time is spent on the overall “look-and-feel” of the program and thus the programmers can concentrate on application functionality. Armed with this perspective, the OWN interface has been designed to place the user’s focus on the objects and tasks he needs to operate on rather than on the applications.

The OWN interface is divided into three classification windows and menus: (i) primary window and container menu, (ii) workspace windows and (iii) active object windows. Each of these enhances the usability of the interface by defining the appropriate user interface as each user activates or deactivates an object.

4.3.2 The Main or Primary Window

As indicated earlier, the OWN interface comprises several forms (or windows). The Main window gives the user access to the global commands of the system such as starting a new network, opening an existing one, saving and closing networks. It is displayed at all times regardless of which object is active.

The Main window is a multiple document interface (MDI) type window with all the other windows as its children. This feature allows the system to manipulate several networks at the same time, which means that the user can have several versions of the same network open, or two or more networks open at the same time.

The Main window provides seven menus to cover all the various tasks supported by the system. These include:

- File: to open, save and close a network
- Edit: for the various editing tasks such as cut, copy and paste.
- View: for the various viewing options, such as hiding and viewing the toolbar, or hiding the results.
- Calculations: for the various calculation options.
- Results: for displaying and printing the results.
- Windows: to view or organize the windows currently open.
- Help: for the various help options.

4.3.3 The Draw Window

The next most important window is the Draw window, which is a child of the Main window. As its name indicates, this window is used to draw the network. Figures 4.1 and 4.2 show the Main window, and the Draw window open.

The Draw window, which is opened for every network, allows the user to draw the topology of the network by simply using the mouse and clicking on the item to be drawn from the menu of hydraulic elements then clicking on the draw area where it is to be placed. A pipe can be drawn between any two items simply

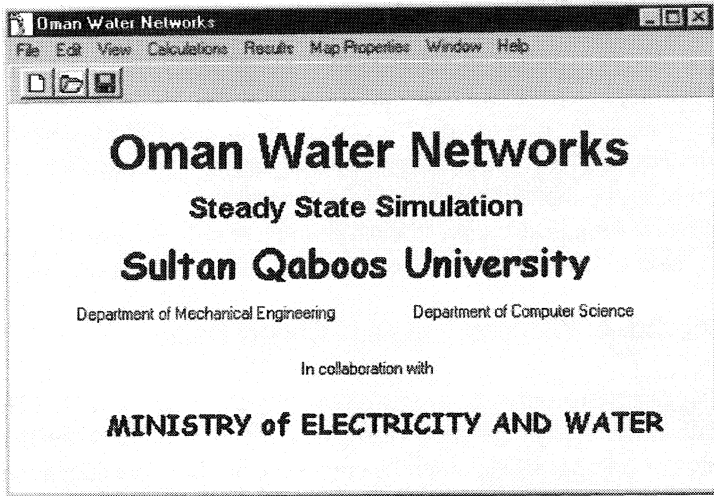


Figure 4.1 Main multiple document interface window.

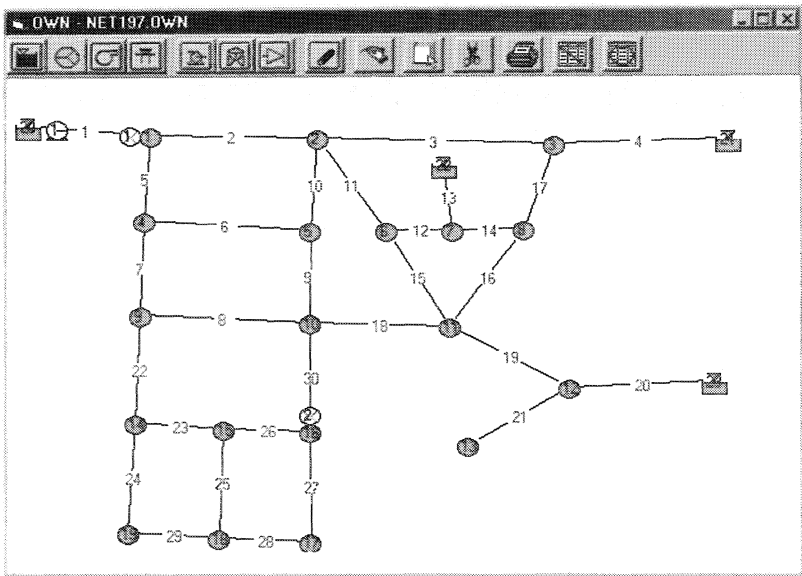


Figure 4.2 The draw window with the Network of Boulos & Wood (1990).

by clicking on the pipe button then on the start and end nodes. Elements have attributes associated with them, moving and drawing them is accomplished by sending a Draw message that provides a different implementation for different object classes.

