

Development of a Design Support Tool for Satellite FDIR by Generating State Transition Diagrams and Evaluating Probabilistic Reliability and Complexity

By Takato HATAE,¹⁾ Shunichiro NOMURA,¹⁾ Satoshi IKARI¹⁾ Ryu FUNASE¹⁾ and Shinichi NAKASUKA¹⁾

¹⁾Department of Aeronautics and Astronautics, The University of Tokyo, Tokyo, Japan

(Received March 1st, 2023)

With the diversification of satellite missions, there is an increasing demand for FDIR designs that realize not only safety but also mission continuity. However, such complex FDIR designs require considerable effort and time. To address this issue, this study developed a design support tool that enables designers to rapidly iterate on FDIR designs. The tool comprehensively evaluates the validity of the FDIR logic along with multiple quantitative Metrics, quantifying the trade-offs underlying the design process. Consequently, FDIR designers are provided with clearly defined decision criteria, thereby facilitating more efficient FDIR designs.

Key Words: FDIR, Design Support Tool, Safety, Nano-satellites, Systems Engineering

1. Introduction

1.1. FDIR Design for Small Satellites

Fault Detection, Isolation, and Recovery (FDIR) refers to a system that autonomously detects anomalies when they occur, prevents the spread of the fault, and takes appropriate corrective actions accordingly. In small satellites, FDIR plays a vital role in fault-tolerant design due to the limited possibility of hardware redundancy. Traditionally, FDIR for such satellites has centered around simple and safe strategies, such as resetting the system or immediately transitioning to a safe mode upon fault detection.

However, recent trends show increased utilization of small satellites for deep space exploration and commercial services through satellite constellations. These applications demand not only safety but also the ability to autonomously continue the mission even in the presence of faults. Consequently, there is a growing need for design methodologies that enable the development of high-performance FDIR systems—ones that ensure both safety and mission continuity—within the constraints of limited development time and cost.

1.2. FDIR Design Process

The design process of FDIR can be summarized as shown in Fig. 1. It generally consists of the following steps:

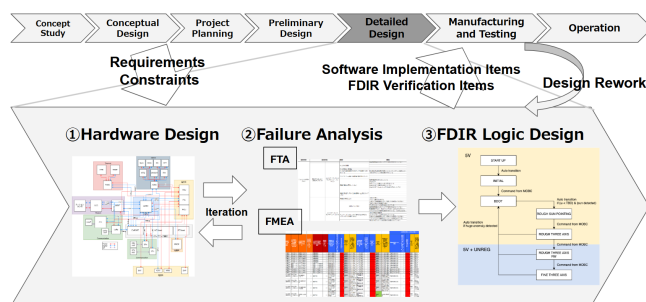


Fig. 1. Satellite development process focusing on FDIR

- 1. Hardware Design:** Based on the mission requirements and constraints, a preliminary hardware design is developed. During this stage, redundancy is assigned to specific

subsystems or components within the limitations of mass and volume.

- 2. Failure Analysis:** Once the hardware design has reached a certain level of maturity, system-level failure analysis is conducted using techniques such as Fault Tree Analysis (FTA) and Failure Modes and Effects Analysis (FMEA). Potential faults are identified based on past data, documentation, and the experience of developers.
- 3. FDIR Logic Design:** In this step, detection and recovery actions are defined for the faults identified in step 2. This involves designing detection mechanisms and corresponding recovery actions, forming the core of the FDIR logic.

Once the logic is defined, quick verification is carried out to ensure that all relevant faults can be detected and that no false detections occur from other detection mechanisms. Finally, the FDIR functions are implemented as part of the flight software and tested on the actual hardware. If any issues arise during this phase, the design must be revisited. Therefore, it is essential to verify the validity of the FDIR logic even during the design phase through preliminary validation methods.

1.3. Challenges in FDIR Logic Design

This study focuses on the challenges in Step ③: FDIR Logic Design, the core of FDIR development. Specifically, it addresses how to design the detection and recovery logic for various anomalies. The following challenges are commonly encountered during this process:

- 1. Lack of quantitative criteria** Qualitative, experience-based discussions dominate, making decisions subjective and time-consuming.
- 2. Too many competing factors** Designers must weigh fault severity, occurrence frequency, action risk, verification cost, mission continuity, etc., often without a clear priority scheme.
- 3. Exhaustive verification burden** Ensuring that the logic avoids false detections or simultaneous triggers across all anomaly cases demands significant verification effort.

1.4. Need for Quantitative Metrics on Safety–Autonomy–Implementation Feasibility

FDIR logic must strike a balance among three key perspectives: (i) **Safety**, (ii) **Autonomy**, and (iii) **Implementation Feasibility**. Yet designers currently rely on qualitative judgment when weighing these aspects. To enable objective trade-off analysis, this study aims to define quantitative indicators for each axis and visualize their interactions.

