Ontology for computer-aided fault tree synthesis

Allan Venceslau, Raphaela Lima, Luiz Affonso Guedes, Ivanovitch Silva
Department of Computer Engineering and Automation
Federal University of Rio Grande do Norte
CEP 59078-970 Natal/RN, Brazil
Email: allanrsv@dca.ufrn.br, raphaela@dca.ufrn.br, affonso@dca.ufrn.br, ivan@imd.ufrn.br

Abstract—**The fault tree is a technique of high importance for reliability analysis in a industrial process. This paper describes an application that was developed for the automatic fault tree synthesis. In order to do so, the knowledge about industrial processes was structured through the development of a domain ontology. With this ontology is possible to represent the relationship between each component of the system and all its failure situations. Once the knowledge is structured in an ontology, an automatic procedure is executed to create a Fault Tree. The proposal was validated with a water tank system and allowed the construction of its fault tree quickly and efficiently.**

Keywords—Fault Tree, Industrial Automation, knowledge, Ontology.

I. Introduction

The increasingly stringent economic, environmental and safety requirements on the operation of industrial processes made its activities sufficiently complex, which created demand for use of computational solutions to aid in complex functions such as alarm management, fault detection or malfunction in equipment, diagnosis of anomalies and asset management.

Failures are inevitable, but the consequences of failure, i.e. the collapse of the system, the interruption in the supply of the service and data loss can be avoided by proper use of viable techniques.

The Fault Tree Analysis (FTA) is one of the most common techniques for reliability evaluation of systems, both for qualitative and quantitative analysis [1]. The FTA is widely used to investigate the root causes of unwanted events when the system fails [2]. They are widely used in various industrial sectors primarily in the aerospace, nuclear, chemical and oil and gas industry. However, the manual construction of a fault tree is a time-consuming task which needs much effort, and is susceptible to human errors at all stages of its development [1].

Thus, in recent decades many researchers have focused on developing computational techniques to automate the construction process of Fault Trees [3], [4], [5], [1], [2]. Automated fault tree generation is faster and easier than the manual approach, and is more suitable for being used in design phase, where the designers may want to try lots of different configurations to find the most reliable one.

A important step in automating the process of construction of fault trees is how to model the system in an appropriate way for being used in algorithmic procedures. One of the challenges is to find an efficient system modeling approach, which can support modeling of different types of systems in a straightforward and easy way to understand [1].

Previous studies have used different methods for system modeling, including digraphs, decision tables, state diagrams, semantic network modeling, and other methods [6], [5], [1]. A very interesting method was developed by Majdara (2009). In his work, a component-based system modeling method approach is presented to model various features and components of an industrial process.

The aim of the present work is to create an ontology capable of modeling a wide variety of industrial processes. Once all knowledge is structured in an ontology, an automatic procedure is executed to create a Fault Tree. The classes and properties of the Ontology were created based on definitions proposed by Majdara (2009).

The ontology presented in this study has already been used for applications in the alarm management area [7] and in process economic analysis [8]. New concepts and properties were added to this ontology to allow the generation of fault trees automatically.

The remainder of the paper is organized as follows: Section 2 show the basic concepts and elements of the system model. Section 3 gives a general overview of the ontology used for semantic modeling of industrial processes and explains how the ontology was used to model industrial processes. Section 4 presents the case study. And finally, Section 5 describes the conclusions and the target for future studies.

II. System Model

The algorithm proposed by Majdara (2009) for automatic generation of fault trees uses the following concepts:

- Component: any elements of a system. This can include all the electrical, mechanical devices, human operators, etc.
- Function tables: describes the relationship between the inputs and outputs of the components and may involve a working condition or status of a particular component.
- State transition tables: used for some components for which it is necessary to map state transitions. They are used in conjunction with the Function tables to map the state changes according to inputs.

Using these concepts it is possible to model a wide variety of systems, and represent the most varied operating conditions. Every component must necessarily have a function table that describes the relationship between the inputs and outputs. However, only particular components possess a state transition table, for example, a valve can be in one of the two states: open or close.

A. Methodology for Automatic Generation of Fault Tree

The algorithm for automatic generation of Fault Tree described in [1] can be summarized as follows. It starts with the selection of the top event which is associated with one of the system components. The next step is to determine whether it refers to an output or state of the component. If it is an output, the function table is analyzed, but if it is a state, the state transition table is analyzed.

The analysis of the function table starts checking which rows of the table have the output value of interest. If there is a functionality condition of the component leading to that output value, a basic event is added to the tree. If more than one row has the value of interest, then a logic gate ”OR” is added and each rows with the output value of interest are set as its inputs. The algorithm trace-back for each component input that leads to the desired output

For each line, if more than one column is contributing to the desired output, then these conditions will be added to a logic gate ”AND”. This procedure is continued until system boundaries are reached.

