

# Multiple Fault Diagnosis from FMEA

Chris Price and Neil Taylor

Department of Computer Science  
University of Wales, Aberystwyth  
Dyfed, SY23 3DB, United Kingdom  
cjp{nst}@aber.ac.uk

## Abstract

The Failure Mode and Effects Analysis (FMEA) design discipline involves the examination at design time of the consequences of potential component failures on the functionality of a system. It is clear that this type of information could also prove useful for diagnostic purposes. Unfortunately, this information cannot be fully utilised for diagnosis when FMEA has been performed by human engineers, because of inconsistencies in effect descriptions. The FMEA process is also very time consuming, with the consequence that the engineer can only deal with single point failures. Automation of the electrical FMEA process facilitates information reuse for diagnosis by providing consistent descriptions of failure effects, and by speeding up the FMEA process to such an extent that it becomes feasible to examine multiple failures. This paper introduces the advantages that automated FMEA provides for diagnosis, and describes its use for generating fault trees from the FMEA report. The paper examines the current limitations of FMEA use for diagnosis, and reports on how these limitations may be overcome.

## Introduction

Failure mode and effects analysis (FMEA) is a design discipline where an engineer examines and records the consequences of any (usually only single point) failure on the operation of a system. The purpose of the analysis is to highlight any significant problems with a design and, if possible, to change the design to avoid those problems. Intuitively, the results of such an exercise should be useful for diagnosis, but in practice they are only used as a source of ideas for the kinds of faults that could occur.

Two of the main reasons why FMEA results are under-utilized are:

**Inconsistency of failure effects.** An engineer might write down different descriptions for different occurrences

of exactly the same effect, or the same description for two slightly different effects. This lack of consistency makes it difficult to arrange all of the failures which could manifest a particular set of fault symptoms for use in a diagnostic investigation.

**Lack of multiple fault information.** FMEA is usually only carried out for single faults. If the results of such an FMEA were used to build a diagnostic system, then they would be equivalent to using a model based diagnostic system with the single fault assumption, as discussed in (Genesereth 1985). Such diagnostic systems are insufficiently powerful to deal with real world problems (de Kleer and Williams 1987).

Using results from an automatically generated FMEA can address both of these problems. The consistency of the results from an automated FMEA enables the reordering of failures into a tree according to symptom, providing effective ordering of the diagnostic investigation. Automated FMEA also makes the problem of multiple fault information more tractable. While no engineer could be expected to generate effects even for all pairs of failures in a system, an automated FMEA can explore many more possible failures.

This paper describes the use of output from an automated FMEA for generating diagnostic trees, showing how the results of the FMEA can be organised for maximum diagnostic benefit. This explanation is grounded in an example from the automotive domain: fault-finding on the lighting system of a typical family car.

## Automated Failure Mode and Effects Analysis

AutoSteve, a C++ reimplement of the FLAME system for performing automated FMEA (Price et al. 1995), is in regular use at several automotive manufacturers and generates reports that are comparable with those produced by an engineer without automated help. Its major benefit as a design tool is that it significantly reduces the engineer time needed to perform an FMEA analysis, turning a task

that might have taken an engineer several months into one that can be performed within a day. This makes it possible to perform an FMEA much earlier in the design lifecycle while design changes are comparatively cheap to make. Repeated analysis of a changed design using such a tool has even higher levels of time saving (Price 1996).

AutoSteve uses qualitative reasoning on electrical circuits (Pugh and Snooke 1996) and knowledge of intended system functions (Price and Pugh 1996) to generate electrical FMEA reports. It imports circuits directly from electrical CAD tools and generates a textual report on the effects of each failure mode. The report includes estimates of the severity, likelihood of detection and likelihood of occurrence for a particular failure.

Currently AutoSteve has been integrated with PowerView, a commercial electrical CAD tool. From within the CAD tool the engineer can verify the behaviour of the circuit model by changing the state of switches and sensors. Reasoning about circuit behaviour is based on a path-finding algorithm through a resistive network (Lee and Ormsby 1993) at the lowest level. Resistances can take one of three values: 0 (no resistance), Load (finite resistance) and Inf (infinite resistance). The path finding algorithm enables each component in the circuit to be labelled ACTIVE or INACTIVE. Above the resistive network is a component centred representation with an algorithm for regeneration of the resistive network as the state of the circuit changes (e.g. when a relay closes). AutoSteve has definitions for common components but can be extended further by defining the qualitative behaviour of new components; these definitions include how the component behaves under its possible failure modes.

