Defining a Modelling Language to Support Functional Hazard Assessment

Daniel Hulse

Seydou Mbaye

Lukman Irshad (KBR, Inc)

Robust Software Engineering

Intelligent Systems Division

NASA Ames Research Center

ASME IDETC/CIE 2024

August 26, 2023, 2:10-3:50 PM (ET)

DAC-08-01: Design for Resilience and Failure Recovery

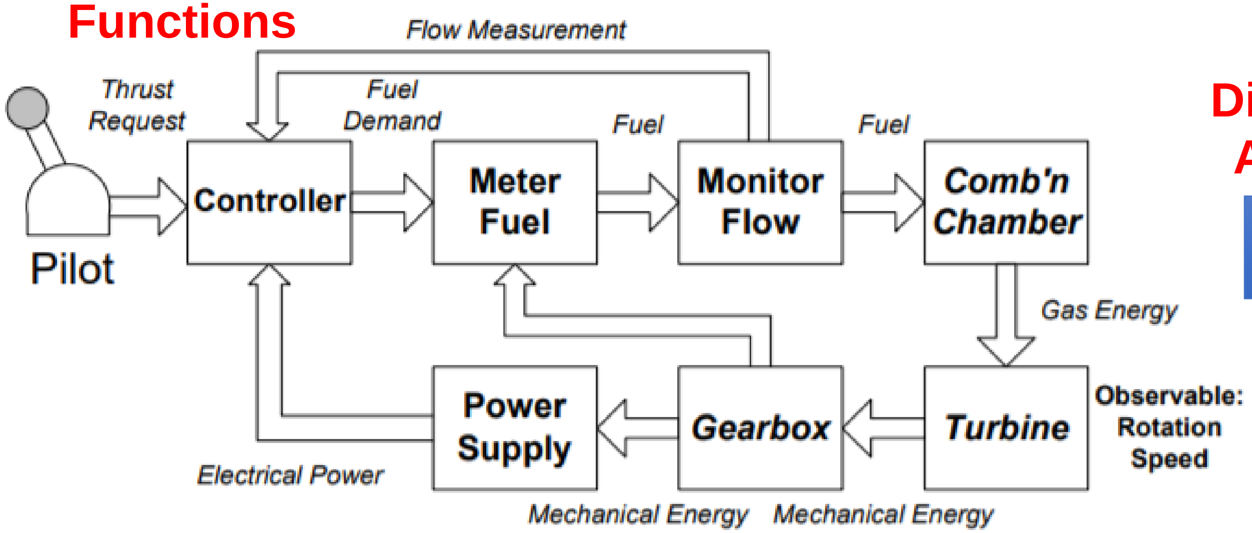
Washington, DC

Paper No: **IDETC/CIE2024-143549**

Motivation: Early Failure Analysis

Identify Functions → Identify Hazardous Conditions → Determine Effects → Derive Requirements and Mitigations

Diagram of System Functions



Discursive Analysis →

Functional Hazard Assessment

Function	Condition	Effect

FHA-Related Standards

History of early "functional" failure analysis

- **MIL-P-1629:** Original 1949 FMEA military standard
- **ARP 4761:** 1996 Civil aviation safety standard calls for FHA
- **ISO 26262:** 2011 Automotive standard on "functional" safety
- **MIL-STD-882E:** 2012 military standard calls for FHA

Generally, modern standards

(1) Call for Functional Hazard Analysis

(2) Define the "what" of the FHA table

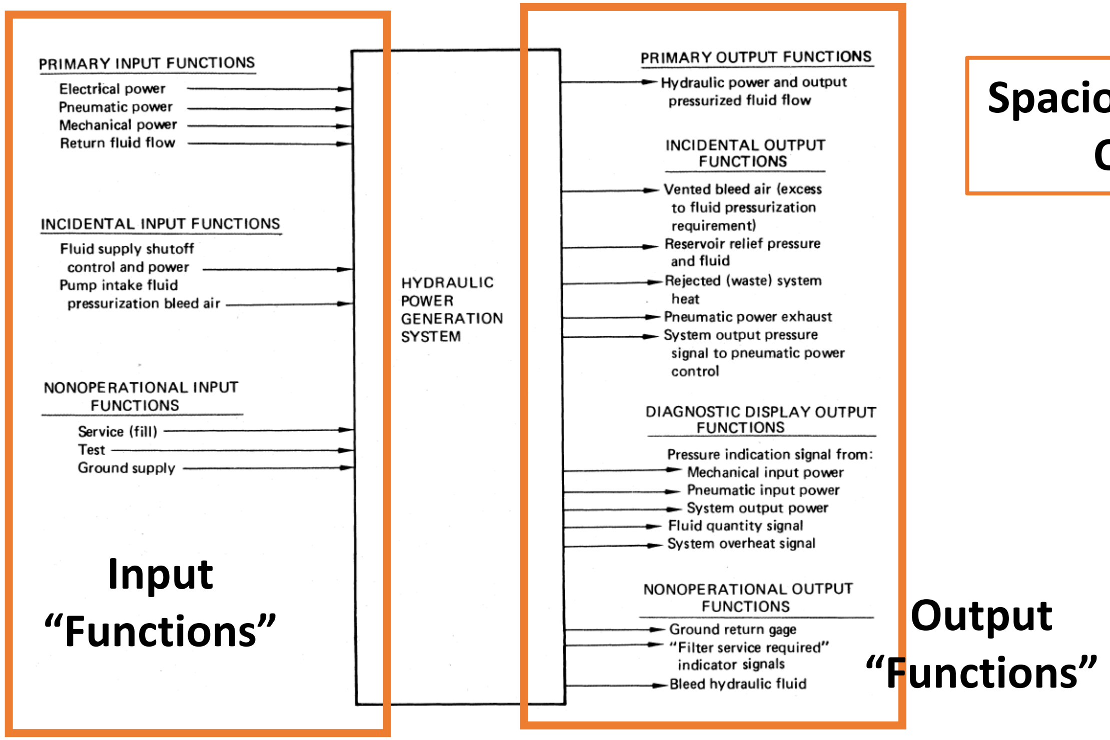
(3) Don't really define the process or how to generate it

