TOWARDS A CONCEPTUAL DATA MODEL FOR FAULT DETECTION, ISOLATION AND RECOVERY IN VIRTUAL SATELLITE

Sascha Müller [(1)], Andreas Gerndt [(2)]

[(1)] German Aerospace Center (DLR), Brunswick, Germany, sa.mueller@dlr.de
[(2)] German Aerospace Center (DLR), Brunswick, Germany, andreas.gerndt@dlr.de

ABSTRACT

Reliability engineering is an integral part in the design of safety critical systems. Especially spacecraft that cannot receive physical maintenance once delivered into orbit heavily require a fault tolerant design approach. In order to overcome these challenges, concepts from the domain of Fault Detection, Isolation and Recovery (FDIR) are employed. With this paper we present our approach for bringing Model Based Systems Engineering into the realm of reliability engineering using the Virtual Satellite (VirSat) framework. The tool we are developing for this purpose is called VirSat FDIR. In this paper, we discuss a Conceptual Data Model for modelling important aspects of the FDIR domain that we have conceived and implemented for VirSat FDIR. It supports modelling of FDIR faults, recovery, analysis and requirements. We further discuss how these models can be actively used for the purpose of generation of FDIR artefacts and the process of Verification and Validation.

1. INTRODUCTION

In the past years a lot of effort has been invested into enabling Model Based Systems Engineering (MBSE) for the whole life cycle of a spacecraft. Part of these efforts is Virtual Satellite 4 (VirSat4) [1]. VirSat4 is a software framework that allows for the integration of various different engineering processes across the individual phases of spacecraft design and operation, as well as the different disciplines.

An important discipline in the design of safety critical systems such as spacecraft is reliability engineering. No matter how well designed a system is, it must always be able to deal with the presence of faults to some extent. In order to raise trust in handling such faults, concepts from the domain of Fault Detection, Isolation and Recovery (FDIR) are employed.

With this paper we present our approach for bringing MBSE into the realm of reliability engineering using the Virtual Satellite framework. The tool we are developing for this purpose is called VirSat FDIR. Virtual Satellite provides a generic systems engineering language in which a Conceptual Data Model (CDM) capturing one specific engineering aspect can be described. In this paper, we discuss such a Conceptual Data Model for the FDIR domain that we have developed for VirSat FDIR.

The tool currently focuses on the modelling of faults by means of Fault Trees (FT). Fault Tree Analysis (FTA) is a commonly used methodology for performing state-of-the-art failure analysis [2]. The resulting Fault Trees are acyclic graphs that describe how faults propagate through the components and subsystems of a system and eventually lead to a top level failure. VirSat FDIR supports the graphical modelling of Fault Trees and the import of textural descriptions of Fault Trees for integrating supplier data. Furthermore, it also supports the generation of Failure Modes and Effects Analysis (FMEA) tables based on the ECSS standards.

In conjunction to fault modelling, the tool also features modelling support to deal with the recovery related aspects of FDIR. For this purpose we have introduced a concept we call Recovery Automaton. It models the underlying decision process guiding which recovery action should be executed upon observing some fault. The tool also implements the synthesis procedure that we have described in [3]. It takes as input a modelled Fault Tree and aims to generate recovery strategies optimized towards reliability. The focus is in particular in regards to redundancy management.

Due to being conceptualized with the generic engineering language, VirSat FDIR can be used to annotate any Virtual Satellite study with fault and recovery information without requiring domain specific knowledge about the models that are being annotated. This also means that the tool can be used as soon as in the early phase A studies and also in the later phases of the spacecraft life cycle. Furthermore, as Virtual Satellite is made with concurrent engineering in mind, VirSat FDIR inherits this capability and can be employed in parallel to the creation of the main system model.

With the initiative of the VirSat FDIR software we not only want to model FDIR concepts but also actively employ these models to assess the FDIR design and perform verification and validation (V&V) on it. Towards this goal, we support performing two forms of analysis: Reliability Analysis, a quantitative form of analysis that requires precise quantitative information such as the failure rates of the base faults. And Minimum Cutset Analysis, a qualitative form of analysis that only requires the underlying Fault Tree structure.