The following trade-off relationships typically arise in FDIR logic design:

- **Safety vs. Autonomy:** Entering a strict safe mode guarantees fault containment but relies on ground commands for recovery, reducing on-board autonomy.
- **Safety vs. Implementation Feasibility:** Triple-redundant sensors improve fault tolerance but may be infeasible for 6U-class satellites due to constraints in mass and power.
- **Autonomy vs. Implementation Feasibility:** On-board artificial intelligence enables self-reliant recovery but may exceed the computational or memory capacity of the actual flight computer.

These trade-offs highlight the need for a unified design support framework. The next section reviews related work and identifies the remaining gap that this study addresses.

2. Proposed Approach

2.1. Related Work and Remaining Gap

Karakawa¹⁾ introduced evaluation metrics such as reliability and complexity to support FDIR design decisions. The COMPASS toolset developed by the European Space Agency²⁾ enables formal verification of failure propagation and false detections using mathematical modeling.

However, these approaches address only individual aspects of the FDIR design process. None provide a unified framework that captures the multi-dimensional trade-offs among safety, autonomy, and Interpretation feasibility. This study fills the gap by proposing a method that visualizes and quantifies these competing objectives for practical and holistic FDIR logic design.

2.2. Objective of the Design Support Tool

Building on the qualitative discussion of FDIR trade-offs, this study introduces a design-support tool that quantifies the three indicators defined in Section 1.4: safety (tolerance to false detections, malfunctions, and cascading faults), autonomy (ability to recover without ground intervention), and implementation feasibility (compatibility with on-board hardware and software).

The tool automatically generates probabilistic state-transition diagrams from the fault–detection–action database and derives multiple metrics aligned with these three indicators. By visualizing the results, designers can numerically assess trade-offs, identify weaknesses, and iteratively refine FDIR logic. As small-satellite missions demand ever higher levels of autonomy and reliability, the proposed approach reduces design effort and supports objective, data-driven decision-making.

3. Overview of the Design Support Tool

An extended framework was developed on the basis of Ito's original approach.³⁾ The original framework evaluates mission continuity and safety using probabilistic calculations derived from the input FDIR logic; by iteratively modifying and evaluating the logic, it enables rapid design cycles. In the present work, while preserving the core algorithms for generating and evaluating state-transition diagrams, additional evaluation functions and logic-verification features were incorporated.

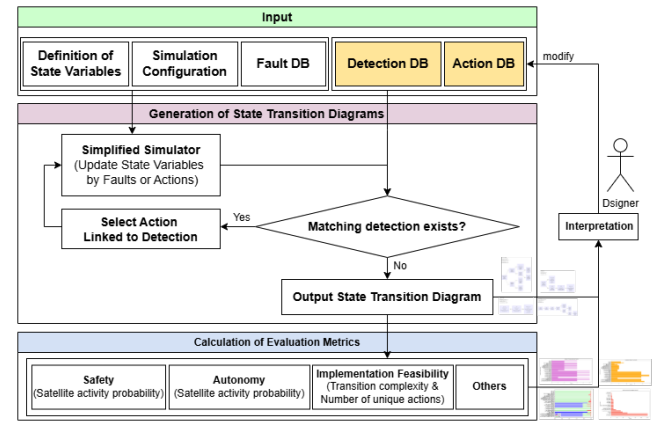


Fig. 2. Overview of the FDIR Design Support Tool

3.1. Data Input

The support tool requires two primary inputs: 1. definitions of the satellite state variables for the system under design, and 2. databases of faults, detections, and recovery actions. The satellite state variables include physical quantities such as battery state of charge, attitude, angular momentum, and the status of individual components.

The databases for faults, detections, and actions define how the state variables are updated. As simulations proceed, satellite states evolve in response to the faults and actions defined in the database.

The databases of detections and recovery actions themselves constitute the FDIR logic, and improving them corresponds to the design of the FDIR logic. In Fig. 2, the green area at the top represents the input information from these databases.

Table 1. Examples of defined state variables.

Category	Example state value
Satellite mode	nominal mode
Attitude control	fine three-axis control
Angular momentum	low
State of charge (SoC)	90%
Device operation status	stx_switch : high (S-band transmitter in high-power mode)
Device health status (status)	adcs.status : hung up (ADCS: system hang-up)

3.2. Generation of probabilistic state-transition diagrams

The generation of a state transition diagram begins with the update of satellite state variables based on fault definitions in the

fault database. When a fault triggers a change in the satellite's state, the system checks for matching detection conditions from the detection database. If a match is found, the corresponding recovery action is executed, which in turn updates the satellite state again.