FHA-Related Standards: ARP 926C

- Fault/Failure Analysis procedure
  - Descendent of early civilian FMEA standards
- Gives recommendations on performing “Functional” F/FA, including diagrams:

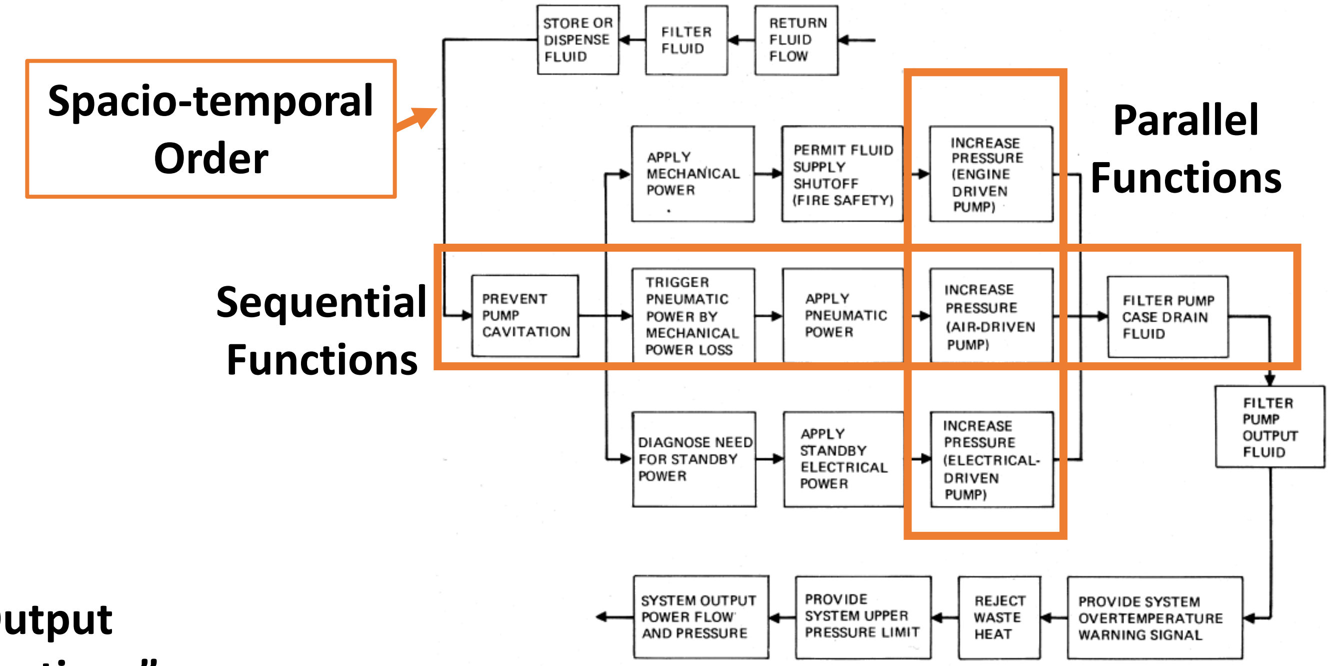
Overall Function Diagram:

Used to identify high-level functional failures



Function Block Diagram:

Used to identify how sub-functional failures cause high-level failures



Our Motivation: Resilience in FHA

Development of Simulation-based Resilience Analysis tools

Overview

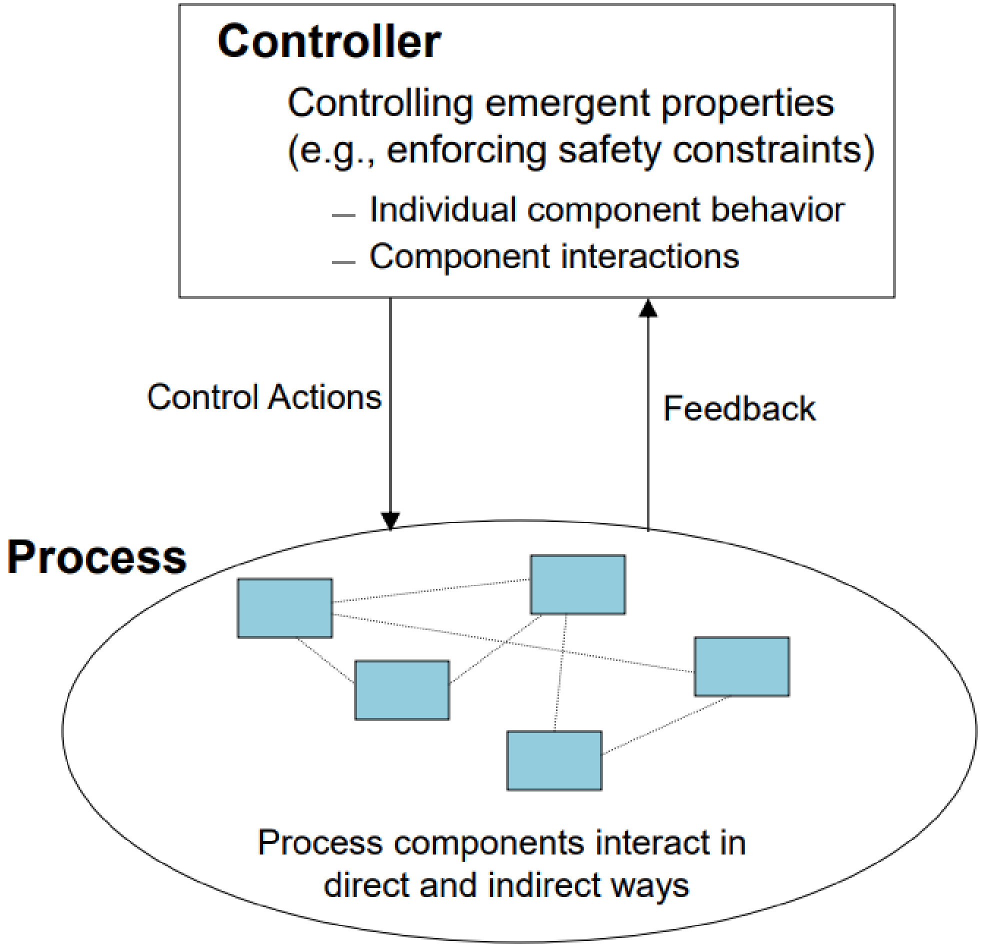
Python 3.11 | pypi v1.3.3 | GitHub Release v1.3.3 | tag v2.0-rc-6 | codefactor A | tests 301
coverage 88.40% | License NOSA | Software Class E

fmdtools (Fault Model Design tools) is a Python library for modelling, simulating, and analyzing the resilience of complex systems. With fmdtools, you can (1) represent system structure and behavior in a model, (2) simulate the dynamic effects of hazardous scenarios on the system, and (3) analyze the results of simulations to understand and improve system resilience.

- **Resilience/Simulation** needs and lessons learned:
  - Need for propagation of hazardous behavior between functions as well as over time
    - System behavior (and thus hazards) vary significantly over control modes
  - Importance of high-level Human, System, and Environmental interactions

FHA-Related Literature: STAMP/STPA

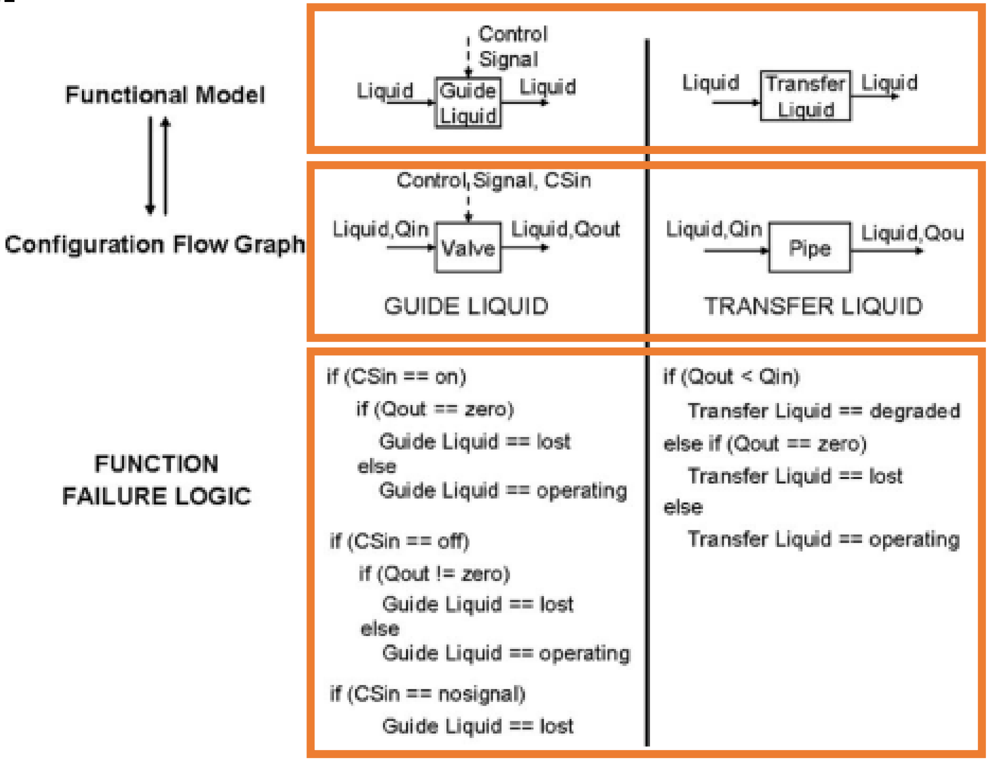
- Key contribution: **importance of control structures in "accident"-type failures**
  - Human interactions
  - Organizational influence
  - Control systems and automation
- Has achieved influence
  - Buy-in and interest from industry
  - Some talk about incorporating STAMP into standards for FHA
  - A lot of guidance and resources!
- **Gap:** Only helpful for accident-type failures, not a general language



Leveson, N. (2011). Engineering a safer and more secure world. *MIT*.

FHA-Related Methods: Tumer et al.

- Key contribution: **EMS functional models** for hazard analysis
  - Descends from design literature
  - **Functions:** tasks (noun-verb pairs) performed by the system
  - **Flows:** Energy, Material, and Signals passed between functions
- Lots of variants in research area
- **Gap:** Doesn't incorporate control loops very well. Limited by spacio-temporal flow representation



Functions

Embodied by **Components**

With given **Behavior**

Kurtoglu, T., & Tumer, I. Y. (2008). A graph-based fault identification and propagation framework for functional design of complex systems.