2. PAPER STRUCTURE

The paper is structured as follows: Section 3 of this paper gives an introduction over the topic of FDIR and fault trees. An introduction on Virtual Satellite and its generic engineering language is given in Section 4. VirSat FDIR and the conceptual data model driving it are then introduced in Section 5. Here we also discuss how to deal with other interesting aspects such as employing configuration control on the level of fault trees. Going into the topic of generation, we continue in Section 6 with what kind of artefact data we can generate from our models. We finally conclude in Section 7 with a summary of this paper and follow up with an outlook to future plans.

3. BACKGROUND ON FDIR & FAULT TREES

The purpose of FDIR lies in keeping a system in a stable and operational state, even in the presence of faults. A *fault* can be any kind of system anomaly. Examples for such faults can be equipment failures, wrong sensor readings, external interferences, random bit flips and many more. However, not every fault is necessary a *failure*. A failure is an actual loss of a mission critical function. The task of FDIR is to find faults in the system and prevent them from turning into failures. While some of the following steps are optional and sometimes omitted, performing FDIR generally means applying the following procedural approach [4]:

- Monitor the system to detect the occurrence of faults.
- Identify the fault and localize it within the system.
- Isolate the fault and prevent further propagation into other parts of the system.
- Perform recovery actions to reconfigure the system and return it into a stable state.

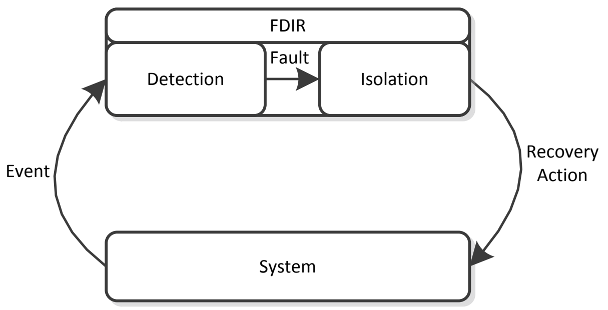

Figure 1. Interaction between system and FDIR

In order to derive how faults relate to each other and eventually lead to a system wide failure, failure analysis techniques such as Fault Tree Analysis can be employed. In general, Fault Trees are graphs consisting of two types of nodes representing events and gates. The root node, or top level event (TLE), usually represents the event of a system failure whereas the leaves of the tree model the event of individual components failing. The leaves are also called basic events (BE). They correspond to a Boolean variable where false represents the initial state of no failure. The variable is considered true in case of a failure event. The branches of the trees are represented by the gates performing operations on the events. Fault propagation in FTs starts at the BEs points over the gates and ends in the TLE.

One of the very basic types of FTs are Static Fault Trees (SFT). They employ Boolean algebra to combine various different failure events by AND and OR operations, often graphically represented as gates, until they sum up to the overall system failure. The failure events are usually related to faulty components of the system. Applying this methodology, statements such as “The system fails if component A and component B fail” can be modelled and refined to arbitrary levels of precision.

A particular extension is the notion of Dynamic Fault Trees (DFT). It introduces temporal understanding and new features to analyse redundancy concepts known as spare management. Accordingly DFTs define a new SPARE gate to model that some faulty component or subsystem is replaced by a spare from a set of redundant parts. In the common understanding of DFTs, the order in which such a spare is chosen is deterministic and defined at design time by the reliability engineer.

With the addition of spares, DFTs also introduce a new node state. In SFTs nodes only have two states: Failed or operational. In DFTs a node can be either failed, active (operational) or dormant (operational). A node that is an unactivated spare is dormant, all other nodes are activated. Together with this state, failure rates for failing actively and failing dormantly can be defined for every BE. These rates are then used for calculating measures of interest such as the probability of the top-level failure after some time (reliability).

4. VIRTUAL SATELLITE 4

Virtual Satellite 4 is a concurrent engineering tool used at the concurrent engineering facility (CEF) at the German Aerospace Center (DLR). It implements an MBSE approach envisioned to cover the whole lifecycle of a satellite, starting from its initial design to the operational phase.

A cornerstone for ensuring modularity, reusability and a high level of semantical precision is the notion of a conceptual data model (CDM), or simply “concept”. A CDM is a meta-model providing the language for capturing and defining a specific aspect in the satellite model. In contrast to generic modelling languages such as SysML or UML, a CDM may be specific to a certain phase or to a certain engineering discipline. In the technical memorandum ECSS-E-TM-10-23 provided by the European Cooperation for Space Standardization (ECSS) a CDM is defined as a

“data model that captures the end-user needs in the end-user terms”.

