

# Architecture and design of a diagnostic information fusion system

KAI GOEBEL\*

GE Corporate Research & Development, Information Systems Lab, Niskayuna, NY

## Abstract

This paper introduces an architecture for aggregation of output from different diagnostic tools. The diagnostic fusion tool deals with conflict resolution where diagnostic tools disagree, temporal information discord where the estimate of different tools is separated in time, differences in information updates where the classifiers are updated at different rates, fault coverage discrepancies, and integration of a priori performance specifications. To this end a hierarchical weight manipulation approach is introduced which creates and successively refines a fused output. The performance of the fusion tool is evaluated throughout its design. This allows impact assessment of adding heuristics and enables early tuning of parameters. Results obtained from diagnosing on-board faults from aircraft engines are shown which demonstrate the fusion tool's operation.

**Keywords:** Information Fusion, Diagnosis, Classification, Design of Experiment, Monte Carlo Simulation

## 1. INTRODUCTION

Today, service providers and manufacturers develop different tools to accommodate particular diagnostic requirements. The latter originate in the need for better fault coverage or for better diagnostic performance. The stipulation to use several tools arises because often times any one tool cannot deal with all faults of interest at the desired level of accuracy. In part this results from environmental changes, which has a different impact on the diagnostic capabilities of individual diagnostic tools. In addition, some tools cannot easily be expanded to include new faults and other tools need to be added to fill the gap. Finally, while some tools are good detecting certain faults they might be virtually worthless for others. The resulting patchwork approach achieves optimization at the local level, but ignores benefits gained by taking a system-level view. Therefore, it seems logical to take the next step and use a system-level scheme that gathers and combines the results of different diagnostic tools to maximize the advantages of each one while at the same time minimizing the disadvantages. Such a fusion scheme holds the promise to deliver a result that is better than the best result possible by any one tool used. In part this can be accomplished because redundant information is available that when combined correctly improves the estimate of the better tool and compensates for the shortcomings of the less capable tool. However, there is no substitute for a good diagnostic tool and, ordinarily, multiple, marginal-performance tools do not necessarily combine to produce an improved result and in fact may worsen the outcome (Hall and Garga, 1999).

There are several traditional approaches that deal with fusion of information. Weighted averaging attempts to compensate for bad tool decisions by smoothing out the poor performers. However, the trade off is that good information succumbs to the bad information in the process and a particular tool's superior performance for some faults is sacrificed. In voting schemes, the tools decide jointly on the final output through a majority vote but encounter similar problems as the weighted averaging because several poor performers can always outvote a good tool. Bagging and boosting (Freund and Schapire, 1999) tries to address some of those problems. Pattern recognition approaches such as neural networks can be employed to recognize patterns of behavior that may lead to correct decisions. However, if the input to the tools is not available to the fusion tool and the output pattern looks

---

\* Reprint requests to: Kai Goebel, GE CR&D, ISL, K1-5C4A, One Research Circle, Niskayuna, NY 12309; email: goebelk@crd.ge.com

similar for different input scenarios, the neural fusion may not perform satisfactorily. Dempster-Shafer reasoning is widely used for fusion tasks where several information sources are fused using Dempster's rule of combination. However, it is imperative to properly fix the meaning of the underlying belief functions because the suitability of the rule depends on the correct context (Smets, 1994). Model-based approaches consist of a sequence of steps for validation and conflict resolution, among others. The method shown by Nelson and Mason (1999) uses multiple models of known (or suspected) behavioral patterns to establish degrees of compatibility between data and hypothesis. It enforces preferences over the set of candidates (by removing candidates that violate these preferences), and iterates through a cycle of merging and deleting within a set of associated hypotheses for that conflict. Sequential and parallel multi-layered configurations (Rahman and Fairhurst, 1998) employ a number of diagnostic tools in a sequential and parallel fashion for the refinement of the initial fault found utilizing a priori probabilistic performance information about the tools which is used to calculate an error probability. The individual classifiers use the current input pattern as well as the fault index of the preceding layer as input variables. A fuzzy fusion schemes described in Loskiewicz-Buczak and Uhrig (1994) utilizes the generalized mean and an  $\alpha$ -cut. The fusion scheme fuses the first two sensors, defines the confidence of the fused decision, and then continues to fuse additional sensors. If the confidence drops, the step is reversed. Finally, an  $\alpha$ -cut (depending on the particular class) determines the exact fault

## **2. BACKGROUND**

Our work was motivated by the diagnostic task of aircraft engine gas path faults. On a very coarse level, service providers to aircraft engines – both commercial and military – are strongly interested in reducing off-wing time and shop time for engines. There are several benefits in savings for the actual repair cost as well as the increased up-time. In addition, improved system reliability leads to a higher success rate for missions in case of the military engine. To accomplish that goal, it would be desirable to obtain reliable in-flight diagnosis that can perform system state estimation throughout the operation of the engine and deliver the results to a maintenance crew during the landing phase thus avoiding lengthy diagnosis after landing. A realistic goal was determined for this particular case to be fault detection capability of greater than 95%, i.e., less than 5% false negatives (missed faults) in addition to less than 1% false positives (false alarms) (Ashby and Scheuren, 2000). Based on traditional tool performance, it was anticipated that this goal could not be met by any one diagnostic tool alone. However, it was expected that a scheme aggregating the information from several diagnostic tools would be able to achieve the desired performance. To that end, relevant diagnostic and other on-board information sources were designed to produce diagnostic estimates and secondary supporting information covering all faults of interest with maximum overlap of fault coverage and time of diagnosis. Irrespective of those goals, the final diagnostic tool suite exhibited sometimes substantial differences in the fault coverage (not all tools covered all faults of interest), fault diagnosis performance (some tools were better than others at performing the diagnosis), flight operation regimes (some tools operated during certain phases, e.g., climb & cruise vs. takeoff only), and operating rates (e.g., 1 Hz, 30Hz, once per flight). Other design requirements were that the operation had to be performed in real-time and on-board the aircraft during flight. Since this work was performed as a dual application for both a commercial and military engine, it necessitated a flexible design that allowed the use of the fusion tool for both applications. In addition, it was the intent to be able to add diagnostic tools at a later time to the fusion scheme thus calling for a modular design.

## **3. ARCHITECTURE AND DESIGN OF FUSION TOOL**

In the sections below we will discuss the information used for the fusion tool, the architecture of the proposed fusion scheme, and the design of the tool.

### **3.1 Information used in fusion scheme**

The fusion tool uses a priori information and the output coming from the diagnostic and non-diagnostic information sources. The proposed scheme relies heavily on information about tool performance implying that this information is attainable through experiments or simulations.