Defining the Functional Reasoning Design Language

Goals:

- Take lessons from simulation and outside research and use them to improve FHA-supporting diagrams
- Create a way so that, in the future, we can define simulations of systems resilience as diagrams rather than code

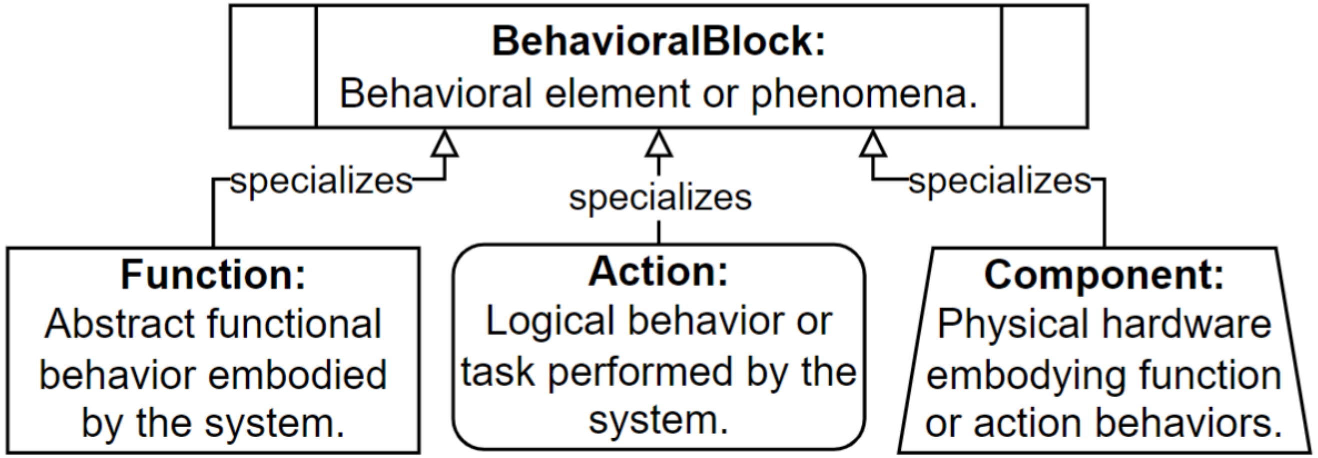
Major Elements:

- **Blocks:** e.g., functions
- **Flows and Relationships:** Connections between blocks
- **Architectures:** Overall diagrams

Behavioral Blocks

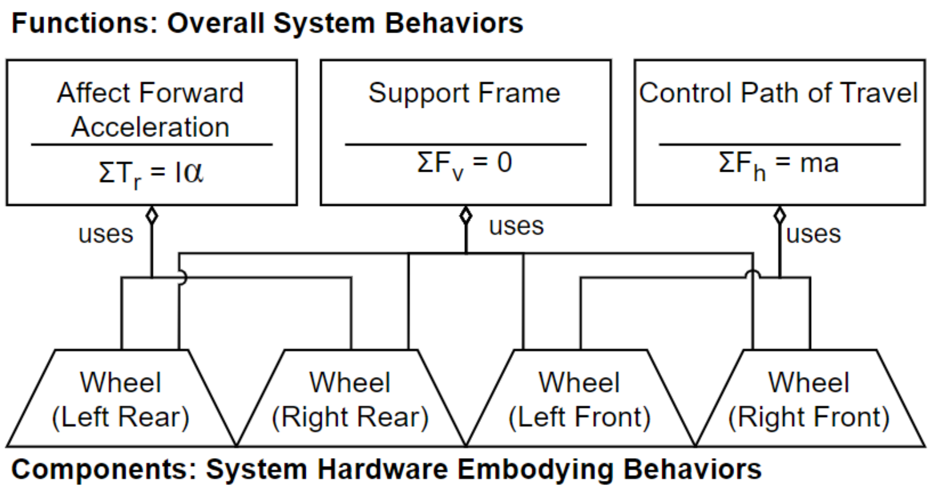
Main idea: Blocks represent *behavior*, of which there are three types.

These blocks can be annotated with tags to better inform analysis



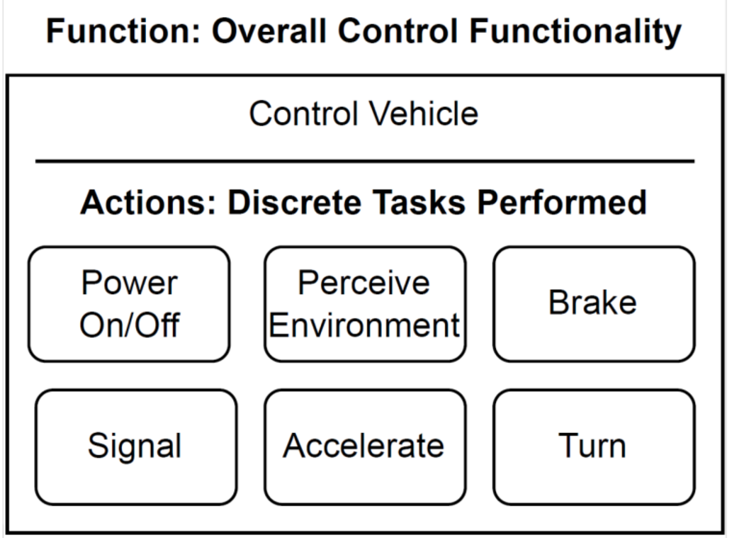
Functions, Components, and Actions

Functions versus Components



- Functions are the *behaviors the system is to embody*
- Components *realize these functions*
  - Mapping likely not be one-to-one
- Similar to EMS/FBED idea of function/component mapping