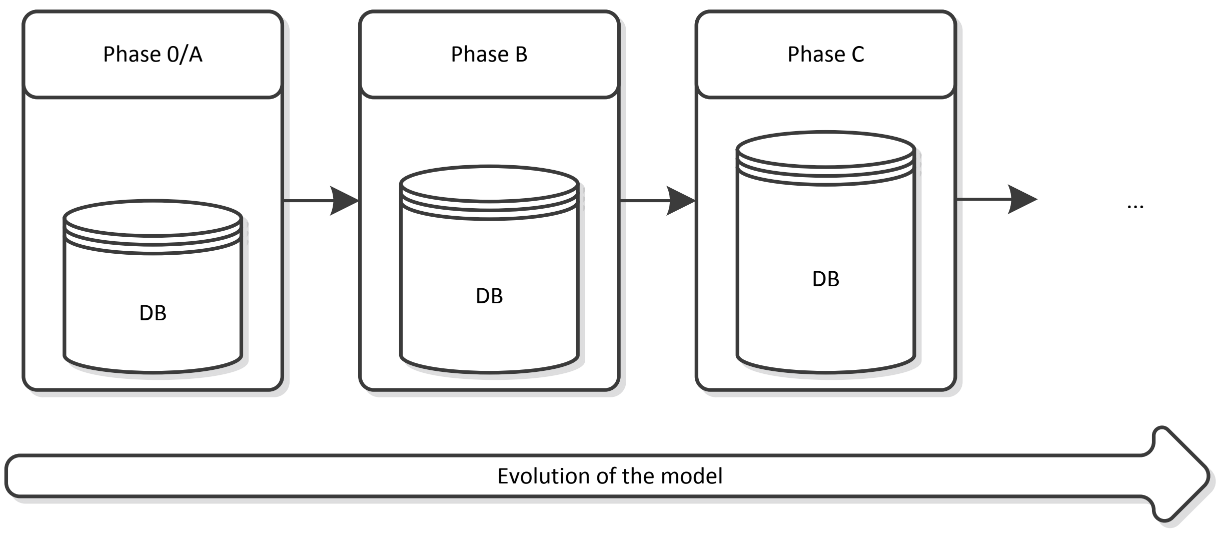

Figure 2. Virtual Satellite database growing along the phases

The high-level of specialization enables CDMs to be semantically precise and restricts models such that those with an unclear interpretation cannot be created. An example for a CDM that is actively being employed in the CEF is described in [5]. This CDM is used for creating Phase 0/A satellite models. Within the framework of the S2TEP project, recent advances have been achieved for bringing VirSat into phase B studies.

In VirSat4, CDMs can be described using its Generic Systems Engineering Language (GSEL) [6]. The GSEL features two types of elements: *StructuralElements* and *Categories*.

- StructuralElements are used to describe system decomposition into its various subsystems and parts and relate parts with each other. An example for a relationship between StructuralElements is a product in a product list typing its actual instantiation in the satellite model.
- Categories, on the other hand, are used for tagging parts with the actual data information. Examples for attachable Categories are mass values, power consumption, interfaces or relevant for this work FDIR information.

To enable concurrent engineering, each instance of a StructuralElement is tagged with an owner. Only the owner is allowed to edit this instance and assign Categories to it. By this manner, merge conflicts are avoided.

A *VirSat4 extension* is a VirSat4 application equipped with a set of CDMs. VirSat4 extensions may share common concepts or be completely independent. When a VirSat4 extension accesses a repository, its concepts are stored in the repository alongside the satellite data model. This enables different VirSat4 extensions that are equipped with different sets of CDMs to communicate with each other. Fig. 3 depicts the architecture of having different VirSat4 extensions operating on a common repository.