### 3.1.1 Confusion matrix:

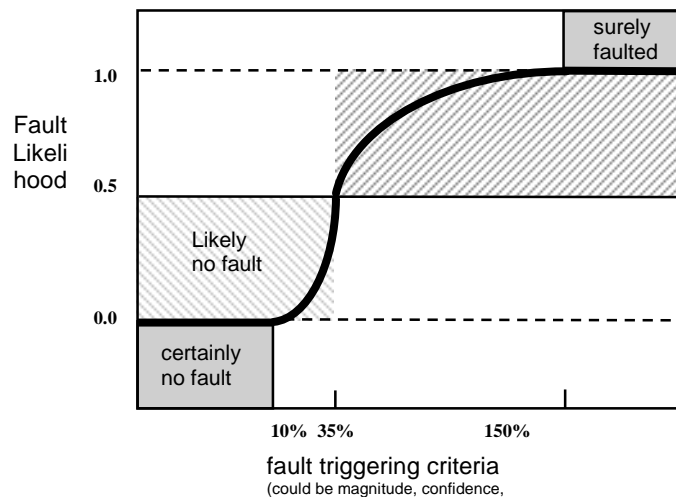
The confusion matrix of the diagnostic tools is the primary source of a priori information for the information fusion tool. As a performance measure for the individual diagnostic tools, the confusion matrix lists the observed faults versus the estimated faults. Because all faults are enumerated, it is possible to obtain information not only about the correctly classified states, but also about the false positives (FP), false negatives (FN), and false classified (FC) states. In our representation of the confusion matrix, the rows list the actual faults, the columns list the estimated faults. Note that the fault  $F_0$  represents the no fault condition. The diagonal entries of the confusion matrix represent the correctly classified cases. The first row – except the first entry – contains the FP. The first column – except the first entry – contains the FN. The off-diagonal elements – except the FP and FN – are the FC. Table 1 shows the normalized confusion matrix for a diagnostic tool where the result was divided by the number of experiments for each class. The faults are denoted as  $F_n$  where  $n=\{0, \dots, 6\}$ .

**Table 1: Confusion Matrix used as Input for both Design and IFM run-time version**

	$\hat{F}_0$	$\hat{F}_1$	$\hat{F}_2$	$\hat{F}_3$	$\hat{F}_4$	$\hat{F}_5$	$\hat{F}_6$
$F_0$	0.7525	0.0010	0.0150	0.0290	0.0320	0.0020	0.0710
$F_1$	0.0020	0.6733	0.1440	0.0290	0.0860	0.0010	0.0080
$F_2$	0.0580	0.0010	0.7921	0.0190	0.0190	0.0020	0.0160
$F_3$	0.0260	0.0010	0.0340	0.4851	0.4220	0.0020	0.0030
$F_4$	0.1170	0.0010	0.0060	0.1600	0.6337	0.0010	0.0460
$F_5$	0.0040	0.0010	0.0590	0.0620	0.0270	0.7327	0.0280
$F_6$	0.1410	0.0010	0.0200	0.0040	0.0150	0.0200	0.7228

### 3.1.2 Relevance matrix

Relevance is an assignment of whether a tool can perform a diagnosis on a per fault basis. It can be derived from the confusion matrices but can also be used in the conceptual phase of the diagnostic tools to determine which tool will or can cover which fault. Relevance is indicated by “1”, no relevance by “0”. Relevance is summarized for all diagnostic tools in a so called “relevance matrix”.



**Fig. 1: Mapping of diagnostic tool output into 0-1 domain**

Table 2 shows an example for 3 tools  $t_1$  through  $t_3$  and 7 faults  $F_0$  through  $F_6$ .

**Table 2: Relevance assignment of tools**

	$F_0$	$F_1$	$F_2$	$F_3$	$F_4$	$F_5$	$F_6$
$t_1$	1	1	1	1	1	0	0
$t_2$	1	1	1	1	1	1	1
$t_3$	1	1	1	1	1	0	1

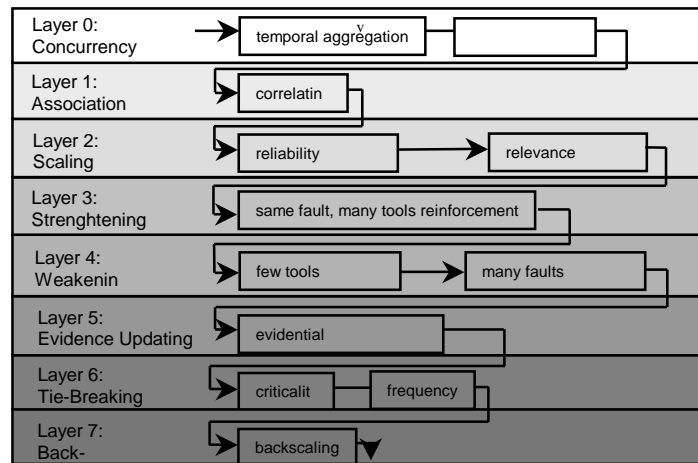
### 3.1.3 IFM Input

Primary input to the information fusion is the output of the classifiers, i.e., the fault vector of each diagnostic tool for the respective faults considered. The information fusion tool is built on the premise that it can utilize information that led to the classification. In other words, it will not only consider the final fault assignment but also the underlying relevant fault strength. Depending on the diagnostic tool employed this can be a distance measure (for example for a k-means classifier), probability (for example for a Bayesian Belief Net), weight (for example for a neural net), membership (for example for a fuzzy knn), etc. This individual assignment criterion is then scaled between zero and one using an appropriate classifier specific non-linear function. The implicit interpretation is that a level closer to one means that the fault is increasingly more likely while a confidence level less than 0.5 is increasingly not likely. Thereby we avoid the step of needing a parametric model for fusing heterogeneous data (Hathaway et al., 1996) and instead impose this task on the designer of the diagnostic tools who has to provide the mapping from diagnostic output to confidence level. Fig. 1 shows the diagnostic output to information fusion mapping.

Other system information not diagnostic in nature (that can be used to support a diagnostic opinion) is also provided as input for the information fusion tool. This is information that would not in itself give rise to an action but helps the diagnostician in understanding and confirming a diagnostic opinion.

## 3.2 Fusion algorithm architecture

We propose a hierarchical architecture that manipulates the information from the classification tools (initially) and the fused estimate (later) sequentially until the most likely candidate fault has been refined. In particular, this is a process that increases and decreases the weight given to the faults according to the heuristics implemented in the respective layers of the fusion process. This implies also that it is possible for some faults to emanate as winners from one layer, only to be overruled in the next layer. The architecture displayed in Fig. 2 gives an overview over how the fusion is performed conceptually. In particular, there are 8 layers. The order of the layers follows the general flow of preprocessing (layer 0-2), analysis (layers 3 – 7) and post-processing layer 8). Layer 7 is a tie-breaker and is therefore located at the end of the analysis section. The layers were initially arranged according to expert reasoning. However, further analysis may show that a different order may lead to better performance.



**Fig. 2: Diagnostic Information Fusion Process Map**

### 3.2.1 Concurrency

The concurrency layer performs aggregation of information over time and collects information before the information fusion is triggered. In addition, this layer performs a fading of information operation which is performed when certain temporal conditions are met. A challenge in dynamic systems is providing a different reaction to situations where decisions agree and to situations where decisions disagree. When decisions agree, and in the absence of evidence to the contrary, we assert there is no reason for the fusion agent to change the collective opinion within that time step (there is still a minute chance of joint incorrect classification). However, if tools disagree, the fusion main module has to decide whether one tool is correct and the other is not (and which) or – if they are disjoint in time – whether an event has occurred in between the time the two tools’ opinions were made. To help in this situation, we try to support the fusion analysis modules by removing outliers and generally by smoothing decisions of individual tools in situations of agreement and by updating decisions quickly when a changed event is indicated. We implemented the concept of decision smoothing via an exponential averaging time series filter with adaptive smoothing parameter (Khedkar and Keshav, 1992; Goebel and Agogino, 1996). Changes of the smoothing parameter will allow weeding out noise and outliers when no fault has occurred but reacting quickly to changes from one event to another. Therefore, if everything is calm, the smoothing parameter is large (and chiefly serves to remove noise); on the other hand, if some change is encountered, the strategy is to be more cautious and to reduce the smoothing parameter value. This reduction is accomplished with the help of a sentinel that monitors system changes. The sentinel checks if the status of a decision shifts and keeps track of these variations. In particular, a change counter establishes whether there is a change of opinion compared to the last opinion. If the change counter is greater than zero, an intermittency factor is computed dividing the change counter by some user defined sentinel window. The sentinel window is a running window that is small initially to allow operation even when there is only one measurement. This implies that initially the parameter will be more receptive to changes. When more values become available, the window increases and the decision smoothing becomes more temperate. This also means that in general more information is better under this paradigm because longer and more refined smoothing can take place. Finally, a rebate factor is employed that has the effect of reducing the sensitivity of the parameter when no changes have been observed for a while thus allowing the algorithm to settle to its smoothing task. The rebate factor is multiplier with a value less than 1.