Each action defines both success and failure outcomes, causing the transition diagram to branch accordingly. In addition, the success and failure probabilities of each action are stored. If no detection matches the current state, the transition ends and the final diagram is produced. Since a transition diagram is generated for each fault, defining n faults results in n corresponding transition diagrams. In Fig. 2, the red-highlighted section indicates the transition generation process.

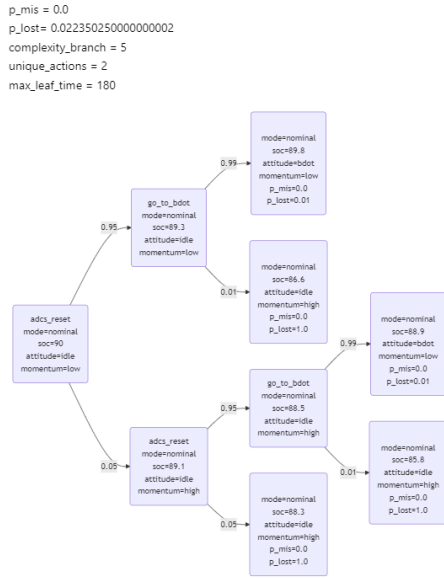


Fig. 3. Example of a transition diagram (attitude system anomaly in a 6U Earth-orbiting satellite model)

Fig. 3 shows an example of a transition diagram. The input fault in this case is an anomaly in the precision gyroscope, represented as `adcs.status = hung_up`. Each node in the diagram includes, in order: the name of the executed action, satellite mode, state of charge (SoC), attitude, and angular momentum. The edges represent the transition probabilities resulting from each action.

Terminal nodes, where no further actions are triggered, list the final satellite state along with two evaluation metrics: p_{mis} (the probability that the mission can continue) and p_{lost} (the probability of total satellite loss). These probabilities are calculated based on the satellite state, as defined by the designer. For example, the state of charge, attitude control mode, and angular momentum are used to determine whether the final state is mission-capable, degraded, or lost.

Table 2. Definitions of satellite activity probabilities.

Final state	Probability
Mission-capable state	p_{mis}
Intermediate state (e.g., safe mode)	$1 - p_{\text{mis}} - p_{\text{lost}}$
Lost state	p_{lost}

In Fig. 3, the root node represents detection of an anomaly in the attitude control system, with a full reset of the ADCS assigned as the recovery action. There is a 0.95 probability of

successful recovery and a 0.05 probability of failure. The next nodes display the updated satellite states and their associated actions. For the successful recovery case (top node in the second column), the system transitions to B-dot control.

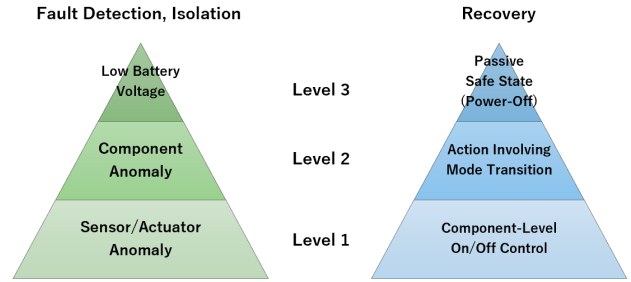


Fig. 4. Hierarchical FDIR implementation in 6U CubeSat

During transition generation, if multiple detections are triggered simultaneously for a single satellite state, a warning is issued. This indicates overlapping logic in the FDIR design, where conflicting actions may be assigned to the same state. In such cases, the designer must revise the logic to resolve the conflict.

However, even when multiple detections occur, it is possible to define prioritized actions—as in hierarchical FDIR schemes—so that only the highest-priority action is executed. This allows consistent transition diagrams to be generated.

```
==== fault_id : accurate_gyro_hangup_moderate ====
Warning: Multiple detections with the same max fdir_layer 2: ['bdot_err', 'go_to_bdot']
Warning: Multiple detections with the same max fdir_layer 2: ['adcs_err', 'go_to_bdot']
Warning: Multiple detections with the same max fdir_layer 2: ['bdot_err', 'go_to_bdot']
Warning: Multiple detections with the same max fdir_layer 2: ['bdot_err', 'go_to_bdot']
Warning: Multiple detections with the same max fdir_layer 2: ['adcs_err', 'go_to_bdot']
```

Fig. 5. Detection conflict warning (overlapping detections at the same level in a hierarchical FDIR scheme)

