INTERNATIONAL DESIGN CONFERENCE – DESIGN 2024
https://doi.org/10.1017/pds.2024.197

Extending the function failure modes taxonomy for intelligent systems with embedded AI components

Felician Campean [1,✉], Unal Yildirim [2], Aleksandr Korsunovs [1] and Aleksandr Doikin [1]

[1] University of Bradford, United Kingdom,
[2] Hubei Key Laboratory of Automotive Power Train and Electronic Control, China

✉ F.Campean@Bradford.ac.uk

Abstract

Early consideration of failure modes in the feature development process is essential to identify and trace risks across the physical and embedded AI components of intelligent systems, to enhance the robustness of the feature delivery as well as trust in the AI. This paper introduces an extension of the AIAG/VDA function failure modes taxonomy, to facilitate the integrated analysis of complex intelligent systems with embedded AI. A case study of an autonomous driving feature is discussed as validation of the proposed taxonomy.

Keywords: robust design, systems engineering (SE), autonomous systems, embedded AI, failure mode and effect analysis (FMEA)

1. Introduction

Across industries, many systems and products that are currently developed, are increasingly relying on embedded AI components to deliver enhanced functionality and support robotic and autonomous control features. Referring to the field of autonomous driving, and in particular advanced driver assist systems (ADAS) features, we have seen significant advancements made possible by the introduction of machine learning (ML) and AI algorithms, embedded with the physical components to deliver functionality for enhanced safety and driver comfort. One consequence from the introduction of AI is that previously unconnected systems are becoming significantly interdependent, which brings new risks, often difficult to manage with conventional approaches (Koopman & Wagner, 2016).

Conventional design methods have been developed and refined over time to understand and characterise the behaviour of systems, and on this basis take design actions to improve and prove the robustness and reliability of the systems (Eifler et al, 2023). For example, Failure Modes and Effects Analysis (FMEA) is a common method used across industries to manage the design process to provide assurance by design for the system performance objectives, including safety and robustness. The issue of robustness of ML and AI components embedded within engineering systems is currently receiving a lot of attention from researchers. Much of this research (see for example Shafique et al, 2020) focusses on vulnerability as an effect of lack of robustness, with explicit links to systems attributes including safety, security, dependability, etc. Many of these approaches concentrate on the uncertainty associated exogeneous factors, with relatively little attention given to the behaviour of the physical system itself. Yet assurance for safety and robustness in the performance of the function is required across the whole system, and is the focus of new standards such as the ISO 21448:2022 safety of the intended function for road vehicles.

From a systems engineering perspective, early consideration of failure modes in the feature development process is essential to identify and trace risks across the physical and embedded ML/AI components of intelligent systems, to enhance the robustness of the feature delivery as well as trust in the AI.

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

So far, taxonomies for failure modes have been largely domain dependent; e.g. Tumer et al (2003) for electromechanical systems, Avizienis et al (2004) for computing systems, and Thieme et al (2020) for software systems. The AIAG&VDA (2019) FMEA handbook, integrates approaches from previous standards and guidelines from the US-based Automotive Industry Action Group (AIAG) and the German Association of the Automotive Industry (VDA), and is now commonly used as reference across the automotive industry. This provides a concise taxonomy for function failure modes, which has the advantage that can be deployed early in the system and feature development, as unlike other domain specific taxonomies, is centred on the function rather than the structure of the system. However, there has been no attempt to investigate the applicability of the AIAG&VDA or any other taxonomies to address the whole system, including embedded AI intelligent features.

This paper aims to address this gap. We first carry out a review of the existing taxonomies and related research, before focussing on the AIAG&VDA (2019) with a critical appraisal. We focus on the AIAG&VDA methodology because it is widely used in industry, and its functional basis makes it suited for the early feature analysis. The novel contribution of this work is the proposed extensions to the AIAG&VDA function failure modes taxonomy, underpinned by theoretical considerations of the state-transition basis of the function representation and the typology of variables associated with the flows of energy, materials and information, and bringing insight from the contrast with other taxonomies, in particular that of Avizienis et al (2004). A case study of an ADAS feature (adaptive cruise control based on traffic sign recognition) is considered to evaluate the applicability of the proposed taxonomy to a smart feature that includes embedded AI.