The idea of decision forgetting is to discount information as it ages when tools disagree at different times (and no new update of both tools can be obtained). We force the older information to “fade” as a function of time passed (Goebel, 2000). This is necessary to account for the fact that an event may have occurred in between the time two different tools issued their respective system estimates. We postulate that the later decision needs to be given more weight in case of decision disagreement to account for the possibility of occurrence of the event indicated by the later tool. The question is how much more weight that later decision (or how much less weight the earlier decision) should have. We further propose that the discounting is a function of time passed. The reason is that there is a higher possibility for the event to occur when the period is larger and vice versa. Again, we assume that the

decisions are scaled between 0 and 1 and propose to change the forgetting factor as the confidence value increases. The forgetting factor is used within a power operator and has a value close to 1 for small tool confidence and rises to 1.1 (in this example) as tool confidence increases. The particular slope and upper saturation is system dependent.

### 3.2.2 Associating

As mentioned before there is information in a preferred misclassification. That is, if it is known beforehand that a tool B misclassifies fault 1 often as fault 2, then it appears to be prudent to integrate that information into the reasoning process if fault 1 is indicated by tool A and fault 2 is observed by tool B. This information is contained in the asymmetric entries of the confusion matrix. This relation can be utilized by correlating (or associating) the classifier output according to those preferred misclassifications. The correlation is performed by pre-multiplying the classifier output with the respective tool specific association matrix. The latter is calculated by using the negative normalized confusion matrix with unity diagonal elements set to one (Goebel and Mysore, 2001). The effect of this operation is that side-effects of cross-correlation are factored out before the first fused estimate is calculated.

For illustrative purposes, an example case with 2 tools and 3 possible fault cases will be presented. The readings from the tools are (arranged as fault 0, fault 1, and fault 2) are listed in Table 3.

**Table 3: Input to Association layer**

	Fault 0	Fault 1	Fault 2
Tool A	0.6	0.8	0.3
Tool B	0.5	0.4	0.7

The output from the association layer is given in Table 4.

**Table 4: Output from association layer**

	Fault 0	Fault 1	Fault 2
Tool A	0.582	0.720	0.100
Tool B	0.227	0.195	0.700

This modification needs to be viewed in the context of the confusion matrices that for this example are enumerated for both tools (tool A and tool B) in Table 5 and Table 6, respectively.

**Table 5: Confusion matrix for tool A**

Tool A	Est. Fault 0	Est. Fault 1	Est. Fault 2
Fault 0	0.95	0.01	0.04
Fault 1	0.10	0.80	0.10
Fault 2	0.20	0.10	0.70

**Table 6: Confusion matrix for tool B**

Tool B	Est. Fault 0	Est. Fault 1	Est. Fault 2
Fault 0	0.60	0.01	0.39
Fault 1	0.39	0.60	0.01
Fault 2	0.01	0.01	0.98

Using the results from tool B as an example, we can see that the input estimate for fault 2 was the highest (0.7), followed by fault 0 (0.5), and fault 1 (0.4). The confusion matrix for tool B shows that fault 2 is correctly classified in 98% of the cases. Fault 1 is correctly classified only in 60% of the cases and fault 1 is misclassified as fault 0 in 39% of the time. As a consequence, the association procedure recognizes this “pattern” and downgrades the estimates for fault 0 and fault 1 proportional to their misclassification (while also taking into account the classification and misclassifications for fault 0), resulting in a significant downgrading of the confidence for fault 0 and fault 1, but not for fault 2. The same procedure is undertaken for tool A.

### 3.2.3 Scaling

The scaling layer implements functionality allowing a tool that is more reliable for a particular fault to be weighted more heavily than a tool with a worse track record for that same fault (Jeon and Landgrebe, 1999). Clearly, there should be higher trust in a more reliable tool, even when a less reliable tool indicates a strong confidence in its own opinion. However, even the more reliable tool fails occasionally and when the less reliable tool happens to be correct, it would be a mistake to always use the more reliable tool’s diagnostic output exclusively. Then again, it is not known when a particular tool is correct and when it is not. Therefore, we propose to use each tool’s information scaled by the degree of its reliability which we derive from the diagonal entries of the confusion matrix. Following the above illustrative example, the output of the scaling layer is as shown in Table 7.

**Table 7: Output of scaling layer**

	Fault 0	Fault 1	Fault 2
Tool A	0.496	0.380	0.024
Tool B	0.047	0.047	0.646

For interpretation we now look at the modified output of tool A. We again have to examine the confusion matrices where we find that fault 0 is correctly classified most often (95%), followed by fault 1 and fault 2 (80% and 70%, respectively). The output of the scaling procedure reflects these relative classifications where the confidence for fault 0 is discounted least (from 0.582 to 0.496), followed by the confidence for fault 1 (discounted from 0.720 to 0.380) and the confidence for fault 2 (discounted from 0.100 to 0.024).

### 3.2.4 Strengthening

The strengthening layer reinforces an opinion about a fault if it is indicated by several tools. Fault opinions expressed by different tools which all agree should lead to a more confident assessment of the system state. This is the trivial case where coinciding opinions are rewarded. The rewarding scheme is accomplished by calculating the fused value as the sum of the classifier outputs that are in agreement. Recall that the diagnostic output is a value expressing a likelihood for that particular class. That is, it is typically a weighted value.

For the illustrative example, the output for the strengthening layer is presented in Table 8.

**Table 8: Output of strengthening layer**

	Fault 0	Fault 1	Fault 2
Fused value	0.543	0.427	0.670

This output is the first actual fused output where the two tools are aggregated. Note that in this case the tools actually disagree on their state estimate, so the reinforcing character of the strengthening tool does not get triggered.

### 3.2.5 Weakening

The weakening layer discounts information if several tools disagree on the diagnostic state. It also discounts information if any tool indicates several faults at the same time. The weakening layer performs part of the conflict

resolution in cases where tools disagree on the diagnostic state of the system. This is performed – in contrast to the strengthening layer – by discounting entries. Discounting was chosen because conflicting information lowers the conviction in a correct outcome. Therefore the fused value for the conflicting fault(s) will be decreased. Similarly, if several faults are indicated at the same time, their contribution will also be discounted (for systems with a one-fault assumption). The discounting is accomplished in both cases by a multiplier smaller than 1.

