

A NEW GRAPHICAL TOOL FOR BUILDING LOGIC-GATE TREES

Terry F. Bott and Stephen W. Eisenhower
Los Alamos National Laboratory

Jonathan Kingson and Brian P. Key
Innovative Technical Solutions, Inc.

ABSTRACT

Tree structures that use logic gates to model system behavior have proven very useful in safety and reliability studies. In particular process trees are the basic structure used in a decision analysis methodology developed at Los Alamos called Logic Evolved Decision modeling (LED). *LED TOOLS* is the initial attempt to provide LED-based decision analysis tools in a state of the art software package. The initial release of the software, Version 2.0, addresses the first step in LED – determination of the possibilities. *LED TOOLS* is an object-oriented application written in Visual Basic for Windows NT based operating systems. It provides an innovative graphical user interface that was designed to emphasize the visual characteristics of logic trees and to make their development efficient and accessible to the subject matter experts who possess the detailed knowledge incorporated in the process trees. This eliminates the need for the current interface between subject matter experts and logic modeling experts. This paper provides an introduction to *LED TOOLS*. We begin with a description of the programming environment. The construction of a process tree is described and the simplicity and efficiency of the approach incorporated in the software is discussed. We consider the nature of the logical equations that the tree represents and show how solution of the equations yield natural language “paths.” Finally we discuss the planned improvements to the software.

INTRODUCTION

Tree structures that use logic gates to model system behavior have proven very useful in safety and reliability studies. The fault tree, in particular, has seen wide application as an analysis tool for risk assessment. In our research we have extended the basic ideas used in the fault tree to create a related class of logic-gate trees that we call process trees [1]. Process trees provide a framework for deductively generating possible causes of a final state or possible outcomes of an initial state.

Process tree construction has a proven record in successfully addressing a wide range of complicated and difficult problems. Process trees have been used to describe complex physical processes, such as cook-off of a heated high explosive [2]. They have been used in a forensic mode to determine what the possible causes of an observed outcome could have been [3]. They have been used to identify what features a system needs to perform its mission successfully [4] and the order in which upgrades should be performed to assure a process’ continued success [5]. Process trees have been used in analysis of spare parts stocking priorities [6] and in setting research priorities in a safety program. They have been used to enumerate the attack modes of a saboteur or terrorist [7] and the ways a spy could obtain protected information. Process trees have also been used to describe abstract processes. For example, we used a process tree to determine the analytical procedure for evaluating the risk of different information compromise scenarios during a visit by a foreign delegation to a secure facility (the information compromise scenarios were also identified by a different process tree) [8].

Process trees are the basic structure used in a decision analysis methodology developed at Los Alamos called Logic Evolved Decision modeling (LED) [6,9]. The basic elements of a decision model are

- Determine the possibilities or alternatives
- Select the metric to rank the possibilities
- Design an inferential model for the metric
- Rank the possibilities according to the metric
- Express the degree of uncertainty in the results
- Express the results in a form useful to the decision maker

LED uses coupled process trees to develop a comprehensive set of possibilities and to evaluate them in a consistent and traceable manner. In the past LED analyses have been performed using a combination of general-purpose software as well as a fault tree

drawing application, *SEATREE* [10]. This approach was acceptable when the number of analyses being performed was small and much of the emphasis was on methodology development. However it has become clear that in order to perform analyses more efficiently and to make the LED approach to decision analysis available to a wider audience, application specific software would be needed.

LED TOOLS is the initial attempt to provide LED-based decision analysis tools in a state of the art software package. The initial release of the software, Version 2.0, addresses the first step in LED – determination of the possibilities. *LED TOOLS* is an object-oriented application written in Visual Basic for Windows NT based operating systems. It provides an innovative graphical user interface that was designed to emphasize the visual characteristics of logic trees and to make their development efficient and accessible to the subject matter experts who possess the detailed knowledge incorporated in the process trees. This eliminates the need for the current interface between subject matter experts and logic modeling experts.

This paper provides an introduction to *LED TOOLS*. We begin with a description of the programming environment. The construction of a process tree is described and the simplicity and efficiency of the approach incorporated in the software is discussed. We consider the nature of the logical equations that the tree represents and show how solution of the equations yield natural language “paths.” Finally we discuss the planned improvements to the software.

