1/1/2025

CAMBRENSIS

AN "INTELLIGENT" FMV (FRAM MODEL VISUALISER)

for estimating probabilities of outcomes in complex systems. | David Slater and Rees Hill

AN "INTELLIGENT" FMV, (FRAM MODEL VISUALISER) FOR ESTIMATING PROBABILITIES OF OUTCOMES IN COMPLEX SYSTEMS.

David Slater - dslater@cambrensis.org

Rees Hill - rees.hill@zerprize.co.nz

ABSTRACT

The Functional Resonance Analysis Method (FRAM) has emerged as a valuable tool for modeling and understanding the dynamic behaviour of complex socio-technical systems. While traditionally used as a qualitative method, recent advancements in the FRAM Model Visualizer (FMV) have introduced quantitative capabilities, enabling the systematic analysis of functional interactions and variability within a probabilistic framework. This paper explores the potential of FRAM to bridge the gap between human factors specialists, who prioritize qualitative insights, and engineers, who demand numerical rigour for system reliability and safety predictions.

Drawing parallels between FRAM functions and neural processes, we develop a "Neuron FRAM" model that integrates probabilistic aspects of function coupling, inspired by the McCulloch-Pitts neuron model. By representing FRAM functions as computational units capable of transmitting, coupling, and adapting probabilistic metadata, this approach facilitates predictive modeling of emergent system behavior under variable conditions. Demonstrations include the prediction of success and failure probabilities for safety-critical systems, showcasing its practical relevance in real-world applications such as healthcare, aviation, and AI systems.

The proposed framework highlights FRAM's versatility as a system modelling methodology capable of quantifying emergent properties while maintaining its core focus on functional interactions and variability. This integration offers an innovative pathway to enhance the resilience, transparency, and reliability of complex systems, paving the way for broader adoption in domains that require both qualitative and quantitative insights.

Key words – Safety, Risk, Bow Ties, LOPA, FRAM, AI

INTRODUCTION

The Functional Resonance Analysis Method is being increasingly utilised by a wide variety of users for a range of applications. It is becoming of interest to safety professionals who have traditionally relied on methods such as Root Cause, Failure Modes and Effects and Bow ties to try to understand how modern engineering systems will behave in practice. But to this group, the need to quantify the potential failure frequencies is a major requirement. Traditionally this was done by a full Probabilistic (Quantitative) Risk Analysis (PRA/QRA) of the system. But in today's cost conscious environment the resources needed are generally too scarce and expensive for routine applications so alternatives such as Layers of Protection Analysis (LOPA) and semi quantitative assessments such as Risk Matrices or Heat maps have to suffice. It's because none of these alternatives feel rigorous enough, that confidence in their predictions is not high among the professionals using them. It is for this reason that the FRAM approach now seems worth exploring.

To most users, the FRAM approach relies on producing a "picture" (a metamodel) of the complex systems involved, sufficiently rigorous to be able to trace and predict the effects of the interdependencies between the functions interacting in the system to produce the outcomes expected and unexpected. To the engineers this is more like a traditional HAZOP of Process and Instrumentation Diagrams (P&ID's) to trace the systemic behaviour in variable real-life conditions of operation. Both find FRAM approach valid and very helpful ("as done") safety "audits" of ("as imagined") designs and procedures.

So for the mainly "human factors" users, quantification is not necessary and irrelevant, while to engineers, the lack of quantification disqualifies FRAM as a "serious" approach worth looking at. But recently the systematic rigour of modelling the details of the functional interactions that the FAM method enables, has been formalised to include the tracing of the properties and behaviour of individual functions as a process unfolds or develops, quantitatively. This feature is now incorporated in the software (the FRAM Model Visualiser, FMV), employed by most FRAM users.

While this feature is proving invaluable to academic research groups, particularly in Healthcare, aviation, and AI applications such as autonomous vehicles and robot surgery, the main body of users is as yet unaware, or sees no need for this feature, and are discouraged by the need to understand the full details, of yet another acronymed safety modelling method (lost in the alphabet soup).

But to any users of the FRAM approach, an ability to predict the system behaviour in ideal or under varying conditions is its main attraction. But an ability to communicate this is mainly based on conveying a visual understanding of what the FRAM models are implying. Having a more quantitative, consistent measure of these behaviours has obvious appeal, if only as relative ratings for scrutiny and comparison (what if?).

To achieve this acceptance, the basis of the quantification needs to be self-explanatory and acceptable as valid by a wide range of users, from Human factors specialists to engineers.

The Basis of the FRAM Methodology

FRAM is built on the premise that complex systems are defined by their dynamic interactions, inherent variability, and emergent outcomes. Hollnagel maintains that a FRAM built model can be manipulated formally and correctly as a Production Systems were widely used defined as follows:

"A production system (or production rule system) is a computer program typically used to provide some form of artificial intelligence, which consists primarily of a set of rules about behaviour, but it also includes the mechanism necessary to follow those rules as the system responds to states of the world"

Slater has recently suggested that a FRAM built model could be described as a collection of linking interdependent Turing machines. This classical Turing machine analogy bridges concepts from systems engineering and computational theory. Interestingly, this description can also encapsulate the essence of a FRAM function.

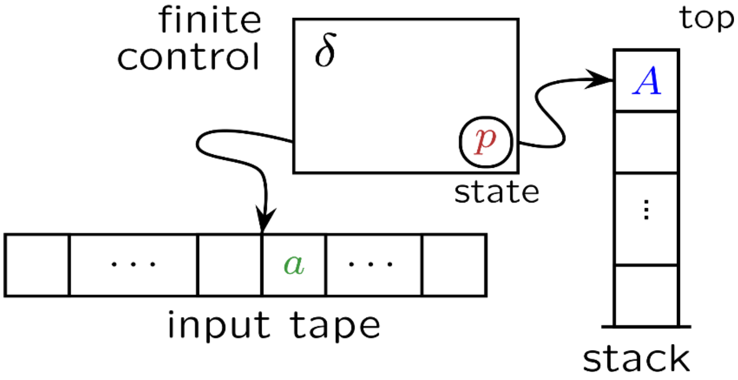


Fig. 1 – A Turing Machine

In FRAM, a function is defined by six aspects: Input, Output, Precondition, Resource, Time, and Control. These aspects form a network of interdependencies that govern the variability and interactions within a system. Conceptually, this aligns closely with the automaton structure. The **Input** in FRAM corresponds to the automaton's set of inputs (**I**), while the **Output** parallels the automaton's outputs (**O**). The states (**S**) of the automaton can be likened to the function's endogenous processing of the inputs to produce the outputs in the operational context, shaped by Precondition, Resource, Time, and Control; information transmitted in the metadata of the functions and processed by the algorithms used. The transitions between states (λ) they produce, and the outputs derived (δ) mirror the dynamic coupling and emergent behaviour captured in FRAM models.