When all of the gates and events are developed until the point they reach a basic event or an event that cannot be developed anymore, then the fault tree synthesis is finished.

III. Ontology for the Modeling of Industrial Processes

In this paper is presented an ontology specially developed to model the domain of industrial process in order to support the industrial automation applications with failure detection, anomaly diagnostics, and intelligent monitoring of processes. Special attention was given to the characterization of the components that make up the industrial plants and the energy flow that goes through such processes. Initially, the following concepts were determined:

- Equipment: element that makes some physical or chemical transformation in the fluids that circulate through the plant.
- Pipe: closed tube used for fluid transportation.
- Instrument: element of automation and instrumentation that can be connected to plant equipment.
- Alarm: visual and/or resonant signaling that identifies the occurrence of a problem that must be noticed quickly.
- Accessory: represents any other component that is not equipment, a pipe, an instrument or an alarm.
- RawMaterial: used to represent the possible raw material used in the industrial processes.
- Utility: used to represent the possible utilities used in the industrial processes, like vapor and water.

Based on these generic definitions the first classes of the ontology emerged, as we can see on Figure 1. In this case,

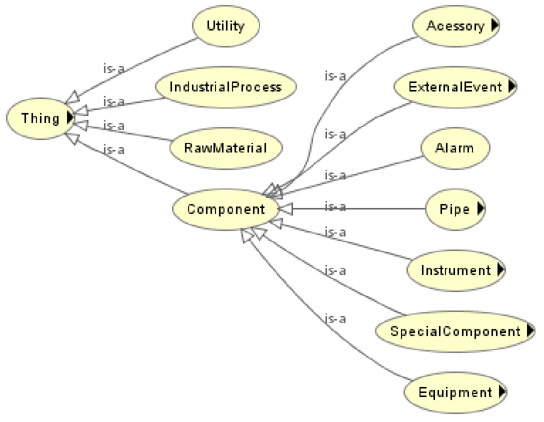

Fig. 1. First classes of the ontology.

the *Thing* class is characterized as the root class of the hierarchy. The *Instrument*, *Alarm*, *Equipment*, *Pipe*, and *Accessory* classes were grouped in subclasses of the *Component* class, as they all have some common characteristics. Following this hierarchical structure, other elements of an industrial process were defined, such as *Controller*, *Actuator* and *Sensor* as subclasses of *Instrument* and *Turbine*, *Pump* and *Tank* as subclasses the *Equipment*.

The ontology also defines which properties (characteristics) the individuals of each class would need to have. As an example of properties, we can mention the *hashTag* that stores the instrument, equipment or alarm TAG value and *hasDescription* to store a generic textual description of the associated element.

Aiming to use the ontology for automatic construction of Fault Tree, it was necessary to create new entities and properties related to the concept of the algorithm proposed by Majdara (2009), shown in previous section.

First the *ComponentState* and *FuncionalityCondition* classes were created to represent, respectively, the States and the Functionality Conditions of the components. Other classes were created to model the concepts of the function table (*FunctionTable*) and states transition table (*StateTable*). Furthermore, it was also necessary to map the items from these tables (*StateTableItem, FuncionTableItem*). The new classes can be seen in figures 2 and 3.

Finally, New properties were also created to make the construction of Fault Tree from the proposed ontology possible. Some of these properties are shown in Table I.

IV. Case Study

Based on the representation of the knowledge of the industrial process and the fault tree methodology, an applicative was developed to help with this activity. The case study was done based on a water tank system, adapted from [9].

The system consists of a tank and a tray. There are two sensors, S1 and S2, which allow us to analyze if the tank level is appropriate. In addition, these sensors send information to their respective controllers (C1 and C2), respectively. The system is shown in Figure 4.

TABLE I. SOME PROPERTIES RELATED TO THE ONTOLOGY CLASSES.

Property	Domain	Data Type	Description
hasState	Component	boolean	Indicates whether or not the component has states to be analyzed, and may take the values true or false.
isFault	Funcionality Condition	boolean	Indicates whether or not the functionality condition is a failure, and may take the values true or false.
hasFunctionalityCondition	TableItem	FunctionalityCondition	Used when creating a Functionality Condition Table to set the operating condition associated with the row of the table.
hasStateTableItem	Function TableItem	ComponentState	Used in the creation of a state transition table. Sets the associated table row state.
hasInputTable	TableItem	InputTableItem	Set to a table item the inputs associated.
hasOutputTable	TableItem	OutputTableItem	Set to a table item the outputs associated.