3.3. Calculation of evaluation metrics

3.3.1. Satellite activity probability:

Based on the values of p_{mis} and p_{lost} at the terminal nodes described in Section 3.2, the overall probabilities for the entire transition diagram can be defined. Let p_i be the probability associated with terminal node i , and let $q_{k,i}$ denote the probability that the k -th action along the path to node i successfully leads to that node. The overall probability p_{all} is then expressed as:

$$p_{\text{all}} = \sum_i \left(p_i \prod_k q_{k,i} \right)$$

In Fig. 3, the transition probabilities from the root node to each terminal node are multiplied by the respective p_{mis} and p_{lost} values at each node, and then summed to obtain the overall values. These represent, respectively, the mission continuation probability and the satellite loss probability in response to a particular fault. Whether the FDIR logic is tuned to prioritize increasing p_{mis} or decreasing p_{lost} depends on the designer's strategy and mission requirements.

Fig. 6 shows a bar graph where the vertical axis lists possible faults, and the horizontal bars represent the calculated satellite activity probabilities. For each fault, the support tool generates a transition diagram and computes the associated p_{mis} and p_{lost} values, which are visualized in the graph.

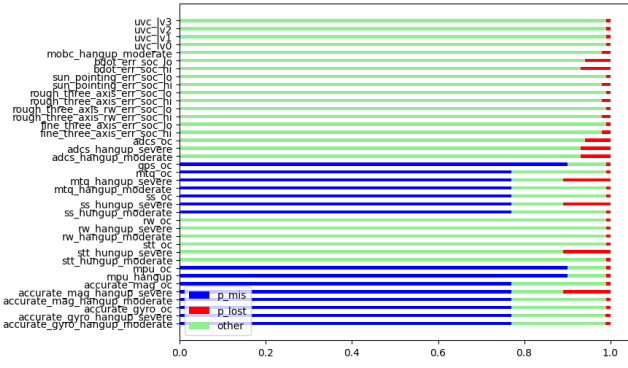


Fig. 6. Satellite activity probabilities for each fault under the FDIR function (blue: mission continuation p_{mis} , red: satellite loss p_{lost} , green: intermediate state)

3.3.2. Transition complexity C_{branch} :

Transition complexity is defined as the number of leaf nodes in a transition diagram. Let v represent a node, and $\deg^+(v)$ the number of outgoing transitions from node v . Then the complexity C_{branch} is given by:

$$C_{\text{branch}} = |\{v \in V \mid \deg^+(v) = 0\}|$$

In the example shown in Fig. 3, this corresponds to the number of branches in the transition diagram, such as $C_{\text{branch}} = 4$. Fig. 7 shows a bar graph where the vertical axis lists fault types and the horizontal axis shows their corresponding C_{branch} values. Notably, faults related to reaction wheels (appearing in the center of the graph) show higher complexity due to transitions involving changes in attitude control laws.

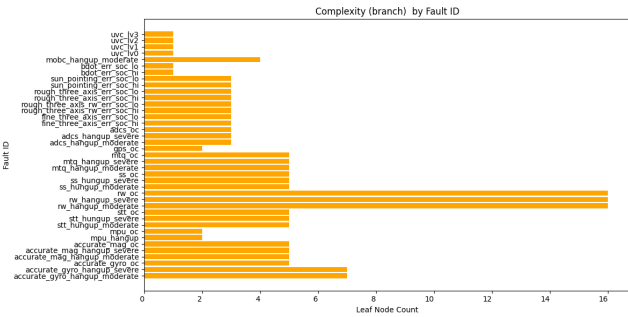


Fig. 7. Transition complexity for each fault under the FDIR function (vertical: fault type, horizontal: number of branches)

FDIR logic that includes multiple conditional branches—such as resetting components in sequence rather than a simple system-wide reset—tends to result in a higher C_{branch} . In practice, verifying all such conditional branches increases development and testing costs, making high-complexity logic less desirable for implementation.

Thus, C_{branch} can be interpreted as an indicator of post-implementation verification cost. Furthermore, if unintended transitions are present in the logic, the transition complexity often becomes anomalously large, making C_{branch} useful for identifying such cases.

3.3.3. Number of unique actions A_{unique} :

This metric represents the number of unique recovery actions that appear in the transition diagram. It is defined as:

$$A_{\text{unique}} = |\{\text{unique}(v) \mid v \in V\}|$$

Here, $\text{unique}(v)$ denotes the recovery action associated with node v . In the example of Fig. 3, two unique recovery actions are observed: an ADCS reset (`adcs_reset`) and a transition to B-dot control (`go_to_bdot`), giving $A_{\text{unique}} = 2$.