In order to produce meaningful FMEA reports, functional states are employed to interpret the results of the qualitative reasoning. Figure 1 shows the tool that an engineer uses to describe how AutoSteve can identify when a function is being achieved. At the bottom of the window is the definition of four component states that need to be ACTIVE for the full\_headlights\_on function to be achieved. Functional states are reusable between different designs for the same automobile subsystem, and so the engineer only needs to check that the existing functions are appropriate for the new design, and add any extra functions that are needed.

When the engineer has designed the circuit and linked it to the functional states, the AutoSteve system performs automated generation of an FMEA report in the following way:

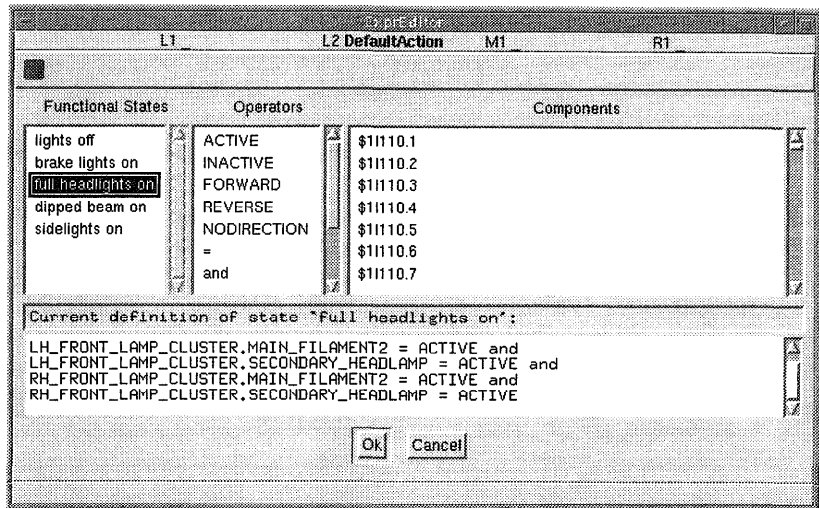


Figure 1 : Functional States Expression Editor

**Obtain the correct behaviour.** Simulate the circuit through its possible changes, operating switches and changing sensor states. The resultant behaviour of the circuit is abstracted by recognizing the operation of functions (e.g. that when the sidelight switch is thrown, the sidelight function and the licence light function operate).

**Make a list of failures that can occur in the circuit.** The possible failures modes of each type of component are defined. The complete list of possible failures for the circuit is made up of the possible failures of each component in the circuit.

**Obtain faulty behaviour of the circuit.** For each possible single point failure, impose that failure upon the circuit and repeat the simulation and abstraction work done for the correct circuit.

**Compare the faulty and correct abstracted behaviour.** Functions that occur when they should not, or which do not occur when they should, describe the significant incorrect behaviour of the circuit for a fault. Discrepancies of this kind form the basis of the textual report for the effects of the failure, shown in figure 2.

For a typical car exterior lighting system, AutoSteve examines 214 single failures on 90 components, taking about a minute to complete on a Sparc 20 computer. An engineer can use AutoSteve to produce an FMEA report for a system of this complexity within a day, where it might have taken several weeks to produce by hand.

Item / Function	Potential Failure Mode	Potential Failure Effect	Sev	Potential Failure Cause	Occ	Current Design Controls	Det	RPN
(710)	Regardless of any event change, the "sidelights on", "dipped beam on" and "full headlights on" functions were never achieved.	No sidelights. No dipped beam. No main beam - legal requirement.	6	The component WIRE53 has failure burned_out, C-1104 has failure blown and component LH_FRONT_LAMP_CLUSTER has failure	1		2	12
(703)	When IGNITION_SWITCH was set to Ign_on (2) the "full headlights on" function was achieved unexpectedly. Also, when BIP_SWITCH was set to Full Beam	No dipped beam. The main beam headlights were on. No main beam - legal requirement.	6	The component WIRE53 has failure burned_out and component LH_FRONT_LAMP_CLUSTER has failure main blown.	1		2	12

Figure 2 : Example FMEA results for multiple failures

### Multiple failure FMEA

The industrial use of AutoSteve for generating FMEA reports on electrical designs has only explored single failures so far. This has been because the primary aim was to be able to reproduce the process already being carried out by engineers and they mostly only consider single faults in FMEA. However, the qualitative simulator used in AutoSteve is not limited to analysis of single failures.

The failure generation algorithm has been amended to generate any combination of failures with a likelihood greater than  $1:10^n$ , where  $10^n$  gives some desired level of reliability; the default value of  $n$  is 9. The likelihood is computed by multiplying the occurrence ratios which are associated with each fault on a component. Some permutations are not possible because they are contradictory failures on the same component (e.g. a wire being both unconnected and shorted to ground). Such combinations are ignored, so reducing the overall number of permutations slightly.

