Proceedings of the ASME 2024 International Design Engineering Technical Conferences
Computers and Information in Engineering Conference
IDETC/CIE2024
August 25-28, 2024, Washington, DC, USA

IDETC/CIE2024-143549

DEFINING A MODELLING LANGUAGE TO SUPPORT FUNCTIONAL HAZARD ASSESSMENT

Daniel Hulse[1,*], Seydou Mbaye[1], Lukman Irshad[2,1]

[1]Intelligent Systems Division, NASA Ames Research Center, Moffett Field, CA 94035
[2]KBR Inc., Moffett Field, CA 94035

ABSTRACT

Functional Hazard Assessment (FHA) is a key early-stage engineering process that supports the incorporation of safety in design by identifying the high-level functional hazards the system may encounter. While many FHA-like methodologies have been proposed in the design engineering literature, many of these methodologies have had difficulty becoming accepted industry practice. Industry standards, on the other hand, either provide too little recommendation on how to represent the function of the system to perform FHA, or rely on existing design artefacts which insufficiently support the goals of the process. This paper presents some of the problems with current modelling languages (both proposed and used) for FHA which limit the scope, expressiveness, flexibility, and precision of the analysis. It then outlines desirable principles an FHA-supporting analysis language should embody, and introduces the Functional Reasoning Design Language (FRDL), a formal modelling language for describing the functional elements of a system and their interactions, which aims to satisfy these principles. To demonstrate the use of this language, the modelling and hazard analysis of a disaster response drone is presented. While this case study is limited in scope, it highlights how FRDL can represent system function while reducing the ambiguity present in typical FHA-supporting functional modelling languages.

Keywords: Hazard Assessment, Functional Modelling, Risk Analysis, Model-Based Systems Engineering

1. INTRODUCTION

Functional Hazard Assessment (FHA) is one of the first analyses performed in the safety assessment process and is used to identify the high-level functional hazards which could occur in a system [1]. This FHA process is a key part of the safety assessment and safety-informed design process for two reasons. First, because it is conducted prior to design activity, it can have a significant impact on driving requirements and decisions supporting system safety (which one would expect to have more impact on the overall system development process [2]). Second, it is performed not just at the product level, but for each system and subsystem [1, 3], meaning the process will be undergone several times in design process as the design becomes more detailed. However, despite the key nature of the FHA in establishing a safety strategy, FHAs are often not revisited in the design process, but instead feed into more detailed analyses of safety such as preliminary system safety analyses (PSSAs) and system safety assessments (SSAs) [4], which can create problems when these assessments are supposed to be cross-checked against the FHA in the Verification and Validation process.

Given its importance, improving the FHA process has significant potential to the overall effectiveness of the safety-informed design process. While FHA-like "function-level" hazard analyses have been performed from the origin of the safety engineering field (as a part of MIL–P–1629 FMECA processes and their derivatives), the delineation of FHA as its own process is somewhat more recent, first being featured in the aviation safety standard ARP-4761 [1] in 1996 and then in the military safety standard MIL-STD-882E in 2012 [3]. In essence, the designer identifies the major functions to be accomplished in the system and the hazardous conditions which could affect them. While the standards do not specify the use of any model or diagram to inform the process, doing so is often necessary for understanding the effects of failure in highly integrated systems [5]. However, the lack of a defined formalism to support this process means that different analysts may create different types of diagrams which abstract the system differently–whether that be the flow charts (or "functional flow block diagrams" [5]) used as examples in ARP-926C [6], containment hierarchies (see examples: [7, 8]), or the control block diagrams used in STAMP/STPA [9].

Over the past 20 years, two major lines of research have proposed various improvements to the system representation used to support early design hazard analysis. In the engineering design literature, much has been done to tie the idea of functional failures (identified in FHA) to functional modelling frameworks (i.e., energy-materials-signals models described in Ref. [10]) proposed

*Corresponding author: daniel.e.hulse@nasa.gov

by design theory [11]. Particularly, models in which the functions of the system act on flows of energy, material, and signals have been demonstrated in the context of identifying faults and their propagated effects through the system [12–14]. In the field of system safety, on the other hand, there has been much activity applying the ideas of system science to the understanding of how hazards arise [15, 16]. One of the most influential methodologies has been Systems Theoretic Process Analysis (STPA) [17], which is based on the Systems Theoretic Accident Model and Process (STAMP) hazard model that considers the feedback structure between processes and their controllers (e.g., software, operators, etc) [18]. STPA frameworks have seen significant interest from industry because of their ability to understand accidents by pushing the scope of hazard analysis from the designed system (i.e., inputs and outputs of a function) to the high-level interactions between the system and its operators.

While efforts in the literature have provided important theory for improving the FHA process, they often run into limitations as the design scope increases from the initiating realm of research interest (e.g., mechanical systems, socio-technical interactions, etc.) to more general applications. In previous work, the authors have developed simulation tools for modelling hazards arising from operators, physical/mechanical behaviors, as well as their interactions [19, 20]. As a practical necessity, this has led to the development of *generic, abstract modelling structures* for representing the functions of a system in a way that is amenable to simulation. This paper applies the insights from this development work to motivate and develop the Functional Reasoning Design Language (FRDL) for representing function structures in the FHA process. The aim of FRDL is to rationalize the development of FHA while accommodating desirable features identified in the literature. The major contributions of this work are thus (1) The identification of issues present in current early design representation used (in practice and in literature) for FHA, (2) The identification of principles that functional modelling languages should embody to address these issues, and (3) The introduction of the FRDL language which embodies these principles.

The organization of this paper is as follows. First, some basic background on Functional Hazard Assessment and related research is presented in Section 2. Then, the issues with these approaches, definition of principles, and proposed FRDL modelling constructs will be presented in Section 3. Next, FRDL will be demonstrated in Section 4 and compared with existing approaches for FHA. Finally, conclusions will be presented in Section 5, along with lessons and avenues for further work.

2. BACKGROUND

This section discusses the state of standards and ongoing research in FHA to motivate and contextualize the development of FRDL in Section 3.

2.1 FHA Standards

The FHA process is defined in the military standard MIL-STD-882E [3], as well as the aerospace standard ARP-4761 [1] and FAA Advisory Curricular AC.23.1309-1E [21], as a process supporting the identification of product-level hazards which may occur, as well as their high-level effects. Generally, these standards do not describe the process of performing the analysis itself as much as the format of the output (e.g., tables, standard

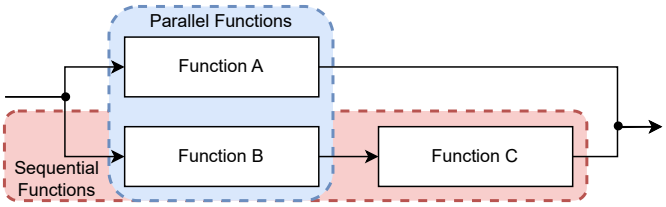

FIGURE 1: Template Functional Model in ARP-926

severity classifications, etc). The analysis process for performing FHA is similar to other discursive hazard analysis approaches such as Failure Modes and Effects Analysis and Hazard and Risk Analysis (defined in the ISO-26262 "functional safety" standard for automotive industries [22]), except that it doesn't include an assessment of rate or probability, since that information is not yet available.

As such, the analysis aspect of FHA may be gleaned from the Fault/Failure Analysis (F/FA) process described in ARP-926 [6], which is meant to generalize FMEA-like failure analyses. ARP-926 describes a "Functional Approach" to F/FA, which may be performed in a "top-down" or "bottom up," sense. In the "top-down" approach, functional analysis is performed that views the overall function of the system as a single, with its input functions on the left of the block and output functions on the right of the block, as exemplified in Figure 15. This is used to determine the hazardous conditions (failure modes, poor inputs, etc.) and their effects in the context of the overall system function (i.e., inputs and outputs from the function). The other, low-level "bottom-up" analysis creates a functional flow block diagram of the system, where the overall function is split into a set of interacting functions in which the outputs of one function feed into the inputs of the next functions, as shown in Figure 3. While these diagrams are similar to EMS models shown in Figure 3 and 2 (e.g., [10, 23]), they apply a somewhat similar convention where the arrows connecting functions are not flows of energy, materials, and signal, but merely represent temporal sequence. Forks in arrows thus represent functions performed in parallel with each other, rather than in sequence.