Fig. 8 shows a bar graph with fault types on the vertical axis and the number of unique actions on the horizontal axis. Faults with fewer actions—such as GPS failures that do not involve attitude transitions—result in smaller values. In contrast, faults such as reaction wheel overcurrents, which affect attitude or power conditions, tend to produce higher A_{unique} values.

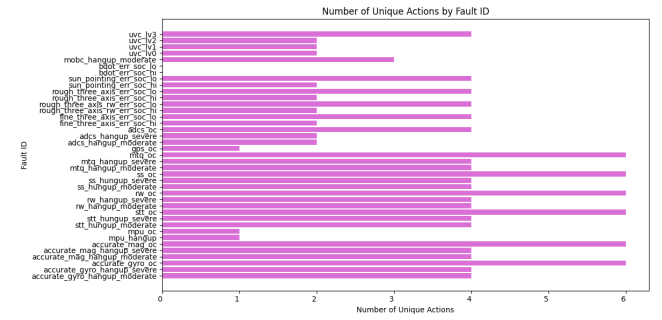


Fig. 8. Number of unique actions per fault under the FDIR function (vertical: fault type, horizontal: number of unique actions)

When implementing the FDIR system, logic that repeatedly uses a single recovery action may result in a high C_{branch} but still maintain $A_{\text{unique}} = 1$. In such cases, verification effort remains low due to the simplicity of the recovery actions. Therefore, A_{unique} can also be treated as a complexity metric.

Additionally, as with C_{branch} , if a fault results in a disproportionately large A_{unique} despite the use of only one type of recovery action, it may indicate the presence of unintended transitions.

3.3.4. Recovery time T :

Each node v in a transition diagram is associated with an elapsed action time, denoted as $\text{time}(v)$. The recovery time T is defined as the maximum total elapsed time along all possible paths P in the transition diagram:

$$T = \max_{P \in \mathcal{P}} \sum_{v \in P} \text{time}(v)$$

where $\mathcal{P}$ is the set of all paths from the root to terminal nodes.

This metric helps identify recovery sequences that, while eventually successful, may take too long for time-critical missions. The support tool includes this evaluation to help designers avoid overly slow FDIR strategies.

In the example of Fig. 3, T corresponds to the longest cumulative time to reach a terminal node. Fig. 9 visualizes recovery time per fault using a bar graph. When compared with Fig. 8, a trend can be observed where faults with a high number of unique actions tend to have longer recovery times. This is due to the execution of time-consuming recovery actions such as attitude resets.

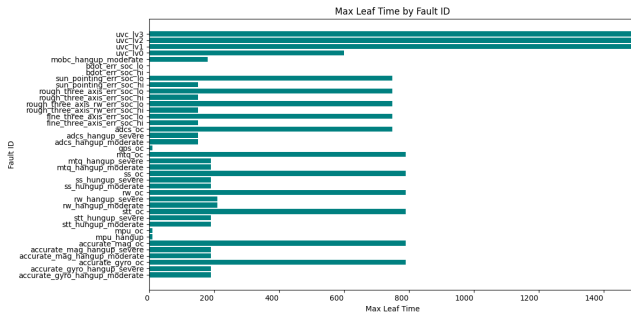


Fig. 9. Recovery time per fault under the FDIR function (vertical: fault type, horizontal: recovery time [s])

3.3.5. Action occurrence frequency $F(a)$:

Each fault i is assigned an occurrence frequency $f_i \in \{1, 2, 3, 4, 5\}$. Let r_i denote the number of times a specific recovery action a appears in the transition diagram corresponding to fault i . Then, the weighted occurrence frequency of action a is defined as:

$$F(a) = \sum_i f_i \cdot r_i$$

This metric represents how frequently a given action is executed across all transition diagrams, weighted by the occurrence frequency of each fault. Actions with higher $F(a)$ values are more likely to be executed in orbit and should therefore be prioritized for verification during FDIR implementation.

Fig. 10 shows a bar graph where the vertical axis indicates the index of each recovery action, and the horizontal axis indicates the weighted occurrence frequency. Actions such as B-dot control transitions and ADCS resets appear most frequently across all diagrams.

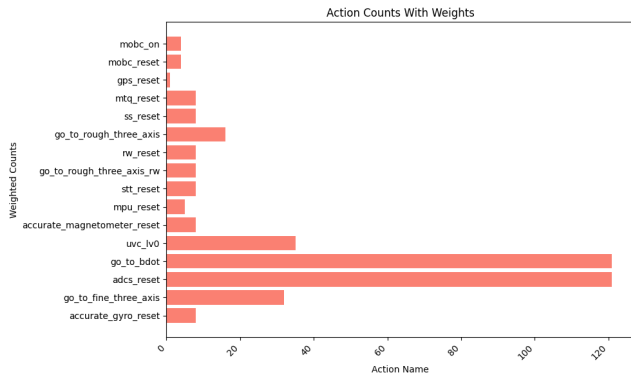


Fig. 10. Weighted occurrence frequency of recovery actions under the FDIR function (vertical: action index, horizontal: frequency. B-dot transition and ADCS reset are the most frequent)

3.4. Interpretation of Outputs

This section summarizes how designers can interpret and utilize the outputs of the design support tool described in the previous sections.