This analogy is particularly compelling because it frames a FRAM function as not merely a static representation, but an active element in the computational fabric of a system. Each function, influenced by variability, transitions dynamically between states, producing outcomes contingent on its interactions with other functions and the broader system context. This perspective elevates the role of a FRAM function to that of an automaton-like entity, capable of representing and interpreting the complexity inherent in socio-technical systems.

Such an approach can be viewed as a generic system modelling approach and emphasises the validity of the FRAM approach as a wider system modelling solution than for just safety applications. But to make it also contextual for the majority of current FRAM users in Healthcare and Human factors we can also use a model familiar to most users of the way information is transmitted in the nervous system of an organism, through the functioning of neurons.

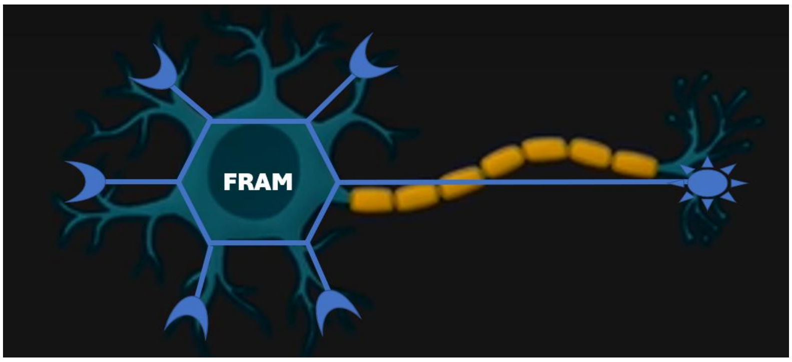


Fig. 2 – The Neuron FRAM function parallels.

THE NEURON MODEL

In 1943, Warren McCulloch and Walter Pitts introduced the concept of the artificial neuron in their ground-breaking paper, "A Logical Calculus of Ideas Immanent in Nervous Activity." This seminal work laid the foundation for the development of neural networks and modern artificial intelligence. The McCulloch-Pitts neuron, commonly referred to as the M-P neuron, was the first formal model to describe how neurons in the brain might work and process information. Its significance lies not only in its innovative approach to understanding neural behavior but also in its role in inspiring decades of research into artificial intelligence and machine learning.

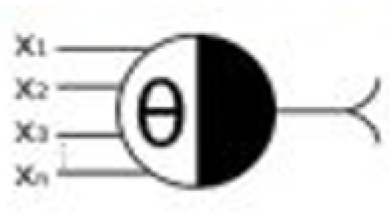


Fig. 3 – The M - P Neuron model

The M-P neuron is characterized by several key features that defined its revolutionary approach. Each neuron processes binary inputs (0 or 1) and produces a binary output (0 or 1), reflecting a simplified model of how biological neurons might operate. Inputs to the neuron are weighted, assigning varying significance to each one. The sum of these weighted inputs is compared to a threshold, and if the sum exceeds this threshold, the neuron "fires" and outputs 1. If not, it remains inactive, outputting 0. This threshold-based mechanism allowed the M-P neuron to perform logical operations, such as AND, OR, and NOT, thereby demonstrating that networks of these neurons could compute any computable function. This insight bridged the gap between biological and computational systems, showing that the brain's mechanisms could inspire computational models capable of vast possibilities.

The introduction of the McCulloch-Pitts neuron marked the beginning of an era of exploration in artificial intelligence. This foundational model directly influenced Frank Rosenblatt's development of the perceptron in 1958, the first artificial neural network capable of learning from data. The perceptron was a pivotal advancement, introducing the concept of binary classification and demonstrating the ability to distinguish between two linearly separable classes of data. Inspired by biological neurons, the perceptron processed multiple inputs, each associated with a weight, and combined these inputs through a weighted sum. This sum was then passed through a threshold activation function to produce a binary output. Crucially, the perceptron included a learning rule that adjusted its weights based on errors encountered during training, enabling the model to refine its performance iteratively.

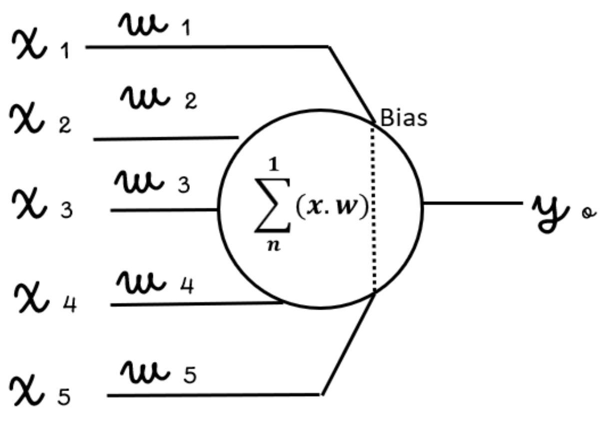


Fig. 4 – The Perceptron Neuron

Despite its ground-breaking nature, the perceptron had limitations. Marvin Minsky and Seymour Papert's 1969 critique revealed that perceptrons could not solve problems that were not linearly separable, such as the XOR problem. This revelation led to a temporary decline in interest in neural networks. However, the 1980s brought a resurgence of research into artificial intelligence, driven by the development of multi-layer networks. These networks overcame the limitations of single layer perceptrons by introducing hidden layers, enabling the modeling of complex, non-linear relationships in data. The introduction of the backpropagation algorithm by David Rumelhart, Geoffrey Hinton, and Ronald Williams in 1986 further revolutionized the field. This algorithm allowed multi-layer networks to adjust weights across multiple layers by propagating errors backward through the network, facilitating the training of deep neural networks.

Multi-layer networks laid the groundwork for the modern era of deep learning. Advances in computational power and the availability of large datasets enabled the evolution of these networks into sophisticated architectures such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs). CNNs, designed for image processing, are now integral to tasks like image classification and object detection, while RNNs, suited for sequential data, excel in applications such as language modeling and speech recognition. More recently, the introduction of transformers and attention mechanisms has further advanced the field, with models like GPT (Generative Pre-trained Transformer) showcasing the ability to generate coherent and contextually appropriate language.