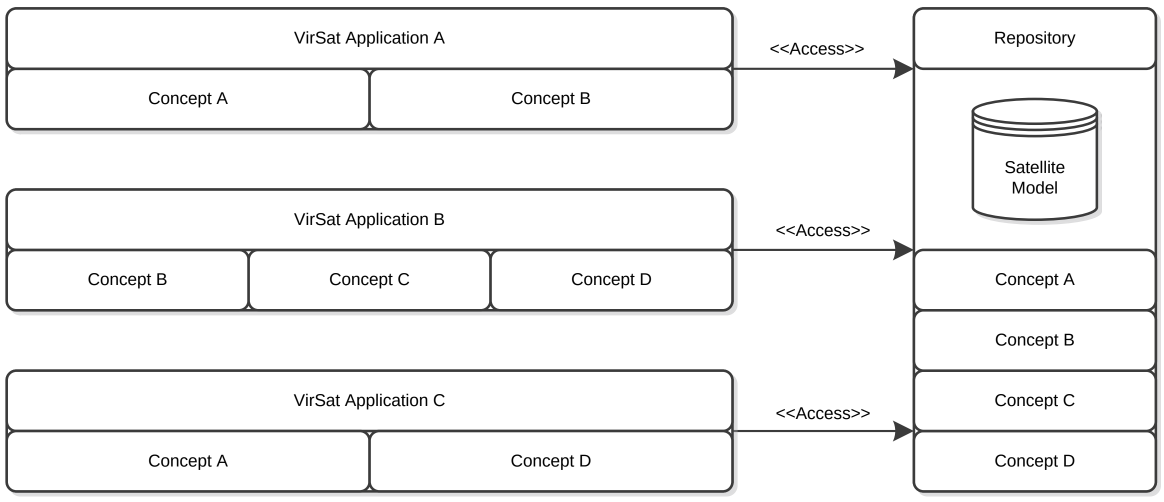

Figure 3. Virtual Satellite 4 architecture with different VirSat4 extensions operating on the same repository

5. FDIR CONCEPTUAL DATA MODEL

In this section we present the main contribution of this paper: Our Conceptual Data Model for the FDIR domain. The CDM deals with mainly two FDIR aspects: Modelling faults and modelling the recovery from them. Detection and Isolation are not considered in the FDIR CDM. However, due to the high importance of detection, it is one of the major future goals to support modelling it as well. Overall, the current FDIR CDM can be split up into of three sections:

- The Fault CDM,
- The Recovery CDM
- And the Requirements and analysis CDM.

The CDM is independent of the concrete structural decomposition of the system and only contains Categories. The actual system decomposition in terms of StructuralElements has to be defined in a separate concept. In the following the word *Component* is used to refer to any element of such a structural decomposition. To provide out of the box modelling capabilities, VirSat FDIR is equipped with the FDIR CDM and a default concept for modelling the system decomposition. The default structural decomposition and how it can be used for configuration control is discussed in section 5.2

The core element of the Fault CDM is the Fault Category. It can be assigned to any Component. To model the cause of a fault, a meta-model following DFTs is employed. The BasicEvent Category models direct causes of a Fault and is supplied with a failure rates and, if it is a transient event, with a repair rate. For indirect causes, every Fault is also equipped with an FT; the fault being the root of the FT. Every FT contains its local graph data, i.e. its edges and the gates describing the propagation from the lower level faults to the root fault. Fig. 4 summarizes the Fault CDM and illustrates the relations between the Categories.

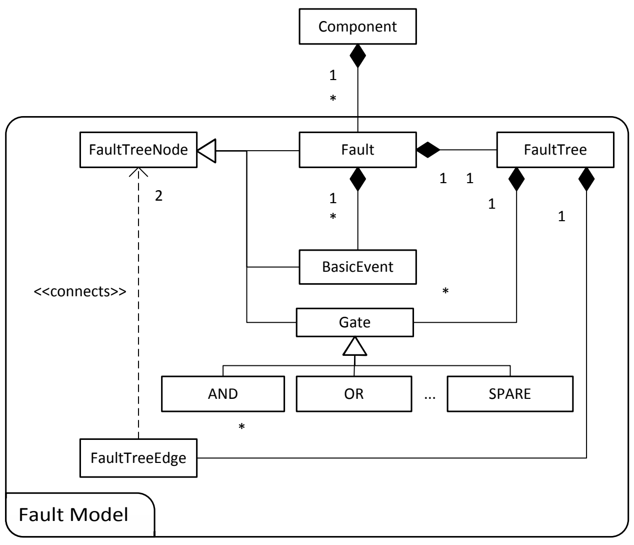

Figure 4. Section of the FDIR CDM for modelling faults

To support provision of fault information from suppliers, VirSat FDIR can import (and export) FTs described in the textual Galileo format [7]. The language has been implemented as a Domain Specific Language using XText. The textual format is also used to convert the FT model into an input representation for external FT analyser tools. VirSat FDIR comes with a native but slow implementation for analysing FTs. For high performance analysis, the tool supports using the STORM [8] tool as a solver backend.