Detection Overlap Warning (Section 3.2)

If detection logic is triggered redundantly, it may indicate overlapping detection conditions. Designers should refine the detection logic to eliminate overlaps and ensure that each detection event corresponds uniquely to a fault. This represents

the *safety* aspect, signalling how reliably the system can avoid inconsistent or conflicting detections.

Transition Diagram (Section 3.2)

Transition diagrams help designers understand how satellite states evolve in response to anomalies and what recovery actions are applied. Analyzing these diagrams allows verification of the intended recovery logic and its structure.

Satellite activity probability (Section 3.3.1)

The values of p_{mis} and p_{lost} represent the probability of mission continuation and satellite loss, respectively. These metrics should be adjusted to match the desired trade-off between safety and continuity based on the design strategy. This metric reflects both the *safety* and *autonomy* aspects.

Transition Complexity (Section 3.3.2)

Transition complexity, represented by the number of leaf nodes in the diagram, reflects the verification cost after implementation. High complexity increases verification burden and may indicate unintended transitions. This metric represents *implementation feasibility*, because higher complexity directly translates into greater testing and integration effort.

Number of Unique Actions (Section 3.3.3)

The number of distinct recovery actions (A_{unique}) affects verification effort. A smaller number of unique actions simplifies testing and is preferable when aiming for safe implementation. This metric represents *implementation feasibility*.

Recovery Time (Section 3.3.4)

The maximum time required to complete a transition to a terminal node (T) is evaluated to ensure timely recovery. For time-sensitive missions, it is important to verify that recovery actions complete within acceptable limits.

Action Occurrence Frequency (Section 3.3.5)

Actions with high weighted occurrence frequency ($F(a)$) are more likely to be executed on orbit. These should be prioritized during testing to ensure their correctness and safety under actual fault conditions.

4. Comparison Between Satellite Models

To assess how differing FDIR design policies influence transition diagrams and evaluation metrics, the proposed design-support tool was applied to two satellite models. System states and databases were prepared for an Earth-orbiting mission and for a deep-space probe, both subject to the same attitude-control fault and both based on a 6U CubeSat platform.

For the Earth-orbiting model, the simulated FDIR policy prioritizes a transition to safe mode (B-dot) and subsequent recovery via ground commands. In contrast, the deep-space probe model assumes a critical orbit-transfer maneuver; its FDIR policy is configured to persist with mission operations even when anomalies occur.

Earth-Orbiting Satellite

Upon anomaly detection, the attitude-control system is reset and the satellite transitions to a safe mode (B-dot). Recovery to nominal mission mode is performed through ground commands once the fault is resolved.

Deep-Space Probe

Because mission success depends on executing a critical orbit-transfer maneuver, the system persistently attempts recovery. Even after anomalies persist, repeated resets are applied so that the mission can continue without waiting for ground intervention.

Fig. 11 and Fig. 12 show the transition diagrams for each model under attitude control system anomalies. Additionally, Table 3 lists the computed values of each evaluation metric.

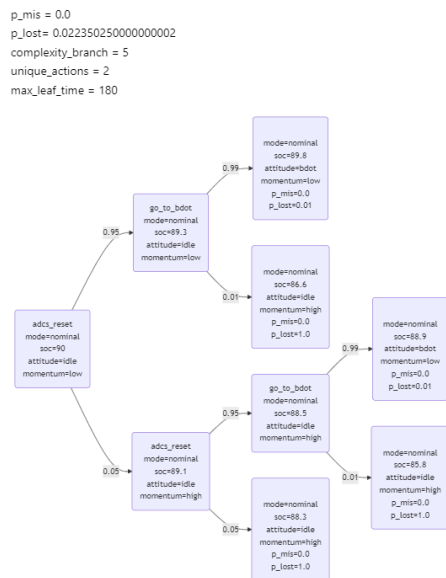


Fig. 11. Transition diagram for the Earth-orbiting satellite model

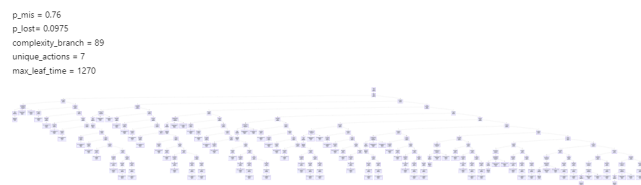


Fig. 12. Overview of the transition diagram for the deep space probe model.