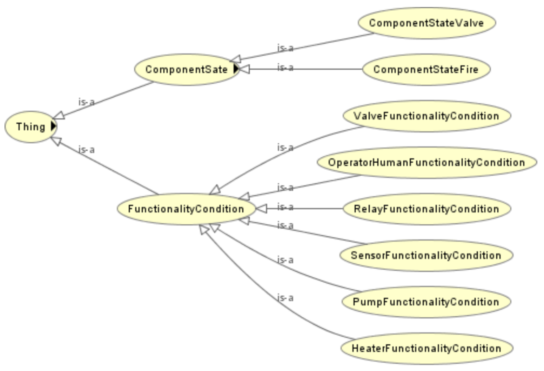
Thing
ComponentSate
ComponentStateValve
ComponentStateFire
FunctionalityCondition
ValveFunctionalityCondition
OperatorHumanFunctionalityCondition
RelayFunctionalityCondition
SensorFunctionalityCondition
PumpFunctionalityCondition
HeaterFunctionalityCondition
is-a

Fig. 2. Functionality conditions and component state.

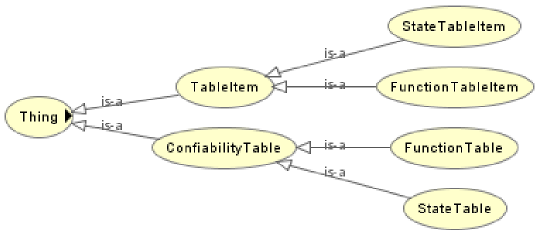

Fig. 3. Condition Table and State Transition Table.

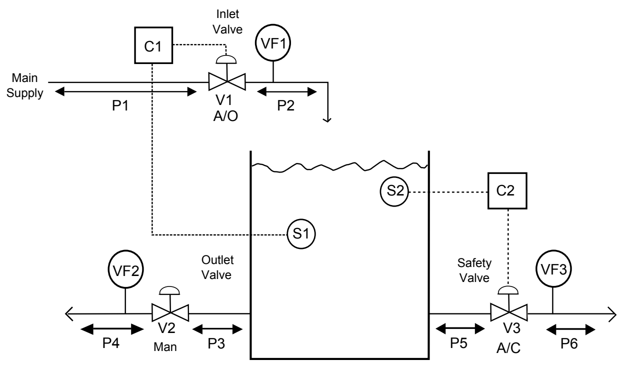

Fig. 4. Water tank system.

For the correct execution of the process, the tank level should not be less than that set by S1 and no greater than that set by S2. The C1 controller is responsible for controlling the valve (V1) while C2 is responsible for controlling the safety valve (V3). The purpose of the tray is to collect any water leaking through a fracture or overflowing from the top, this is located under the tank. It is equipped with a sensor (SP1) that indicates whether there is presence of water or not. The valve V2 is also present in the system, but this is operated manually. Pipelines numbered P1 to P6 are the means through which the liquid flows, and these may also are subject to failures, such as cracks and obstructions. Finally, there are three flow sensors VF1, VF2 and VF3.

Under normal conditions, the valve V2 is open drawing water from the tank and V1 is open to replace the water flowing from V2. The V3 valve will remain closed until the tank reaches a critical level.

The first step for the automatic fault tree construction is to create an instance of the ontology, modeling each component of the system. The behavior of each element was mapped in their Function Tables and State Transition Tables. An example of these tables are shown in tables II and III.

TABLE II. CONTROLLER 1 FUNCTION TABLE.

Input 1	Functionality Condition	Output 1
Below	ok	1
Above	ok	0
-	Failed Low	1
-	Failed High	0

TABLE III. VALVE 1 STATE TRANSITION TABLE.

Input 1	Initial State	Functionality	Final State
1	-	ok	open
0	-	ok	close
0	open	Fail to close	open
1	close	Fail to open	close

Once all the knowledge is represented on the ontology, the fault tree can be constructed. An example of automatically generated fault tree is present in Figure 5. In this case, the selected top event corresponds to the situation of water overflowing, causing the tray to receive water.

In order to facilitate the visualization, only few nodes are displayed in Figure 5. Analysing this figure, it is noticed that the situation of failure can be caused by several different