For modelling the recovery aspect we define an object called a *RecoveryAutomaton* [3] (RA). An RA is a Mealy automaton that listens to the events produced by an FT and outputs a list of recovery actions. Through the Fault CDM, the system is abstracted to a pure fault perspective. In our model, this abstracts the System-FDIR interaction initially depicted in Fig. 1 to a FT-RA interaction. Fig. 5 illustrates the simplified view.

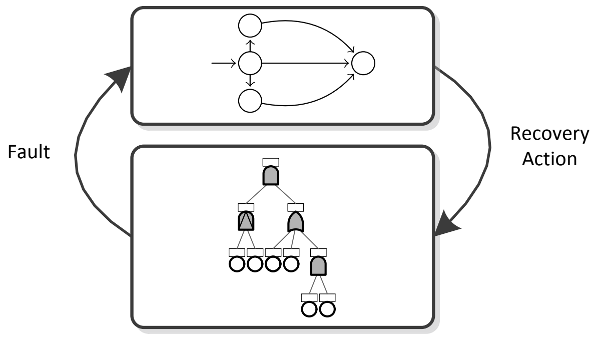

Figure 5: Interaction between Recovery Automaton and Fault Tree

The RA uses guarded transitions. If the events listed by the guards occur in the FT, the RA transitions to a new state and outputs a list of recovery actions. A RecoveryAction is an abstract Category providing an interface for modelling domain specific recovery behaviour. It is envisioned that more RecoveryActions from specific domains can be added by extending the FDIR CDM. The overall Recovery CDM and its relation to the elements of other CDMs are illustrated in Fig. 6.

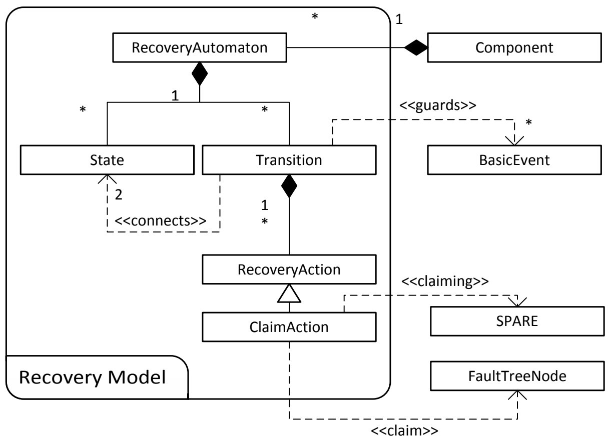

Figure 6. The CDM section for modelling recovery

Per default, the FDIR CDM provides the ClaimAction. The ClaimAction represents the action of a SPARE gate claiming a spare. Going forward, we hope to be capable of extending the palette of default recovery actions included in the FDIR CDM. Current efforts involve are going towards providing a RecoveyAction modelling a transition into the Satellite Safe Mode.

Since both FTs and RAs are objects designed with graphical representation in mind, modelling them in a graphical manner is highly desirable. VirSat FDIR supports a connection to the Graphiti framework for providing diagrams for FTs and RAs respectively.

5.1 FDIR Analysis

In VirSat FDIR, just as the fault and recovery models, Components can also be tagged by analysis and requirements Categories. The analysis Categories reference the fault element to be analysed. Furthermore, the requirements Categories reference the analysis elements and impose expected values on them.

VirSat FDIR currently supports two forms of analysis:

- ReliabilityAnalysis. Reliability is a quantitative property. It is determined by the probability that a system is still functional after a given timeframe t has passed. Also computed for this Category is the Mean Time To Failure (MTTF). The MTTF is the expected time until the fault under analysis occurs. For this analysis all BasicEvent Categories require to be supplied with failure rate data.
- MCSAnalysis. The Minimum Cut Set (MCS) Analysis is a qualitative form of analysis. A MCS is a minimum set of BasicEvents that can lead to the occurrence of the fault under analysis. The fault model does not need to be refined to the point of having failure rate data available. Also computed in this analysis is the fault tolerance: The size of the smallest MCS.

