ANNALS of the ORADEA UNIVERSITY.
Fascicle of Management and Technological Engineering, Volume VI (XVI), 2007

A SOFTWARE IMPLEMENTATION OF A CALCULUS ALGORITHM OVER THE SECTIONS OF A LOOP

Ichinur OMER [a] Dumitru Ion ARSENIE [a]
Claudiu Stefan NITESCU [a] Cristina GHERGHINA [a]

[a] *"Ovidius" University of Constanta, Civil Engineering Faculty, 22 B Unirii Street, 900524*
email: ichinur.omer@univ-ovidius.ro

Keywords: loop, head loss, flowchart, graphical interface.

Abstract: This paper describes a software implementation of an algorithm that calculates flow capacities and head loss over the sections of a loop. The application is written in Java, a well known object-oriented programming language, and it may run on all platforms. It is simple to use, the data being processed quickly and the results being presented in a table format and registered in a text file on disk.

1. Theoretical Basis

We consider a loop with *N* pipes, limitated by *N* nodes. Number 1 is the node through the water enters, from the network and the final node (*N+1)* coincides with the first one (*N=1*).

The water 's volume entering into loop is equal with the water 's volume going out from nodes (the continuity flows equation):

$$Q_1 = \sum_{i=2}^{N} Q_i \qquad (1)$$

We note the flow through between the nodes number one and number two (the flow through the pipe 1-2) with (*x*) and applying the continuity equations we express the flow through the other pipes:

$$\begin{aligned} &Q_{23} = x - Q_2 \\ &Q_{34} = x - Q_2 - Q_3 \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &Q_{i,i+1} = x - \sum_{j=2}^{i} Q_j \end{aligned} \qquad (2)$$

We calculate the head loss with usuall formula, using hydraulic resistance modulus $M_{i,i+1}$:

$$h_{r_{i,i+1}} = M_{i,i+1} Q_{i,i+1}^2 = M_{i,i+1}\left(x - \sum_{j=2}^{i} Q_{j,j+1} \right)^2 \qquad (3)$$

where $M_{i,i+1}$ can be calculated with one of the following formulas:

$$M_{i,i+1} = \lambda_{i,i+1} \frac{l_{i,i+1}}{d_{i,i+1}} \frac{v_{i,i+1}^2}{2g} = \frac{8\lambda_{i,i+1}\ l_{i,i+1}}{g\pi^2 d_{i,i+1}^5} \tag{4}$$

$$M_{i,i+1} = \frac{l}{C_{i,i+1}^2 A_{i,i+1}^2 R_{i,i+1}} = \frac{64\ l_{i,i+1}}{C_{i,i+1}^2 \pi^2 d_{i,i+1}^5} \tag{5}$$

$$C_{i,i+1}^2 = \frac{1}{n_{i,i+1}} R_{i,i+1}^{\frac{1}{6}} \tag{6}$$

$$R_{i,i+1} = \frac{d_{i,i+1}}{4} \tag{7}$$

The head loss can be expressed by the function:

$$f_r(x) = Ax^2 - 2Bx + C$$

where

$$A = \sum_{i=1}^{k-1} M_{i,i+1} - \sum_{i=k}^{N} M_{i,i+1}$$
$$B = \sum_{i=2}^{k-1} a_i M_{i,i+1} - \sum_{i=k}^{N} a_i M_{i,i+1} \tag{8}$$
$$C = \sum_{i=2}^{k-1} a_i^2 M_{i,i+1} - \sum_{i=k}^{N} a_i^2 M_{i,i+1}$$

Finally we obtain the flow through the pipe 1-2 (Q_{12}) and using the relations (2) we can calculate the other flows.

Notation List:
Q – flow rate; h_r – head loss; d – diameter of pipe; v – velocity; g – gravity acceleration; A – cross section pipe; l – lengh of pipe; n – coefficient of roughness; C – Chezy coefficient; λ – Darcy coefficient; R – hydraulic radius; M – hydraulic resistance modulus.