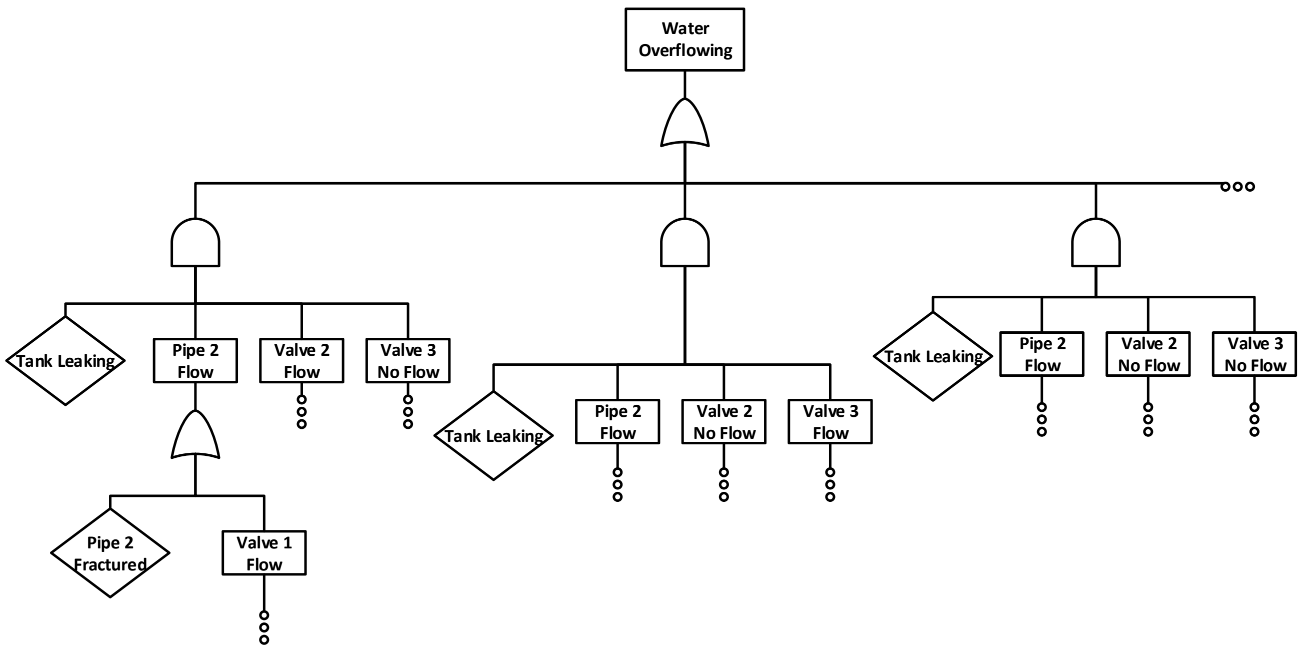

Fig. 5. Fault Tree of the Water Tank.

scenarios. Analyze each of these scenarios manually is a exhaustive task that would take too long.

Ess metologia

V. Conclusion

When an industrial process is projected, the reliability analysis is indispensable. In this study, was presented an application for automatic fault tree construction through the structuring of knowledge in the industrial automation domain with ontology. The application was validated using a water tank system.

The main contribution of this work is to allow the user to represent the knowledge of a system with a high level of abstraction. This reduces the chance of a human error, and allows designers to test different configurations in order to choose the most reliable.

As the next step, is intended to implement the whole approach in a real large-scale systems, so the features of the approach could be better evaluated.

Acknowledgment

The authors would like to acknowledge the support of the Brazilian research agency CAPES that partially funded this work.

References

[1] A. Majdara and T. Wakabayashi, "A new approach for computer-aided fault tree generation," *Systems Conference, 2009 3rd Annual IEEE*, pp. 308–312, March 2009.

[2] Y. Wang, T. Teague, H. West, and S. Mannan, "A new algorithm for computer-aided fault tree synthesis," *Journal of Loss Prevention in the Process Industries*, vol. 15, no. 4, pp. 265 – 277, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950423002000116

[3] A. Bossche, "Computer-aided fault tree synthesis i (system modeling and causal trees)," *Reliability Engineering & System Safety*, vol. 32, no. 3, pp. 217 – 241, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/095183209190001N

[4] R. Ferdous, F. Khan, B. Veitch, and P. Amyotte, "Methodology for computer-aided fault tree analysis," *Process Safety and Environmental Protection*, vol. 85, no. 1, pp. 70 – 80, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957582007713885

[5] S. A. Lapp and G. Powers, "Computer-aided synthesis of fault-trees," *Reliability, IEEE Transactions on*, vol. R-26, no. 1, pp. 2–13, April 1977.

[6] J. ANDREWS and J. HENRY, "A computerized fault tree construction methodology," *Process Mechanical Engineering*, vol. 211, no. 3, pp. 171 – 183, 1997.

[7] R. Lima, G. Leitao, L. Guedes, J. Melo, and A. Duarte, "Semantic alarm correlation based on ontologies," in *Emerging Technologies Factory Automation (ETFA), 2013 IEEE 18th Conference on*, Sept 2013, pp. 1–4.

[8] R. G. Lima, G. Leitao, L. A. Guedes, V. Bezerra, and A. Venleslau, "Economic evaluation of industrial processes with ontology," in *Networking, Sensing and Control (ICNSC), 2014 IEEE 11th International Conference on*, April 2014, pp. 322–327.

[9] M. Lampis and J. D. Andrews, "Bayesian belief networks for system fault diagnostics," *Quality and Reliability Engineering International*, vol. 25, no. 4, pp. 409–426, 2009. [Online]. Available: http://dx.doi.org/10.1002/qre.978