While these advancements have transformed artificial intelligence, they have also raised questions about transparency and trust. Deep learning models, often described as "black boxes," can produce outputs that are difficult to interpret and audit. This lack of transparency poses challenges in domains where trust and accountability are critical, such as healthcare and finance. Efforts to address these concerns have included alternative approaches like the Functional Resonance Analysis Method (FRAM).

THE FRAM RELEVANCE

The functional processes of neurons firing in the brain and the way functions couple in FRAM share striking similarities. Both systems operate in dynamic, non-linear environments where interactions—whether between neurons or system functions—generate emergent outcomes that shape overall behavior. Neurons communicate through electrical impulses transmitted via synapses, influenced by synaptic strength, neuronal state, and network conditions. This context-sensitive coupling mirrors the adaptability of the brain, where minor changes in inputs can produce significantly different outcomes.

Similarly, FRAM functions represent activities within a system, defined by six aspects: input, output, precondition, resource, control, and time. These aspects determine how functions couple, and their variability can lead to diverse outcomes. Like neurons, FRAM functions do not operate in isolation but depend on their network context. Variability in FRAM functions, akin to synaptic variability in neurons, can propagate and resonate across a system, producing emergent properties like resilience or failure.

Both neural networks and FRAM models exhibit emergent behaviours and rely on feedback mechanisms. In neural networks, feedback loops adjust synaptic connections, enabling learning. In FRAM, feedback influences the performance of functions, driving system adaptations over time. Non-linearity and context dependency are shared traits, with both systems demonstrating that small inputs can lead to disproportionately large outputs depending on network conditions.

Exploring the parallels between neuron firing and FRAM function coupling offers insights into the principles of complexity and adaptability that underlie both biological and socio-technical systems. FRAM's emphasis on variability and coupling provides a framework for understanding dynamic systems, bridging biological and artificial domains. By applying these principles, researchers can enhance the resilience and transparency of AI systems, paving the way for innovations that balance performance with accountability.

For example, Nomoto et al. utilized a FRAM model to predict currency exchange rates by weighting FRAM function aspects. This approach demonstrates the potential for combining neural-inspired modeling with FRAM's transparency to create AI systems that are not only effective but also trustworthy. Such applications highlight the importance of aligning AI advancements with principles of transparency and trust, ensuring their benefits are realized across critical domains like healthcare and finance.

THE IMPLEMENTATION IN THE FMV

We know that a FRAM function represents an activity, or process that changes something in the system, and this change can have an effect on any other function in the system with which it interacts. The way the function operates to interact with the rest of the system depends on not just the presence of the necessary Aspects but also what information on upstream functions' status (metadata) is made available through the couplings.

So in our neuron model, we can say that the probability of the output of a function depends on the probability that the correct Aspects are present times the probability of the function itself firing given the information form the upstream couplings. This is shown in Fig.

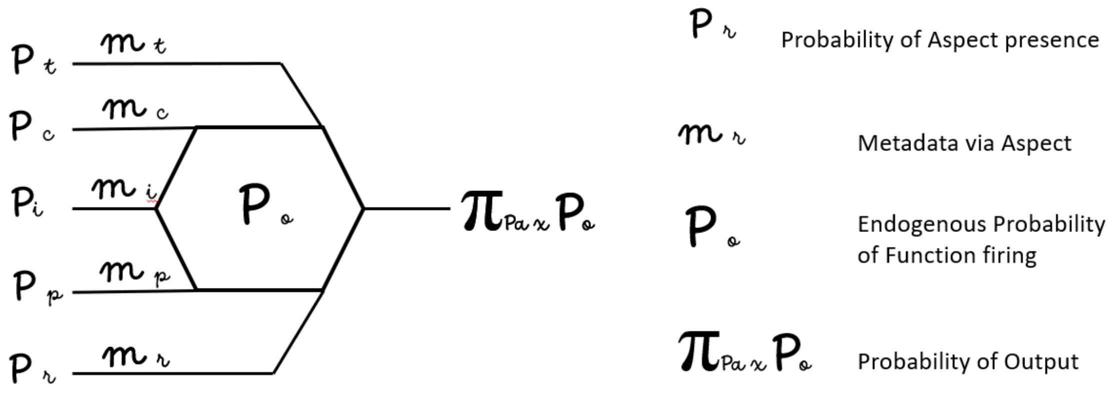


Fig. 5 – The FRAM neuron

The metadata facility sets out to allow us to see the effect of these changes and to follow the consequences of these events. Consider a function has a property that can be modified after its interaction with another function. call this property/parameter a "VALUE" and identify it with a "KEY". The FMV allows us to specify these key–value pairs as metadata (figure 2).

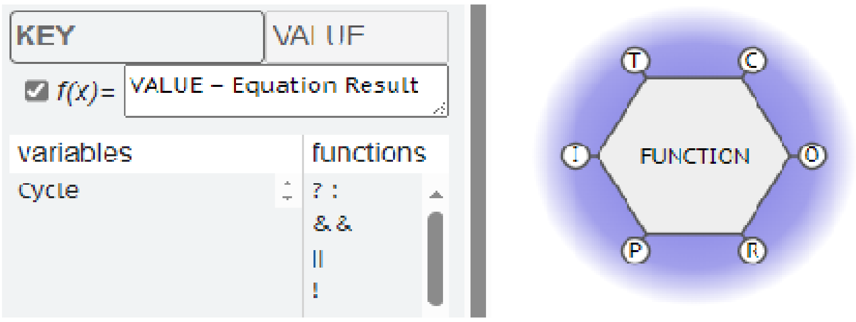


Fig. 7 - metadata key–value pair

So, for our Neuron FRAM Function, the properties we are interested in are: -

- what kind of Function is it, (human, H, Technological T, Organisational O, AI, or User specified X)?

- what is the probability it will transmit, or not a viable output that can link as an aspect to another Function when activated? (Psd/PFD)

- What is the probability that that aspect will be present when needed to activate the next Function? Paspect = Pso (the probability of the linking function producing a viable aspect.

- The probability of failure is then the complement of its probability of success (Pfd)

The details of the way this is processed in the FMV software is set out in Appendix A – The Theory, but if the user is more interested in testing the predicted performance of specific sets of interacting functions under variable limits and constraints of real life conditions, then the software can produce indicative performance predictions in the form pf probabilities of success, for mainstream FRAM users or for estimating probabilities of failure of safety critical systems and barriers "on demand". This will serve to compare and contrast with the simplistic LOPA estimates currently popular but increasingly questioned.