OVERVIEW OF THE *LED TOOLS* INTERFACE

LED TOOLS uses a hierarchical format familiar to users of Windows to display tree structures. A Multiple Document Interface allows multiple files to be opened and viewed at the same time. Two windows are displayed for each project as shown in Fig. 1. One is the main tree window and the other is a replicants window, to be discussed in more detail below. The associated project name appears at the top of each window. In the main tree window, the tree starts with a top node and branches down. This is called the tree stem. The tree stem is populated by gates, terminal nodes, and replicant nodes. Each node has an icon representing its logical functionality and a text line that is the node display name. The output of each node is shown by means of light dashed connecting lines. Levels of development within the tree are shown by the extent of indentation. This format allows very large logic-gate trees with many levels of hierarchy to be displayed in a compact format.* Sub-sections of the tree are expanded or collapsed by double clicking on the top node for that sub-tree. Double clicking a top node in the main window will collapse the tree structure below the selected node and replace the node with a diamond symbol. A collapsed sub-tree can be expanded by double clicking on the diamond symbol. Double clicking on a replicated node will collapse the sub-tree below that node to a triangle.

Just below the standard menu is the gate palette. A gate can be dragged from the palette to the tree in a manner similar to a graphics program. Control-dragging a gate onto an existing node changes the gate type. The preferred method of tree construction is

visual, using drag and drop. However these operations can also be performed using commands on the EDIT menu or by right clicking. All of the gates used in the process tree are either of the OR-family or the AND-family. However as we shall see below, process trees are not restricted to the logic gates used in fault trees. The *LED TOOLS* architecture is designed to allow for the addition of new logic gates as needed.

Because the software will be used by subject matter experts who are unfamiliar with many of the details of mathematical logic and graph theory, an extensive Help system is available via the HELP menu. The Help system contains a detailed tutorial on tree construction as well as descriptions of the various logic gates available.

CONSTRUCTING A PROCESS TREE USING *LED TOOLS*

As noted earlier, process trees have been used in many different applications. In this paper we discuss the construction of a process tree for a problem of current interest [8]. In this case the possibilities to be deduced are associated with the question – What are possible attack scenarios using weapons of mass destruction (WMD) against a set of targets? Figure 2 shows an *LED TOOLS* process tree for the WMD problem. This tree is logically hierarchical in form; most of the hierarchy has been collapsed in this view. Development of the process tree requires the deduction of a series of logical steps from cause to effect or from effect to basic causal events and one could consider this to be a very preliminary version of the final tree.

A WMD attack requires a weapon and an attacker to use it. These requirements are indicated by the use of an AND gate at the top of the tree with the two inputs *utilizing* and *The attack is perpetrated by*. Each of these is in turn developed in more detail using deduction: successive deductive steps produce the characteristic hierarchical logic structure. The gates describe how each deduction is related to the previous one. We can quickly deduce that various attackers are possible and these appear under *The attack is perpetrated by*. This set of possibilities forms a taxonomy. Taxonomies are often encountered in process trees and a TAXONOMY gate is available in *LED TOOLS* for this purpose. It can be considered to be a specific type of an EXCLUSIVE OR (EOR) gate. As noted earlier many other gate types are available to express logical relationships. The taxonomy here has three members; each appears as a solid circle. This indicates that they are terminal nodes. That is, no further development occurs below it at present. Note that two of them are colored blue and one is green. Blue indicates that a node has been designated as excluded. The effect of an exclusion will be discussed below. Returning to the input, *utilizing*, we see that it is also a taxonomy with three terminal nodes as members. Two of these have been excluded as well. The third member, *radionuclides* is shown as a solid green diamond. A diamond indicates that this node has been contracted – there are additional nodes below this one. The fact that the diamond is solid is the visual indication that this node has been terminated. That is, the nodes that appear below it in the logical hierarchy are all considered to be neglected. Terminal nodes are also, by definition, terminated. They mark the end of a logical development, and this is why they appear as solid.

* This outline format is also used in *SEATREE*.