Having generated the combinations (including single failures) that match the above criteria, AutoSteve runs the simulation on each combination in order to calculate the effects. In the lighting circuit there are 90 different components with a total of 214 different single failures. The total possible number of permutations is  $2^n - 1$ , where  $n$  is the number of different single failures, giving approximately  $2.6e+64$  possible multiple failures in this example. Only generating candidates with a likelihood greater than  $1:10^9$  reduces this to 6,735 failure combinations.

The FMEA simulation on this many failure combinations takes about 1.5 hours to run. It is impractical to expect the engineer to examine all of the produced report as is done with the single failure report. Fortunately, many of the multiple failure cases are uninteresting as they

provide no more information than the single failure reports, and so the final report can be pruned, without reducing significant information that is presented to the engineer, by the following method.

For each multiple failure combination a comparison is made with the effects of each single failure that comprise the multiple failure. Only interesting multiple failure cases are retained according to the following test:

Let the fault symptoms for a component failure  $f$  be a set of function differences  $D_f$ , where a function difference is either the unexpected operation of a function, or the absence of expected operation of a function.

For two failures  $X$  and  $Y$ , the multiple failure  $X \wedge Y$  is not interesting if:

$$D_{X \wedge Y} = D_X \text{ or } D_{X \wedge Y} = D_Y \text{ or } D_{X \wedge Y} = D_X \cup D_Y$$

For example, if the effect of failure 1 is that the main headlights do not work when expected, and the consequence of failure 2 is that the dipped headlights do not work when expected, then the combination of the two failures being that neither the main nor the dipped headlights work when expected is not interesting.

The criteria for interestingness show for pairs of failures can be extended from pairs to the general case. Pruning the results from the lighting circuit simulation on interestingness criteria resulted in 342 interesting failure combinations (214 single, 128 multiple) from the 6,735 failure combinations that passed the likelihood criteria

There are few enough interesting multiple failure items in the FMEA report that it is realistic to expect the engineer to examine them at the same time as examining the usual single failure FMEA details.

## Generating fault trees from FMEA results

The automated FMEA report contains consistent failure mode and effect descriptions of the type shown in figure 2. Therefore, for each combination of component failures, we can determine the effect it has on the system in terms of its function differences (functions that failed to occur and functions that occurred unexpectedly). From such information it is possible to identify all failures in the FMEA report that have the same effect (e.g. all failures which can cause the car's dipped beams to not light when they should, or all failures which cause both the main beams and the dipped beams to be lit when they should not be).

The fact that the FMEA report has been checked by an engineer when the circuit was designed is attractive for diagnosis, as it means that the model-based information has undergone a degree of verification by engineers.