2. Literature review

2.1. Failure mode taxonomies for physical systems

The early engineering approach to failure modes classification was largely based on mechanics of materials failures, or physics of failure. For example, Collins (1993) discussed the phenomenology of failure based on: (i) manifestation – e.g. elastic or plastic deformation, rupture or fracture, or material change; (ii) the impact of external acting agents such as forces and environmental factors, and their intensity and frequency; and (iii) topology of the part that determines the location of failure. For mechanical components this approach is still used to characterise failure behaviour, with mathematical models employed to quantify the physics of failure based on material properties and topology, to predict reliability attributes. Tumer et al. (2003) extended this approach towards the development of a more comprehensive taxonomy covering the electromechanical domain, arguing that physics-based description can provide an understanding of the failure modes at their most “elemental” state, and thus provide practitioners with a true understanding of a way in which the failure occurs.

Taking the view of more complex systems, which are made up of multiple components with different failure behaviours, requires a systems view and approach to classifying failure modes. The common approach is to relate the definition of failure modes to the ways in which a system can fail to deliver its intended functional requirements. Blischke & Murthy (2000) discussed failure modes with respect to functions of a system/component by referring to the taxonomy of Blache & Shrivastava (1994) which includes extended failures and intermittent failures. Extended failures are categorized into partial failures and complete failures, with further classification into sudden and gradual failures. This led to the definition of "catastrophic failure” as “complete and sudden failure”, and “degraded failure” as a “gradual and partial failure”, respectively.

The consideration of failure modes is particular important to the identification and management of risks in the design process. The AIAG&VDA (2019) guidelines for Failure Mode and Effects Analysis (FMEA) is based on function analysis, describing what the system or element of the system is intended to do, and function failure mode analysis guided by a concise taxonomy which includes seven categories of function failure: no function; partial function; degraded function; intermittent function; unintended function; exceeding function; and delayed function. The AIAG&VDA (2019) provide guidelines and examples on how this taxonomy can be used across different product development domains, as “Design FMEA”, “Process FMEA” and “supplemental FMEA for monitoring and system response (FMEA-MSR)”. The implementation of this methodology requires a detailed consideration of the way in which

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

functional performance needs to be described and specified. For example, Clausing and Frey (2004), have introduced a measurable way of describing performance of functionality by relating the functionality to a set of conditions, lower or upper limit (one-sided failure mode) or both (two-sided failure mode), under which the system operates without failure, which has been effective in use in conjunction with the FMEA function failure approach.

Focussing on addressing the dependability of embedded systems, Avizienis et al. (2004) have introduced the most comprehensive failure modes taxonomy, by considering service failure modes, development failure modes, and dependability and security failure modes, with further categorization into sub-failure modes from multiple viewpoints. In terms of failure domain viewpoint, service failure modes are categorized into content failure modes, early/late timing failure modes, halt failures and erratic failures. “Service” and “Dependability and Security” failure modes can also be associated with performance of functionality as they refer to “limit” and “time” in the definition of failure modes, for example, early/late timing failure mode is defined on the basis of the delivery of the service being too early/late. In order to more effectively support the approach to functional safety, ISO26262 (2018) introduced a failure mode classification based on the type of hardware elements which are categorized into non safety-related or safety-related hardware elements. Each element is related to a fault type: i) “Non safety-related” faults are associated with non safety-related hardware elements; ii) “safe”, “detected multiple-point”, “perceived multiple-point”, “latent multiple-point” and “residual/single-point” faults are considered in the context of safety-related hardware elements.

2.2. Failure mode taxonomies for software and AI

The proliferation of AI components in the architecture of complex systems, in particular associated with robotic and autonomous systems, has made it necessary to consider a holistic approach to the understanding of failures of systems, including both the physical and embedded AI components. There are differences between AI failure modes and software failure modes from a technology perspective (Kumar et al., 2022). Traditional software is about performing a task based on the programme which it is set up, whereas ML -a subset of AI- uses algorithms to automatically learn insights and recognize patterns from data.