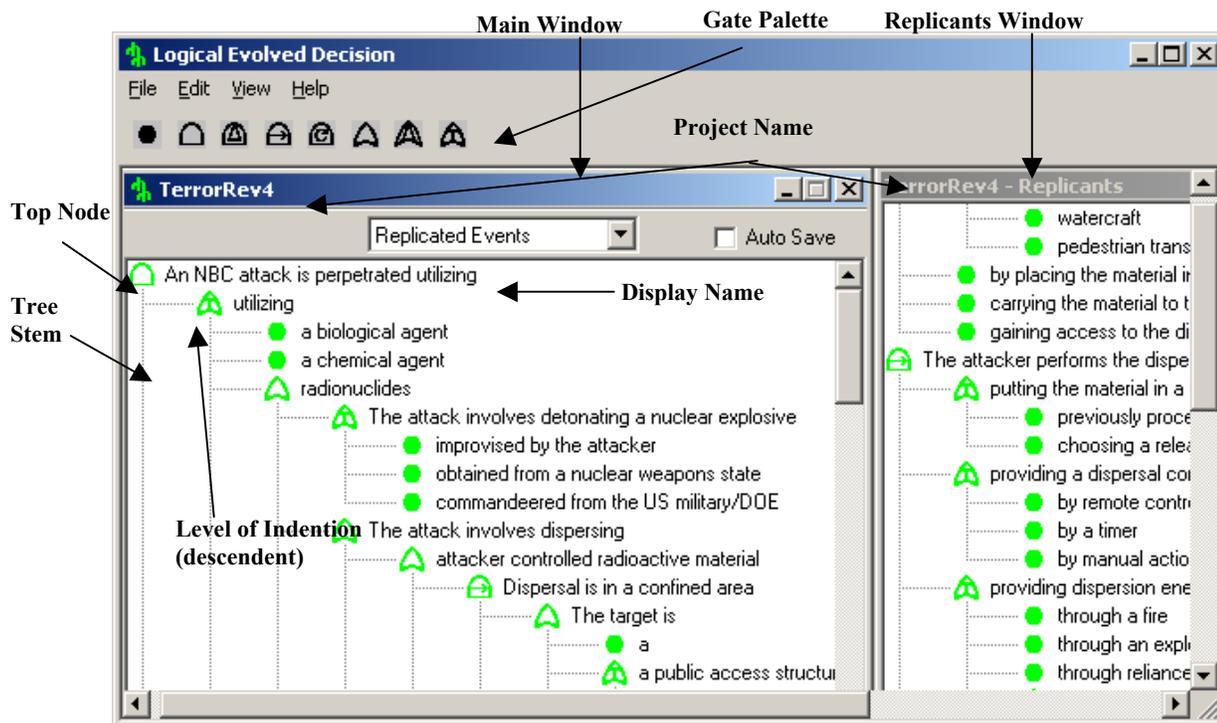


Figure 1. The LED TOOLS programming environment

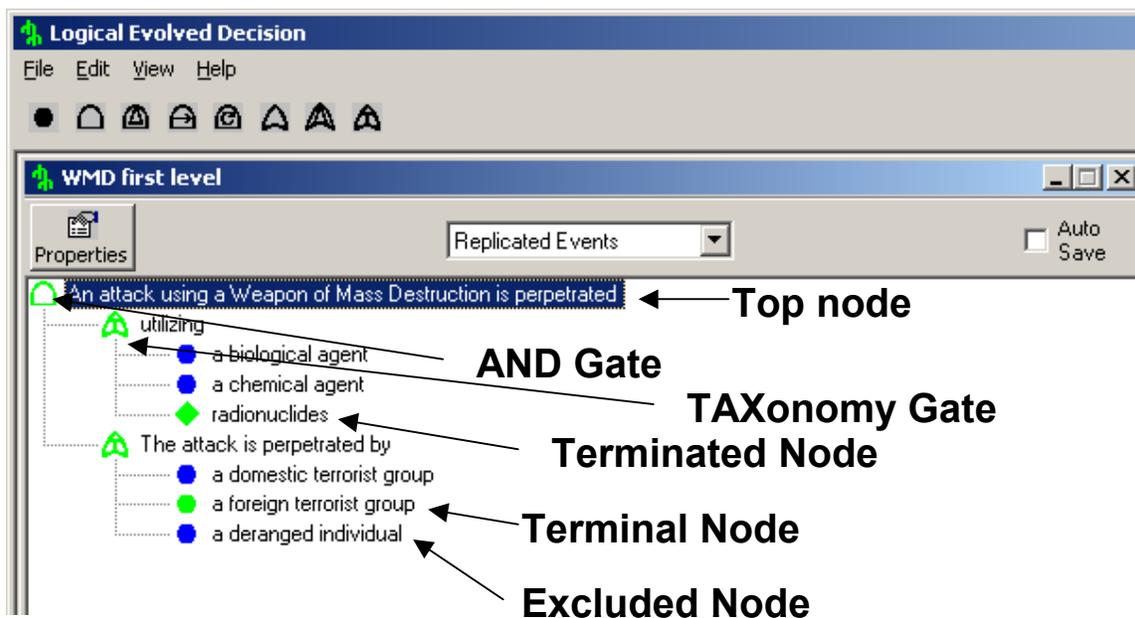


Figure 2. Top level of process tree for WMD attack possibilities.

Of course the process tree that we've build so far isn't particularly interesting. Recall that the node *radionuclide* is collapsed. Figure 3 shows what happens when this node is changed from terminated to included and is then expanded (by double clicking) – new levels of logical relationships are revealed. At the first level under *radionuclides* appear three nodes connected by EXCLUSIVE OR logic that describe what the material source of the radioactive material is, e.g. a nuclear explosive. Note that this node is blue so it is excluded and is an open diamond. An open diamond indicates that the node is collapsed. In Fig. 2, the node *The attack is perpetrated by* is collapsed so the details about the attacker groups that we saw in Fig. 1 are now hidden. The use of the expand and collapse functions allows the tree builder (or viewer) to concentrate on particular aspects of the tree as desired.

In Fig. 3 the expand/contract feature has been used to emphasize the logic associated with the sub-path

...The attack involves dispersing -- attacker controlled radioactive material—Dispersal occurs in an unconfined location....

The last node in this sub-tree, *Dispersal occurs in an unconfined location*, is connected to its five inputs by a CAUSAL gate. A CAUSAL gate is a variant of an ordered AND gate where it is understood that the order of the siblings, from top to bottom represents a causal chain that taken together produces the parent [11].

The first input of this CAUSAL gate is *The released material contaminates a region that includes* and we see that the only region contained in this taxonomy that is included with the tree in its current state is a downtown area. Note that the TAX gate is enclosed by a box outline. This indicates that *The released material contaminates a region that includes* is a replicant.