However, diagramming conventions vary by domain. Notably, for digital systems and equipment (ARP 1834 [24]), functional block diagrams show the interaction of electricity and/or signals between functions. In this domain, signal connections are often specified using *bidirectional arrows*, since communications may go back and forth between the individual functions or components using the same channel. This is different from a functional flow block diagram, where the arrows only flow in one direction through the system. Additionally, fault probability in the domain of software is generally considered to be "impossible to quantify" since they are solely the result of design errors and not physical processes [25]. Recently, in the autonomous vehicle domain, ISO-21448 has introduced more methods to assist with autonomy-related hazards, including the use of systems-theoretic models to examine hazards arising from the interactions between the driver, the vehicle, and the environment [26]. This has been motivated by an increasing interest in accounting for complex interactions that happen at the system/vehicle level to understand the risks posted by autonomous systems.

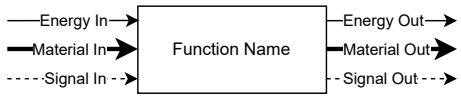

FIGURE 2: Template high-level functional model

To summarize, while FHA-related standards agree in the general approach of identifying the system functions, hazards, and resulting effects, the underlying models used to analyze the system in support of this are not properly standardized and vary by domain, especially as FHA is applied to increasingly complex and autonomous systems. The aim of this paper is to address this issue by providing a single language which can be used consistently across domains while accounting for the complex interactions present in autonomous and highly-integrated systems.

2.2 FHA Literature

Literature on FHA can be broken down into examples of FHA to novel use-cases (such as aeroelastic wings [27], virtual control towers [28], software [29], and AI/ML functionality [30]) and the development of methodology to support the FHA process. While standards do not impose a specific means to model the functions of the system to inform the FHA process, research has developed formal methodologies to improve the process. While the traditional FHA process only requires listing the functions of the system to identify hazards and their effects, it is considered good practice to inform the process using function-flow block diagrams (see Figure 1) [31], especially when designing highly-integrated systems [5]. Methodologies for FHA generally focus on ways to improve the underlying language used to represent functionality, by applying formalized languages and/or modelling and simulation techniques to the analysis.

One of the most common conventions for representing function in the FHA literature is with a hierarchical containment model. In this convention, primary functions of the system are decomposed into sub-functions and organized in a hierarchical tree until an acceptable level abstraction is achieved [7, 32]. One of the obvious failings of this representation is that it doesn't capture the behavioral interactions between functions, and is thus no different than a hierarchical list. As such, methodological extensions to this approach have enabled the representation of relationships between functions. The Goal Tree Success Tree Master Plant Logic Diagram (GTSC-MPLD) provides a more successful consideration of interactions which provides more operations for physical interactions using "AND/OR" gates in the context of an overall function/component hierarchy as well as task analysis for human operators [33]. However, in doing so, requires much more detail and thus loses some of the advantage of a functional perspective. Other frameworks augment the hierarchical model with separate (functional-block-diagram-like) diagrams which include "input/output" and "method/constraint" arrows [34]. While these methods enable some ability to assess interactions, they are also somewhat complex methods and tend to push the designer towards breaking the system down into detail, rather than focusing on high-level hazards one would consider in FHA.

In the engineering design field, there has been much interest in using formal functional modelling languages to support FHA-like analyses. Particularly, energy/materials/signals (EMS) models such as the Functional Basis Engineering Design (FBED) [23] and others [10] have a similar structure to function-flow block diagrams, as shown in Figure 15. The main difference is that when decomposing the overall function into a functional model, they apply a "spacio-temporal" representation that ties the sequence of functions to their input-output relationships, as illustrated in Figure 3, enabling a better understanding of their physical interactions. Using these models has been shown to assist with the generation of FMEA-style hazard tables [35], and have been extended to enable the representation of human actions and errors in the design process [36, 37]. However, EMS-based modelling approaches are somewhat deceptive in the context of failure analysis because they represent the system as a directed graph, which neglects bi-directional propagation behavior and makes it difficult to understand important feedback loops between the system and its environment/user (since these are generally considered to be outside the system boundary).

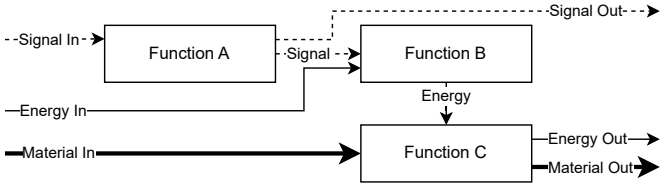

FIGURE 3: Template functional decomposition

Modelling and simulation has been an active area of interest for formalizing, informing, and iteratively developing the FHA. Many of these simulation approaches have been based on EMS-type models. Early methods for simulating hazards in functional models involved creating component models of the system to propagate to the functional level [12]. These models have further been augmented with action sequence graphs to simulate function failures and human error propagation in tandem to support joint errors [38]. A key insight from simulation has been that functions often have inherent associated behaviors which can be modelled without fully specifying components [13], as illustrated in Figure 6. That is, functions represent defined physical behaviors which may be realistically modelled without fully specifying components. This has led to modelling frameworks that rely primarily on function information [19], which in turn have been augmented to further represent human behaviors and hazard by including human-oriented modelling constructs such as action sequence graphs, performance shaping factors, and information networks [20, 39]. Outside of the engineering design literature, SysML has been used to inform FHA [40]. While this has been helpful for enabling a model-based hazard analysis paradigm, SysML has no concept of "function" and one is often left using diagrams (e.g., activity or block diagrams) which don't fully communicate the concept of functionality as having interacting structural and behavioral properties. In the past, this problem has been addressed to some extent by creating an interacting simulation that uses an EMS-based modelling paradigm [41]. Similarly, state charts have been used to simulate behavioral models to inform FHA [42], which is helpful for formalizing behavior but not representing structure, making it difficult to derive failure propagation without modelling it explicitly via states.

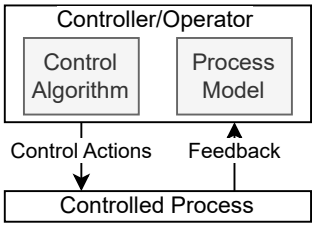

FIGURE 4: Template STAMP model used in STPA

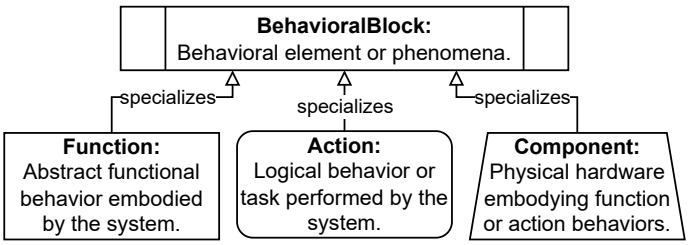

FIGURE 5: Types of blocks representing system structure.

More recently, systems theory-based hazard analysis approaches like Function Resonance Analysis Method (FRAM) and System Theoretic Process Analysis (STPA) have gained attention due to their ability to represent complex interactions between different system elements [43]. STPA [44] uses a control structure where the system is considered a collection of interacting control loops where controllers, controlling processes, and support systems are captured through blocks and interacting feedback arrows, as shown in Figure 4. This enables the assessment of poor control actions–the archetypical cause of "accident"–type catastrophes which have increasing impact as systems increase in complexity. FRAM [15, 45], on the other hand, uses a functional representation where a function is described using six characteristics; inputs, outputs, preconditions, resources, times, and control. the interconnection between functions are represented through the connections between function characteristics (e.g., output of one function is a resource to another) and hazards are considered to emerge due to the variability of these interactions. Because of its representation of time-based dynamics and event sequence, FRAM is somewhat more flexible for understanding the dynamic behavior of systems necessary for understanding resilience [46]. Both STPA and FRAM resolve a major problem in EMS-based models, in that they enable the representation of high-impact hazards that arise from the operator and external environment that would otherwise considered to be outside the system boundary (and thus, out of scope of the analysis). However, one major limitation of these methods is that they tend not to treat the physical/technical aspect of system behavior with as much rigor as EMS-based methods. Nevertheless, these methods have seen increasing interest from industry, with STPA being suggested in ISO-21448 as a way to analyze hazard relating to autonomous and semi-autonomous driving.