O’Halloran et al. (2012) pointed out that software failures are non-physical, and therefore they can be classified under signal/information flow of functionality. Chu et al. (2011) and McNelles et al. (2017) provide system-specific taxonomies of hardware and software failure modes which can be related to material/energy flow of functionality and signal/information flow of functionality respectively. Chu et al. (2011) introduced a taxonomy for digital components for the purposes of probabilistic safety assessments by mapping failure modes across different level of details (system, division, etc.). McNelles et al. (2017) focused on the introduction of a failure mode taxonomy for Field Programmable Gate Arrays (FPGA) which are a form of programmable digital hardware configured to perform digital logic functions. They categorized failure modes under “Design” and “Operation" which are further classified into other failure modes. Both Chu et al. (2011) and McNelles et al. (2017) include elements of the AIAG&VDA taxonomy (e.g. loss of function) and Avizienis et al. (2004)`s taxonomy (e.g. failure to actuate in time). Chu et al. (2011) and McNelles et al. (2017) also introduced failure modes for specific system levels. For example, “Random/Unknown Signal” is shown as a software failure mode at sub-component level. Thieme et al. (2020) proposed a structure for software functions, and introduced a taxonomy consisting of 29 failure modes categorized based on the proposed software function structure, e.g. a taxonomy for value-related and timing-related failure modes in relation to the output of a software function. While the applicability of this taxonomy on AI systems has not been tested yet, they suggest that the taxonomy has the potential of underpinning software/design FMEA.

A considerable amount of literature has been published on the use of AI (i.e. ML) for failure mode identification (e.g. Hu et al., 2023), however little is known about the failure modes of AI/ML per se and their impacts on autonomous driving tasks except for a few studies. Scott and Yampolskiy (2020) referred to four dimensions in the classifications of AI failures: consequences; agency; preventability and stage of introduction in the product lifecycle. Kumar et al. (2022) approached ML failure modes from a cybersecurity perspective. Specifically, they point out failure modes induced by attacks against the ML systems and classify them into intentional and unintentional failures.

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

3. Methodology

3.1. Background to AIAG&VDA reasoning about function failure modes

The AIAG&VDA (2019) recommends a taxonomy of failure modes construed as "*failures of a function*" including the seven types of potential failure modes:

1. No function [NF] or loss of function (e.g. inoperable, fails suddenly);
2. Partial function [PF] (e.g. performance loss)
3. Degraded function [DgF] (e.g. performance loss over time);
4. Intermittent function [IF] (e.g. random operation starts/stops/starts);
5. Unintended function [UF] (e.g. operation without command, or at the wrong time);
6. Exceeding function [EF] (e.g. operation above the acceptable threshold);
7. Delayed function [DF] (e.g. operation after the intended time interval).

While the standard indicates that this is not an exclusive list, the benefit of a concise taxonomy is that it provides the practitioners with a concise heuristic for reasoning about the ways in which a function could fail. An important update in this updated standard, compared to previous versions of the standard, is the introduction of the last two categories - exceeding function and delayed function. In particular, *delayed function* is important because it enables the explicit consideration of requirements and specifications for timing to be considered within the FMEA analysis.

From an engineering point of view, in order to operationalise this taxonomy, the failure modes need to be judged in relation to the attributes and specifications of the function. In practice, this principle is often quoted as "*the failure modes are measured in the same unit as the function*", exemplified in the standard with graphical illustrations for each failure mode type. It follows that in order to carry out a robust failure mode analysis, the functional architecture needs to be defined and specified, at least to a level where the functional requirements can be quantitatively identified.