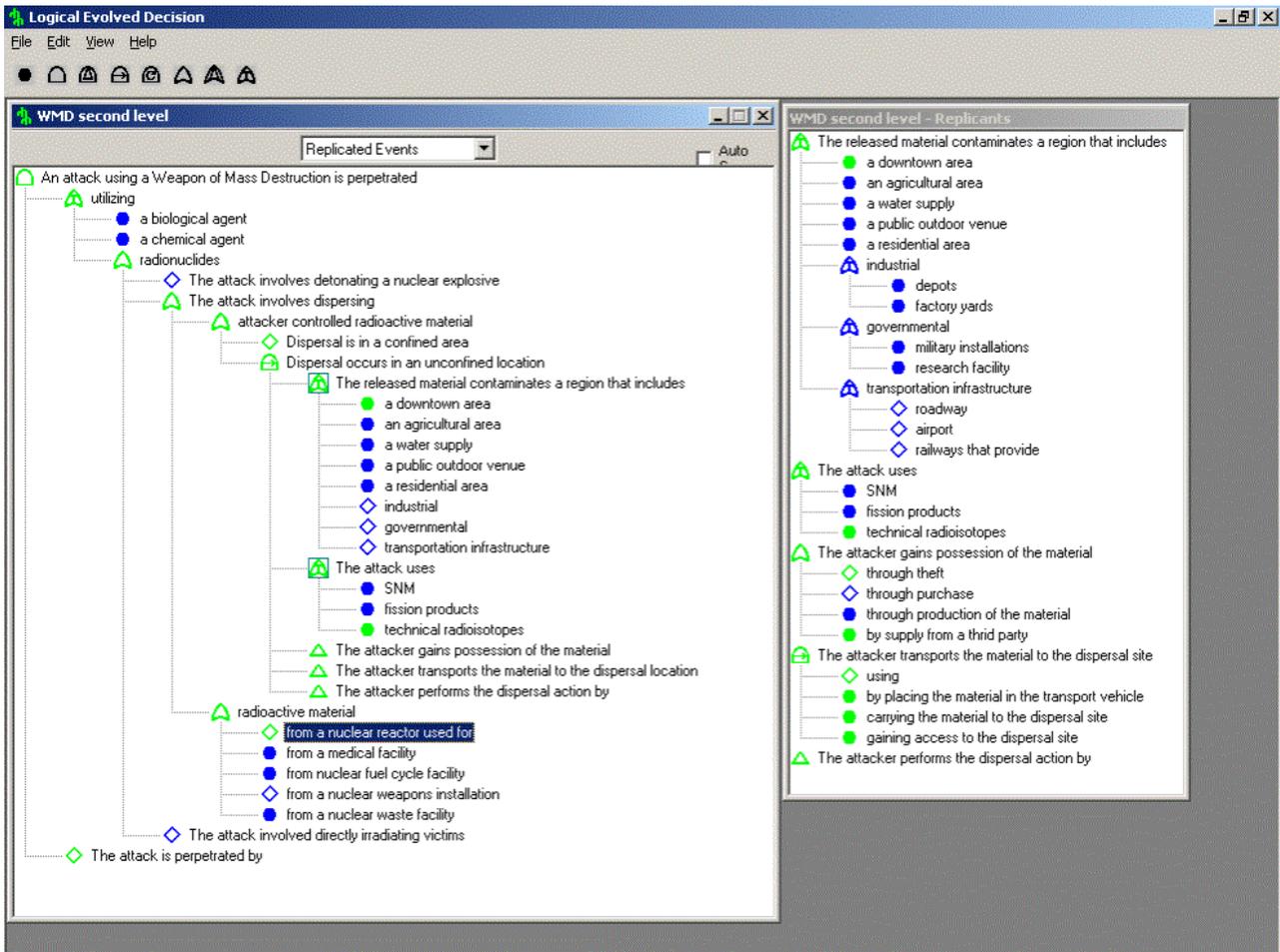


Figure 3. Second level development for WMD process tree. File: WMD second level.

USING REPLICANTS

A replicant is a sub-tree that has the property that it can be used more than once in the main tree.* The ability to create and use replicants is a very powerful feature of *LED TOOLS*. The individual replicant sub-trees are contained in a separate window with “- Replicants” appended to the main window file name. *The released material contaminates a region that includes* appears at the top of the replicant window in Fig. 3. The image of a replicant in the main window is referred to as an instance. All editing of a replicant occurs in the replicant window. Any edit operation thus performed is immediately reflected in all associated instances in the main window. A replicant can only be added to the main tree by dragging it from the replicant window to the desired location in the main window. This is to avoid possible confusion as discussed below.

A collapsed replicant is denoted by a triangle in either window. Note that the degree of expansion may differ between the replicant and an instance; the logic remains the same. Terminations and exclusions performed in the replicant window are also applied to all the associated instances as well. Individual instances of a replicant may be terminated as well. This option is available so that repetitions of a sub-tree in a path can be suppressed to improve readability.

SOLUTIONS AND PATHS

The properties of inclusion, exclusion and termination are associated with two very important properties of a process tree, the paths and the implicants. A process tree is a set of simultaneous logical equations. Nodes that have been excluded or terminated in a tree will be ignored when the equations are solved. With the tree as shown in Fig. 2, there are three logical equations:

1. *An attack using a Weapon of Mass Destruction is perpetrated AND utilizing, The attack is perpetrated by*
2. *utilizing TAX a biological agent, a chemical agent, radionuclides*
3. *The attack is perpetrated by TAX a domestic terrorist group, a foreign terrorist group, a deranged individual*

Here blue indicates the node that is the logical result (output) of the inputs in black operated upon by the logic gate in red. The set of path solutions of these equations with the given exclusions and termination has only one element, the path

An attack using a Weapon of Mass Destruction is perpetrated utilizing radionuclides. The attack is perpetrated by a foreign terrorist group.