2. Java Programming Language

The presented application is written in Java, a high-level object-oriented programming language, developed by JavaSoft, a group inside the Sun Microsystems company. The main characteristics are:

- Simplicity: it removes the so called benefits that may cause an ambiguous code (for instance, the multiple inheritance, the overloaded operators, etc);
- Robustness: it eliminates the most frequent sources of errors that may appear when writing code and it accomplishes this by removing pointers and by self-managing the memory through the instrumentality of garbage collector program that runs in background and removes those objects that aren't used anymore. A Java program that passes the compilation step will never break the system which it runs on;
- Completely object-oriented - it removes the procedural programming style;

- Very secure: it's one of the most secure programming language available, offering high level security tools like dynamic code checking, imposing strict rules for running programs on remote computers, etc;
- Portability: Java is an independent-platform programming language, namely, the same application can run without modifications on different systems like Windows, UNIX or Macintosh. This property brings hefty profits to companies that develop network application over the Internet;
- Dynamic language
- Compiled and interpreted: Java source code compiles into portable bytecodes that require an interpreter to execute them; this interpreter is called JRE (Java Runtime Environment).

3. Application flowchart

The execution flow of the application is presented by the following flowchart (fig. 1):

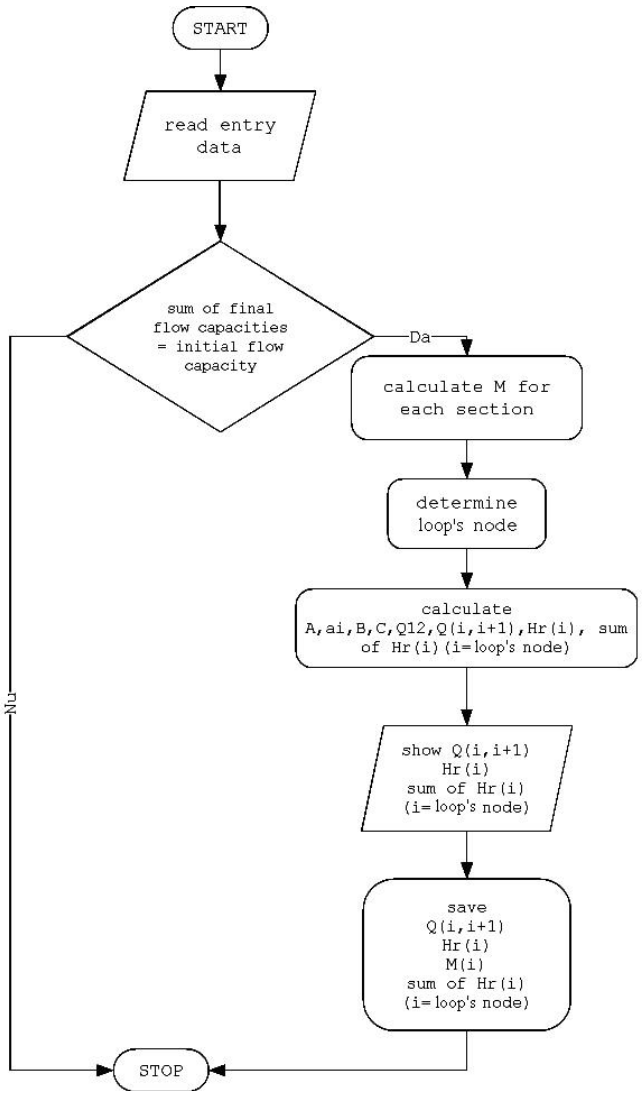


Fig.1.

4. Application Description

The application named "Loop", represents a Java implementation of the formulas presented in this paper, that calculate the flow capacities and loss of charge over the sections of a ring. Due to Java portability, the program may run on all platforms (Windows, UNIX, Macintosh), the only condition being that the Java Runtime Environment package, which may be downloaded from the official site www.sun.com (version 1.5.0 or later), has been installed on the user computer and it's working properly.

The graphical interface of the program is very friendly and simple to use in order to make the necessary steps to execute the calculus algorithm.

The application offers support for:

- Loading the entry data (number of ring's nodes, the initial flow capacity, the final flow capacities, the section's characteristics) from a text file
- Capturing the entry data from the user input (fig. 2.)