First build your FRAM model

For first time users, this always feels challenging, so for the less experienced, there are now a number of aids to getting started using AI – see the detailed HOW TO papers (   )Most analysts will use the Standard software, the FRAM Model Visualiser (FMV), to display the initial set of functions. As with all good modelling approaches, this is an iterative process as the models will get better and better as the FRAM model enables a deeper understanding of what the system is doing. This model can now be used to input the required metadata. This is set out in the figure below as a series of steps.

STEP 1 - To use the software "short cut", the user needs access to the sandbox version which provides the function templates as prepopulated metadata. So the metadata symbol is clicked.

STEP 2 - This opens the screen shown on which the FMV "Toolbox" symbol can be selected.

STEP 3 - This lists the range of special applications, that the software can facilitate appears – on this select the option FMI Control Neuron.

STEP 4 – This reveals a set of Function "Agents" in the classic FRAM terminology, H = a Human agent, O = a function of the responsible or supporting organisation, T = a function delivered by physical engineering or technical hardware, AI = a function delivered by AI or software.

Clicking on the function in the FMV display area and then on the appropriate HOT button, automatically populates the metadata and algorithms need.

These steps are illustrated in figure 6 – below.

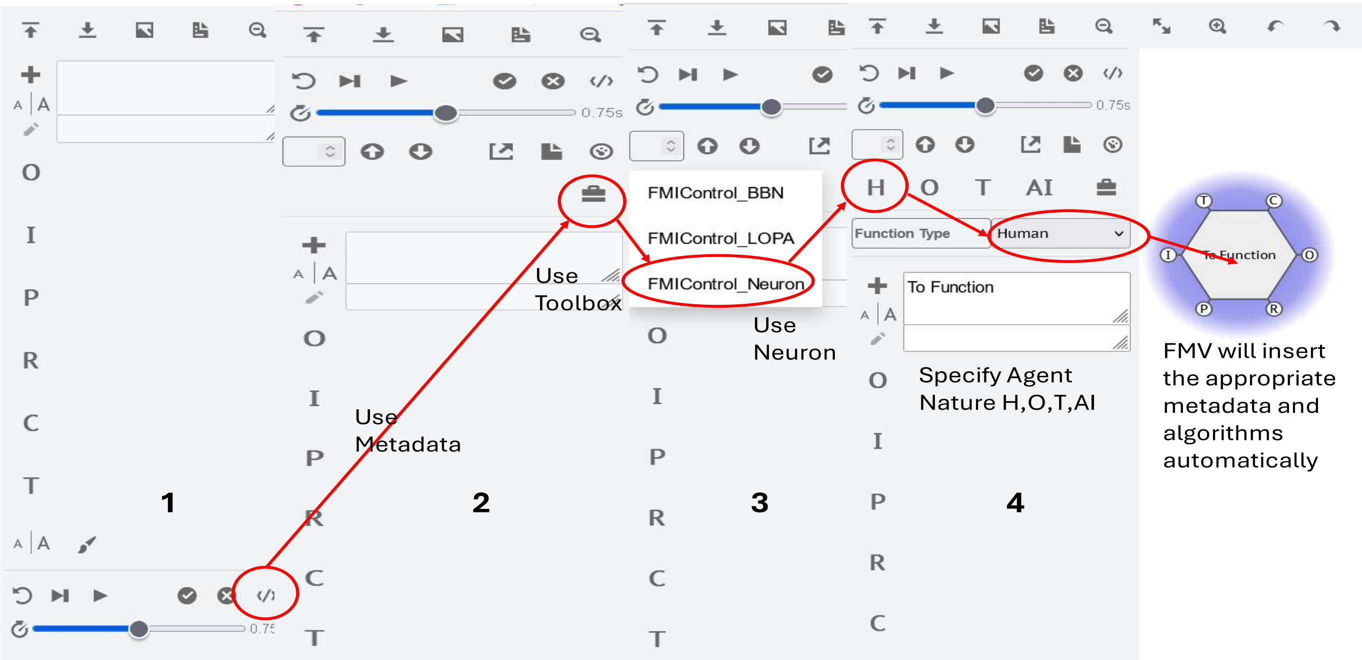
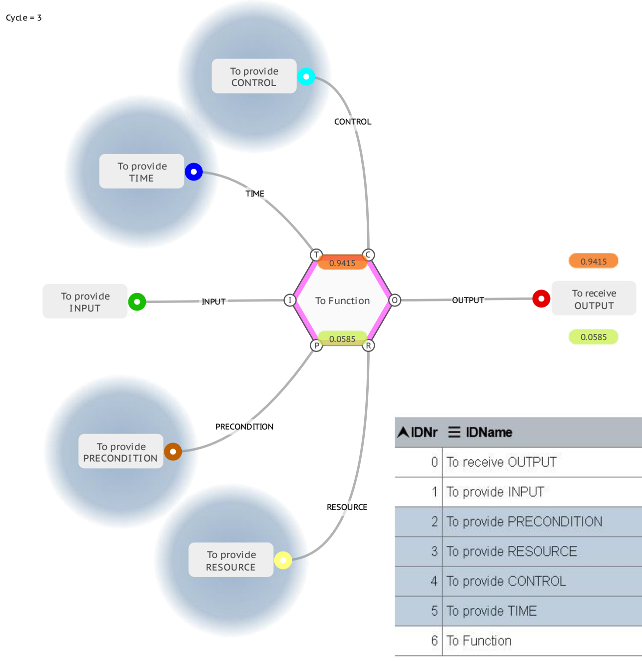


Fig. 8 – The FMV input process

A sample run on a single Human function is shown below in Figure 7.



IDNr	IDName	PFD	Psd	Pbias	Pso	Pfd1	Pfd2	Activate
0	To receive OUTPUT				0.9414801494009999	1.0000000000000002E-12	0.0585198505990011	1
1	To provide INPUT	0.01	0.99	0.9	0.99	0.01	0.010000000000000009	1
2	To provide PRECONDITION	0.01	0.99	0.9	0.99	0.01	0.010000000000000009	1
3	To provide RESOURCE	0.01	0.99	0.9	0.99	0.01	0.010000000000000009	1
4	To provide CONTROL	0.01	0.99	0.9	0.99	0.01	0.010000000000000009	1
5	To provide TIME	0.01	0.99	0.9	0.99	0.01	0.010000000000000009	1
6	To Function	0.01	0.99	0.9	0.9414801494009999	1.0000000000000002E-12	0.0585198505990011	1

Fig. 9 – Single FRAM function illustration

If the finished, populated model is run as normal, it will highlight the probabilities on the individual functions and tabulate them for the analyst to use.