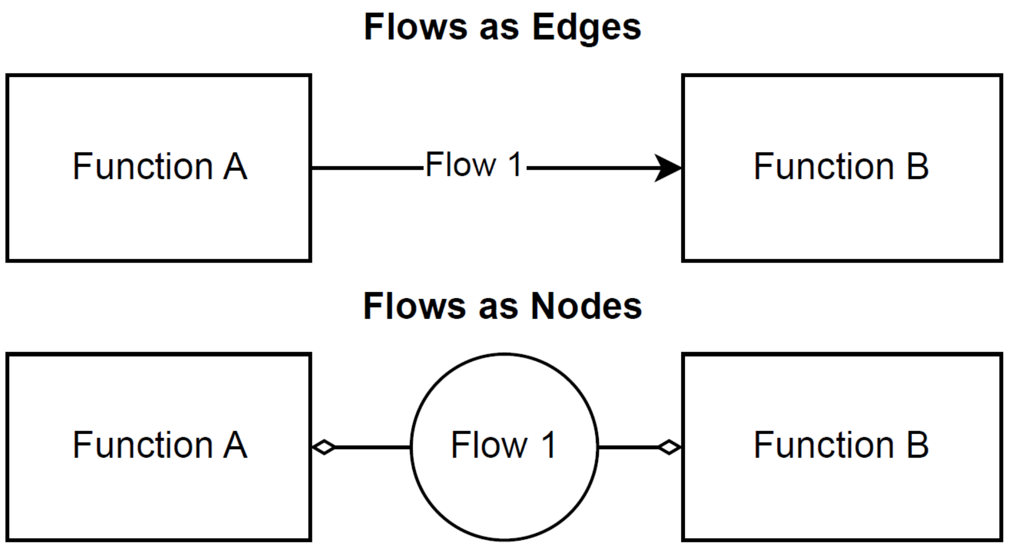
Functions versus Actions



- Agents like operators, controllers, and users are considered Functions
- Discrete tasks performed by these functions are actions
- Enables STAMP idea of the representing control structure

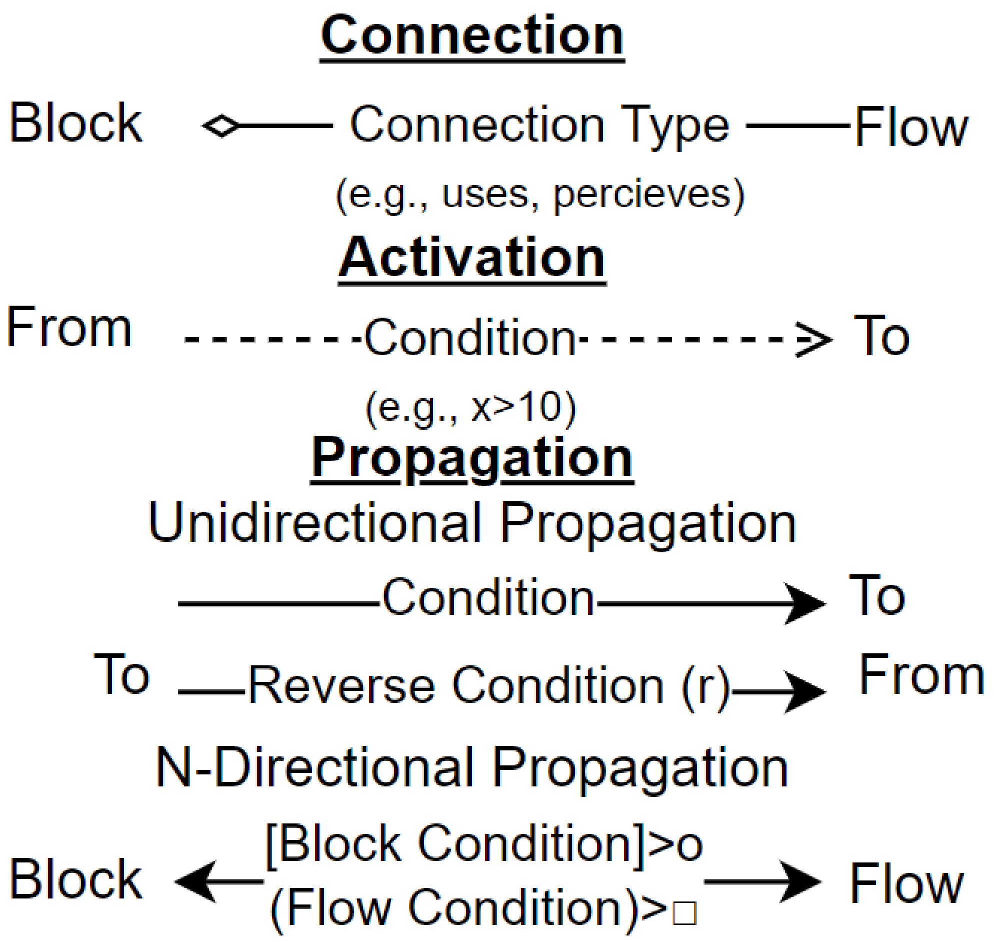
Flows

Main Idea: Flows are *nodes, not edges*



- Flows represent **shared variables** which enable behavioral propagation
- Flows can be **shared by more than one block**, enabling efficient representation of:
  - Communications
  - Multiple agents sharing and interacting in a joint environment
  - Complex (more realistic) failure propagation between different functions