3. DEFINING THE FRDL

The development of the functional reasoning design language is motivated by the inherent issues present in existing system modelling formalisms for informing the FHA. This section highlights some of these issues, identifies desired principles for an FHA-supporting language to fulfill, and uses these principles to justify the modelling constructs which make up the FRDL.

3.1 Defining Behavioral Blocks

One core feature of FHA modelling languages is the representation of the overall system and its decomposition into individual elements. From there, these blocks may be arranged as a part of a hierarchy, temporal block diagram (in ARP-926 [47]), function-flow block diagram (in FFDM [11]), or control structure (in STPA [17]). In general, these approaches agree that a function should be a "form-invariant" representation of the overall "purpose" or "task" performed by the system–a feature they share with EMS-based functional modelling languages. However, defining these functions is often an issue for the FHA process, since there is always some difficulty abstracting what is known about the system into these functions [5], which is a result of Problem 1.

Problem 1: *Existing FHA models impose functional abstractions which may not be appropriate in a general design context.* Specifically, there is confusion between the system being made up of "technical functions," which are *functionalities* of the system (i.e., the framework used in the FFDM literature), "tasks," which are discrete modes or events the system performs (a framework more suited to function block diagrams), and components, which are the physical elements of the system (a framework which often happens when engineers do not readily conceptualize the functional abstraction). Often, rules imposed by languages enforcing a single type of abstraction (e.g., a function must be a noun-verb pair acting on inputs to produce outputs, etc.) do not appreciably resolve the ambiguities that create this distinction, leading to incoherent models. This is especially relevant to the design of aircraft, where the overall function of the aircraft is not readily expressed in terms of inputs and outputs and utilized functionality may change over the phases of operation.

Principle 1: *Graphical FHA languages should delineate the different temporal and structural aspects of system function.* Specifically, functions which are considered more as "tasks" have a temporal element (e.g., a state in which the task is "complete" or "failed"), whereas functions considered more as "functionality" must be maintained throughout the operation of the system. For example, an aircraft may turn left and right multiple times to accomplish a given mission, tasks which rely on the existence of pitching and general aviation functionality embodied by wing, aileron, and control components. The distinction between these abstractions should thus be clearly delineated so that they are not confused. As an example, the SysML language differentiates structural and behavioral elements of the system as "blocks" and "activities," respectively, creating a conceptual distinction which avoids confusion. This principle is embodied in Solution 1.

Solution 1: *In FRDL, functions are represented as a type of behavioral element which may contain actions or components.* Functions, components, and actions are abstract behavioral elements of the system, meaning that they *embody phenomena*, as shown in Figure 5. Examples of these phenomena include physical equations (e.g., equations of motion, force balance, etc), logical operations, and tasks (e.g., pressing a button or taking an object to a location). Behavioral blocks are differentiated by the type of behavior and structure they represent. In this formalism, functions thus represent *abstract functionality* which itself may

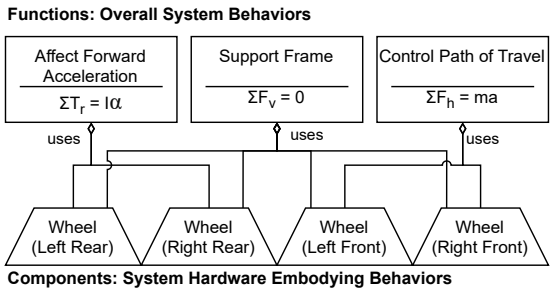

FIGURE 6: Relationship between functions (high-level behaviors) and components in an automotive wheelbase.

be further embodied by actions or components. Actions in turn represent the *logical behavior* or sequential tasks performed by the system over the course of an operation, such as taking off or landing an aircraft. Components, on the other hand, represent the hardware elements which will physically embody the system. The relationship between component and function is illustrated in Figure 6 in the case of wheels in a car, where the functions "affect forward accelleration", "support frame," and "control path of travel" are each based on aggregating individual behaviors from each of the wheels. This behavioral perspective clarifies the role of functions compared to components while imposing a level of formality on what a function can be. Specifically, functions are concrete system behaviors (e.g., movement, acceleration, force balance) which will be in the system regardless of the specific component architecture. However, while delineating behavioral block types enables a more expressive language for representing system behavior, more information may further inform analysis, which motivates Problem 2.

Problem 2: *Existing FHA languages do not convey certain relevant aspects of function behavior.* Generally, FHA-informing languages represent functions solely as boxes with a name for the function, with no further indication of known characteristics of the function. One exception to this is STPA, which specifies one box as a "controlled process" and another as a "controller," which in turn delineates the type of behavior of each box (as controlling versus controlled behavior) and informs the identification of hazardous behavior associated with each function [17]. Given the other aspects of functions (timing, embodiment, and control structure), solely specifying functions as boxes represents a missed opportunity to use these properties to inform hazard analysis, justifying Principle 2.

Principle 2: *FHA languages should include the flexibility to convey structural and behavioral attributes to inform analysis.* Relevant structural and behavioral attributes should be able to be expressed when representing functions to better inform hazard identification. Properties like timing, embodied components, parameters, can all improve the understanding of hazardous behavior. Thus, to inform analysis, designers should be given a means to convey this information–not as a requirement of the FHA process, but as an option to provide more detail. Providing this capability would further help address Problem 1 by giving designers a means to not lose relevant information when applying the functional abstraction. This motivates Solution 2.

Solution 2: *In FRDL, behavioral blocks may be annotated to convey relevant behavioral and structural details.* In order to specify hazard-effecting details which may inform analysis, FRDL enables the use of tags and annotations. These tags are shown in Figure 7 and explained in the next sub-sections.

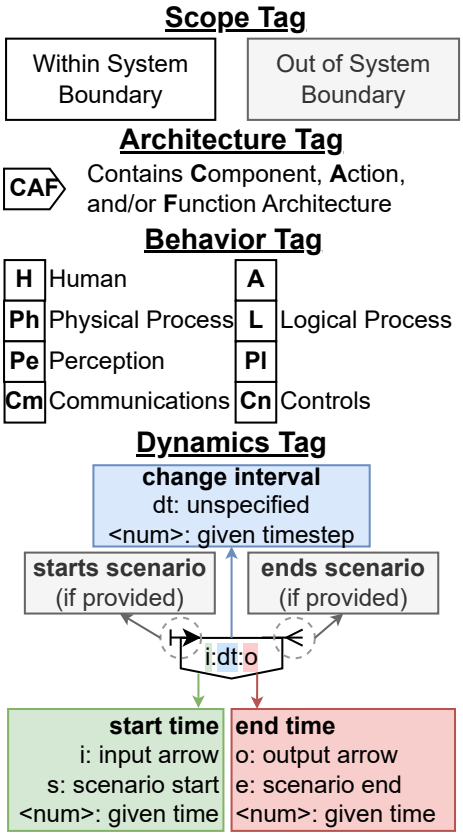

FIGURE 7: Proposed tags for annotating functions

3.1.1 Design Scope. With the increasing autonomous operations of complex engineered system in safety-critical environments (e.g., autonomous vehicles), it is becoming increasingly important to extend the scope of hazard analysis to consider interactions between the system and the environment. However, conveying this from a functional perspective may be confusing–typically, there is guidance to think of functions as intended functionality or "tasks" the system must perform [10]. To consider external interactions outside of the design scope, there should thus be a means to clarify which functions are external versus designed. In FRDL, this may be represented using the "Design Scope" tags shown in Figure 7, where a white box represents a designed aspect of the system, and a grey box represents an external function the system interacts with.

3.1.2 Architecture. One key feature of the functional abstraction is that functions may be broken down into further subfunctions, or, as specified in Solution 1, embodied by components or actions. As this is performed, the overall function may thus be considered to contain architectural information. If these details have been represented somewhere (e.g., on a lower-level diagram), it may be important to convey that information so they may be referenced in the overall hazard analysis. In turn, it should be possible to distinguish functions which have been broken down in detail like that (e.g., a pre-designed function) as opposed to

being treated at a high-level (a function to design). In FRDL, this may be represented using the "Architecture" tags shown in Figure 7, where a C represents a component architecture, an A represents an action architecture, and F represents a functional architecture (see Section 3.3 for definitions).