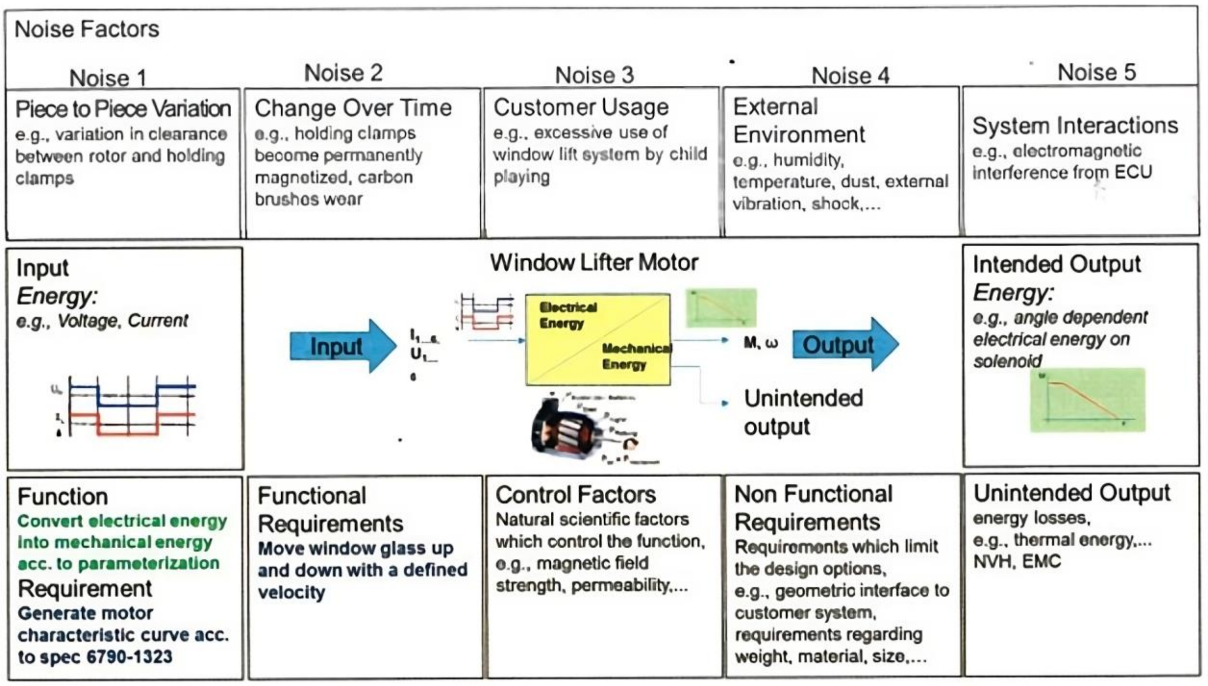
The function model referred to in AIAG&VDA (2019) illustrated in Figure 1, is based on an enhanced P-Diagram based on Taguchi's framework for robust design (Taguchi, 1986). At the core of this is a black-box model of the system, with inputs and outputs identified in terms of flows of energy, material and information. The function, functional and non-functional requirements are explicitly identified, enabling to identify a specification for the *intended output*. The controls, noise factors and diverted output are also identified, but this presumes that the design solution is known, which is needed for the *failure causes* reasoning, not necessarily needed for the function failure reasoning. Within the AIAG&VDA (2019) this framework is illustrated with good level of detail at a component level, but insufficient guidance provided for how the analysis should be carried out at a system level.

3.2. Proposed framework for extending the failure mode taxonomy

Early conceptual analysis of new features as system-of-interest (SOI): the early analysis of new features, based on activity diagrams and use case analysis, would seek to identify functionality and functional requirements based on the analysis of operations - e.g. the way the user will interact with the system. Operations can be characterised as exchanges, i.e. identifying the key attributes and variables involved, with requirement specifications reflecting the performance expected, thus reflecting the goals and intentions, rather than structure behaviour (Vermaas, 2009). The first level of decomposition focuses on logic functionality, with achievement of the high level functions explicitly linked to the achievement of lower level functions. From a mathematical representation viewpoint, this is akin to establishing decomposition on the basis of first order logic models (Barwise, 1977). This approach is useful from the point of view of multidisciplinary systems (including hardware and embedded software and AI), because the functionality can be abstracted to operations on flows, thus enabling an integrated treatment across the different nature of components.

In terms of basic function representation, we refer to the modified statechart approach of Yildirim et al (2017, 2023), in which the function is defined as a state transition or transformation, referring to the input and outputs states of objects, described in terms of measurable attributes (Figure 2). This facilitates the representation of functions with multiple attributes, including time.