An edit button (between the Pipe and the Print buttons on the toolbar) is available for the user to toggle between the draw mode and data entry mode. In draw mode the user can add items or pipes to the network. Co-ordinates of the elements are displayed on the screen and are rounded up to the nearest grid point. In data entry mode the user can double click on an item to bring up the data entry form to enter or modify its parameters. The user can also double click on the pipe number to bring up the data entry form for the pipe.

The Draw form is also used to display some results, such as flow rates and pressures, next to the nodes and pipes once the calculations have been done.

4.3.4 The Object Windows, Pipe and Control Data

The package supports only one general data entry form for the nodes. However, during data entry, the system shows only the relevant data items for the type of item being considered; all the other data items are hidden. The data items shown are re-organized to make the form look better. Figure 4.3 shows the data entry form for the node as an example. The data entry form for the pipes is shown in Figure 4.4.

Structural data				
Node ID	5	Node type	node	
Node Seq. No.	5	Zone Identifier		
Grid Coordinate X	9947.09	m	Grid Coordinate Y	6095.238
Variable data				
Elevation (m)	0	Demand (l/s)	0.03469	
Minimum head loss (m)				

DK

Figure 4.3 Data entry form for a node.

When all the relevant data has been entered, the user can run the package by pressing the calculation button or choosing the calculate option from the pull-down menu. In either case the package brings up a control data entry form through which the user can enter the information necessary to calculate all the flow rates, pressures etc. in the network.

The image shows a software dialog box titled "Pipe Data". It is divided into two sections: "Structural data" and "Variable data".

Structural data:

- Pipe ID: 5
- Start node: 6
- End node: 5

Variable data:

- Length (m): 853.4
- Roughness size (m): 120
- Internal diameter (m): 0.203
- Local Pressure loss (Pa): (empty field)

At the bottom center of the dialog box is a button labeled "OK".

Figure 4.4 Data entry form for a pipe.

4.3.5 Viewing Results

Once the calculations are performed, the results can be viewed in three formats: on the Draw area, either as bars or numbers, on a separate graph or on a table form. A toggle button permits to switch between either flow rates or pressures or both. Care has been taken to reduce further processing of the results.

4.4 Development and Testing

During software development, testing remains the most difficult and time-consuming aspect. During the development process, which usually takes a long time, it is important to choose a programming language adequate to the task. It is recognized that to attain the acceptable quality standards, a flexible modular software architecture is needed (Meyer, 1988). The notion of modularity implies that software modules are relatively autonomous and organized in robust architectures. Equally, particular attention has to be paid to elements such as self-describing data structures, internal error checking and diagnostics and dynamic memory handling. Dynamic memory allocation is used to mean objects are created only when needed. It is also imperative that the development process be carefully planned and monitored and properly documented for future reference and error tracking if need be. Documentation is an important aspect if during development members of the team become unavailable.

The testing process should evolve in two steps. First, individual components and routines are tested for accuracy and robustness preferably in an automated manner. In the second phase, which is complex and challenging, the combination of code features is tested. In this phase, the end user, which in our

case was the Water Department at the Ministry of Electricity and Water, needs to play a definite role in testing and evaluating the code and its interface (user-friendliness).

Initial evaluation of software by developers tends to be for only a very limited number of cases. Subsequent application to different situations can be full of surprises and reveal drawbacks. In addition, it is also difficult to find any set of benchmark data that are based on real networks. This is particularly true in developing countries. Initiatives are however being taken to develop benchmark models. For example, the Water Software Systems Group in De Monfort University and the Water Operational Research Centre of Brunel University in the United Kingdom are engaged in a joint EPSRC (Engineering and Physical Sciences Research Council) research project funded by the European Union. The aim of the project is to develop benchmark models for different classes of algorithms, such as hydraulic simulation, optimal scheduling, demand prediction and state and parameter estimation. The lack of data is a real situation confronting developers as was experienced in this project, where most of the examples used were taken from previously published work in the literature. On a commercial front, the selection is often based on discussion with vendors and on demonstrations of sample simulations often idealized to highlight features of the software. In the future, the benchmarks will inevitably ease the evaluation of the computational capabilities of calculation programs.