For the illustrative example, the output is enumerated in Table 9:

**Table 9: Output of weakening layer**

	Fault 0	Fault 1	Fault 2
Fused value	0.543	0.341	0.536

Here, the confidences are discounted as a result of the weakening strategy responding to the conflicting class estimates.

### 3.2.6 Evidence updating

The evidence updating incorporates information that does not originate from a classifier. Besides primary diagnostic information, many systems are rich with secondary evidential information, much of which is encoded as expert knowledge. Evidential information is information that, in conjunction with the primary diagnostic finding, confirms the diagnostic finding. However, it is not diagnostic in nature. That is, no action would be taken based on evidential information alone. We implemented this strategy by using evidential information only to support a diagnostic opinion. If the evidential information is found not to corroborate the diagnostic finding, it will not be used to discount the diagnostic finding. Take, as an example, the thermal history of a compressor. A compressor fault might make more sense in light of information indicating many cycles on the compressor. However, few cycles per se do not discount a compressor fault indicated by the diagnostic tools.

Similar to the relevance matrix, we propose the use of an evidence matrix that relates evidence to faults. In particular, in the presence of  $n$  evidence items, a  $n \times f$  matrix is formed where  $f$  is the number of faults considered with “1” and “0” entries for relevance and no relevance, respectively. An evidence item may have relevance for more than one fault. A relevance item may must have at least one relevance entry. This evidence matrix provides a convenient way to map evidence to faults. The evidence matrix is then post multiplied by the evidence input. The evidence input is a  $1 \times n$  vector containing the current evidence scaled between zero and one, depending on the strength of the evidence. While desired, the scaling is not required for the operation of this layer and can be omitted where strength of evidence information is hard to come by. To realize the boosting character of the evidence, the fused value is increased by the entries of the evidence matrix and the evidence input, weighted by a scaling factor greater than 1. The scaling factor is tuned to keep the effect of the evidence within bounds. 5%-10% was a good value for our system, but other values may be chosen, depending on the application.

The output of the fused value for the illustrative case is modified by the evidence updating as shown in Table 10.

**Table 10: Output of evidence layer**

	Fault 0	Fault 1	Fault 2
Fused value	0.599	0.376	0.591

Integrated in this result is information from the evidence matrix (see Table 11):

**Table 11: Evidence matrix for illustrative case**

	Fault 0	Fault 1	Fault 2
Evidence 1	1	0	0
Evidence 2	0	1	0
Evidence 3	0	1	1
Evidence 4	0	1	0

The evidence matrix expresses the correlation of the evidence (if observed) with a particular fault. The evidential observation for the particular case was as shown in Table 12.

**Table 12: Evidential observation for illustrative case**

	Evidence 1	Evidence 2	Evidence 3	Evidence 4
Evidential Observations	1	1	0	0

### 3.2.7 Tie-breaking

In case the likelihood magnitude of the two highest ranking values of the fused fault vector are very similar a further refinement can be performed by discounting the less critical fault and/or by discounting the less frequent fault. For that purpose, a ranking of the faults was designed based on criticality and frequency of occurrence. We define criticality as the impact that a particular fault has on the mission. Criticality is scaled between 0 (no impact) and 1. The algorithm checks for a close call of the top ranking faults and then modifies the values in question using a dilation operation that has the effect of increasing the value it is applied to using a power operator. The exponent used in the dilation is a value  $0 < x < 1$  and is calculated as a function of criticality and frequency. Both criticality and frequency of occurrence are derived from statistics of data in the past where frequency is defined per system run and criticality is a subjective measure.

The illustrative case was chosen such that a (typically rare) tie-breaking situation was encountered. The subjective criticality and objective frequency of the fault cases are given in Table 13.

**Table 13: Criticality and frequency of cases**

	Fault 0	Fault 1	Fault 2
Criticality	0.01	0.9	0.3
Frequency	0.9918	0.008	0.0002

With that information, the fused value is modified as shown in Table 14.

**Table 14: Output of tie-breaking layer**

	Fault 0	Fault 1	Fault 2
Fused value	0.599	0.376	0.610

### 3.2.8 Back-Scaling

The fused value has undergone several rounds of modifications in the previous layers which leaves it – while internally coherent – in a state which may not be easily interpretable by a user. To that end, a sequence of backscaling steps is employed. First, the output for each fault is scaled by the number of tools for that class. This increases the relative weight of faults for which there are only few classifiers. This is necessary because those cases are undervalued in the strengthening and weakening layers. Next, the fused value undergoes a dilating operation. The final result is expressed within the 0-1 domain and the faults are ranked from more likely to less likely.

For the illustrative case, the output of the back-scaling layer is enumerated in Table 15.

**Table 15: Output of back-scaling layer**

	Fault 0	Fault 1	Fault 2
Fused value	0.460	0.289	0.583

### 3.3 Fusion Architecture Design

During design of this fusion tool, it was important to establish that each heuristic implemented in the layered architecture did contribute to an improvement of the result. To that end, we broke down the development process into smaller steps and adapted a strategy of complete design cycles for each heuristic. That is, each step encompassed iterations of conception, implementation, and testing. This was done to reduce the re-design necessary after completion of the whole fusion architecture which at that stage would have been more difficult because the influence of individual heuristics would have been masked in the overall architecture. At the onset, it was not clear what effect the addition of new modules of the algorithm had on the overall performance and whether the chosen implementation of that heuristic would move the error into the desired direction. Similarly, the parameters were not known beforehand and at least coarse tuning needed to be carried out for each heuristic. To appraise the performance of the algorithm during development, we needed a metric to assess incremental enhancements to the code. As a result, we established an error metric using expert judgements. It became immediately clear that the design space had enormous size and that it would be difficult to elicit expert opinions for the entire space. To cut down on the number of fault permutations to look at and to at the same time cover the entire design space as exhaustively as possible, we employed a Design-of-Experiment (DoE) approach (Fowlkes and Creveling, 1995) for a conceptual set of faults. Because the diagnostic expert was exposed only to the tool output without knowledge of the faults we call this step the “bottom-up DoE”. The purpose was to capture expert’s reasoning and to get a training set that could be used to design the reasoning tool. Using the experts’ heuristics, the DoE set, and the associated ratings, we implemented successively the heuristics for the fusion tool with constant evaluation of performance. Implementations of heuristics not leading to an error reduction were redesigned and could be rapidly re-evaluated. This cut down significantly on development time and ensured a successful architecture. In a second step, we ran full Monte Carlo simulations (Fishman, 1996, Sobol, 1994) with known inputs (“top-down Monte Carlo”). The Monte Carlo runs were carried out in batch mode for all faults, i.e., a desired (typically large) number of runs was employed and the results were returned in cumulative fashion and automatically converted into FP, FN, and FC. This is particular useful for simulations that operate on the far ends of the probability tails, i.e., for events that do not occur very often. The Monte Carlo simulations were no longer constrained to the conceptual fault set but were used on the actual faults and tools under consideration for the application. This step helped to understand the complex behavior of the various diagnostic tools, establish that the fusion tool worked properly and aid in fine tuning of the algorithm and parameters.