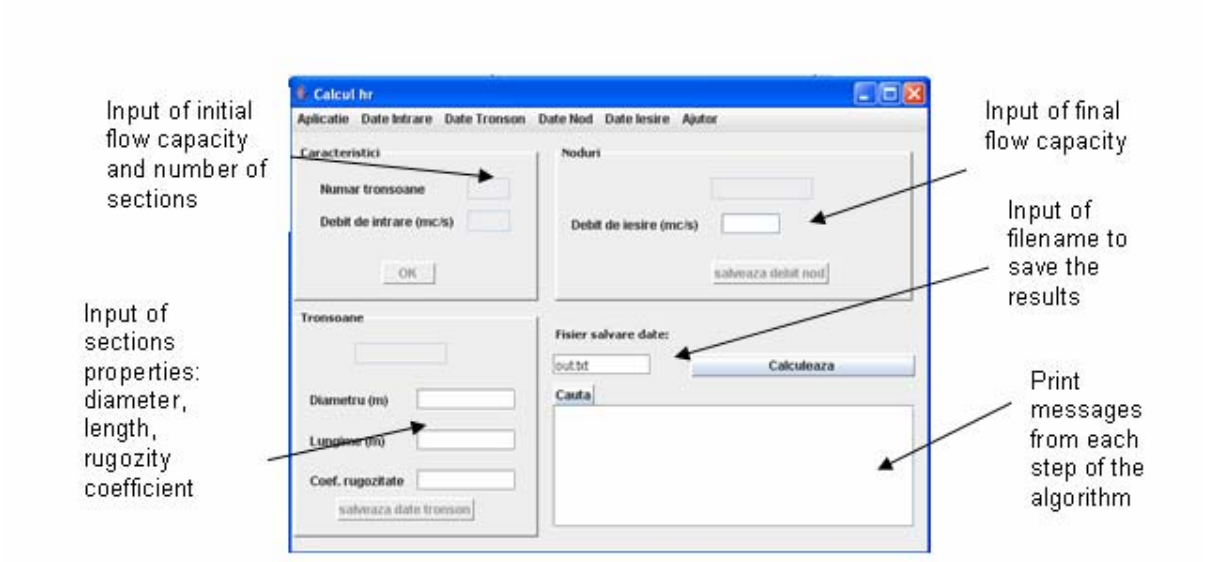
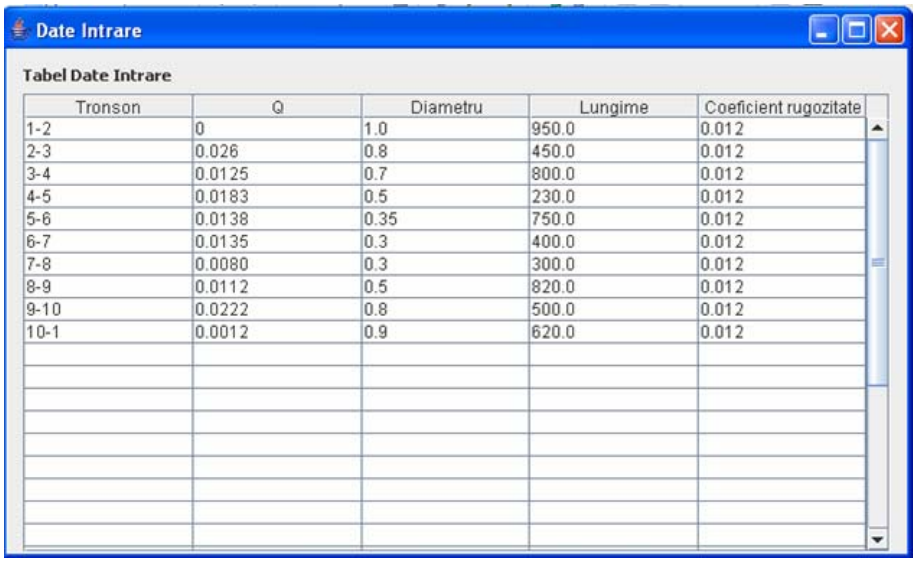


Fig.2.