In the testing sequence of the software package, several networks taken from the existing literature were used (Wood and Charles, 1972; Featherstone, 1983; Boulos and Wood, 1990; Chin et al., 1978). The performance of the code confirmed good and robust convergence properties for all of the above cases and the results are assembled in Table 4.1. The criterion of convergence for the linear theory method was that the norm of the vector difference between the new and old flow vectors (resulting from the previous iteration) normalized by the norm of the old flow vector be less than 0.0001. For the Newton loop method convergence was assumed when the norm of the loop and pseudo-loop flow

Table 4.1 Summary of results.

Case	pipes	nodes	pumps	valves	reservoirs	iterations	
						Newton	Linear Theory
Chin et al (1978)	74	48			2	10	12
Boulos & Wood (1990)	30	19	1	2	4	11	12
Wood & Charles (1972)	19	9			1	7	12
Featherstone (1983)	16	11			1	7	7

correction vectors is below 0.0001. Both calculations used an initial flow distribution based on the minimum resistance spanning tree and zero chord flows as described in Epp and Fowler (1970). As can be seen in Table 4.1, convergence is achieved after few iterations and the discrepancy between the results of both calculation methods was negligible.

The case of Boulos and Wood (1990) is presented here as an illustrative example, see Figure 4.2. The network represents a typical municipal water distribution network. It contains 19 nodes, 30 pipes, 4 fixed-grade nodes, two valves with one of them closed and a pump. The geometrical and parametric detail can be found in the original reference, however Table 4.2 presents the results obtained from the original reference and the present work. As can be observed there is excellent agreement between the two. In general, agreement with previously published test cases was excellent throughout.

Table 4.2 Flow rate results.

Pipe	Flow rate (L/s) Boulos & Wood (1990)	Flow rate (L/s) OWN
1	275.01	274.81
2	147.10	146.98
3	72.66	72.58
4	38.90	38.81
5	127.91	127.82
6	38.23	38.19
7	89.69	89.63
8	39.69	39.63
9	31.48	31.46
10	6.75	6.73
11	61.19	61.14
12	26.37	26.35
13	35.05	35.00
14	8.68	8.67
15	4.81	4.80
16	24.92	24.89
17	13.76	13.78
18	71.17	71.09
19	26.06	26.00
20	16.06	16.00
21	10.00	10.00
22	50.00	50.00
23	27.52	27.52
24	22.48	22.48
25	6.66	6.66
26	20.86	20.86
27	0.86	0.86
28	19.14	19.14
29	12.48	12.48
30	0	6.3414554E-07

4.5 Conclusions

Developing a software package for simulation of water distribution networks involves consideration of several aspects that have a direct influence on quality assurance:

1. The developers have the responsibility to select through pertinent and detailed literature search the best robust and reliable numerical methods and graph theoretic algorithms and embody them in a highly efficient code.
2. There is a clear mandatory need to subject the calculation programs to a wide range of stringent tests. Lack of benchmark data has been identified. Therefore it is imperative that benchmarks are developed to provide a common ground for testing for users/buyers and developers. This can perhaps be best achieved through collaboration between universities and industry.
3. The increasing use of such packages in industry by less specialized personnel means that such packages have to be produced so that operators who have only a superficial understanding of all ingredients can use them. This requires therefore development of good user interface facilities. Nowadays however undue emphasis is usually placed on the interface and ability to integrate packages and the accuracy side is often overlooked. To avoid this inclination, it is necessary to strike an acceptable balance between the performance of the solver algorithm and the interface.
4. Substantial code changes may take place during development. The best answer lies in the use of programming language that allows upward migration as alluded to above and a combination of automated testing and efficient programming.
5. It is planned to extend the code capability in the future to include extended period simulation, explicit determination of network parameters, and the capability to simulate unsteady conditions.

Acknowledgment

Financial support through a grant, jointly provided by the Ministry of Electricity and Water of Oman and Sultan Qaboos University, is gratefully acknowledged.