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press



Noise Factors

Noise 1 – Piece to Piece Variation: e.g., variation in clearance between rotor and holding clamps

Noise 2 – Change Over Time: e.g., holding clamps become permanently magnetized, carbon brushes wear

Noise 3 – Customer Usage: e.g., excessive use of window lift system by child playing

Noise 4 – External Environment: e.g., humidity, temperature, dust, external vibration, shock,...

Noise 5 – System Interactions: e.g., electromagnetic interference from ECU

Input – Energy: e.g., Voltage, Current

Window Lifter Motor

Intended Output – Energy: e.g., angle dependent electrical energy on solenoid

Function – Convert electrical energy into mechanical energy acc. to parameterization. Requirement – Generate motor characteristic curve acc. to spec 6790-1323

Functional Requirements – Move window glass up and down with a defined velocity

Control Factors – Natural scientific factors which control the function, e.g., magnetic field strength, permeability,...

Non Functional Requirements – Requirements which limit the design options, e.g., geometric interface to customer system, requirements regarding weight, material, size,...

Unintended Output – energy losses, e.g., thermal energy,... NVH, EMC

Figure 1. P-Diagram as function context in AIAG&VDA (2019)

Function failure reasoning can be carried out in relation to the achievement of the specified output state attributes, given the input state attributes. Following the AIAG&VDA (2019) guidance, if the input state attributes are incorrect, this should be treated as a failure mode of the preceding function in the chain and dealt with within that function failure analysis. The robustness approach, based on the noise factors framework (illustrated in Figure 1) to consider the impact of exogeneous variables on the input state attributes offer a powerful complementary approach.

In order to provide stronger guidance for the function failure modes reasoning and articulation of failure modes, we focus on the *types of variables* that are used to measure the attributes that describe the states. The two fundamental types are *continuous variables* and *categorical variables* (which include binary). The flows of *energy* and *material* are typically associated with measurable attributes described by *continuous variables*, corresponding to fundamental units; nonetheless, categorical variables could also be encountered in some cases. Conversely, an *information* flow is predominantly described with *categorical variables*, for the function analysis at higher levels of resolution in system decomposition, where functions are not explicitly linked to a specific way of achievement.

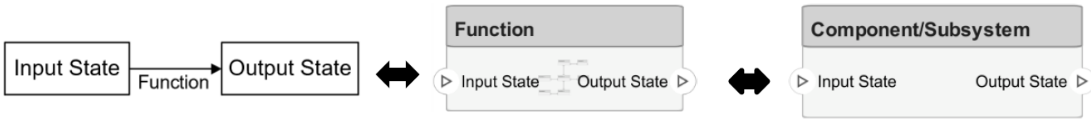


Figure 2. Function abstraction as state transition (Yildirim et al, 2023)

In relation to the application of the AIAG&VDA (2019) function failure mode taxonomy, consideration of the type of variable describing the function is relevant, because the description of the function failure modes, and even the choice of a type of function failure mode, can be explicitly related to the type of variable used to measure the output attribute.

On this basis, Figure 3 illustrates the proposed approach for extending the AIAG&VDA failure modes taxonomy, by considering the type variable associated with the attribute of the flow, with prescriptive indication of how the failure mode would manifest in relation to the observed variables.