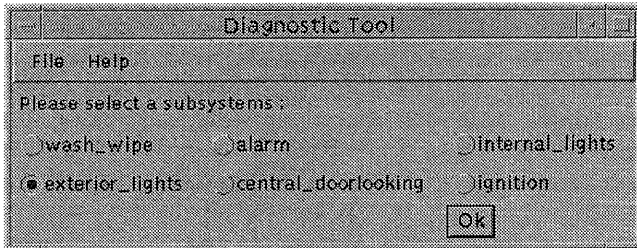


Figure 3 : Selecting a subsystem

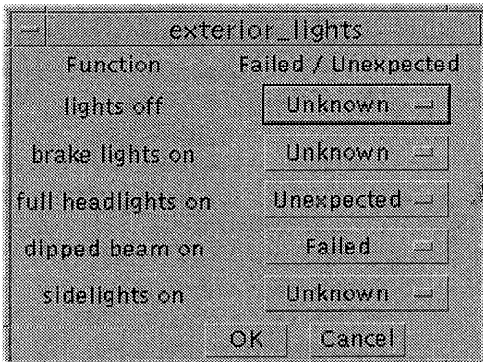


Figure 4 : Identifying function differences

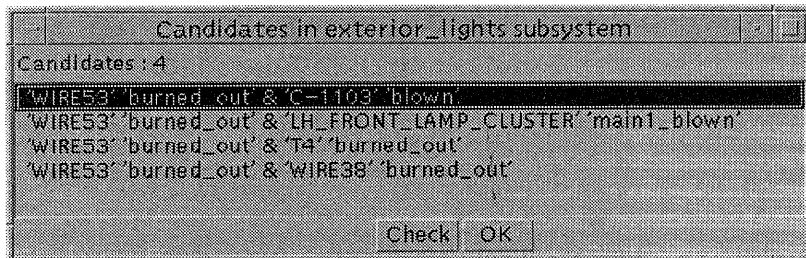


Figure 5 : List of candidates

Another significant advantage of using AutoSteve FMEA output as input to a diagnostic system is that the effect descriptions are based on the unexpected presence or absence of the intended functions of the design being analyzed, and so can be used to elicit a user-level description of the problem. In the car lighting circuit, for example, the intended functions of the circuit are at the level of user-identifiable symptoms (full headlights on, sidelights on etc.). Information about which functions operate as expected can then be used to focus on the candidate faults that fit the user's description. The remaining candidates can be ordered by likelihood of occurrence (available from the FMEA report). No pair of failures is more likely than any single failure, so in practice this will place the single failures at the head of the candidate fault list.

An apparent exception to the rule that multiple faults are less likely to occur is the case of *dependent faults*, where one failure causes another (for example, a short circuit often causes a blown fuse). However, AutoSteve computes the consequences of each failure during its simulation (for example it would detect that the short circuit had caused a fuse to blow and record any effects this had on the system functions). As a result, dependent fault cases are covered by the relevant single fault cases and will be dealt with as single failure (high probability) cases.

Once the failure has been localised using information about function, tests can be used to discover whether the most likely candidate fault is the actual fault. For example, if *high beam relay stuck open* is the most likely candidate fault, then testing the high beam relay will either blame or exonerate it. If high beam relay is not stuck open, then all multiple failure candidates that include that failure can also be deleted. In this way, the remaining possibilities can be addressed in a reasonable manner, and can be quickly pruned to select the right solution.

FMEA reports are generated for each major electrical subsystem in a car, and so this strategy can be extended to form the basis for a diagnostic scheme covering the whole of the car electrical system, as shown in figure 3. This is a simple example diagnostic system using FMEA output. It shows a set of car subsystems that can be selected and facilitates fault localization on the basis of the subsystem where the problem is (*exterior lights* in this case). Next the set of functions within the chosen subsystem is shown to the user for indication of which symptoms are present. Figure 4 shows the set of functions available for the exterior lights subsystem. Selecting the functions which have either failed or occurred unexpectedly allows the system to determine a list of candidates for the given fault signature, ordered by likelihood (figure 5).

Finally the diagnostic system would guide the user through the possible diagnoses.

This is not a completely novel strategy and builds on previous work in generating fault dictionaries by simulation. Previous work which stands out in this area includes (Bratko et al. 1989) and (Mauss and Neumann 1995). However, there are two significant differences between this work and those previous efforts to build diagnostic fault trees from a fault dictionary.

**Method of generating candidates.** By constructing the candidates from verified FMEA output, the method described in this paper generates candidates whose behaviour has been verified by an engineer. This minimises model-building effort, and gives a much higher degree of confidence in using model-based reasoning.

**Method of constructing a tree.** The two papers cited both generate all candidates and then form a tree using ID3 to compress the tree and to order candidates merely by ordering tests on information gain grounds. The method described in this paper prunes the number of candidates where the candidate can be deduced by combining simpler candidates, leading to a significant reduction in the size of the tree. It then orders candidates primarily by the function they affect, allowing quick focusing by observed symptoms. The partitioning described here is on the basis of domain knowledge about partitioning of car functions at the top two levels, and by abduction at the lowest level. As the domain partitioning was originally done for design purposes, this is an efficient as well as effective use of the domain knowledge.

### Limitations of this work

This paper describes a promising direction for practical model-based diagnostic reasoning. This section identifies where there are opportunities for further research, and suggests work that might be profitably pursued.

**It is impossible to generate all combinations of failures for large systems.** This is a perennial problem of model-based diagnosis. However, in practice, many combinations of four or more failures have a vanishingly small probability of occurring. The strategy used here is to generate only combinations of failures with a higher likelihood of occurrence than a given threshold. This seems to work well for practical situations.

**Some problems go across subsystem boundaries.** The scheme outlined in this paper assumes that there is a clear separation between subsystems. That assumption is false for some of the more challenging problems to diagnose, where a single failure in each of two subsystems will interact to cause a complex overall problem. This is also an issue for FMEA generation, because some of the worst design problems are caused by interactions between subsystems. We are developing methods of hierarchical reasoning about automotive circuits which may help with this problem by identifying which other subsystems may be affected by a particular failure. This would allow us to add such multiple subsystem candidate faults in the relevant fault trees.

**There is insufficient information in the tree to order candidates correctly.** The leaf nodes of the tree are only ordered on likelihood of occurrence. Other information such as cost of test would be needed to produce an effective ordering of candidates. However, some commercial automotive diagnostic tools do not provide detailed ordering, leaving the fault identification decisions to the mechanic using the system. If detailed ordering was required, then a realistic prospect would be to link the work here to a diagnostic tool such as TestBench, and to order the candidates within the commercial tool.