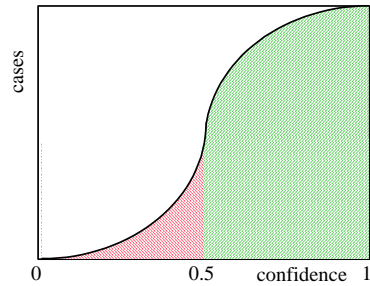
#### 3.3.1 Performance Evaluation

During the conceptual phase of tool development, we used the sum squared error (SSE) as a metric for the DoE phase. To confirm that the fusion tool led to actual performance improvement in the second phase, we devised a benchmark algorithm that performed a maximum wins strategy on transformed and normalized input scaled by the a priori reliability of the tool. An overall performance index was created by weighing the individual components false positives (FP), false negatives (FN), and false classified (FC) where the weightings of the individual components were driven by the application at hand as shown in equation (1).

$$0.6*(1-FP)+0.3*(1-FN)+0.1*(1-FC) \tag{1}$$

The benchmark performance index was set to zero. An increase in performance is measured as the fraction of improvement from that baseline to perfect performance, expressed in percent.

We simulated diagnostic tool output by assuming a Gaussian distribution of fault recognition. That is, if the reliability of the tool for a specific fault was given as 0.8 in the diagonal entry of the confusion matrix, then 80% of the faults were simulated between confidence 0.5 and 1. The other 20% were simulated between 0 and 0.5. Fig. 3 illustrates that example.



**Fig. 3: Simulation of diagnostic tool output**

Note that all modules are evaluated on a stand-alone basis, where possible. This was done to evaluate the contribution of each module in isolation and to ensure that only modules were added that had a positive impact on reducing the overall error.

### 3.3.2 Bottom up DoE

To deal with the combinatorial explosion, we started out with a conceptual problem of 2 diagnostic tools, 1 evidential tool, and 2 possible fault states. Each one of these states could take on 3 possible values (confidences). This amounts to 729 solutions. Even for this very constrained problem, 729 was a number too large for evaluation and hence had to be reduced. This was accomplished by performing a half factorial DoE (Fowlkes and Creveling, 1995) which cuts that number to 45 experiments as can be seen in Table 16. The result of the expert evaluation is seen in the columns labeled “Target 1” and “Target 2” which represent the diagnosis for Fault 1 and Fault 2, respectively. This set was then used to evaluate the first generation fusion algorithm in an iterative fashion as seen in Fig. 4 where the SSE was used as error metric. The highlighted section of the algorithm (with labels provided next to it for better legibility) corresponds to a particular heuristic added (or re-visited) as outlined in the architecture in Section 3.2.

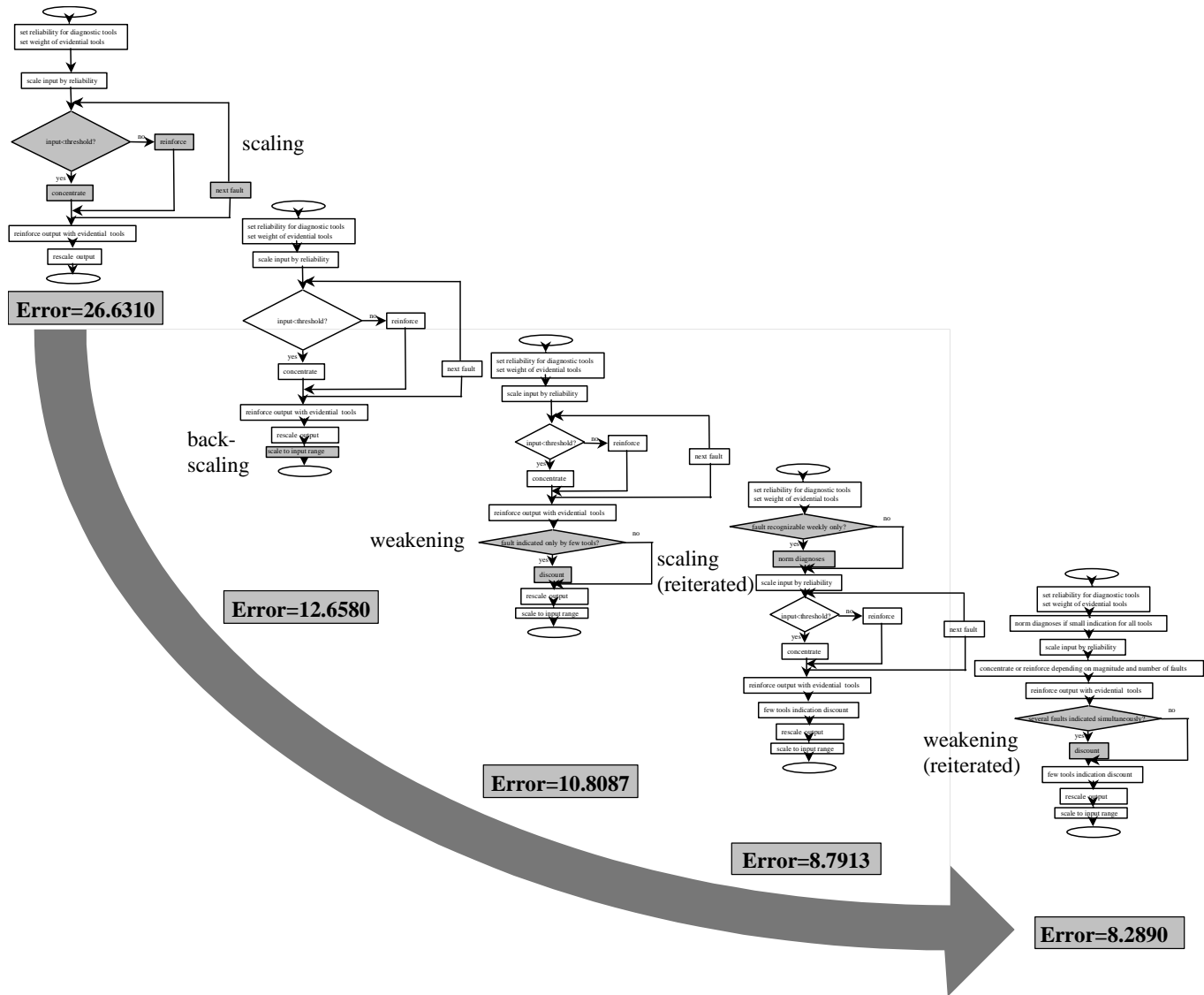
**Table 16: DoE set and expert specified target output**

Tool A Fault 1	Tool B Fault 1	Evidence Fault1	Tool A Fault 2	Tool B Fault 2	Evidence Fault 2	Target 1	Target 2
0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0.6
0	1	0	0	0	0	1	0.3
1	1	0	0	0	0	0	0.9
0	0	1	0	0	1	0	0
1	0	1	0	0	0	0.8	0
0	1	1	0	0	0	0.4	0
1	1	1	0	0	1	1	0
0	0	0	1	0	1	0	0.1
1	0	0	1	0	0	0.5	0
0	1	0	1	0	0	0.3	0
1	1	0	1	0	1	0.9	0.1
0	0	1	1	0	0	0	0
1	0	1	1	0	1	0.5	0
0	1	1	1	0	1	0.3	0