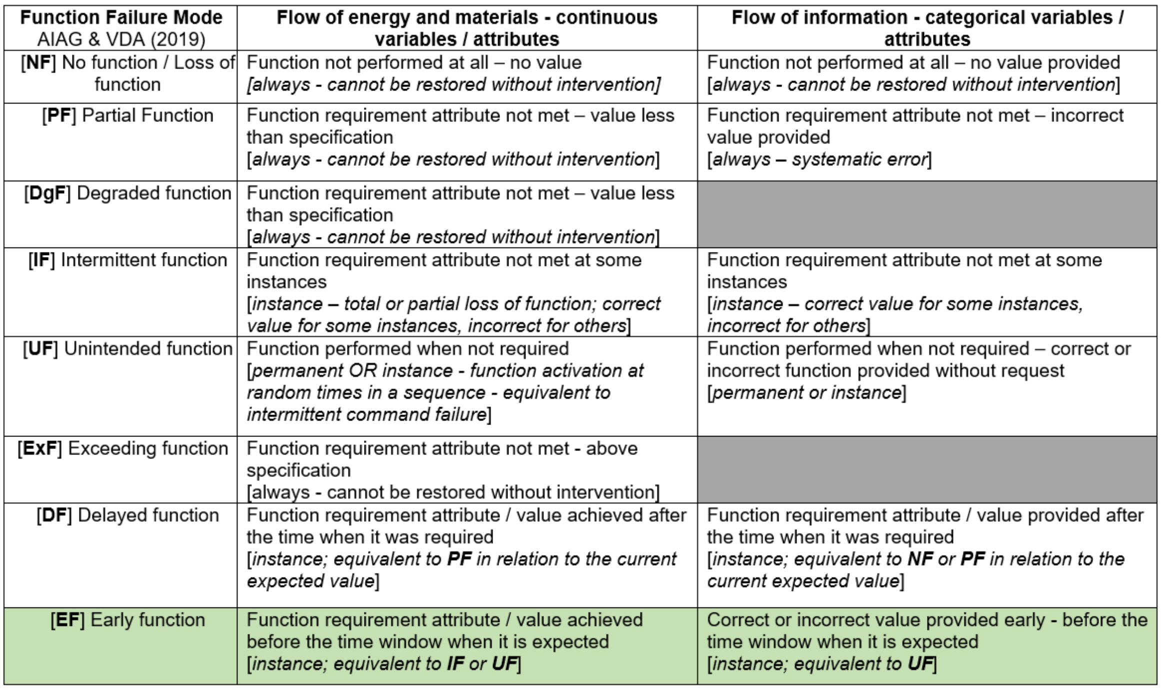
The analysis in Table 3 also considers the fact that the context of most engineering functions (in particular those associated with intelligent control features, representing the main focus of this paper) is dynamic or continuous in time. Thus, assuming that the function of the system relates to a continuous sequence of repeated events (e.g. a sensor in the loop with the embedded AI decision support system

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

and an actuating system, implementing a *Sense - Plan - Act* sequence), we make distinction based on the phenomenology of the failure modes; i.e. failure modes that can occur in one or several instances or in relation to an event, or permanently - across the whole sequence.

Figure 3 provides heuristic guidance for the implementation of the taxonomy in relation to both continuous and categorical variables. Focussing in particular on the categorical variables, most likely associated with embedded AI functionality as information flow / exchange, the analysis shows that not all AIAG&VDA taxonomy categories are applicable. For example, *DgF* and *ExF* are not considered relevant, since degradation of information content is the same as incorrect value provided - which has been categorised as *PF*; same logic with *EF*. Figure 4 summarizes the alignment between the proposed categorisation of information flow failure modes in Figure 3 and the reference framework proposed by Avizienis et al (2004). We consider that the categorical value of a function output state attribute denotes the information "content" in relation to Avizienis' framework.

The consideration of this alignment prompted the broader consideration one whether "Early Function" should be added as a category in the AIAG&VDA taxonomy. For a categorical variable, providing a correct value early would constitute an "unintended function" (same if an incorrect value is provided early). For a continuous variable, if a correct value is provided early, in relation to the instance at which the judgement is made, this would be either an "intermittent failure" (e.g. if this represents a temporary increase in the operating level), or an "unintended function" (e.g. if the current level of operation was no function). However, by introducing "Early Function" as a new category in the failure mode taxonomy in Figure 3, thus extending the AIAG&VDA taxonomy, this enriches the process heuristic (in the sense that it prompts the engineers to consider early function provision as a potential failure mode), and reduces the chance of variability in the categorisation, i.e. where different teams provide different taxonomical interpretation for the same event.