3.1.3 Behavior Type. As mentioned previously, different types of behavior may have different types of hazards associated with them. In particular, physical behaviors can be modified by physical conditions, and thus have a variety of physical mechanisms associated with them that can cause them to fail. Logical behaviors (i.e., control logic), on the other hand, cannot be modified, except insofar as they are embodied by physical processes (e.g., memory, bit-flips, etc) or implementation (design errors). This information can thus be used to inform the analysis, as exemplified in the literature (see: [17, 20]). FRDL represents this behavioral information using the "behavior type" tag shown in Figure 7, which delineates human/autonomous agents representing types of operators, physical/logical types of elemental behaviors, and planning, perception, communications, and control as sub-operational behaviors. Note that the list of tags provided may be expanded as more behavioral distinctions are identified, and that multiple tags may be used for each function to represent the full scope of behavior within the function.

3.1.4 Dynamics. Finally, an important and related aspect of functional behavior is the dynamics involved. Behavioral dynamics determine important hazard-effecting properties like fault opportunity (i.e., in which phases of operation a fault may arise), timing errors (highlighted in STPA [17]), and resulting effects (as explored in [48]). This time-related information is further important for understanding the resilience of the system (i.e., capacity for recovery). To specify this dynamic information, FRDL uses the "Dynamics" tags shown in Figure 7, which state whether the function or action activates at a given time, updates continuously over a given timestep, and deactivates at a particular time. Note that these specifics may not be known early in the FHA process and thus are not required to be specified in full, and should be considered notional, abstract assumptions used to support an analysis, rather than concrete system requirements. For example, tagging a the function "accumulate water" with a timestep of 1 minute means that the task of accumulating water progresses in the scale of minute, not that it must literally accumulate the water every minute. Further means of specifying the dynamics of function behavior in the context of its interactions are defined in Section 3.2.2.

3.2 Defining Flows

In FHA-supporting languages, flow arrows represent a few distinct but related properties. In function-flow block diagrams, flow arrows represent sequence, while in EMS-based functional models, flow arrows represent "spacio-temporal" information–meaning, the transference of energy, material, and signal and the sequence implied by that transference [10, 23]. This presents ambiguities and limitations which make it difficult to represent bi-directional interactions which unfold over time, as stated in Problem 3.

Problem 3: *FHA languages apply ambiguous definitions of functional flow which conflate "what" with "when" and "how".* Specifically, arrows can represent causality or sequence, but can also represent input-output relationships of energy, materials, and signals. These arrows confuse the analyst's understanding of system failure propagation, because functional interactions are often bi-directional, meaning the functional failures may not solely impact "downstream" functions later in the sequence of arrows, but also "upstream" functions. For example, a short in a light bulb not only affects the optical energy output, but also affects power input from electricity sources. This potential for bidirectional causality is not represented in current FHA-supporting languages in part because they confuse EMS linkages with causality, motivating Principle 3.

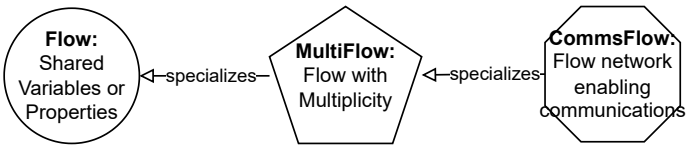

FIGURE 8: Types of flows

Principle 3: *FHA languages should delineate between causality and connections between functions.* It is important to represent both connections (i.e., the "what"–energy, materials, signals and/or shared variables or properties) and causality (i.e., the "how" specifying what properties cause a change in behavior) as linkages between functions. Causality is important for understanding how a change in one block may lead to changes in other (e.g., for advancing to the next step in a sequence of tasks). Connections, on the other hand, are important for representing the mechanisms by which a change in one function's behavior may effect other functions. Both sets of information should be included to support analysis, but they should be delineated to avoid their conflation.

Solution 3: *Causality and connections may represented via flow nodes and connection, activation, and propagation relationships.* Flows in FRDL are defined as nodes which connect behavioral blocks in an overall architecture. Flows represent the shared variables which connect functions, which may be energy, materials, signals, or other shared aspects (e.g., components, objects, aggregations of properties, etc.). As nodes (as opposed to edges), flows can belong to more than two behavioral blocks, which represents an advancement over traditional representations of function (e.g., EMS models). This enables the representation of more complex interactions between functions typical of complex systems, such as variable coupling (e.g., multiple functions in an aircraft sharing the same position, velocity, attitude, etc). The notion of perception and communication relationships is further expressed by delineating types of flows, as shown in Figure 8, which follows the definitions described in previous work [39] for representing distributed situational awareness properties. Here, MultiFlows represent flows which contain multiple copies (e.g., ones perceived by individual functions), while CommsFlows may represent an entire flow network enabling communications (i.e., flow copies communicated between different functions). These containment arrows and flow types provide a rich language for understanding how shared properties flow between functions.

Three main types of relationships are defined to related behavioral blocks and flows, as shown in Figure 9, which represent the connection of the block to the flow, how the flow activates

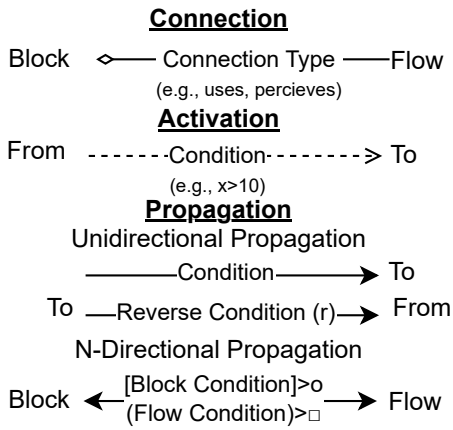

FIGURE 9: Options for representing flow containment and propagation relationships.

blocks (and vice versa), and how the propagation of flow characteristics between blocks. These are described in detail in the following subsections:

3.2.1 Aggregation. With flows defined as variables and properties shared between behavioral blocks, aggregation connections (a relationship borrowed from SysML) may be used to describe whether a flow belongs to a given block. These connections are represented with a diamond at the block end and a line at the flow end, as shown in Figure 9. Connection type annotations may additionally be used to give more insight into how tightly the flow couples the blocks it connects. For example, a "uses" annotation could be used to convey that the block takes the flow as input and modifies it (which may be a strong coupling), while a "perceives" annotation could be used to convey that the block copies the input (which may be a weaker coupling).

3.2.2 Activation. In FRDL, the activation behavior may be specified separately from sharing of flows, to enable a clearer picture of the interactions between functions. Activation refers to how the behavior in one function (or values of a flow) changes or updates the behavior of a connected function. This idea originates from the simulation of tightly-coupled behavioral models, where simulating one functional block often requires re-simulating connected function to propagate failures and achieve consistent behavior. In previous work, this has been accommodated via "static propagation" algorithms in fmdtools functional models [19] and conditions in action sequence graphs [20]. Outside of simulation, conveying propagation has potential to rationalize hazard analysis process by making the tabulation of effects from initiating causes more legible and explicit. To achieve this, FRDL provides the activation arrows shown in Figure 9, which may be used to describe how a condition from one block will result in changed behavior in another block via their flow connections.