IMPLEMENTATION

A simplified model of a system of four safety critical barrier functions (Figure ---.), was used to demonstrate how the above assumptions determine the probability propagation predicted.

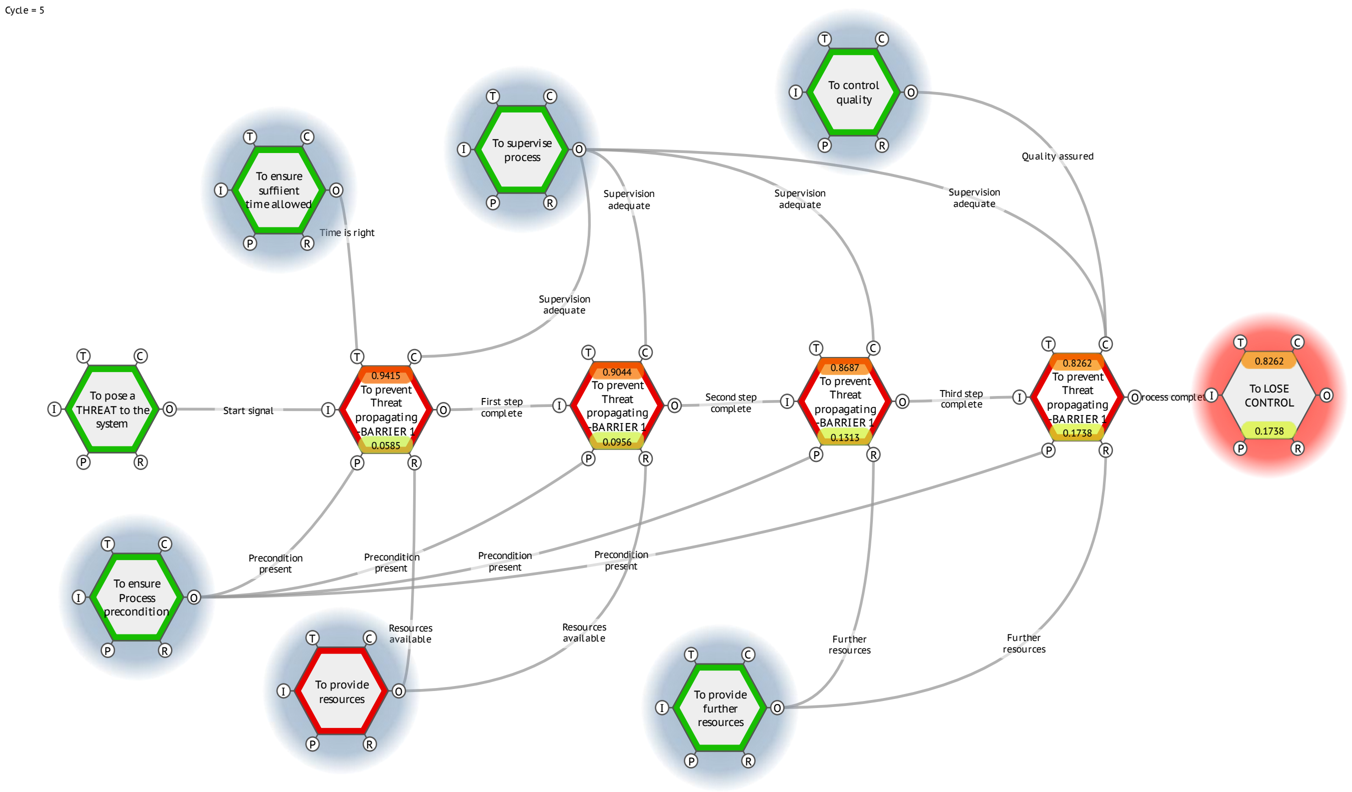


Fig. 10 – LOPA Barrier performance prediction

The Table below sets out the metadata parameters needed to test the viability of the approach.

Table 1. The Predicted probabilities from the unbiased model

Cycle	IDNr	IDName	PFD	Psd	Pso	Pfd1	Pfd2
0	1	To ensure Process precondition	0.01		0.99		
0	2	To pose a THREAT to the system	0.01	0.99	0.99	0.01	0.010000000000000009
0	3	To ensure suffiient time allowed	0.01		0.99		
0	4	To provide resources	0.01		0.99		
0	6	To supervise process	0.01		0.99		
0	8	To control quality	0.01		0.99		
0	10	To provide further resources	0.01		0.99		
1	0	To prevent Threat propagating -BARRIER 1	0.01	0.99	0.9414801494009999	0.0001	0.05851985059900011
2	5	To prevent Threat propagating -BARRIER 1	0.01	0.99	0.9043820750088043	1.0000000000000002E-06	0.0956179249911957
3	7	To prevent Threat propagating -BARRIER 1	0.01	0.99	0.8687458127689781	1.0000000000000002E-08	0.1312541872310219
4	9	To prevent Threat propagating -BARRIER 1	0.01	0.99	0.8261686238355865	1.0000000000000002E-10	0.17383137616441346
5	11	To LOSE CONTROL			0.8261686238355865	1.0000000000000002E-10	0.17383137616441346

But if we now set a tolerance limit for the probability of Aspect presence on all functions of say 90%, we get the result below in Figure 2. where the second Barrier fails and stops the sequence. The individual probabilities are again shown in the Table 2.