---

1	1	1	1	0	0	1	0
0	0	0	0	1	1	0	0.3
1	0	0	0	1	0	0.5	0.2
0	1	0	0	1	0	0.2	0.1
1	1	0	0	1	1	0.8	0.2
0	0	1	0	1	0	0	0.2
1	0	1	0	1	1	0.5	0.1
0	1	1	0	1	1	0.3	0.2
1	1	1	0	1	0	0.9	0.1
0	0	0	1	1	0	0	0.6
1	0	0	1	1	1	0.4	0.5
0	1	0	1	1	1	0	0.4
1	1	0	1	1	0	0.6	0.4
0	0	1	1	1	1	0	0.7
1	0	1	1	1	0	0.5	0.5
0	1	1	1	1	0	0.2	0.5
1	1	1	1	1	1	0.8	0.5
0	0.5	0.5	0.5	0.5	0.5	0.3	0.3
1	0.5	0.5	0.5	0.5	0.5	0.7	0.3
0.5	0	0.5	0.5	0.5	0.5	0.3	0.3
0.5	1	0.5	0.5	0.5	0.5	0.5	0.2
0.5	0.5	0	0.5	0.5	0.5	0.4	0.3
0.5	0.5	1	0.5	0.5	0.5	0.4	0.3
0.5	0.5	0.5	0	0.5	0.5	0.4	0.2
0.5	0.5	0.5	1	0.5	0.5	0.4	0.4
0.5	0.5	0.5	0.5	0	0.5	0.4	0.3
0.5	0.5	0.5	0.5	1	0.5	0	0.4
0.5	0.5	0.5	0.5	0.5	0	0.4	0.3
0.5	0.5	0.5	0.5	0.5	1	0.4	0.3
0.5	0.5	0.5	0.5	0.5	0.5	0.4	0.3

---



**Fig. 4: Error reduction during sequential implementation phase of heuristics using DoE set**

### 3.3.3 Top-Down Monte Carlo

Next, we simulated the actual system fault states (“top down”) for the full input set with 3 diagnostic tools, 5 evidential tools, and 7 fault states to fine tune the algorithm and parameters. Each fault state was allowed to take on any value between 0 and 1 per architecture design requirements. For achieve realistic tool behavior, we utilized the confusion matrices to create the system response. There were 3 major steps involved:

- For a given fault case, compute the associated z score; the z score is a statistical measure relating to how many standard deviations away from the mean the particular value is located. It is computed by interpreting the associated entries in the confusion matrix as the area under the curve of a Gaussian distribution from which the z-score is obtained through iterative integration.
- Assign random values based on the z score. As an example, let us assume that the reliability of a tool is 95%. This value is obtained from the diagonal entries of the confusion matrix, e.g., 0.95. The interpretation is that the tool classifies correctly in 95% of the cases. Therefore, 95% of the output cases are generated between 0.5 and 1 and 5% of the output cases are generated between 0 and 0.5.
- Carry out the information fusion

The procedure described is repeated many times and the results are compiled in confusion matrix format. The next step involves the evaluation of the results. If necessary, the parameters and/or algorithm for a particular heuristic are adjusted, and the process is repeated until desired system response is obtained.

### 3.3.4 Sensitivity Analysis

During the design stage of the fusion tools the final version for the diagnostic tools were not available yet. This meant that the input to the fusion was not completely known. To evaluate how variations in the performance of the individual diagnostic tools would effect the outcome of the fusion tool we performed a simplified sensitivity analysis. We considered two effects: worse than expected diagnostic tool performance and worse than expected data quality (which is also assumed to result in poorer diagnostic tool output). Although the two effects are not independent in reality, we feel that this consideration will give a notion for the impact of different types of degradation. To keep complexity to a minimum we assumed a uniform degradation of the tool reliabilities and data degradation effects. The confusion matrices were manipulated to reflect the changed behavior and the Monte Carlo was re-run as shown in Fig. 5 where the x-axis and y-axis show the deterioration in percent/100. The z-axis shows the effects on the FP rating. As can be seen, the fusion deterioration is not linear and is able to recover some of the diagnostic tool deterioration experienced. The effect on FN and FC was similar.

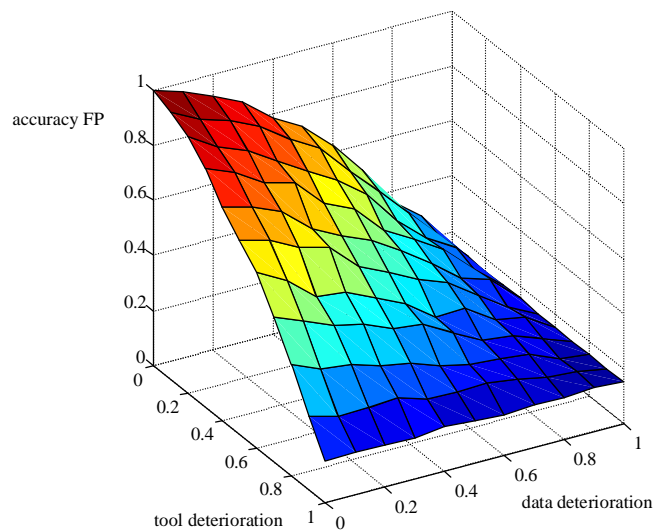


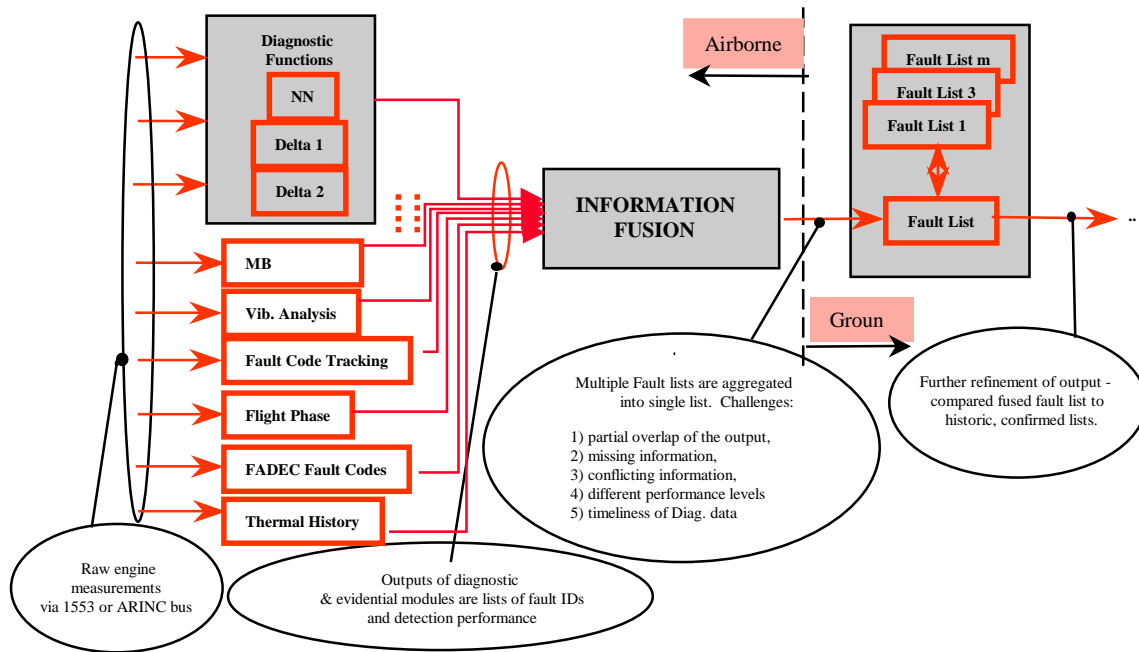
Fig. 5: Sensitivity analysis for FP fusion performance established via input variations on tool and data