3.2.3 Propagation. Finally, propagation arrows, shown in Figure 9 represent both the connection of blocks and flows, as well as the activation or update behavior that is carried by the flow. Propagation arrows are represented with an arrow along with the activation condition(s) it carries, may be unidirectional or n-directional, and can optionally additionally be annotated with type tags (e.g., `<uses>`) at the end of the edge corresponding

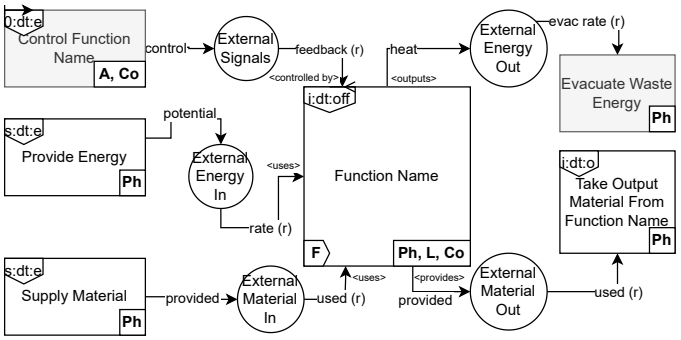

FIGURE 10: Function-In-Context Diagram for Figure 2

to the behavioral block, as shown in Figure 10. Unidirectional arrows convey a single reversible direction of propagation, with activation conditions propagating along the direction specified by the arrow and reverse activation conditions (marked with (r)) propagating in the reverse direction.

N-directional propagation arrows carry multiple activating conditions between blocks and flows using the notation shown in Figure 9, where `[condition]>` ∘ represents a block condition activating a flow and `(condition)>` □ represents a flow condition activating a block. While propagation arrows convey the same information as activation and connection arrows, they are recommended for use when flows connect more than two blocks or when a block is expected to be decomposed in a lower-level architecture diagram, since they concisely group multiple related flow properties.

3.3 Architectures

In the FHA processes, various representations are used to show how an overall function is embodied by sub-functions. These sub-functions are in turn used to identify hazardous scenarios which may effect the overall function. As described in Problems 1 and 3, this is in part due to inadequate function and flow abstractions, which may be solved (Solutions 1 and 3) by defining different types of blocks (function, component, and action) in terms of behavior and structure and separating flow containment from the propagation behavior. When taken together, the resulting language provides a means of describing system architecture described in Solution 4.

Solution 4: *System architectures are represented with bipartite graphs of blocks and flows, where edges may represent containment and propagation relationships.*

In FRDL, architectures are represented using bipartite graphs of functions and flows to better express behavioral interactions between functions. A template for this representation is shown in Figure 11. In this representation, flows may be shared by multiple functions without (on its own) conveying an input-output relationship. Instead, the propagation information is specified explicitly via propagation arrows. This is an important consideration for modelling highly-coupled physical systems, where different properties flow though the system in different directions. For example, in a valve system, water may flow in a defined spacio-temporal direction, but a blockage of water may cause a backup of pressure, leading to faults considered "previous" in the functional flow. This overall system representation may be specialized by block type to further describe different types of system architectures.

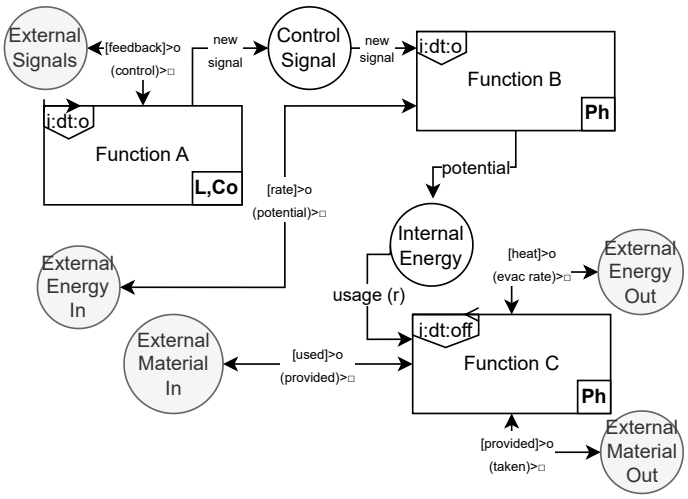

FIGURE 11: Functional Architecture Corresponding to Figure 3

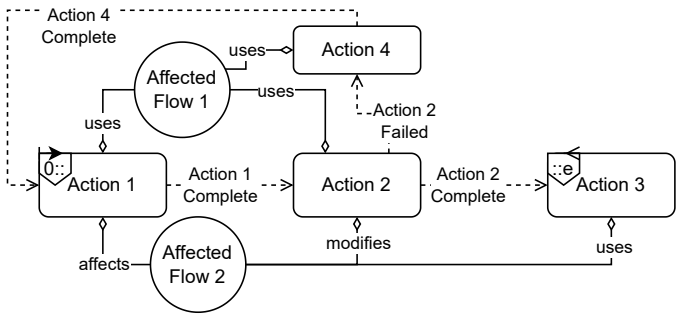

FIGURE 12: Template Action Architecture Diagram

3.3.1 Architecture Types. Considering the three types of behavioral blocks presented in Section 3.1, three main types of system architecture diagrams may be defined–functional architectures, action architectures, and component architectures.

Functional Architecture Diagrams show the functions of the system and their interactions via flows and propagation/activation arrows. These diagrams are important for conveying the high-level behaviors of the system and their interactions with each other, as illustrated in Figure 10 and Figure 11. One common characteristic of function architecture diagrams is that interactions between functions may be based both on updated information (e.g., if a flow changes in one function, it changes another connected function) as well as the unfolding of behavior over time. As a result, it can be useful to use the time-based behavior tags (in addition to propagation arrows) to specify how these functions may be activated. Two major functional architecture diagrams may be used to support the FHA process. In the initial top-down analysis of hazards, a Function-In-Context Diagram, which shows the overall functionality provided by the system and its interactions with external functions (see Figure 10) may be developed to support the process. In the bottom-up analysis of hazards, on the other hand, the Functional Architecture diagram may be used to analyze the individual functions of the system and their interactions with each other (see Figure 11).

When analyzing the logical behavior of the system, **Action Architecture Diagrams** may be constructed to convey the sequence of tasks performed by the system over time. A template

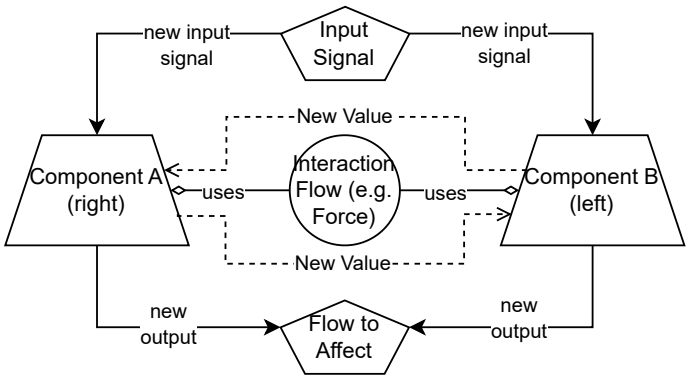

FIGURE 13: Template Component Architecture Diagram

of this diagram type is shown in Figure 12. One unique property of action architectures is that the propagation arrows are often based on completion or performance of tasks, rather than new inputs or output flow values. This is because action architectures are generally used to represent how the system transitions between states over time.

Finally, when analyzing the interactions between specific components in a function or set of functions, an **Component Architecture Diagram** can be constructed, as shown in Figure 13. Note that component architectures may be similar to functional architectures, since they show a static view of behavioral propagation across physical properties of the system, rather than tasks accomplished over time. The main difference is that the functional view summarizes an overall behavior (e.g., locomotion) fulfilled by a component architecture (e.g., wheels on a car). Since components may contribute to multiple functions, components may be featured in multiple component architectures contained in different functions which would take on different characteristics (e.g., wheels on a car both contribute to moving the passengers as well as supporting them vertically).

These architectures represent the propagation of behavior between different aspects of the system. In general, these views of the system may be used to form an overall hierarchy of abstraction, where the functional decomposition is used to represent the overall, integrated behavior of the system, while component and action architectures are used to represent the behavior of individual functions–specifically, component architectures are used to represent the *embodiment of functions as hardware* while action architectures are used to represent the *tasks or sequence of operations defining controller and/or operator functions*.

3.4 Summary and Process

FRDL thus provides a unified, integrated model for understanding behavior in complex systems at varying levels of abstraction to support design activities. Using this language, one can define the structure and behavior of the system to inform hazard analysis both in early design (when one is interested in analyzing overall behavior and interactions) and carry it into the later design stages (when one is interested in analyzing the behavior of subsystems and components). Using FRDL to support the traditional hazard analysis process would thus involve going through the process shown in Figure 14 (though it should be noted that FRDL is merely a language and could, in theory, support many different types of analyses).