**Tests are not always specific enough.** The electrical qualitative simulator used to generate the FMEA report only reasons in terms of ACTIVE and INACTIVE parts of the circuit. In practice, faults can develop gradually, and so a wire might be active, but below the threshold for activating a relay, for example. In such cases, it might be necessary to have a more specific test to perform on the circuit. Work has been done by our collaborators at Ford Research Center, Dearborn (Montgomery et al.) into using a quantitative circuit simulator for generating the effects of failures. Linking the results of the two kinds of simulation should enable the generation of more specific tests.

### Conclusions

Model-based diagnosis has been identified as a promising technology because it takes away the effort of building fault trees by hand, and provides a more complete coverage of possible faults. It has typically been portrayed as the execution-time generation and ordering of candidate faults. That approach is very flexible and powerful, but rarely results in practical diagnostic systems. The problems with that approach have been in model building, and in

providing the user with a sensible ordering of tests to be carried out.

The work described in this paper presents an alternative strategy for building a model-based diagnostic system. Diagnostic trees can be compiled off-line from models originally intended for design analysis, rather than generated at run-time. This work has several significant positive features:

**It is not based on models built specially for diagnosis.**

Generation of the diagnostic tree reuses the models that were constructed in order to produce an FMEA report. This means that the effort of producing a first attempt at a diagnostic system is minimal. As the model has been developed for design analysis, the diagnostic system can be available very early in the product lifecycle.

**It provides fault localization.** The design-time partitioning of a car into subsystems can be used to focus the diagnostic system on the area where the fault has occurred.

**It provides an appropriate vocabulary for symptom description.** The design-time identification of the functions of each subsystem provides a vocabulary which the diagnostic system can use to focus the search for candidate faults. This is not provided by most model-based diagnostic systems, and would have to be another layer constructed on top of the diagnostic system.

**It is efficient at run-time.** In many cases, it is impossible on performance grounds to execute models of the system at run-time in order to generate likely candidates. Because all likely candidates are generated earlier and then compressed, this scheme produces diagnoses efficiently.

**It allows user verification and adjustment of the tree.**

One of the practical worries about execution-time generation of candidates is that such a diagnostic system cannot be properly tested to ensure that it works. Examining an automatically generated diagnostic tree is much less effort than generating one by hand, but still allows the developers to decide whether tests are being done in the correct order.

The research described in this paper provides the basis for a cost effective way of building practical, powerful model-based diagnostic systems.

## References

Bratko, I.; Mozetic, I.; Lavrac, N. 1989. *KARDIO: a study in deep and qualitative knowledge for expert systems*, MIT Press.

Genesereth, M. 1985. *The Use of Design Descriptions in Automated Diagnosis*, in *Qualitative Reasoning About Physical Systems*, North-Holland.

de Kleer, J.; Williams, B. 1987. *Diagnosing Multiple Faults*, *Artificial Intelligence* 32, 97-130.

Lee, M. and Ormsby, A. 1993. *Qualitative Modelling of the Effects of Electrical Circuit Faults*. *Artificial Intelligence in Engineering* vol. 8, 293-300.

Mauss, J. and Neumann, B. 1995. *Diagnosis by Algebraic Modelling and Fault-Tree Induction*. In *Proceedings of DX-95*, Goslar, Germany, 73-80.

Montgomery, T.; Pugh, D.; Leedham, S.; and Twitchett, S. 1996. *FMEA Automation for the Complete Design Process*. In *Proceedings of Annual Reliability and Maintainability Symposium*, 30-36, IEEE Press.

Price, C. J. 1996. *Effortless Incremental Design FMEA*, *Proc. Ann. Reliability and Maintainability Symp.*, 43-47, IEEE Press.

Price, C. J.; Pugh, D. R. 1996. *Interpreting Simulation with Functional Labels*, *Proc. 10th Annual Qualitative Reasoning Workshop*, Stanford Sierra Camp, AAAI Press.

Price, C. J.; Pugh, D. R.; Wilson, M. S.; Snooke, N. 1995. *The Flame System: Automating Electrical Failure Modes & Effects Analysis (FMEA)*, *Proc. Ann. Reliability and Maintainability Symp.*, 90-95, IEEE Press.

Pugh, D. and Snooke, N. 1996. *Dynamic Analysis of Qualitative Circuits*. In *Proceedings of Annual Reliability and Maintainability Symposium*, 37-42, IEEE Press.