Table 3. Evaluation Metrics for Each Model Under Attitude Anomaly

Metric	Earth-Orbiting	Deep Space
p_{mis} (Mission Continuity)	0.00	0.76
p_{lost} (Loss Probability)	0.001	0.0975
C_{branch} (Transition Complexity)	4	89
A_{unique} (Unique Actions)	2	7
Recovery Time [s]	120	1270

The results quantitatively highlight the characteristics of each design. For the deep-space probe, which accepts greater risk to complete its orbit maneuver, a higher mission-continuity probability (p_{mis}) is achieved, but at the cost of an elevated satellite-loss probability (p_{lost}).

Although the transition complexity (C_{branch}) is large—owing to numerous branches in the probe’s FDIR logic—the number of

unique actions (A_{unique}) remains relatively small. This outcome reflects a qualitative policy of “persistently attempting the same recovery actions,” a behaviour accurately captured by the tool’s automatic transition-diagram generation and metric evaluation.

5. Case Study on FDIR Logic Design

To demonstrate how the proposed design support tool enables FDIR logic design from scratch, a case study was conducted using the Earth-orbiting satellite model. The goal was to design an FDIR logic that prioritizes mission continuity while keeping complexity low, assuming an imaging mission scenario.

Fig. 13 summarizes the outputs of the first iteration, corresponding to the charts in Figs 6 - 10. From these results, the logic for handling the anomaly of decreased battery charge rate was revised to allow autonomous recovery from safe mode without ground command, aiming to improve mission continuity.

Fig. 14 shows the outputs from Iteration 3. Compared to Iteration 1, the mission continuity probability p_{mis} under the charge drop anomaly improved from 0 to 0.76. However, transition complexity increased from 1 to 5. Furthermore, a detection overlap warning appeared, and for attitude-related anomalies occurring during low charge, p_{lost} increased significantly from 0.01 to 1. This indicates that the current FDIR logic is insufficient for handling attitude anomalies under low power conditions and must be improved.

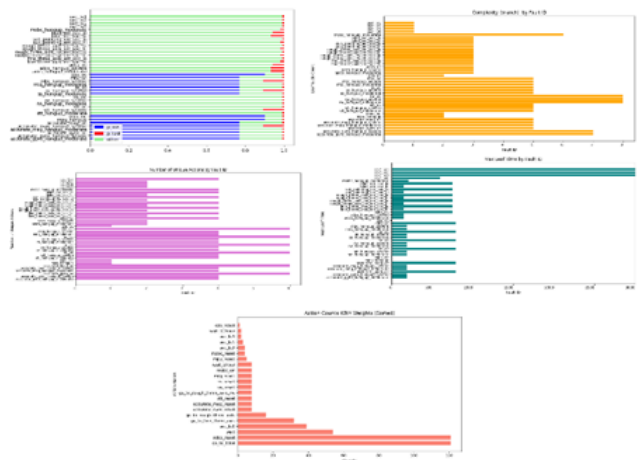


Fig. 13. Overview of Iteration 1: Earth-Orbiting Satellite Model

Specific changes are summarized below.

Detection Overlap Warning

Iteration 3 generated multiple overlap warnings that were absent in Iteration 1.

Mission-Continuity Probability (p_{mis} p_{lost})

For the decreased-battery-charge-rate anomaly, the mission-continuity probability p_{mis} increased markedly, indicating that the modification successfully achieved its intended improvement. Conversely, the satellite-loss probability p_{lost} for attitude anomalies under low charge also rose.

Transition Complexity (C_{branch})

Transition complexity grew substantially for the decreased-battery-charge-rate anomaly, reflecting additional branches in the updated logic.

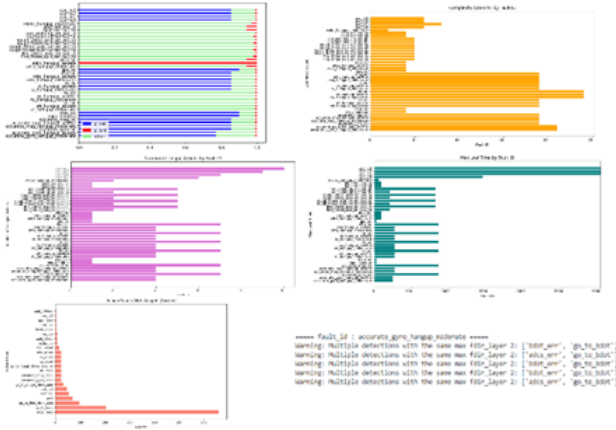


Fig. 14. Overview of Iteration 3: Earth-Orbiting Satellite Model; the figure illustrates overall changes rather than fine details..