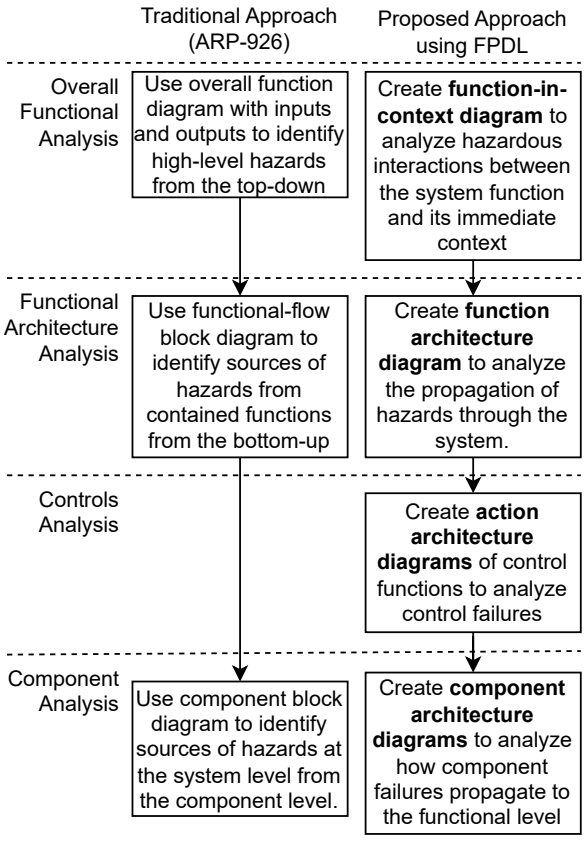

FIGURE 14: Proposed FHA process using FRDL.

This approach has a number of advantages over approaches used in practice as well as in literature. Unlike traditional FHA approaches, the formalism FRDL provides can be used to perform controls analysis, a capability which is becoming more and more important as systems become more automated. While this analysis can already be performed using an approach like STPA [17], FRDL's unified language means that the analysis of controls can be inherently integrated with component and functional analyses, rather than requiring a separate model and process. A major advantage of this unified view is that it means there can be direct, internal consistency between the different types of analyses that carry through the safety analysis process from the earliest (functional stages) through detailed design. This in turn should enable a streamlined V&V process, since high-level and low-level analyses of hazards can be kept consistent throughout design, rather than having high level analyses be abandoned as the design becomes more detailed, only to then need to be brought into consistency as a part of V&V. While this unified view is possible to achieve in simulation (see: [20, 38]), the setup cost for a simulation is relatively high compared to a diagram and existing diagramming approaches have not supported this ability very well. Thus, FRDL has potential to encourage the transformation of safety analysis from its existing document-based approach process into an agile, model-based process.

4. DEMONSTRATION: DISASTER RESPONSE DRONE

This section demonstrates the use of the FRDL language on a drone carrying out a wildfire surveillance mission. In this demonstration the drone is tasked with autonomously flying to an active

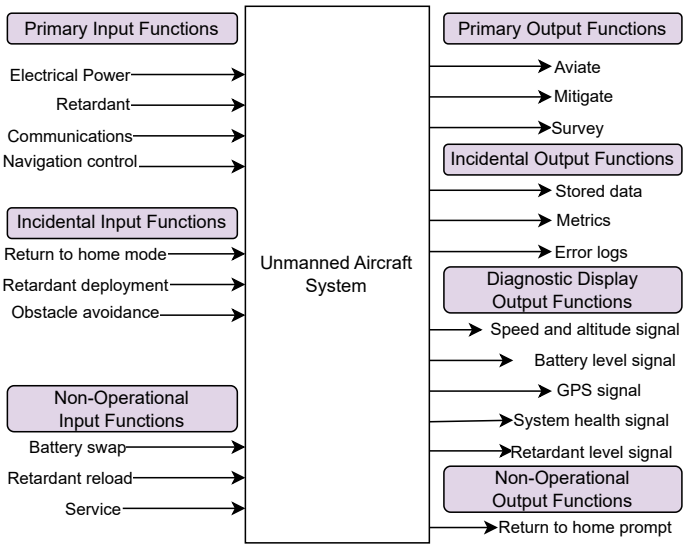

FIGURE 15: Drone Function Diagram

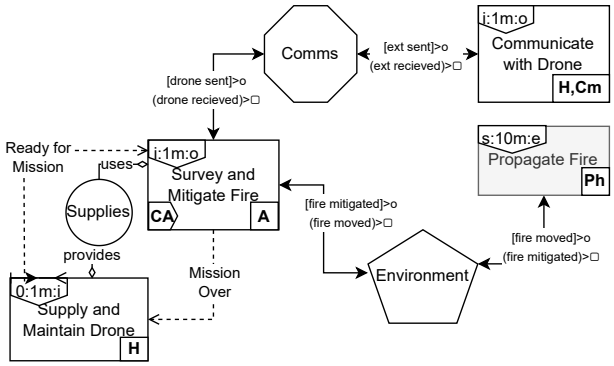

FIGURE 16: Drone Function-in-Context Diagram

fire, mapping the area with onboard surveillance technology, and providing feedback to the ground crew about the evolution of the fire. While the drone acts autonomously, the ground crew is tasked with inspecting the drone and ensuring its airworthiness pre-flight, as well as defining overall mission scope (surveillance, retardant drops, etc.). In-flight, the system is equipped with self-diagnostic tools that will gauge critical parameters such as battery health, amount of available retardant, among others, and initiate a return-to-home command if any of those drops below a safety threshold. The hazards in this drone have been analyzed in previous work, which proposed the use of Model-Based Systems Engineering for Fault Tree Analysis and FMECA [49, 50]. One issue with that work was that it relied on subjective interpretation and judgement to assess the consequences of hazards. The aim of this demonstration is to show how FRDL can better inform this kind of assessment by unambiguously formalizing the behavioral interactions between functions in the context of the overall FHA process shown in Figure 14.

4.1 Overall Functional Analysis

Overall functional analysis defines the function of the system and its inputs and outputs to identify and evaluate high-level hazards. In ARP-926, these inputs and outputs are functions used by the system (primary, incidental, and non-operational) and functions provided by the system (primary, incidental, diagnostic, and

non-operational), as shown in Figure 15. In the FRDL approach, on the other hand, the overall function of the system is presented in the context of its interactions with external functions, as shown in Figure 16. These diagrams have slightly a different scope and properties which will affect hazard analysis–while the drone function diagram in Figure 15 provides a more detailed accounting of all inputs and outputs by eliciting the different categories, it conveys very little in terms of how the function will interact with its environment and operators. Furthermore, the idea of "input" and "output" functions is confusing, since it lacks timing information. On the other hand, the function-in-context diagram in Figure 16, provides much more information about the interactions and timing of the functions it performs, which is helpful for understanding how external factors (e.g., misperception of environment, failure to receive messages) may drive drone failure and why (e.g, sensor failure, communications interference). Note that while this may appear to come at the expense of detail, these details may be further added by further defining the properties of the individual flows (e.g., defining supplies to include battery swap and retardant reload).

4.2 Functional Architecture Analysis

Diagrams conveying architectural information are used to inform the bottom-up analysis of how failures in sub-functions lead to overall functional failure. This would be achieved for the drone per ARP-926 using a function-flow block diagram, as shown in Figure 18. In this work, a function architecture diagram as shown in Figure 17 is proposed to take the place of the function-flow block diagram. While a full assessment of hazards is out of the scope of this short demonstration, it should be apparent how much more informative the function architecture diagram is conveying than the function block diagram. Specifically, by providing much more information in a much more formalized way, the function architecture can be used to trace hazards from their originating function through the system architecture.

As an example, consider a motor failure in the "Aviate" function. As depicted in Figure 17, it can easily be traced to the inability to affect the environment (and thus, drop retardant and execute mission), as well as potential for adverse energy usage. Considering an analogous fault in the "acquire thrust" function of the function block diagram in Figure 18, can only readily be traced to being unable to fly to the fire area and drop retardant. This illustrates how the FRDL-based function architecture can be used to better identify failure propagation paths through the system. However, it should be noted that the FRDL diagram requires more set-up effort to define the diagram. The benefit of this effort is that it provides a more formalized design artifact which can continue to inform the design process as the system becomes more detailed.