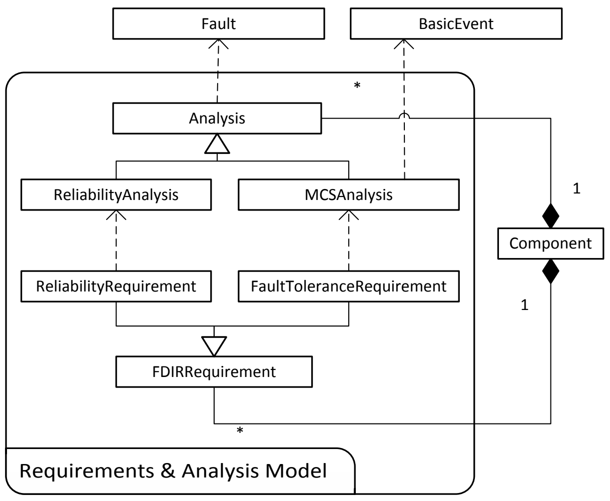

Figure 7. The CDM section for modelling FDIR requirements and analysis

Mirroring the analysis model, the requirements model provides two Categories: One for imposing expected results on the ReliabilityAnalysis and one for imposing expected results on the MCSAnalysis. The overall CDM is depicted in Fig. 7. On this basis, automatic V&V is now performed as follows: Whenever an analysis result changes, VirSat FDIR executes an automatic validator that checks the fulfilment of the FDIR requirements. It then creates warnings for non-fulfilled requirements.

5.2 Configuration Control

When designing space systems, it is often not the case that the entire system has to be designed from scratch. Parts of previous studies, product definitions and many other design artefacts have a high potential for reuse. This holds especially when considering a series of space systems [1]. VirSat4 FDIR comes with a Product Structure CDM for describing the system decomposition allowing for the reuse of designed products. Considered here is a simplified version depicted in Fig. 8.

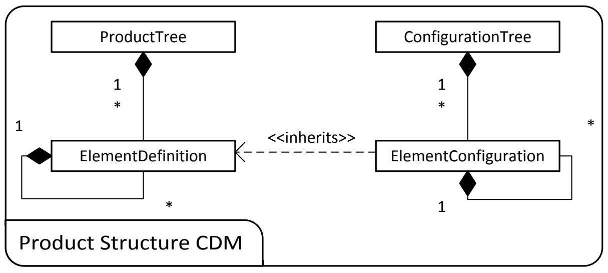

Figure 8. Simplified Product Structures CDM

A ProductTree (PT) represents a container for product definitions, called the ElementDefinitions (ED). An ED abstractly represents a Component (Product). A ConfigurationTree (CT) represents a concrete instance of the system, i.e. the satellite model, and contains ElementConfigurations (EC). ECs can be typed by EDs and inherit their Categories. In particular, fault Categories can be assigned to EDs. In this manner, product level FTA can be reused over multiple ECs and also over multiple missions, each with their own CT.

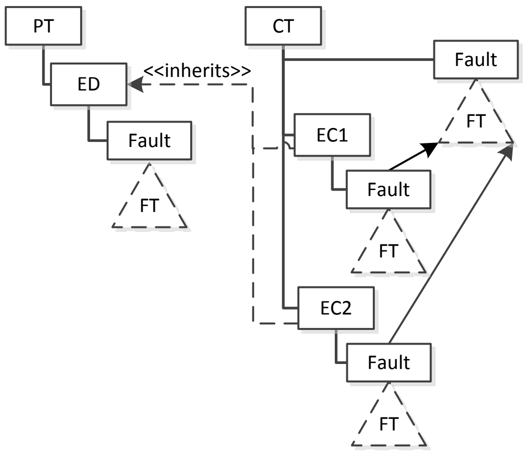

Figure 9. Example of a Product Tree and a derived Configuration Tree

Fig. 9 illustrates an example of constructing an FT over a product structure with two ECs typed by one ED.