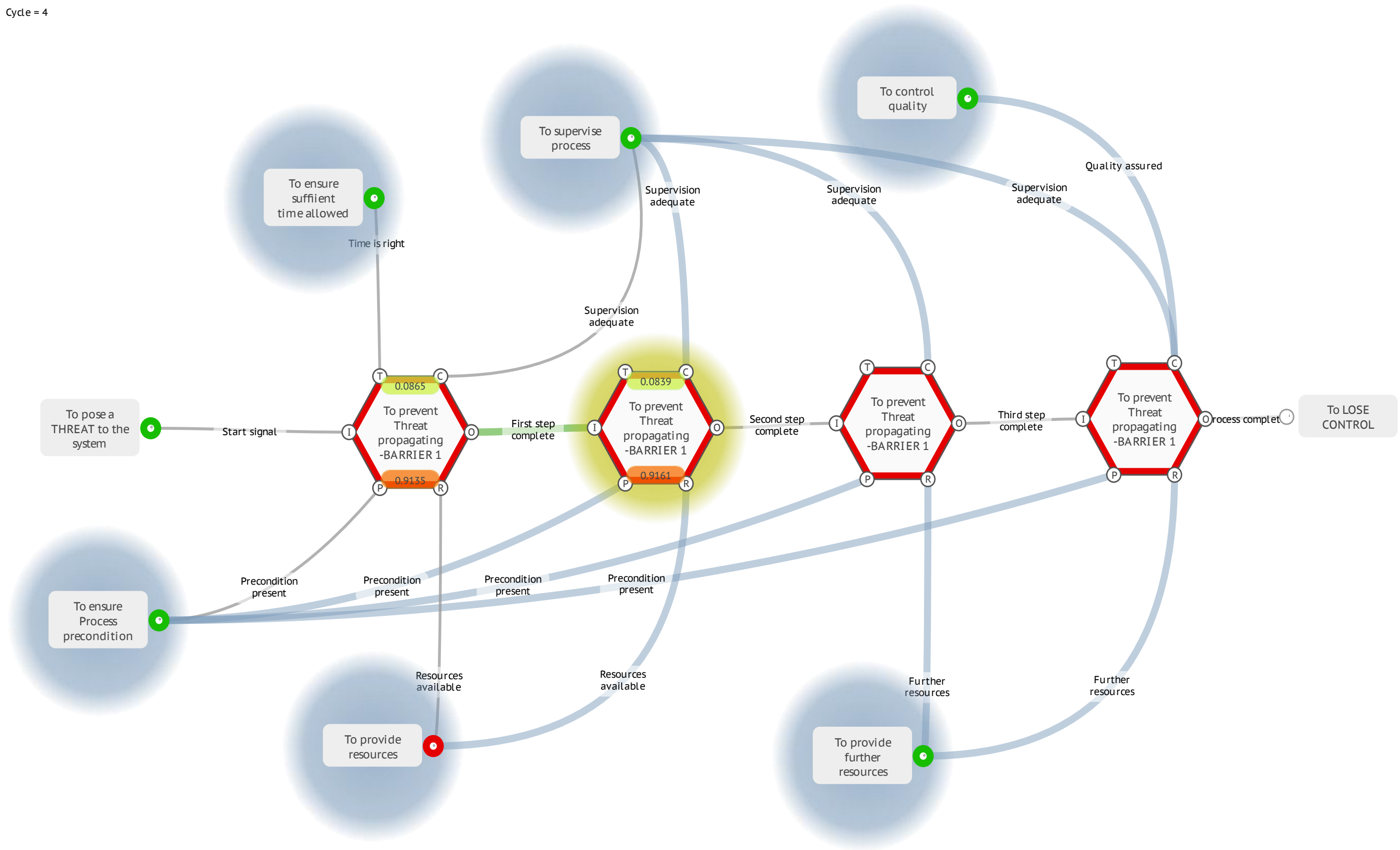


Figure 11. – The biased test model showing the failure at Barrier 3.

Table 2 . The predicted probabilities from the biased test model

Cycle	IDNr	IDName	PFD	Psd	Pso	Pfd1	Pfd2	Pbias	Activate
0	1	To ensure Process precondition	0.01		0.99			0.9	
0	2	To pose a THREAT to the system	0	1	1	0	0	0.9	1
0	3	To ensure suffiient time allowed	0.01		0.99			0.9	
0	4	To provide resources	0.01		0.99			0.9	
0	6	To supervise process	0.01		0.99			0.9	
0	8	To control quality	0.01		0.99			0.9	
0	10	To provide further resources	0.01		0.99			0.9	
1	0	To prevent Threat propagating -BARRIER 1	0.91	0.08999999999999997	0.08645364089999996	0	0.9135463591	0.9	1
2	5	To prevent Threat propagating -BARRIER 1	0	1	0.08388588131162905	0	0.916114118688371	0.9	0
3	5	To prevent Threat propagating -BARRIER 1	0	1	0.08388588131162905	0	0.916114118688371	0.9	0
4	5	To prevent Threat propagating -BARRIER 1	0	1	0.08388588131162905	0	0.916114118688371	0.9	0
5	5	To prevent Threat propagating -BARRIER 1	0	1	0.08388588131162905	0	0.916114118688371	0.9	0
6	5	To prevent Threat propagating -BARRIER 1	0	1	0.08388588131162905	0	0.916114118688371	0.9	0
7	5	To prevent Threat propagating -BARRIER 1	0	1	0.08388588131162905	0	0.916114118688371	0.9	0

CONCLUSIONS

In the search for a system modelling methodology that met some very hard but very desirable criteria has been ongoing for some times. As systems get more complex the challenge gets harder.

- It must be able to model systems that are not just a collection of components, however accurately the components have been described.
- It should recognise that just the involvement of humans makes systems complex by definition.
- It should provide a sufficiently accurate "picture" / overview of how whole systems behave to allow assessment of the effect of changes, intended or unintended.
- It should be versatile enough to apply to a wide range of applications from essentially mind maps to engineering design tools, from Bow Ties to System Integrity Level auditing.
- A significant set of users would require that it is numerate and can forecast results that can be checked against experience.
- The models need to transparently demonstrate a mathematically sound basis for predicting overall system properties.
- In line with the way most safety analysts think about risk, the model should predict expected behaviours in the preferred "language" of the probabilities of occurrence and consequences.
- For safety critical systems it should be capable of predicting system integrity.

For such a demanding specification, the Functional Resonance Analysis Method seems to be the only current candidate that gets near meeting all the criteria.