5. CONCLUSION

This paper highlighted some of the challenges using existing functional representations of systems to support hazard analysis. In hazard analysis standards, the representation of the system is often an informal block diagram or containment structure, which both provide an inadequate understanding of system interactions and behaviors for understanding how hazardous conditions propagate through the system. To address these challenges, this paper proposes the use of the Functional Reasoning Design Language

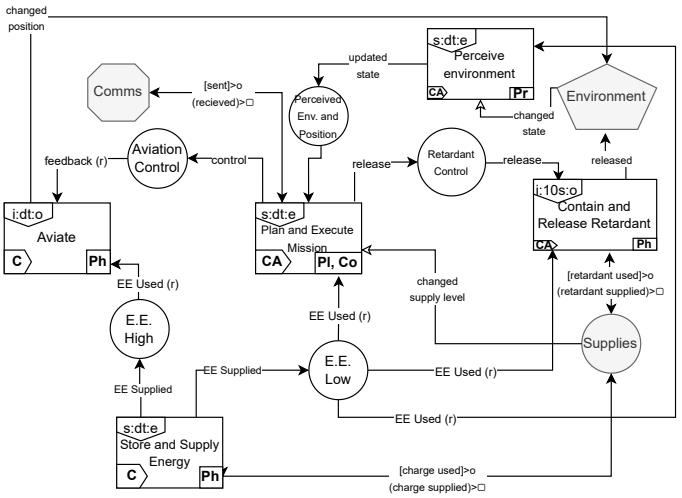

FIGURE 17: Drone Functional Architecture

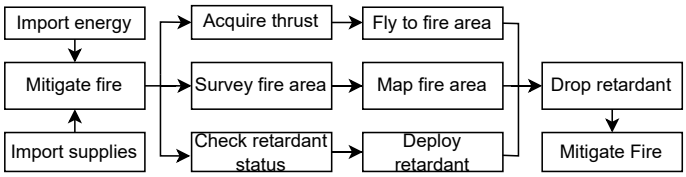

FIGURE 18: Drone Function Block Diagram

(FRDL) to represent system function and interactions in support of hazard assessment. This language was developed based on lessons learned developing high-level simulations of hazardous behaviors, and thus resolves many ambiguities which can lead to poor system specification. We were able to then demonstrate this language on a multirotor drone used for fire response, showing how this language can help represent the functional, operational, and hardware aspects of failure to support overall hazard assessment. This language has a number of advantages over the state of the literature and practice due to its ability to represent each of these aspects in an unambiguous and consistent model.

This work represents a start at developing a coherent language to support the assessment of hazards in the engineering process. In future work, we hope to more comprehensively (quantitatively, rather than qualitatively) study the use of FRDL vis-a-vis existing FHA languages for identifying hazardous conditions, scenarios, and effects. Ideally, future work should empirically study if designers using this language identify more hazards, and effects. Additionally, this language is currently merely a specification–in future work, we further hope to develop a comprehensive toolset for hazard analysis, including a model-based systems engineering tool for developing these models and linking them with explicit analyses (FHA, FMEA, FTA, etc).

ACKNOWLEDGMENTS

This research was funded by the System-Wide Safety project in the NASA Aeronautics Research Mission Directorate. The findings herein represent the research of the authors and do not necessarily the view of the U.S. Government or NASA. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the U.S. Government.

REFERENCES

[1] Committee, SAE International S-18 et al. "ARP4761 Guidelines and methods for conducting the safety assessment process on civil airborne system and equipment." *Warrendale, Pennsylvania: Society of Automotive Engineers* (1996).

[2] Tan, James JY, Otto, Kevin N and Wood, Kristin L. "Relative impact of early versus late design decisions in systems development." *Design Science* Vol. 3 (2017): p. e12.

[3] Smith, Robert E. "MIL-STD-882E." *Department of Defence* (2012).

[4] Joshi, Anjali, Whalen, Mike and Heimdahl, M. "Model-based safety analysis final report." *NASA Techreport* (2005).

[5] Wilkinson, PJ and Kelly, TP. "Functional hazard analysis for highly integrated aerospace systems." (1998).

[6] Aircraft, SAE S-18, Dev, Sys and Committee, Safety Assessment. "SAE ARP926C: Fault/Failure Analysis Procedure." *SAE International* (2018).

[7] Graydon, Mallory, Neogi, Natasha A and Wasson, Kimberly. "Guidance for designing safety into urban air mobility: Hazard analysis techniques." *AIAA Scitech 2020 Forum*: p. 2099. 2020.

[8] Denney, Ewen W. "AdvoCATE User Guide." *NASA V&V Commercial Systems TC-3 Conference and Seminar Series*. 2022.

[9] Leveson, Nancy. "STPA (System-Theoretic Process Analysis) Compliance with MIL-STD-882E and other Army Safety Standards." (2016)URL http://sunnyday.mit.edu/compliance-with-882.pdf.

[10] Pahl, Gerhard and Beitz, Wolfgang. *Engineering design: a systematic approach*. Springer Science & Business Media (2007).

[11] Stone, Robert B., Tumer, Irem Y. and Van Wie, Michael. "The Function-Failure Design Method." *Journal of Mechanical Design* Vol. 127 No. 3 (2004): pp. 397–407. DOI 10.1115/1.1862678. URL https://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/127/3/397/5601418/397_1.pdf, URL https://doi.org/10.1115/1.1862678.

[12] Kurtoglu, Tolga and Tumer, Irem Y. "A Graph-Based Fault Identification and Propagation Framework for Functional Design of Complex Systems." *Journal of Mechanical Design* Vol. 130 No. 5 (2008): p. 051401. DOI 10.1115/1.2885181.

[13] McIntire, Matthew G, Keshavarzi, Elham, Tumer, Irem Y and Hoyle, Christopher. "Functional models with inherent behavior: Towards a framework for safety analysis early in the design of complex systems." *ASME International Mechanical Engineering Congress and Exposition*, Vol. 50657: p. V011T15A035. 2016. American Society of Mechanical Engineers.

[14] Jensen, David, Tumer, Irem Y and Kurtoglu, Tolga. "Flow State Logic (FSL) for analysis of failure propagation in early design." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 49057: pp. 1033–1043. 2009.

[15] Patriarca, Riccardo, Di Gravio, Giulio, Woltjer, Rogier, Costantino, Francesco, Praetorius, Gesa, Ferreira, Pedro and Hollnagel, Erik. "Framing the FRAM: A literature review on the functional resonance analysis method." *Safety Science* Vol. 129 (2020): p. 104827.

[16] Zhang, Yingyu, Dong, Chuntong, Guo, Weiqun, Dai, Jiabao and Zhao, Ziming. "Systems theoretic accident model and process (STAMP): A literature review." *Safety science* Vol. 152 (2022): p. 105596.

[17] Ishimatsu, Takuto, Leveson, Nancy G, Thomas, John, Katahira, Masa, Miyamoto, Yuko and Nakao, Haruka. "Modeling and hazard analysis using STPA." (2010).

[18] Leveson, Nancy. "A new accident model for engineering safer systems." *Safety science* Vol. 42 No. 4 (2004): pp. 237–270.

[19] Hulse, Daniel, Walsh, Hannah, Dong, Andy, Hoyle, Christopher, Tumer, Irem, Kulkarni, Chetan and Goebel, Kai. "fmdtools: A fault propagation toolkit for resilience assessment in early design." *International Journal of Prognostics and Health Management* Vol. 12 No. 3 (2021).

[20] Irshad, Lukman and Hulse, Daniel. "Resilience Modeling in Complex Engineered Systems With Human-Machine Interactions." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 86212: p. V002T02A024. 2022. American Society of Mechanical Engineers.

[21] Administration, Federal Aviation. "AC.23.1309-1E." (2011)URL https://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_23_1309-1E.pdf.

[22] ISO. "ISO 26262 Road vehicles — Functional safety." (2018).

[23] Stone, Robert B and Wood, Kristin L. "Development of a functional basis for design." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 19739: pp. 261–275. 1999. American Society of Mechanical Engineers.