## 4. INFORMATION FUSION APPLIED TO AIRCRAFT ENGINE FAULT DETECTION

We applied the principles of information fusion as outlined above to the diagnostic information fusion task for a gas turbine engine. The main goal was to provide in-flight health monitoring capability for gas path faults. Current diagnostic and condition monitoring systems generate information that, while unambiguous in their specific intended application, will be less accurate as more fault coverage is demanded from the tool and less definite as new diagnostic tools are added to either enhance capability or address new faults. This may lead to: 1) ambiguity in troubleshooting, 2) maintenance personnel making uninformed decisions, 3) erroneous component removals, and 4) high operating costs. The fusion effort is one part of an overall project that addresses these problems through the design and test of a condition-based Intelligent Maintenance Advisor for Turbine Engines (IMATE) system (Ashby and Scheuren, 2000). The overall goal of the information fusion in this context was to combine the relevant diagnostic and other on-board information to produce a fault diagnosis estimate to mitigate each of the aforementioned problems thereby achieving a more accurate and reliable diagnosis than through any individual diagnostic tool.

Key system components considered for this health monitoring scheme are the fan, the high pressure compressor, the high & low pressure turbines, and bearings. Thermocouples, pressure and flow sensors as well as wireless micro electro-mechanical systems (MEMS) measure and process data from the components as depicted in Fig. 6. The information fusion module (IFM) demonstrates dual use capability by being designed, and tested on both a commercial and a military aircraft engine (CFM56 and F110, respectively) (Goebel et al., 2000). The gas path faults considered are:

1. Fan fault – Fan blade damage, typically occurring due to bird strikes or other Foreign Object Damage (FOD) during takeoff.
2. Compressor fault – Compressor blade damage or abnormal operation
3. High Pressure Turbine (HPT) fault – Typically a partial loss of one or more blades, most commonly during high power conditions.
4. Low Pressure Turbine (LPT) fault – Typically a partial loss of one or more blades, most commonly during high power conditions. LPT blade faults are less frequent than HPT blade faults.
5. Customer Discharge Pressure (CDP) fault – Leakage in excess of the desired bleed level commanded by the aircraft and communicated to the Full Authority Digital Electronic Control (FADEC). FADEC does not have control over the CDP valve. The CDP valve takes air off the HP compressor for use in various aircraft functions such as air-conditioning, cabin pressurization, and anti-icing.
6. Variable Bleed Valve (VBV) fault – VBV doors not closing according to FADEC issued command, or one or more doors stuck in a particular position. VBVs are intended to prevent low pressure compressor stalls.
7. Combustor Leak (Leak) fault - Holes burnt in combustor liner and hot gas leaks into the bypass duct.
8. Variable Stator Vanes (VSV) fault – Manual errors in installation resulting in small misalignments in vane angles. The VSVs control the amount of air that goes through the high pressure compressor.
9. Inlet Guide Vane (IGV) fault – Manual errors in installation resulting in small misalignments in vane angles. The IGVs control the amount of air that goes into the fan.



**Fig. 6: Information Fusion Architecture for IMATE**

The combustor leak, VSV, and IGV faults are applicable to the military engine only, while the CDP leak and VBV faults are applicable to the commercial engine only; otherwise, the faults are applicable to both engines.

#### 4.1. Primary Diagnostic Input – Summary

Primary diagnostic input is provided by several diagnostic tools that were developed within IMATE. Several other tools (e.g., Bayesian Belief Nets, Fuzzy Expert System, etc.) were down selected at an earlier stage. The parameters of the remaining tools as well as their structure are different for the two engine types (CFM56 and F110). The tools considered are:

1) *Overall Delta (D1)*

D1 is an approach that compares pre-flight with post-flight conditions to arrive at a system estimate.

2) *Delta-Delta (D2)*

D2 is an approach that compares sensor measurements with modeled values. Deviations (“deltas”) are indicative of faults. The comparison is referenced to pre-flight deltas to remove residual deltas.

3) *Neural Net (NN)*

NN is a pattern recognition approach that takes engine measurements as inputs, evaluates them as patterns and – if found suspicious – renders a fault message. It is an absolute approach that allows the detection of a priori faults that may crop up, for example, due to incorrect maintenance. The other two approaches (D1 and D2) cannot deal with a priori faults. In addition, the NN approach can be used during take off where no other technique operates.

#### 4.2 Other Input – Summary

Other system information not diagnostic in nature (that can be used to support a diagnostic opinion) is also provided as input for the information fusion tool. In particular, the following evidential information was used:

4) *Model Based Engine Deterioration Estimator (MBT)*

5) *Vibration Analysis (VIBE)*

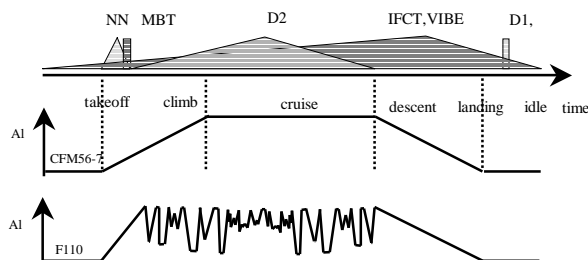
6) *FADEC Fault Codes (FFC)*

7) *Intermittent Fault Code Tracking (IFCT)*

8) *Thermal History (TH)*

9) *Flight Phase Switch (FPS)*

Fig. shows when the diagnostic tools and non-diagnostic information sources are active throughout the flight. The NN is active during take-off only. At the end of the take-off, the MBT gives its estimate. The D2 kicks in after take-off during climb and cruise/mission. The IFCT and VIBE information sources are active throughout the flight and the D1 and TH give their output after landing. The time shown in Fig. 7 line explains also why some of the layers were implemented, in particular the need to deal with temporal information discord which is addressed in the concurrency layer. Also shown in Fig. 7 are the flight profiles (not to scale) of the two aircraft types which are somewhat similar during idle, takeoff and landing, but very dissimilar during the cruise and mission phase.



**Fig. 7: Time line of diagnostic and evidential tools for CFM56-7 and F110**

### 4.3 Output

Table 17 shows the performance gain of the individual layers when tested alone. Using the Monte Carlo approach described in Section 3.3, it was ensured (where possible) that each module led to an error reduction.

**Table 17: Performance gain of individual layers (tested in stand-alone mode)**

Layer	Performance gain	Comments
1	10%	
2	9%	
3	6%	
4	39%	
5	39%	
6	1%	
7	N/A	Evaluation too long
8	0	No performance gain intended

The integrated performance of the layers is not completely additive. For one, each subsequent layer operates on an already improved system. In addition, there is some interdependence of the layers which can in part be explained with conflicting heuristics. The overall performance (established over 10,000 runs) is

- FP=0.01%
- FN=0.20%
- FC=0.20%
- performance index gain =94.71%:

The associated confusion matrix of the fusion module is displayed for the commercial engine in Table 18.