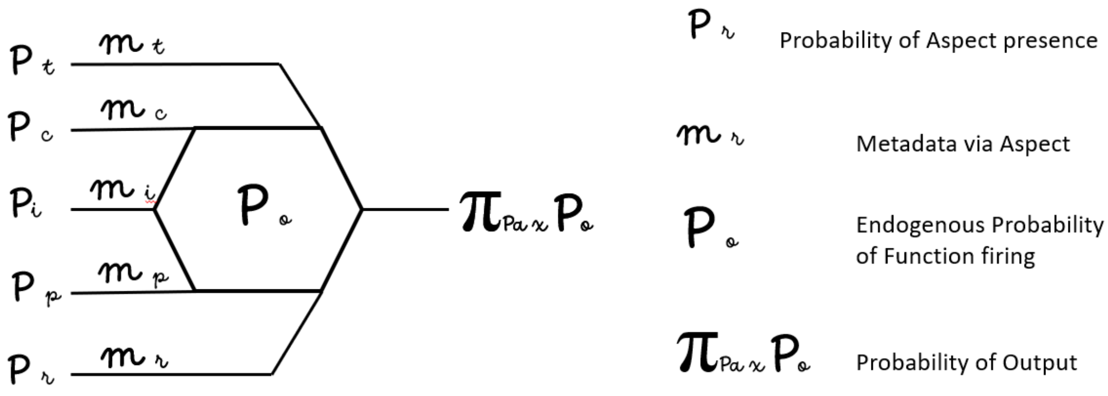
REFERENCES

1. Hollnagel, E. (2012). *FRAM the Functional Resonance Analysis Method: Modelling Complex Socio-Technical Systems*. Boca Raton, FL: CRC Press.
2. Turing, A. M. (1936). *On Computable Numbers, with an Application to the Entscheidungsproblem*. Proceedings of the London Mathematical Society, 2(42), 230–265. doi:10.1112/plms/s2-42.1.230.
3. Slater https://www.researchgate.net/publication/387894837_The_Turing_Analogy_in_FRAM
4. *https://www.researchgate.net/publication/364959115_A_FRAM_HANDBOOK*
5. *https://www.researchgate.net/publication/372079036_FRAM_The_Development_of_the_Complex_System_Analysis_Methodology*
6. *https://www.researchgate.net/publication/385205209_Do_FRAM_functions_mimic_Neurons*
7. *https://www.researchgate.net/publication/372591806_The_FRAM_model_Visualiser_FMV-The_Web_Version*
8. *https://www.researchgate.net/publication/384763526_HOW_TO_USE_THE_METADATA_FACILITY_IN_FRAM*
9. *https://www.researchgate.net/publication/379044204_Using_a_Metadata_Approach_to_Extend_the_Functional_Resonance_Analysis_Method_to_Model_Quantitatively_Emergent_Behaviours_in_Complex_Systems*
10. *https://www.researchgate.net/publication/386905489_The_illusion_of_LOPA_Functional_System_Integrity_Assessment*
11. *https://www.researchgate.net/publication/386503508_The_Paradox_of_PFD's_The_Buncefield_Disaster_A_Case_Study_in_Safety_Failures_and_Systemic_Oversight*

12. *https://www.researchgate.net/publication/385207273_The_Human_in_the_Loop_-the_problem_or_a_solution*
13. *https://www.researchgate.net/publication/385694071_QUANTIFYING_HUMAN_FACTORS_IN_COMPLEX_SOCIOTECHNICAL_SYSTEMS_USING_THE_FRAM*
14. *https://www.researchgate.net/publication/381583186_Building_nonlinear_systemic_Bow_Ties_using_Functional_Barriers*
15. *https://www.researchgate.net/publication/378150797_Real_Time_Barrier_Management*
16. *https://www.researchgate.net/publication/377396290_A_GUIDE_ON_HOW_TO_KWIKSTART_A_FRAM_ANALYSIS_OF_A_COMPLEX_SOCIOTECHNICAL_SYSTEM*
17. *https://www.researchgate.net/publication/387613886_How_can_large_language_models_assist_with_a_FRAM_analysis*

APPENDIX A –

THE THEORY



We can use the Neuron analogy to model the way (and probability) that Functions transmit activating signals - i.e.,

1. Outputs of other functions couple with the FRAM function as Aspects to activate it. All the Aspects required have to be present before it fires. When it fires it makes its output is available to couple with any other Functions in the system needing that **Output** as an essential Aspect.

Output

2. The function has a specific (endogenous), probability of successfully firing, if all the Aspects are present. (**Pout**). For a single function this is the complement of its Probability of Failure on Demand (**PFD**).

Pout = 1 – PFD

3. This affects the downstream Function with which it couples, i.e., the probability of that function successfully firing (**Pso**) is thus, the product of the probability of the Aspect arriving, (**Paspect**) times the probability of that function firing.

Pso = Paspect x Pout

4. For multiple Aspects, the function acts as an AND logic gate and the effective Aspect probability is then the product of the individual Aspects required. The probability of a successful Function Output is now this product (**∏ Paspects),** times the inherent (endogenous) probability that the function will fire.

Pso = ∏ Paspects x Pout

5. If we now add the neuronal Bias, **(Pbias**) and define it as the minimum probability of presence of the Aspects that the function will tolerate, we can add a conditional check that the minimum Aspect probability that the Function will accept as activating is **Pbias.** In which case the metadata key **ACTIVATE = 1** and the function fires. If not, it waits, or seeks an alternative function.

6. The successful output thus becomes its probability of successfully arriving at the next function that requires it.

7. So, we can thus start at the Entry function and calculate the propagation of functions successfully firing until we reach an Exit function. The probability of successfully activating the exit function (**Pso (EXIT**)) represents the overall probability of the system operating successfully.

   Psuccess = Pso (EXIT)

8. The probability that it does not operate successfully, given the bias threshold is met, is then the complement of the above. This has to be the probability of system failure. (**Pfail**),

   Pfail = 1- Psuccess

(Note - Calculating the same entities using the probabilities of failure of the aspects to arrive at the Function times its own PFD, (a LOPA perspective), gives a very different set of figures?)

THE IMPLEMENTATION IN THE FMV

So, for our Neuron FRAM Function, the properties we are interested in is: -

- what kind of Function is it, (human, H, Technological T, Organisational O, AI, or User specified X)?
- what is the probability it will transmit, or not a viable output that can link as an aspect to another Function when activated? (**Psd/PFD**)
- What is the probability that that aspect will be present when needed to activate the next Function? **Paspect = Pso** (the probability of the linking function producing a viable aspect.
- The probability of failure is then the complement of its probability of success (**Pfd2**)
- These are set out in the table below.