Relationships

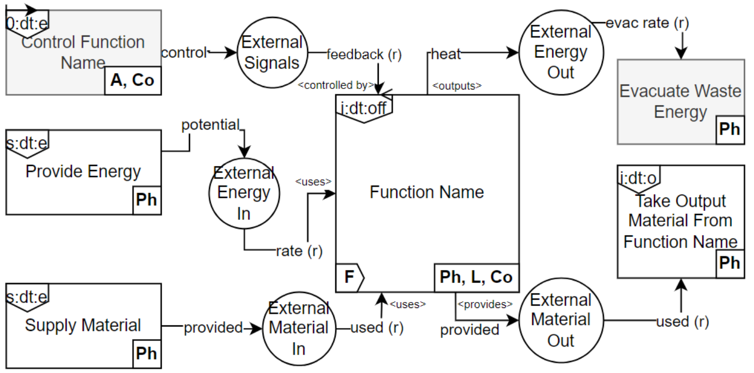


Different ways of relating flows and blocks with each other:

- **Connection:** is a flow in the function(s)?
  - Names/annotations tell us more about what the function is doing
- **Activation:** how a condition in one block changes another block's behavior
- **Propagation:** combination of connection and activation
  - Annotations tell us direction of activation

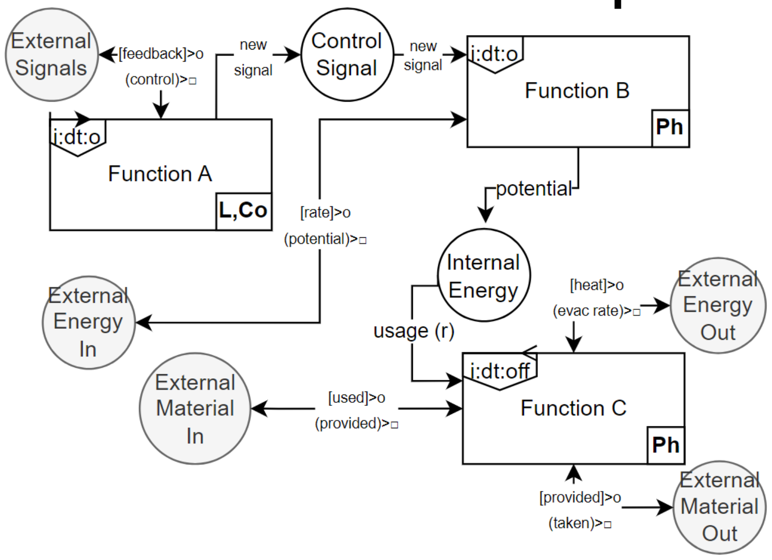
Architectures: Functional

Overall Function Diagram



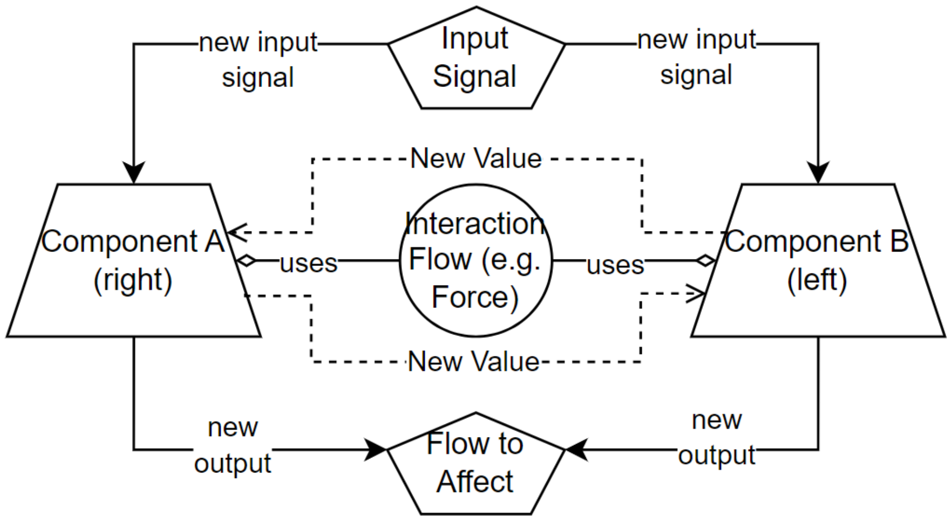
- High-level interactions of system with its environment
- Includes sources for inputs/outputs as well as external controllers

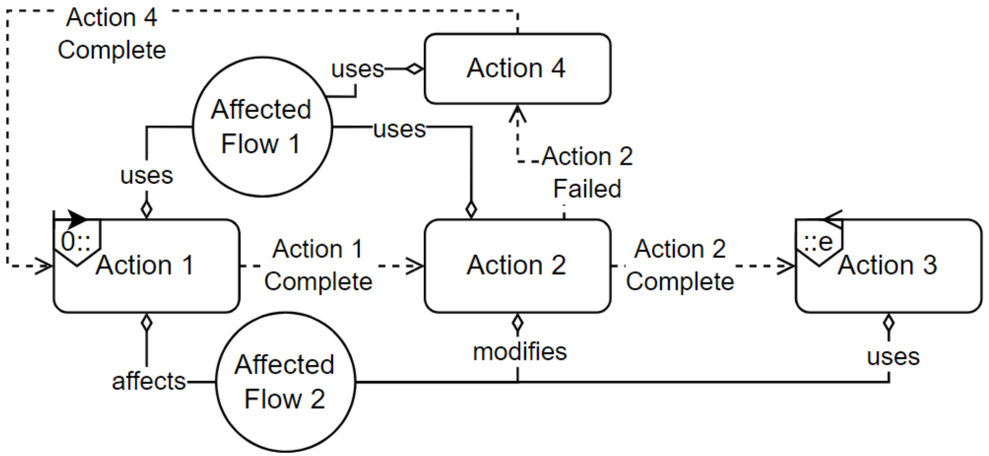
Functional Decomposition



- Breakdown of overall functionality into functions

Architectures: Component and Action





Component Architectures represent component behaviors in the scope of a given function