**Table 18: Confusion matrix for fused results**

est\ real	Null	Fan	Comp	HPT	LPT	VBV	CDP
Null	0.9962	0.0000	0.0002	0.0009	0.0003	0.0011	0.0013
Fan	0.0000	0.9986	0.0003	0.0000	0.0002	0.0004	0.0005
Comp	0.0036	0.0000	0.9873	0.0006	0.0008	0.0012	0.0065
HPT	0.0024	0.0001	0.0001	0.9864	0.0073	0.0028	0.0009
LPT	0.0088	0.0004	0.0004	0.0066	0.9797	0.0016	0.0025
VBV	0.0015	0.0000	0.0080	0.0087	0.0047	0.9638	0.0133
CDP	0.0048	0.0000	0.0010	0.0000	0.0003	0.0161	0.9778

Instrumented rig tests have confirmed the operation of the fusion tool. The overall diagnostic task of the integrated main information fusion module improved the result by about two orders of magnitude compared to any individual diagnostic tool alone from an average of 10% misclassifications to 0.1% misclassifications.

#### 4.4. Sensitivity Analysis

The reliability of diagnostic tools may drop due to sensor degradation, faulty data acquisition, environmental disturbances (electromagnetic, ...), incorrect model correction, etc. These effects are expected to result in a performance drop. The degradation of performance was simulated by imposing an average performance drop on each diagnostic tool of 10% and 20%. We observed a drop of the fusion performance of only 2.46% and 3.20% for the two cases, respectively on the three measures false positives, false negatives, and falsely classified faults. This indicates that the performance of the overall fused output is very robust against deterioration.

### 5. SUMMARY AND CONCLUSIONS

We introduced a system and design for fusion of classifier output. For the fusion system, we proposed a layered weight manipulation scheme. This approach helps in situation with conflicting classifier information, temporal information discord (where the estimate of different tools is considerably separated in time), differences in information updates (where the classifiers are updated at different rates), fault coverage discrepancies, and integration of a priori performance specifications, among other things. If a hierarchical model is not desired or if some layers are not applicable, individual layers can be used separately or in any combination for fusion tasks. For the design, we employed a dual approach that first extracted and used expert knowledge to design the individual layers of the fusion tool and to ensure that error reduction was achieved for each layer. Then, a large-scale simulation was exercised to exhaustively test the input scenarios and the fusion tool's response. The two approaches, here called bottom-up DoE and top-down Monte Carlo, do not always need be engaged in the dual fashion. In some situations, the bottom-up DoE may be more appropriate in isolation, in others the top-down Monte Carlo simulation. The test set obtained from the bottom-up DoE has the potential to be usable for validation of a wide variety of diagnostic fusion situations because it is not application dependent. In addition, we view the IFM modules developed as applicable not only to decision fusion. Rather, they might prove equally useful for processing of information on the feature or data level.

Future work will address formulating the ideas presented here in a different framework (for example in a fuzzy logic context). In addition, individual layers can be refined or added; for example, a discounting redundancy penalty could be implemented which penalizes directly correlated output. Currently we only have an implicit assumption that tools must be sufficiently different to be a productive contributor to the fusion module. Because we could influence the design of the classifiers, we postulated the desired output format thus circumventing the need to deal with aggregation of information in different domains. For existing tools, that may not be possible and work needs to be done to address aggregation of heterogeneous information. In addition, the particular application may drive the need to address specific challenges which may be encoded as separate layers.

### 6. ACKNOWLEDGMENTS

This research was in part supported by DARPA project MDA 972-98-3-0002. Views expressed by the author are not necessarily those by the government. The author also gratefully acknowledges the comments of Malcolm Ashby, Kiyoungh Chung of GEAE and Vivek Badami, Michael Krok, and Hunt Sutherland of GE CR&D.

### 7. REFERENCES

- Ashby, M., and Scheuren, W. (2000). Intelligent Maintenance Advisor for Turbine Engines. *Proc. IEEE Aerospace Conf.*, p. 11.0309.
- Fishman, G. (1996). *Monte Carlo : concepts, algorithms, and applications*. Springer-Verlag, New York.
- Fowlkes, W. and Creveling, C., (1995). *Engineering Methods for Robust Product Design*, Addison-Wesley Publishing Company, Reading, MA.
- Freund, Y. and Schapire, R. (1999) A Short Introduction to Boosting. *J. Japanese Society Artificial Intelligence*, 14(5), pp. 771-780.

- Goebel, K. (2000). Decision Smoothing and Decision Forgetting for Diagnostic Information Fusion in Systems with Redundant Information, *Proc. SPIE; Data Fusion, Techniques, Tools, and Architecture IV, Aerosense*, pp. 438-445.
- Goebel, K., and Agogino, A.M. (1996). An Architecture for Fuzzy Sensor Validation and Fusion for Vehicle Following in Automated Highways. *Proc. 29th ISATA*, pp. 203-209
- Goebel, K. Krok, M., and Sutherland, H. (2000) Diagnostic Information Fusion: Requirements Flowdown and Interface Issues, *Proc. IEEE Aerospace Conf.*, p. 11.0303.
- Goebel, K., and Mysore, S., (2001). Taking advantage of misclassifications to boost classification rate in decision fusion. Accepted for publication *Proc. SPIE: Data Fusion, Techniques, Tools, and Architecture Y, Aerosense*. paper 4385-02.
- Hall, D., A, Garga, A. (1999). Pitfalls in Data Fusion (and How to Avoid Them). *Proc. Second Int. Conf. Information Fusion (Fusion '99)*, pp. 429-436.
- Hathaway, R., Bezdek, J., and Pedrycz, W. (1996). A Parametric Model for Fusing Heterogeneous Fuzzy Data. *IEEE Trans. on Fuzzy Systems*, Vol. 4, No. 3, pp. 270-281.
- Jeon, B., and Landgrebe, D. (1999) Decision Fusion Approach for Multitemporal Classification. *IEEE Trans. Geoscience and Remote Sensing*, Vol. 37, No., 3, pp.1227-1233.
- Khedkar, P., and Keshav, S. (1992) Fuzzy Prediction of Time Series, *Proc. IEEE Int. Conf. Fuzzy Systems*, San Diego, pp. 281-288.
- Loskiewicz-Buczak, A., and Uhrig, R. (1994). Decision Fusion by Fuzzy Set Operations. *Proc. third IEEE Conf. Fuzzy Systems*, Vol. 2, pp.1412-1417.
- Nelson, M., and Mason, K. (1999). A Model-Based Approach to Information Fusion. *Proc. Information, Decision, and Control*, pp. 395-400.
- Rahman, A., and Fairhurst, M. (1998). Towards a Theoretical Framework for Multi-Layer Decision Fusion. *Proc. IEE Third Europ. Workshop on Handwriting Analysis and Recognition*, pp. 7/1-7/7.
- Smets, P. (1994). What is Dempster-Shafer's model? *Advances in the Dempster-Shafer Theory of Evidence*, Yager, R., Fedrizzi, M., and Kacprzyk, J., (Eds.), John Wiley & Sons, New York, pp. 5-34.
- Sobol, I. (1994). *A primer for the Monte Carlo method*. CRC Press, Boca Raton, FL.

**Kai Goebel** received the degree of Diplom-Ingenieur from the Technische Universität München, Germany in 1990. He received the M.S. and Ph.D. from the University of California at Berkeley in 1993 and 1996, respectively. He is currently with General Electric's Corporate Research and Development. His research interests include classification, information fusion, and soft computing. Dr. Goebel is an adjunct professor of the CS Department at Rensselaer Polytechnic Institute (RPI), Troy, NY, since 1998 where he teaches classes in Soft Computing. He is a member of ASME, VDI and AAAI.