Function Failure Mode AIAG & VDA (2019)	**Flow of energy and materials - continuous variables / attributes**	**Flow of information - categorical variables / attributes**
[**NF**] No function / Loss of function	Function not performed at all – no value *[always - cannot be restored without intervention]*	Function not performed at all – no value provided [*always - cannot be restored without intervention*]
[**PF**] Partial Function	Function requirement attribute not met – value less than specification [*always - cannot be restored without intervention*]	Function requirement attribute not met – incorrect value provided [*always – systematic error*]
[**DgF**] Degraded function	Function requirement attribute not met – value less than specification [*always - cannot be restored without intervention*]	
[**IF**] Intermittent function	Function requirement attribute not met at some instances [*instance – total or partial loss of function; correct value for some instances, incorrect for others*]	Function requirement attribute not met at some instances [*instance – correct value for some instances, incorrect for others*]
[**UF**] Unintended function	Function performed when not required [*permanent OR instance - function activation at random times in a sequence - equivalent to intermittent command failure*]	Function performed when not required – correct or incorrect function provided without request [*permanent or instance*]
[**ExF**] Exceeding function	Function requirement attribute not met - above specification [always - cannot be restored without intervention]	
[**DF**] Delayed function	Function requirement attribute / value achieved after the time when it was required [*instance; equivalent to* ***PF*** *in relation to the current expected value*]	Function requirement attribute / value provided after the time when it was required [*instance; equivalent to* ***NF*** *or* ***PF*** *in relation to the current expected value*]
[**EF**] Early function	Function requirement attribute / value achieved before the time window when it is expected [*instance; equivalent to* ***IF*** *or* ***UF***]	Correct or incorrect value provided early - before the time window when it is expected [*instance; equivalent to* ***UF***]

Figure 3. Proposed extension to failure mode taxonomy considering the type of variables

A final point of discussion in relation to the analysis provided in Figure 4, relates to the correct content arriving late; this could be considered either IF - if the arrival of the signal is with a delay that can be compensated, or NF - if the delay is significant. Avizienis et al (2004) have provided a similar discussion, with the suggestion that the delay grading is based on the effect (or consequence) on the service should be considered. While in practice engineering teams would be able to make such judgements, from a methodological perspective this is problematic, as it implies that engineers should consider both failure modes and the effects at the same time, which is not recommended practice, as it

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

might lead to a biased judgement of effects. In the spirit of AIAG&VDA, and good systems engineering practice, the expectation is that the specification for the timing requirement should include both the target and the grading for the delay.

		Timing		
		Correct	Incorrect	
			Early	*Late*
Content	Correct		**UF / EF**	**NF / IF**
	Incorrect	**PF**	**UF / EF**	**UF**
	None	**NF**	**NF**	**NF**

Figure 4. Alignment between the proposed failure mode categorisation for categorical variables and the taxonomy of Avizienis et al (2004)

We illustrate the implementation of this extended taxonomy and evaluate its effectiveness in handling the analysis of systems with embedded AI by considering a case study, presented in the next section.

4. Case study: Adaptive cruise control with traffic sign recognition

4.1. Case study background

Traffic sign recognition (TSR) is nowadays a common ADAS feature in many vehicles, typically providing information to the driver on the speed limit for the road ahead (Farag, 2019). It is the responsibility of the driver to use this information to adapt the velocity of travel of the vehicle to the speed limit to ensure compliance with legislation. TSR in general (i.e. including all traffic signs not just speed limit) has been extensively studied in conjunction with autonomous driving (e.g. Atif et al., 2022), being one of the benchmark problem for ML and AI researchers seeking to demonstrate the robustness of their algorithms (Aslansefat et al, 2021).

Some automotive manufacturers have attempted to build on the TSR capability with enhanced ADAS features based on TSR information input. This includes attempts to implement an adaptive cruise control (ACC) feature based on TSR, referred in the following as ACC-TSR. This enhanced ADAS feature offers the driver the comfort of delegating the responsibility for longitudinal vehicle acceleration control with the facility that the road legal speed limit is monitored and observed though automatic adjustment of vehicle velocity of travel. While this appears as an attractive feature for many drivers, even a cursory search on social media and search engines reveals a significant number of issues reported by both drivers and experts - advising that the feature should not be used even if available. Common complaints relate to (i) dangerous harsh decelerations due to an incorrect low speed limit detected; and (ii) speed restriction signs being missed, with the implication that the vehicle breaks the law by driving at higher speed (e.g. BusinessInsider, 2023). These are major failures of the function, with potential safety- and legal-critical effects. While this problem could be tackled from a safety perspective in relation to an existing vehicle and feature architecture, the position we take in this case study is that of a team conducting the initial analysis of the ACC-TSR as a proposed new feature, without any solution or architecture assumed. This will enable us to reason about function and function failure in an unconstrained manner, which would be a fair test ground for the proposed taxonomy.