References

- Boulos, P. F. and Wood, D. J. 1990, *Explicit calculation of pipe network parameters*, J. of Hyd. Engineering, vol. 116, 1329-1344.
- Chin, K. K., Gay, R. K. L., Chua, S. H., Chan, C. H. and Ho, S. Y. 1978, *Solution of water networks by sparse matrix methods*, Int. J. of Numerical Methods in Engineering, vol. 12, 1261-1277.
- Christofides, N. 1975, *Graph Theory*, Academic Press, New York.
- Colebrook, C. F. 1938, *Turbulent flow in pipes with particular reference to the transition region between the smooth and rough pipes laws*, J. Inst. C. E., (1938-39), 133.
- Dijkstra, E. W. 1959, *A note on two problems in connection with graphs*, Numerische Math., vol. 1, 269-271.
- Epp, R. and Fowler, A. G. 1970, *Efficient code for steady state flows in networks*, J. of the Hydraulics Division, vol. 96 HY1, 43-56.
- Featherstone, R. E. 1983, *Computational methods in the analysis and design of closed conduit hydraulic systems*, in *Development in Hydraulic Engineering*, Edited by P. Novak, applied Science Publishers, London, 111-150.
- Fowler A. G. 1985, *Personal computers and water network simulation*, in *Hydraulics and Hydrology in the Small Computer Age*, Vol. 2, Edited by W. R. Waldrop, ASCE, 927-931.
- Jeppson R. W. 1976, *Analysis of Flow in Pipe Networks*, Ann Arbor Science, Ann Arbor, Michigan.
- Kruskal, J. B. 1956, *On the shortest spanning sub-tree of a graph and the travelling salesman problem*, Proc. Am. Math. Soc., Vol. 7, 48-50.
- Lansley K. E. and Mays L. W. 1989, *Network simulation models*, in *Reliability Analysis of Water Distribution Systems*, Edited by L. M. Mays, ASCE, Chap. 2, 11-36.
- Martin, D. W. and Peters, G. 1963, *The application of Newton's method to network analysis by digital computers*, J. of the Institute of Water Engineers, vol. 17, 115-129.
- McCormick, M. & Bellamy, C. J. 1968, *A computer program for the analysis of networks of pipes and pumps*, The Journal of the Institution of Engineers, Australia, 51-58.
- Meyer, B. 1988, *Object-Oriented Software Construction*, Prentice Hall Inc., Englewood Cliffs, N.J.
- Nielsen, H. B. 1989, *Methods for analysing pipe networks*, Journal of Hydraulic Engineering, vol. 115, 2, 139-157.
- Osiadacz, A. 1987, *Simulation and Analysis of Gas Networks*, Spon, London.
- Press, W. H., Flannery, B. P., Teukolsky, S. A. and Vetterling, W. T. 1989, *Numerical Recipes in Fortran, the Art of Scientific Computing*, Cambridge University Press, New York, N.Y.
- Prim, R. C. 1957, *Shortest connection networks and some generalizations*, Bell Sys. Tech. J., vol. 36, 1389-1401.
- Shamir, U. and Howard, C. D. 1968, *Water distribution systems analysis*, Journal of the Hydraulics Division, vol. 94, 219-234.

- Snell, M. 1995, *Visual development tools*, Computer, Vol. 28, 3, 8-10.
- Swamee, P. K., and A. K. Jain. 1976, *Explicit equations for pipe-flow problems*, Journal of the Hydraulics Division of the ASCE, 102, no. HYS.
- Travers, K. 1967, *The mesh method in gas network analysis*, Gas Journal, November 1, 167-174.
- Thulasiraman, K. and Swamy, M. N. S. 1992, *Graphs: Theory and Algorithms*, John Wiley & Sons Inc.
- Wood, D. & Charles, C. A. O. 1972, *Hydraulic network analysis using linear theory*, Journal of the Hydraulics Division, vol. 98, 1157-1170.
- Wood, D. and Funk, J. E. 1992, *Hydraulic analysis of water distribution systems-Part1: Hydraulic network equations and reliability of solutions*, in "Water Supply Systems, State of The Art and Future Trends", Ed. by Carbrera, E. and Martinez, F., Computational Mechanics Publications, Southampton, Boston, 43-67.
- Wood, D. J. and Rayes, A. G. 1981, *Reliability of algorithms for pipe network analysis*, J. Hydr. Div., ASCE, 107(HY7), 1145-1161.
- Zigrang, D. J. and Sylvester, N. D. 1982, *Explicit approximations to the solution of Colebrook's friction factor equation*, AIChE Journal, vol. 28, No 3, 514-515.