[24] International, SAE. "ARP1834 Fault/Failure Analysis For Digital Systems and Equipment." *Warrendale, Pennsylvania: Society of Automotive Engineers* (2018).

[25] SC-205, RTCA Committee. "DO-178C Software Considerations in Airborne Systems and Equipment Certification." (2011).

[26] ISO. "ISO-21448 Road vehicles — Safety of the intended functionality." (2022).

[27] Noviello, Maria Chiara, Dimino, Ignazio, Concilio, Antonio, Amoroso, Francesco and Pecora, Rosario. "Aeroelastic assessments and functional hazard analysis of a regional aircraft equipped with morphing winglets." *Aerospace* Vol. 6 No. 10 (2019): p. 104.

[28] Meyer, Lothar, Vogel, Markus and Fricke, Hartmut. "Functional hazard analysis of virtual control towers." *IFAC Proceedings Volumes* Vol. 43 No. 13 (2010): pp. 146–151.

[29] Tran, Vu N, Tran, Long V and Tran, Viet N. "Functional Hazard Analysis for Engineering Safe Software Requirements." *2021 4th International Conference on Information and Computer Technologies (ICICT)*: pp. 142–148. 2021. IEEE.

[30] Nagy, Bruce, Edwards, Loren and Sivapragasam, Gunendran. "Functional hazard analysis and subsystem hazard

analysis of artificial intelligence/machine learning functions within a sandbox program." Acquisition Research Program (2021).

[31] Ericson, Clifton A et al. *Functional Hazard Analysis*. John Wiley & Sons, Ltd (2005): Chap. 15, pp. 271–289. DOI https://doi.org/10.1002/0471739421.ch15. URL https://onlinelibrary.wiley.com/doi/pdf/10.1002/0471739421.ch15, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/0471739421.ch15.

[32] Johannessen, Per, Grante, Christian, Alminger, Anders, Eklund, Ulrik and Torin, Jan. "Hazard analysis in object oriented design of dependable systems." *2001 International Conference on Dependable Systems and Networks*: pp. 507–512. 2001. IEEE.

[33] Modarres, Mohammad. "Functional modeling of complex systems using a GTST-MPLD framework." *Proceedings of the International Workshop on Functional Modeling of Complex Technical Systems*: pp. 12–14. 1993.

[34] Rasmussen, Birgitte and Whetton, Cris. "Hazard identification based on plant functional modelling." *Reliability Engineering & System Safety* Vol. 55 No. 2 (1997): pp. 77–84.

[35] Stone, Robert B, Tumer, Irem Y and Van Wie, Michael. "The function-failure design method." *Journal of Mechanical Design* Vol. 127 No. 3 (2005): pp. 397–407.

[36] Sangelkar, Shraddha and McAdams, Daniel A. "Creating actionfunction diagrams for user centric design." *2012 ASEE Annual Conference & Exposition*: pp. 25–355. 2012.

[37] Soria Zurita, Nicolás F, Stone, Robert B, Onan Demirel, H and Tumer, Irem Y. "Identification of human–system interaction errors during early design stages using a functional basis framework." *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering* Vol. 6 No. 1 (2020): p. 011005.

[38] Irshad, Lukman, Ahmed, Salman, Demirel, H Onan and Tumer, Irem Y. "Computational functional failure analysis to identify human errors during early design stages." *Journal of Computing and Information Science in Engineering* Vol. 19 No. 3 (2019): p. 031005.

[39] Irshad, Lukman and Hulse, Daniel. "Modeling Distributed Situation Awareness in Resilience-Based Design of Complex Engineered Systems." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 87295: p. V002T02A050. 2023. American Society of Mechanical Engineers.

[40] Schäfer, Michael, Berres, Axel and Bertram, Oliver. "Integrated model-based design and functional hazard assessment with SysML on the example of a shock control bump system." *CEAS Aeronautical Journal* Vol. 14 No. 1 (2023): pp. 187–200.

[41] Jiao, Jian, Pang, Shujie, Chu, Jiayun, Jing, Yongfeng and Zhao, Tingdi. "An Improved FFIP Method Based on Mathematical Logic and SysML." *Applied Sciences* Vol. 11 No. 8 (2021): p. 3534.

[42] El Ariss, Omar, Xu, Dianxiang and Wong, W Eric. "Integrating safety analysis with functional modeling." *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* Vol. 41 No. 4 (2011): pp. 610–624.

[43] Zikrullah, Nanda Anugrah, Kim, Hyungju, van der Meulen, Meine JP, Skofteland, Gunleiv and Lundteigen, Mary Ann. "A comparison of hazard analysis methods capability for safety requirements generation." *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* Vol. 235 No. 6 (2021): pp. 1132–1153.

[44] Ishimatsu, Takuto, Leveson, Nancy G, Thomas, John P, Fleming, Cody H, Katahira, Masafumi, Miyamoto, Yuko, Ujiie, Ryo, Nakao, Haruka and Hoshino, Nobuyuki. "Hazard analysis of complex spacecraft using systems-theoretic process analysis." *Journal of spacecraft and rockets* Vol. 51 No. 2 (2014): pp. 509–522.

[45] Hollnagel, Erik. *FRAM: the functional resonance analysis method: modelling complex socio-technical systems*. Crc Press (2017).

[46] Toda, Yoshinari, Matsubara, Yutaka and Takada, Hiroaki. "FRAM/STPA: Hazard analysis method for FRAM model." *Proceedings of the 2018 FRAM Workshop. Cardiff, Wales*: pp. 1–17. 2018.

[47] International, SAE. "ARP 926: Fault/Failure Analysis Procedure." (2018).

[48] Hulse, Daniel, Hoyle, Christopher, Tumer, Irem Y, Goebel, Kai and Kulkarni, Chetan. "Temporal Fault Injection Considerations in Resilience Quantification." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 84003: p. V11AT11A040. 2020. American Society of Mechanical Engineers.

[49] Mbaye, Seydou, Jones, Garfield, Infeld, Samantha I., Okon, Shira and Davies, Misty D. *A Model-Based Systems Engineering Evaluation of the Evolution to an In-Time Aviation Safety Management System*: DOI 10.2514/6.2022-3423. URL https://arc.aiaa.org/doi/abs/10.2514/6.2022-3423.

[50] Mbaye, Seydou, Walsh, Hannah S., Davies, Misty, Infeld, Samantha I. and Jones, Garfield. *From BERTopic to SysML: Informing Model-Based Failure Analysis with Natural Language Processing for Complex Aerospace Systems*: DOI 10.2514/6.2024-2700. URL https://arc.aiaa.org/doi/abs/10.2514/6.2024-2700.

[51] Beckers, Kristian, Heisel, Maritta, Frese, Thomas and Hatebur, Denis. "A structured and model-based hazard analysis and risk assessment method for automotive systems." *2013 IEEE 24th International Symposium on Software Reliability Engineering (ISSRE)*: pp. 238–247. 2013. IEEE.

[52] Eisenbart, Boris, Gericke, Kilian and Blessing, Luciënne. "An analysis of functional modeling approaches across disciplines." *AI EDAM* Vol. 27 No. 3 (2013): pp. 281–289.

[53] Hadef, Hefaidh, Negrou, Belkhir, Ayuso, Tomás González, Djebabra, Mébarek and Ramadan, Mohamad. "Preliminary hazard identification for risk assessment on a complex system for hydrogen production." *International Journal of Hydrogen Energy* Vol. 45 No. 20 (2020): pp. 11855–11865.

[54] Irshad, Lukman, Ahmed, Salman, Demirel, Onan and Tumer, Irem Y. "Coupling digital human modeling with

early design stage human error analysis to assess ergonomic vulnerabilities." *AIAA SciTech 2019 forum*: p. 2349. 2019.

[55] Borza, John S. "FAST Diagrams : The Foundation for Creating Effective Function Models." 2011. URL https://api.semanticscholar.org/CorpusID:53612032.

[56] Lawrence, Philip and Gill, Simon. "Human hazard analysis: A prototype method for human hazard analysis developed for the large commercial aircraft industry." *Disaster Prevention and Management: An International Journal* Vol. 16 No. 5 (2007): pp. 718–739.