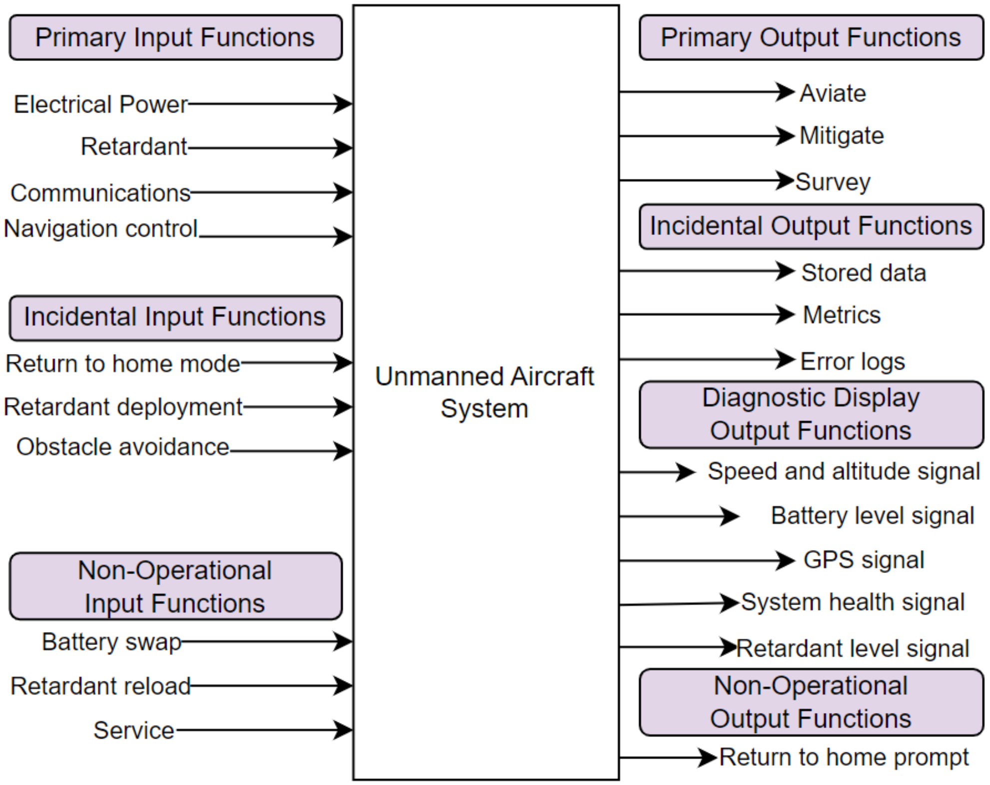
Action Architectures represent sequences of tasks a function performs and their inputs/outputs

- Similar to a state machine

Demonstration—Fire Response UAV

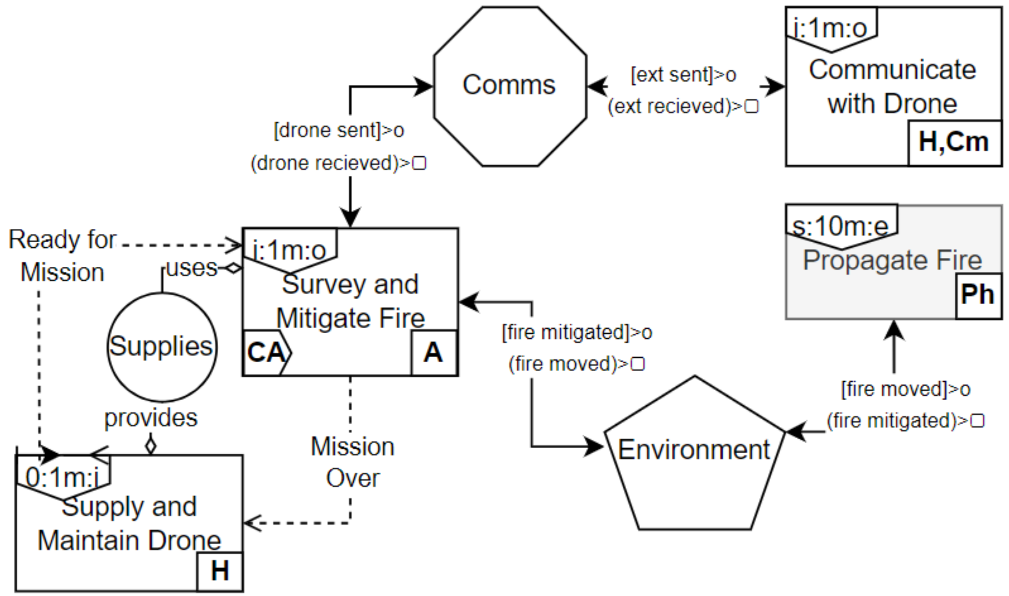
- Goal: Qualitative demonstration/comparison between FRDL and ARP-926C models
  - Not a full analysis or FHA, just a look at what each diagram tells us
- UAV is meant to semi-autonomously fly from a base, conduct surveillance mitigate wildfires while communicating with external operators

