Applied Soft Computing 18 (2014) 82–90

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Fuzzy decision support software for crisis management in gas transmission networks

Hoda Nokhbeh Foghahaayee [a,*], Mohammad Bagher Menhaj [a], Hasan Montazer Torbati [b]

[a] Department of Electrical Engineering, Amirkabir University of Technology, Iran
[b] National Iranian Gas Transmission Network Manager, Iran

ARTICLE INFO

Article history:
Received 25 May 2010
Received in revised form 8 March 2011
Accepted 18 January 2014
Available online 28 January 2014

Keywords:
Decision support system
Software development
Multicriteria decision making
Fuzzy decision making
Gas transmission network

ABSTRACT

In this work, a new software for fuzzy decision support system, with a new decision making structure, to control the crisis conditions in the gas transmission network is developed. This system after receiving both functional variables of the gas transmission network and faults signals makes right decisions to eliminate and repair the conditions of the transmission network according to its database established through experience gathered from experts. These decisions are expressed in the form of some scenarios with different desirability degrees. Desirability degrees measure the outcomes of the decisions taken for the aforementioned conditions; this in turn will assist the managers in choosing the best ones. The user interface properly developed is graphical and provides the manager a good facility to easily use the system without any hesitation at the occurrence time of crisis.

1. Literature review

Today's software technology has an important role to represent the so-called impossibly implementable ideas. Due to its demanding, software engineering has been fully expanded.

Since the 1960s, many descriptions of the classic software life cycle have appeared (e.g., Hosier 1961, Royce 1970, Boehm 1976, Distaso 1980, Scacchi 1984, Somerville 1999). Royce [9] initiated the formulation of the software life cycle using the now familiar "waterfall" chart. How to develop large software systems is still difficult because it involves complex engineering tasks that may require iteration and rework before completion.

These classic software life cycle models usually include some version or subset of the following activities [1]:

- *System initiation/planning*. where do systems come from?
- *Requirement analysis and specification*. It identifies (1) the problems that a new software system is supposed to tackle, (2) its operational capabilities and its desired performance characteristics, and (3) the resource infrastructure needed to support system operation and maintenance.
- *Functional specification or prototyping*. Here the goal is to formalize the computational objects, their attributes and relationships; it also identifies the operations transforming these objects, and the constraints restricting system behavior.
- *Partition and selection* (build vs. buy vs. reuse). In this stage, given requirements and functional specifications, the system is first divided into manageable pieces that denote logical subsystems, and then it determines whether the new, existing, or reusable software systems correspond to the required pieces.
- *Architectural design and configuration specification*. The interconnection and resource interfaces among subsystems, components and modules are properly determined so that their detailed design and overall configuration remain functional.
- *Detailed component design specification*. Here, the procedural techniques through which the data resources within the modules of a component are transformed are defined.
- *Component implementation and debugging*. The preceding specifications into operational source code implementations are encoded and their basic operation is also validated.
- *Software integration and testing*. In this part, the overall integrity of the software system is preserved; this is done by (1) verifying the consistency and completeness of implemented modules, (2) confirming the resource interfaces and interconnections against their specifications, and (3) ensuring the performance of the system and subsystems against their requirements.
- *Documentation revision and system delivery*. The descriptions of the recorded system development should be documented in a systematic way more suitable for users and system support.

* Corresponding author. Tel.: +98 21 64543347.
E-mail addresses: nokhbeh@aut.ac.ir, nokhbeh_h@yahoo.com (H. Nokhbeh Foghahaayee).

1568-4946/$ – see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2014.01.019

- *Deployment and installation*. It provides guidelines for the delivered software installation and configuring operating systems parameters.
- *Software maintenance*. It sustains full operation of a system by providing performance improvements, repairs and conversions.

To stick with the aforementioned chart becomes more necessary when we are dealing with very complex real systems such as gas networks.

As natural gas pipeline systems have grown larger and more complex, the importance of optimum operation and planning of these facilities has increased. The investment and operating costs of pipeline networks are so large that even small improvements in system utilization can involve substantial amounts of money.

The natural gas industry services include producing, moving, and selling gas. Our main interest in this paper is focused on the transmission of gas through the network.

Pipe lines in a gas network, transfer the gas refined in refineries to the various scattered consumption centers in the country. The interior consumers of gas network include cities and industrial centers especially power plants. In order to maintain the gas pressure in required amount for these centers, some pressure fortification compressor stations are placed in the pipeline route. There is a dispatching center for coordinating and controlling whole gas network performances. This center first receives some information continuously about the network situation and then determines new working conditions for each element of the network with respect to a set of predefined goals and some practical limitations of the network.