Table

Parameter	Key	Equation
Probability that function fails to activate. (or failure on demand (PFD) if preferred?)	PFD	User Input in the FMV
Probability of Function/ System producing required output	Psd	= 1 - Pfd
Threshold probability needed to activate the function.	Pbias	User input
Key needed to activate the function.	Activate	= Amin_Pso>Pbias? 1:0
Probability of function successfully succeeding given the probabilities of the linking aspects	Pso	= Aprd_Pso*Psd
Probability of Function failing, given the probabilities of linking aspects.	Pfd1	= Aprd_Pfd1*PFD
Probability of Function failing, as the complement of its probability of success	Pfd2	= 1 - Pso

DATA SOURCES FOR THE PROBABILITIES

In Layers of Protection Analysis (LOPA), each Independent Protection Layer (IPL) is assigned a Probability of Failure on Demand (PFD), representing the likelihood that the layer will fail when required. Typical PFD values for common IPLs are used in the software:

- **Basic Process Control System (BPCS) Trips**: PFD ≈ 0.1 (fails 1 in 10 times)
- **Safety Instrumented Systems (SIS)**:
  - **SIL 1**: PFD between 0.1 and 0.01
  - **SIL 2**: PFD between 0.01 and 0.001
  - **SIL 3**: PFD between 0.001 and 0.0001
- **Operator Intervention**: PFD ≈ 0.1 to 0.5, depending on training and situational factors.
- **Physical Barriers (e.g., dikes, blast walls)**: PFD ≈ 0.01 to 0.001, depending on robustness.

These values are general estimates; actual PFDs should be determined based on specific system designs, operational conditions, and industry standards. For more detailed guidance, refer to the American Institute of Chemical Engineers (AIChE) publications on LOPA and safety instrumented systems.

Additionally, the Centre for Chemical Process Safety (CCPS) provides comprehensive guidelines on initiating events and independent protection layers, which include detailed discussions on PFD values and their application in risk assessments.

See - AIChE (2020) - SAFETY LAYERS AND LAYER OF PROTECTION ANALYSIS (LOPA) - North Jersey Section AIChE

It's essential to consult these resources and industry standards to accurately assess and assign PFD values tailored to specific applications.

For those interested in using more detail in the Human function characteristics

Human Error Probabilities (HEPs) quantify the likelihood of human errors during task execution and are essential in risk assessments like Layers of Protection Analysis (LOPA). Several methodologies provide standardized HEP values for various tasks:

1. Technique for Human Error Rate Prediction (THERP):

THERP offers detailed tables assigning HEPs to specific tasks. For instance, the probability of omitting a step in a procedure might be 0.003, while misreading an instrument could have a probability of 0.01. These values are derived from extensive data and expert judgment.

https://www.oreilly.com/library/view/risk-assessment-tools/9781118309636/c10_level1_4.xhtml

2. Human Error Assessment and Reduction Technique (HEART):

HEART categorizes tasks and assigns nominal human unreliability scores. For example, a task like "Restore or shift a system to original or new state following procedures, with some checking" has a nominal error probability of 0.003. This base rate is adjusted using identified Error Producing Conditions (EPCs) to reflect the specific context.

https://www.epd.gov.hk/eia/register/report/eiareport/eia_2242014/EIA/app/app12.10.pdf

3. Success Likelihood Index Method (SLIM):

SLIM uses expert judgment to rate Performance Shaping Factors (PSFs) such as training, procedures, feedback, perceived risk, and time pressure. These ratings are combined to produce a Success Likelihood Index (SLI), which is then converted into a HEP. For example, a task with an SLI of 5.55 might correspond to a HEP of 0.0007, indicating a low probability of error.

https://en.wikipedia.org/wiki/Success_likelihood_index_method

4. Absolute Probability Judgment (APJ):

APJ relies on expert consensus to estimate HEPs for specific errors. For instance, the probability of a pilot capturing false information about the final approach path might be estimated between 2.0E-05 and 1.1E-03, depending on expert assessments.

https://en.wikipedia.org/wiki/Absolute_probability_judgement

It's important to note that these HEP values are context-dependent and should be tailored to the specific conditions of the task and environment under analysis. Consulting detailed sources and methodologies, such as the THERP handbook or HEART guidelines, is recommended for accurate application in risk assessments.

Table - a summarized table of Human Error Probabilities (HEPs) derived from various methodologies like THERP, HEART, and general industry practice:

Task/Activity	HEP Range	Source/ Methodology	Notes
Simple routine task with clear instructions	0.001–0.003	THERP	Includes tasks like pressing a button or flipping a switch.
Reading or interpreting instruments	0.01–0.02	THERP	Dependent on display clarity and training.
Omitting a procedural step	0.003	THERP	Low likelihood if procedures are clear and followed.
Complex task under time pressure	0.1–0.3	HEART	Influenced by stress and time constraints.
Failing to follow standard procedures	0.1	HEART	Common in high-pressure or ambiguous situations.
Decision error under uncertainty	0.05–0.2	HEART/THERP	Example: diagnosing equipment failure in a complex scenario.
Restoring a system state following procedures	0.003	HEART	Nominal reliability score adjusted by Error Producing Conditions (EPCs).
Misdiagnosing a problem (initial fault detection)	0.1–0.15	THERP/HEART	Includes factors like alarm design and operator experience.
Operator fails to respond to an alarm	0.01–0.1	THERP	Alarm salience and operator workload play a role.
Failure to recognize a hazard during inspection	0.05–0.3	HEART	Variability depends on inspection training and task complexity.

The Cognitive Reliability and Error Analysis Method (CREAM) is a widely recognized framework for estimating Human Error Probabilities (HEPs), particularly in complex socio-technical systems. CREAM provides a structured way to assess human reliability based on context-specific factors and cognitive demands.

Task/Activity	HEP Range	Context Level	Notes
Highly automated tasks with minimal operator input	0.0001–0.001	Strategic Control	Minimal cognitive involvement; tasks mostly involve monitoring systems.
Routine tasks with clear procedures	0.001–0.01	Tactical Control	Example: routine maintenance or periodic checks.
Tasks requiring moderate cognitive effort	0.01–0.1	Opportunistic Control	Includes responding to deviations or alarms without immediate guidance.
High-stress tasks with significant uncertainty	0.1–0.5	Scrambled Control	Situations with high complexity or conflicting demands.
Misinterpreting an alarm or signal	0.05–0.2	Varies by alarm design and training.	Poorly designed alarms lead to higher error rates.
Failure to execute a planned action	0.01–0.1	Opportunistic/ Scrambled Control	Can depend on workload and cognitive distractions.
Failure to diagnose a fault	0.1–0.3	Scrambled Control	Influenced by complexity and operator expertise.

CREAM Control Modes

1. **Strategic Control**: Operators perform tasks as planned, with sufficient time and minimal cognitive load. Errors are rare (low HEP).
2. **Tactical Control**: Operators respond to predictable variations. Errors are slightly more frequent.
3. **Opportunistic Control**: Operators rely on heuristics, often under time pressure or partial understanding. HEP increases.
4. Scrambled Control: Operators act under high stress, uncertainty, or insufficient information, leading to the highest HEP values.