Unique Actions (A_{unique})

The number of attitude-system resets climbed sharply between Iteration 1 and Iteration 3, revealing heavier reliance on that recovery action.

These results demonstrate that improving one evaluation metric for a specific anomaly may degrade others or compromise the validity of logic for different anomalies. By iterating such FDIR logic designs using the support tool, it becomes possible to modify the logic while assessing trade-offs across multiple metrics in a comprehensive manner.

6. Sensitivity Analysis of Satellite-Activity Probability

Satellite activity probabilities, p_{mis} and p_{lost} , depend on the success and failure probabilities assigned to recovery actions. To clarify the robustness of these metrics, a sensitivity analysis was performed by perturbing the reset-success probability, a parameter directly related to the root cause of the representative attitude-control fault.

6.1. Influence of Branch Depth on Sensitivity

This subsection derives the output probability and its sensitivity when a branch with uniform probability is repeated n times in series, thereby showing that the degree of repetition amplifies sensitivity in a directly proportional manner.

If the probability of taking a specific branch is q , the probability Q_n of selecting the same path (the same action) after n consecutive transitions is

$$Q_n = q^n,$$

which decays as a power law with respect to the repetition count n .

Defining the relative sensitivity by a logarithmic derivative with respect to q yields

$$S_n \equiv \frac{\partial \ln Q_n}{\partial \ln q} = n.$$

Thus, any small perturbation in the branch probability is magnified by a factor of n in the output probability.

Including a finite variation δq leads to

$$\frac{\delta Q_n}{Q_n} = n \frac{\delta q}{q},$$

indicating that deeper transition sequences exhibit more pronounced fluctuations.

For example, when the reset action is attempted $n = 3$ times, the output probability scales as q^3 and the relative sensitivity is $S_3 = 3$. In general, a deeper transition sequence (larger n) becomes probabilistically rarer, and the same sequence is also more vulnerable to uncertainty in the branch probability.

6.2. Direct-Cause Sensitivity: Reset Success Probability

Figure 16 shows the response of p_{lost} when the reset-success probability is varied. Because the ADCS reset is executed up to three times along the longest failure path, the sensitivity exhibits a cubic profile. As expected, higher reset reliability monotonically lowers p_{lost} , demonstrating that a direct, single-action recovery reduces the overall loss risk.

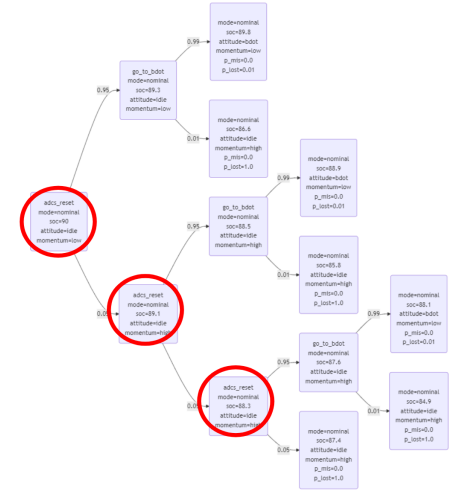


Fig. 15. Transition diagram for a adcs fault. Red circles denote adcs reset.

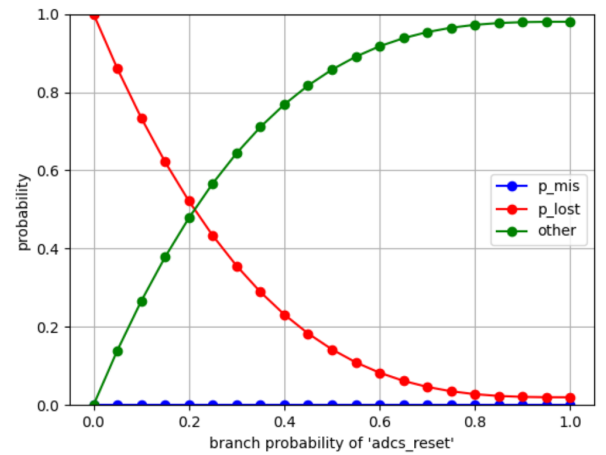


Fig. 16. Sensitivity of satellite activity probability to variations in reset-success probability .

6.3. Single-Step Sensitivity: Mode-Transition Success Probability

Figure 18 illustrates the impact of varying the mode-transition success probability on the satellite activity probability. Since the mode transition occurs only once along the recovery path, the effect of this probability on the overall outcome is minimal. Accordingly, the response curve remains flat, indicating that p_{lost} is