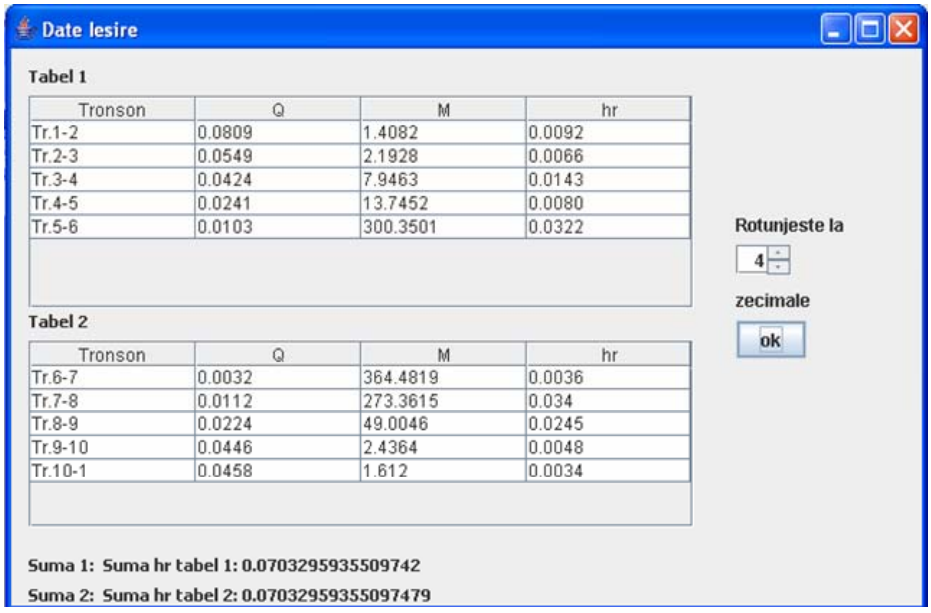
- Modifying the entry data once they are loaded from the text file or after they have been captured from the user input
- Validating the entry data and showing suggestive messages when an error is detected (for instance, when the user input is a string instead of a number)
- Showing the entry data, so that the user may check them (fig. 3.)



Tronson	Q	Diametru	Lungime	Coeficient rugozitate
1-2	0	1.0	950.0	0.012
2-3	0.026	0.8	450.0	0.012
3-4	0.0125	0.7	800.0	0.012
4-5	0.0183	0.5	230.0	0.012
5-6	0.0138	0.35	750.0	0.012
6-7	0.0135	0.3	400.0	0.012
7-8	0.0080	0.3	300.0	0.012
8-9	0.0112	0.5	820.0	0.012
9-10	0.0222	0.8	500.0	0.012
10-1	0.0012	0.9	620.0	0.012

Fig.3.

- Saving the results in a text file
- Loading the results from a text file and showing them to the user in a table format (fig. 4.)



Tronson	Q	M	hr
Tr.1-2	0.0809	1.4082	0.0092
Tr.2-3	0.0549	2.1928	0.0066
Tr.3-4	0.0424	7.9463	0.0143
Tr.4-5	0.0241	13.7452	0.0080
Tr.5-6	0.0103	300.3501	0.0322

Tronson	Q	M	hr
Tr.6-7	0.0032	364.4819	0.0036
Tr.7-8	0.0112	273.3615	0.034
Tr.8-9	0.0224	49.0046	0.0245
Tr.9-10	0.0446	2.4364	0.0048
Tr.10-1	0.0458	1.612	0.0034

Fig.4.

The application is well documented, describing each step the user has to make to apply the algorithm.

In order to install and execute the application, the user has to double click the jar archive Loop.jar.

5. Final Considerations

The application allows the user to execute quickly and efficiently a complex algorithm over a large data set, having the results calculated in a short period of time (depending of the user computer resources) and registered in a text file in a specified location.

BIBLIOGRAPHY

[1] B. Eckel, „Thinking in Java", 3rd Edition, Prentice Hall, 2002.
[2] E. Petac, T. Udrescu, „Java Fundamentals", Ed. MatrixRom, 2004.
[3] Larock Bruce, Jeppson Roland, Watters Gary, „Hydraulics of pipeline systems", CRC Press Boca Raton London; New York Washinton DC, 2000.
[4] Georgescu C. Corneliu, Georgescu I. Lucian, „Calculul si functionarea retelelor hidraulice si a electropompelor aferente", Ed. Matrix Rom, Bucuresti 2006.
[5] Mays W.Larry, „Hydraulic Design Handbook", Ed. Mc Graw-Hill, 1999.