Conceptually, the paths are found by successively substituting into the logical equations and preserving the partial results at each step. The number of paths depends upon the gate logic. For example, if all of the attacker types were included (green indication) there would be three paths differing after “perpetrated by...” If all of the terminal nodes were green there would be nine paths, accounting for all possible combinations of agent and

weapon type. If one takes a path and removes all of the elements except terminals, then the result is the implicant set that is the analog of the cut set for a fault tree.* For the path given above the implicant set is {radionuclides, a foreign terrorist group}.

LED TOOLS was designed to facilitate the construction of logical equations using natural language expressions. In addition to making it easier to perform the basic deductive steps, this feature also results in paths that can be read as paragraphs; each path is a unique and complete story describing how the top node in the tree occurs. One of the paths that results from the solution of the tree in Fig. 3 is.

An NBC attack is perpetrated utilizing radionuclides. The attack involves dispersing attacker controlled radioactive material. Dispersal is in a confined area. The target is a public access structure housing a mass audience sports venue. The attack uses technical radioisotopes. The attacker gains possession of the material through theft from a source of stored material by means of insider diversion. The attacker transports the material to the dispersal site using a road vehicle by placing the material in the transport vehicle carrying the material to the dispersal site gaining access to the dispersal site. The attacker performs the dispersal action by putting the material in a dispersible form by previously processing the material into a dispersible form providing a dispersal command signal by manual action of the attacker providing dispersion energy through reliance on natural dispersive forces providing a dispersion path through the ventilation system. The attack is perpetrated by a foreign terrorist group.

Each path solution is a fairly detailed “story” about an individual WMD attack. The underlined elements comprise the implicant set for this path and can be regarded as a “summary” of the story. The collection of paths can be further elaborated by adding additional structure in the tree. The use of the exclusion and termination features reduces the total number of paths that are obtained by solving the tree and allows the user to control the number and level of detail in the paths. Removing all of the solution restrictions in the complete WMD tree would yield a set of paths that number in the thousands.

Solving a process tree for the paths rather than the cut-sets may appear strange to readers familiar with fault trees. However it is clear that we can address many different questions using process trees beyond ‘How does a system fail?’ In doing so the details and interrelationships amongst the nodes are important. For example, in considering the risk of the scenario above, one would have to consider the capabilities of the attacker relative to the intricacy of the attack. Experts capable of performing such evaluations will need to understand scenario details and how scenarios are related. The combination of the process tree and the solution paths provides the necessary information.

* “Replicant” is taken from the novel *Do Androids Dream of Electric Sheep* by Philip K. Dick., the basis for the movie *Bladerunner*.

* Note however that a process tree path is not the analog of the path set for a fault tree.

WORKING WITH NATURAL LANGUAGE TREES

The combination of natural language nodes and the visual paradigm implemented in *LED TOOLS* has a number of implications that are not obvious from the discussion thus far. In a complicated path it might be useful to use the same phrase several times but where the phrase is a different logic node. For example, the word “utilizing” could appear as the parent in sub-trees associated with biological, chemical and/or nuclear threats. Of course all of these sub-trees are different logically and therefore “utilizing” appears in different contexts. Each *utilizing* is a unique node. *LED TOOLS* allows for context-dependent node names by using the rule that all nodes are born unique. Consider the situation in Fig. 4. In the first OR gate at the second level, each node is a unique logical object, although the words are the same. The second two inputs were created by dropping and dragging the first. All drag and drop, or equivalently copy and paste operations within the main window actually reproduce the structure of the node copied. All drag and drop operations from the replicant window to the main window copy an exact instance of the replicant. All instances of a replicant are logically identical. On occasion it is useful to take an instance of a preexisting replicant and “make it unique.” In this case the instance loses its replicant

status. That is, it can be edited in the main window and can be dragged, copied, etc. like any other non-replicant node.