Demo – Surveillance UAV



Function Diagram per ARP-926

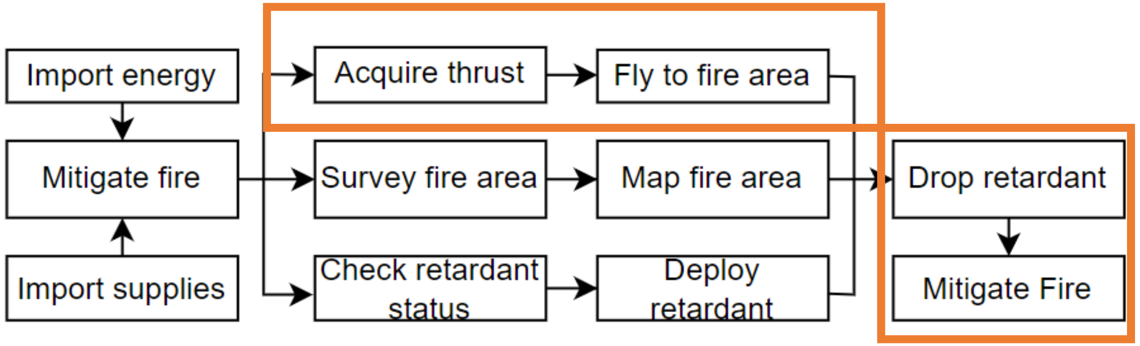
- Good for identifying inputs/outputs



FRDL Function-in-Context

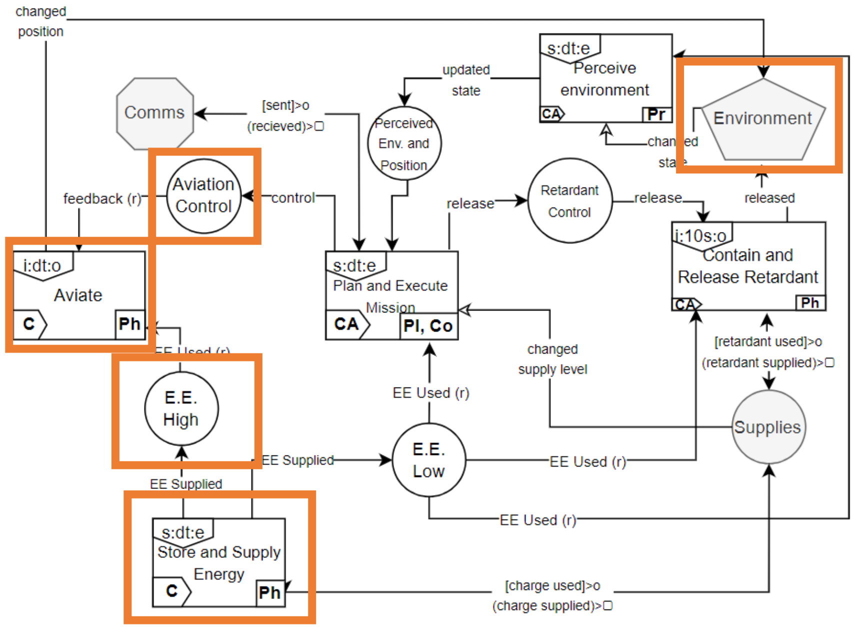
- Better idea of **system**: interactions, dynamics, and usage
- Inputs/Outputs more abstract, but able to be broken down elsewhere

Demo—Surveillance UAV



Function Block Diagram per ARP-926

- Spacio-temporal view makes it **hard to trace full propagation**



FRDL Functional Decomposition

- Much **more specific** about what flows are interacting and how
- Better for tracing failure propagation
- Shows "how" the system would behave, not just that it fails

Example: Electrical fault (short) in thrust/aviate function

- **ARP-926 model:** Unable to fly to fire area and thus complete mission/mitigate fire
- **FRDL model:** Aviate fault causes **adverse change in position** in the environment (i.e., a crash) as well as **adverse energy draw**, and **modified control feedback** propagating to other functions

Discussion/Conclusions

Very initial demonstration

- Didn't go down to the level of component/action architectures
- Didn't provide FHA output
- We will need to address this in future work

However:

- FRDL gives us a much more expressive means to represent propagation of hazardous behaviors
- It integrates multiple perspectives:
  - STAMP/STPA control interactions between operator, system, and environment
  - Physical constraints defining failure propagation in the technical system
- It may also take more input effort from the analyst

Questions?

Daniel Hulse daniel.e.hulse@nasa.gov
google scholar: scholar.google.com/citations?user=fa1S_74AAAAJ&hl=en
ResearchGate: researchgate.net/profile/Daniel-Hulse-4

Seydou Mbaye seydou.mbaye@nasa.gov
google scholar: https://scholar.google.com/citations?user=q7eCRgEAAAAJ&hl=en
ResearchGate: https://www.researchgate.net/profile/Seydou_Mbaye

Lukman Irshad lukman.irshad@nasa.gov
google scholar: scholar.google.com/citations?user=u64zCIEAAAAJ&hl=en
ResearchGate: researchgate.net/profile/Lukman-Irshad

Fmdtools Simulation Package
repo: github.com/nasa/fmdtools
documentation: nasa.github.io/fmdtools/