6. FDIR ARTIFACT GENERATION

To increase the value of created FDIR models and to reduce the effort in creating them, VirSat FDIR aims to generate derivable information from the models. Currently, the tool supports the generation of recovery models focusing on increasing system reliability and on the generation of FMEA tables. In the future, we aim to generate entire FDIR reports from the FDIR model.

6.1 Generation of Recovery Models

In VirSat FDIR, we have implemented the methodology reported in [3]. This allows the generation of RAs from non-deterministic FTs. Typically, gates in FTs are interpreted deterministically. However, this requires the designing engineer to know a-priori the optimal strategy for managing the redundancies. In non-deterministic FTs SPARE gates do not simply claim spares deterministically from left to right. Instead, they claim according to the ClaimActions by an RA. By transforming a non-deterministic FT into a Markov Automaton and optimizing its schedule, a reliability optimal RA can be generated from the fault model.

6.2 Generation of FMEA tables

FTs are one of many ways for describing fault relations. While they are powerful for precisely describing the causes of a fault, in the space industry Failure Modes, Effects and Criticality Analysis is the standard go-to tool demanded by many standards such as the ECSS. In VirSat FDIR we follow the ECSS standard ECSS-Q-ST-30-02C. To guarantee compatibility with these standards, it is necessary to provide a view on the fault model through FMEA tables. The solution for obtaining an FMEA entry for a given Fault equipped with a FT is straightforward. An ECSS compatible FMEA table varies depending on the item under consideration, but commonly it contains at least the following columns:

- Item. The name of the fault under consideration.
- Failure Modes. The direct causes of a fault. In an FT these are the basic events of a fault and the direct child faults.
- Failure Causes. The failure modes of the failure modes. Hence, they can be computed just as the failure modes of the fault under consideration.
- Probabiliy Level. The probability level can be obtained by performing a ReliabilityAnalysis and categorizing the result according to a mission specific probability level table.

7. CONCLUSION AND FUTURE WORK

In this paper we have presented our approach for performing MBSE in the FDIR domain using an FDIR CDM. Furthermore, we have introduced our new tool VirSat FDIR that implements the presented methodology. In the future we plan to expand the CDM to also model the aspects for detection. Other interesting aspects intended for follow up are the consideration of more recovery actions in the recovery model, generation of FDIR reports from the FDIR model and we aim to bring VirSat FDIR into the later phases of spacecraft V&V and design by integrating a simulation based FDIR validation approach into VirSat FDIR.

8. REFERENCES

1. Lange, C., Grundmann, J. T., Kretzenbacher, M., & Fischer, P. M. (2017). Systematic reuse and platforming: Application examples for enhancing reuse with model-based systems engineering methods in space systems development. *Concurrent Engineering*, 1063293X17736358.
2. Ruijters, E., & Stoelinga, M. (2015). Fault tree analysis: a survey of the state-of-the-art in modeling, analysis and tools. *Computer science review*, 15, 29-62.
3. Müller, S., Gerndt, A., & Noll, T. (2017). Synthesizing FDIR Recovery Strategies From Non-Deterministic Dynamic Fault Trees. In *AIAA SPACE and Astronautics Forum and Exposition* (p. 5163).
4. Wander, A., & Förstner, R. (2013). Innovative fault detection, isolation and recovery strategies on-board spacecraft: state of the art and research challenges. Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth eV.
5. Fischer, P. M., Deshmukh, M., Maiwald, V., Quantius, D., Gomez, A. M., & Gerndt, A. (2018). Conceptual data model: A foundation for successful concurrent engineering. *Concurrent Engineering*, 26(1), 55-76.
6. Fischer, P. M., Lüdtke, D., Lange, C., Roshani, F. C., Dannemann, F., & Gerndt, A. (2017). Implementing model-based system engineering for the whole lifecycle of a spacecraft. *CEAS Space Journal*, 9(3), 351-365.
7. Dugan, J. B., Sullivan, K. J., & Coppit, D. (2000). Developing a low-cost high-quality software tool for dynamic fault-tree analysis. *IEEE Transactions on reliability*, 49(1), 49-59.
8. Dehnert, C., Junges, S., Katoen, J. P., & Volk, M. (2017, July). A storm is coming: A modern probabilistic model checker. In *International Conference on Computer Aided Verification* (pp. 592-600). Springer, Cham.