FUTURE WORK

Development of *LED TOOLS* continues. These developments include:

Fast Path Solver: The normal need to solve for all of the paths associated with a particular combination of terminations and exclusions requires a fast solution algorithm. A *LED TOOLS* file contains the complete ordering of the tree. Knowledge of this structure makes it possible to determine the number of paths quickly and to calculate paths much faster than by using a classic successive substitution approach [12].

Digraph Drawing Interface: A process tree is actually a directed graph and the software provides a tool to develop digraphs deductively. The capability to convert an *LED TOOLS* file into the equivalent digraph has been demonstrated. Future versions of the software will have the capability to do this in real time. Together with the fast path solver this will provide the analyst with multiple ways to study the logical structure of a problem.

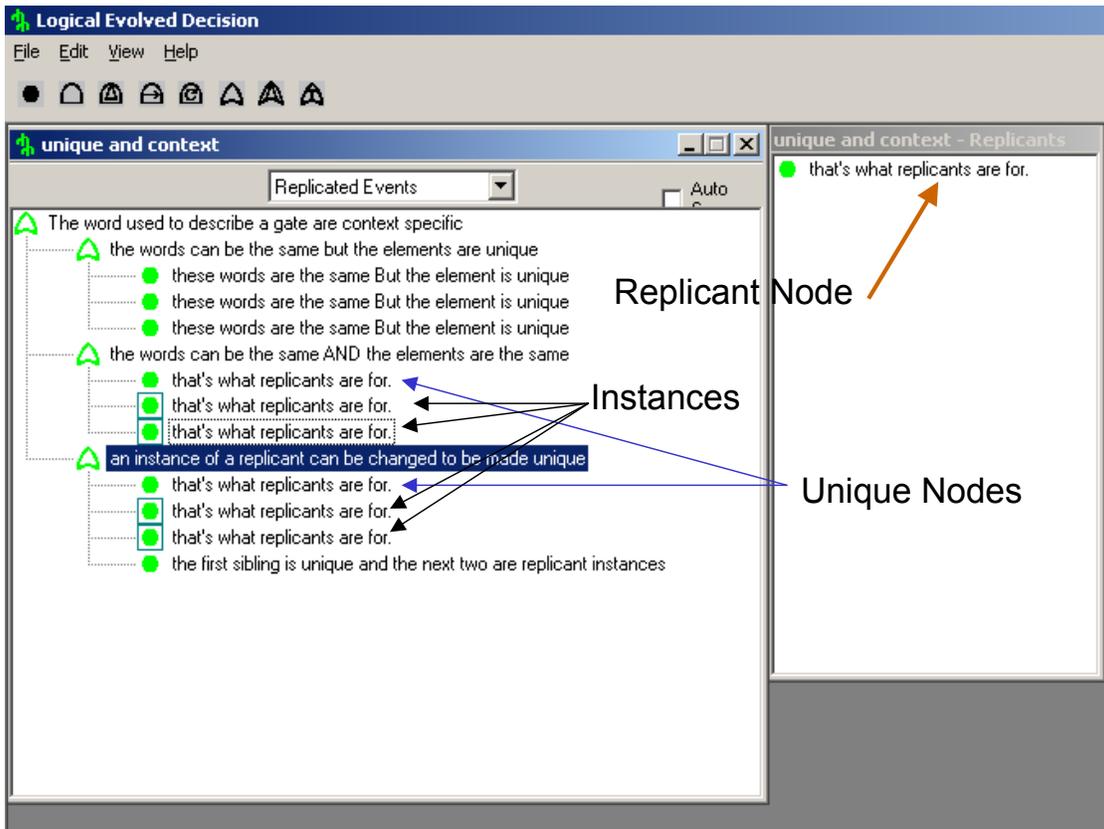


Figure 4. Context specific gate text.

Hyperlink Utility: A process tree is a very powerful database structure. Subject matter experts can use it to organize and archive information associated with a problem. We are currently implementing a hyperlinking feature in the software to provide this capability.

Inferential Process Tree Designer: Inferential models – used to rank order a set of possibility paths, are efficiently represented as process trees. Such models have associated with them additional variables that are not easily represented in tree form. We are developing utilities to make it possible to create and edit these variables while building the inferential process tree.