In large scales systems such as gas transmission networks, it is necessary to note that they have high dimensions and spread performance field which undoubtedly put it among large organizations. In a decision making center, after collecting information from most network nodes, necessary decisions are to be made to improve the network performance so that the defined goals are met [5].

In general, individuals use their own experiences, memory and knowledge to make decisions. Of course, sometimes it is difficult to collect all these information in a short time; therefore, an efficient decision may not be made merely based on the individual knowledge especially in crisis conditions. Furthermore, due to the stress and time shortage the responsible person may not be able to use his/her knowledge to make suitable decisions or there may be no expert person in the center to take over the situation. Consequently, the necessity of equipping each important organization with a Decision Support System (DSS) becomes justifiable.

DSS can be described as "computer-based systems that help decision makers confront ill-structured problems through direct interaction with data and analysis models" [6]. DSSs are designed to increase the speed and accuracy of data analysis, while reducing costs, enabling the effective and efficient analysis of large volumes of quantitative data. DSS are particularly valuable tools in complex situations, where decision makers need to analyze multiple sources of data [2]. Decision support systems can be designed to support decision makers in any level of an organization. These systems will be able to support management operations and strategic decisions.

In our previous paper [2], a new structure for decision making for large-scale systems was introduced. Based on this new structure, this paper presents a new software development for decision making in Iranian gas transmission network as a large scale system.

As far as the authors know, there has been no decision support system designed for gas transmission network and this software is novel.

This software is mainly based on Matlab and Labview. It uses Matlab as the decision making part and Labview software for user interface.

The rest of the paper is organized as follows.

Section 2 describes the fuzzy decision making part of the system. A gas transmission network is investigated in Section 3. Section 4 contains interface types and implemented interface between user and the software, and between Matlab and Labview. In Section 5, designing and applying the designed decision support system to gas transition network is described. Finally, Section 6 concludes the paper.

2. Decision making system

Previous paper of the authors [2] introduced a new structure for decision making based on fuzzy computation. Since a crisis begins with events, using this structure is begun with categorizing the events and making some models for each one. Suppose there are ***n*** different cases corresponding to ***n*** distinguished events each described by some necessary environmental parameters and also having its own decision model consisting of properly developed controllers that compute some possible values for one decision variable. Each event triggers the associated models. These models get their necessary information from the system's database which is consisted of two kinds of data: on-line and off-line data.

After computing all decision variables based on gathered data, all different combinations of the values are produced. However, due to some practical limitations, some of them should be chosen properly based on a mechanism [2]. Finally, they will be used for solving the event already occurred each resulting in a solution with a degree of acceptance through several scenarios produced properly. These values are calculated based on specific user criteria. There is an additional component in the system which provides the user some information about the scenarios to make superb decisions for same cases in the future and to assign higher degrees to the more effective components. Components of the system given in Fig. 1 are briefly discussed below.

The crisis classification block is indeed an associative mechanism that links an input vector **b** (network situation) to one of the prototype cases with the highest degree of association. This mechanism is mainly based on fuzzy computation.

For each case, a membership function is defined. This membership function is then used to express the degree of similarity between the current situation and the situation in the case.

The "sub-controllers-block" contains a set of sub-controllers that are developed to control each decision variable based on fuzzy computations, i.e., each sub-block is represented by a fuzzy logic controller (FLC). The number of these controllers equals the number of decision variables. These controllers can be established either by a set of fuzzy if-then rules or through possibilistic programming that is a linear programming with uncertain coefficients whose

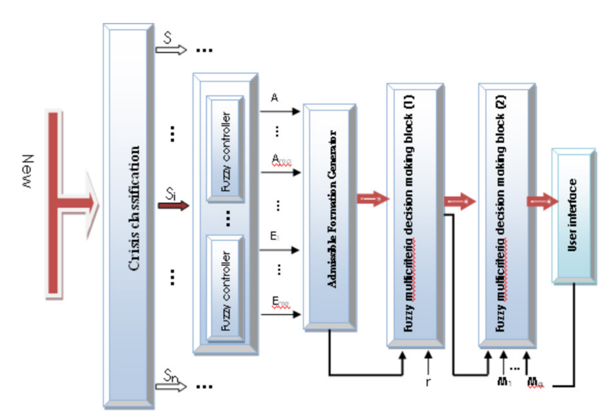

Fig. 1. Decision making structure.