Gate Designer: As noted earlier, the basic software architecture allows for the addition of new gates to the gate palette. A new gate requires the definition of an icon, the basic logical equation and an equivalent digraph form. Preliminary design of a utility to provide these functions is planned for the near future.

CONCLUSIONS

Process trees have proven to be valuable tools in describing complex systems. Applications have included a broad range of technical, physical and even administrative systems. To date, however, all of these applications have required the involvement of the developers of the process tree. This is because the concepts used in process tree development have been under development and the available tree drawing tools required the translation from process tree to fault tree terminology and concepts. The use of the process tree has thus been dependent on the availability of a few people who interact with experts and translate their expertise into process tree logic. This is an inefficient use of the technique. In many cases it would be preferable for the subject matter experts to construct their own process trees with guidance from the process tree specialists only when required. The development of *LED TOOLS* makes this a viable option. The process tree methodology available to a much wider audience of subject matter experts. The software has been designed using a modern, visual approach so that a new user can quickly “learn by doing” without an excessive amount of manual reading. The availability of the software has in turn also spurred new development of the process tree methodology that we think will continue as the tool is more widely used.

ACKNOWLEDGMENT

The authors thank Drs. Philip Howe, Deanne Idar, and Larry Luck for their support in the development of LED Tools.

REFERENCES

- [1] T. F. Bott and S. W. Eisenhauer, “Programme Planning with Logic Trees,” **International Journal of Quality and Reliability Management**, Vol. 6, 1989, pp. 14-24.
- [2] S. W. Eisenhauer, T. F. Bott, L. B. Luck, J. Kingson and B. P. Key, “A Logic Model for Cook-off Phenomenology in High Explosives,” to appear *21st International System Safety Conference*, Ottawa, Canada, August 4–8, 2003.
- [3] S. W. Eisenhauer and T. F. Bott, “Accident Reconstruction using Process Trees,” *Risk and Safety Assessment: Building Viable Solutions*, PVP-Vol. 320/SERA-Vol. 5, SME 1995.

- [4] T. F. Bott, S. W. Eisenhauer, “An Approach to Assessing the Need for High Explosives Replacement in Aging Nuclear Weapons,” *27th International Pyrotechnics Seminar*, Grand Junction, July 2000

- [5] T. F. Bott and S. W. Eisenhauer, “A Logic Model Approach to the Conceptual Design of a Scientific/Industrial Complex,” *ASME-PVP Annual Meeting*, Vancouver, 2002, PVP-444, pp 119-127.

- [6] S. W. Eisenhauer, T. F. Bott, and J. W. Jackson, “Prioritizing the Purchase of Spare Parts Using an Approximate Reasoning Model,” *Proc. International Symposium on Product Quality and Integrity*, January 2002.

- [7] S. W. Eisenhauer, T. F. Bott, and D. V. Rao, “Assessing the Risk of Nuclear Terrorism Using Logic Evolved Decision Analysis,” to appear *ANS Topical Meeting on Risk Management*, San Diego, June 1-4, 2003.

- [8] K. B. Christiansen, T. F. Bott, J. L. Darby, and S. W. Eisenhauer, “The Approximate Reasoning (AR) Based Method for Information Loss Path Analysis,” *Institute for Nuclear Materials Management Annual Meeting*, Orlando, June 2002.

- [9] T. F. Bott, S. W. Eisenhauer, “Evaluating Complex Systems When Numerical Information Is Sparse,” *Words Automation Conference*, Orlando, June 2002.

- [10] B. Bingham, J. Hutchinson, and B. Lopez, “SEATREE Version 2.62 User Manual,” Science and Engineering Associates, Albuquerque, New Mexico, 1994.

- [11] S. W. Eisenhauer, T. F. Bott, “Application of Approximate Reasoning to Safety Analysis,” *Proc. International System Safety Conference*, Orlando, Aug 1999, pp 374-382.

- [12] Richard B. Worrell and Desmond W. Stack, *A SETS User’s Manual for the Fault Tree Analyst*, NUREG/CR-0465, Sandia Laboratories, November 1978.