4.2. ACC-TSR feature analysis

Figure 5 provides a block diagram representation of the ACC-TSR feature, based on a generic high level architecture of the TSR system including a vision sensor (e.g. a camera) and an embedded AI module. The function of the sensor is to convert the scenery image of the environment to a digitised tensor with the required resolution and colour matrix, and the required frame frequency. The task of the embedded AI module is to convert the input from the vision sensor to speed limit (SL) information for the propulsion control module / ACC feature, specifying the upcoming road speed limit (if any changes identified), and distance to the start of enforced speed restriction. At its core, the TSR AI module typically uses ML (convolutional neural networks) to detect, classify and then track through the time sequence of images (Atif et al, 2022). Based on this information, the ACC feature calculates the required

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

control input for the multi-vector vehicle propulsion systems (prime mover, transmission, brakes) to adjust the velocity profile and trajectory such that the desired speed limit is reached with the driveability and comfort expected by the vehicle occupants.

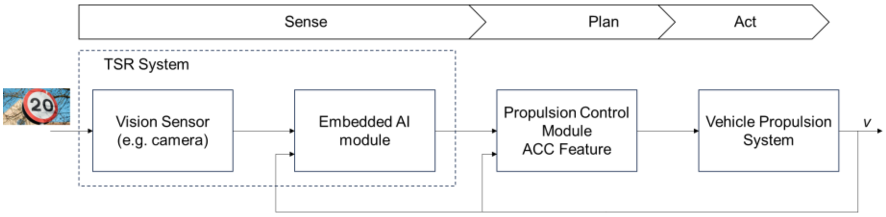

Sense
Plan
Act
TSR System
20
Vision Sensor (e.g. camera)
Embedded AI module
Propulsion Control Module ACC Feature
Vehicle Propulsion System
v

Figure 5. Block diagram of the ACC-TSR feature

From the point of view of our case study analysis, we focus on the function of the TSR system. Figure 6 provides a more detailed functional analysis, indicating the relevant input and output variables, and their measurable attributes. Considering the TSR function at the black box level (top part of Figure 6), the function failure analysis focusses on the output variables; the SL signal is a categorical variable, based on the set categories of traffic sign speed limits available in the jurisdiction; the distance to the start of enforcement (normally the location of the traffic sign) is a continuous variable (measured in meters); and finally, time (in seconds or miliseconds) is also a variable of interest, reflecting the constraint that the total TSR processing time needs to be shorter than the frequency of image acquisition. Figure 7 illustrates the function failure mode analysis for the TSR system, at the black box level, carried out based on the proposed taxonomy, Figure 3. While the failure modes are analysed independently in relation to the two variables, by aggregating the two we obtain a complete list of function failure modes. This analysis is correct given that the two variables, while they are output from the same function block can connected inside the algorithm, they are independent from the point of view of their failure modes, in the sense that the distance information can be correct even if an incorrect SL signal is identified. If the failure modes are dependent, the correlation between the two should be taken into account.

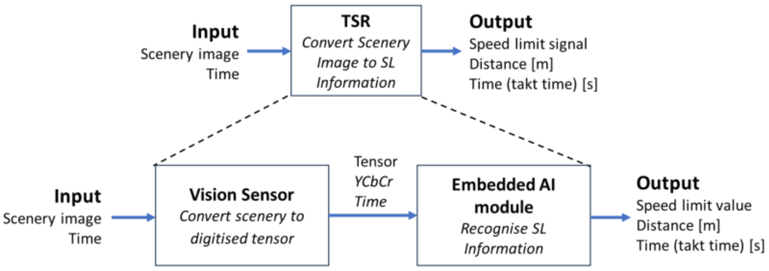
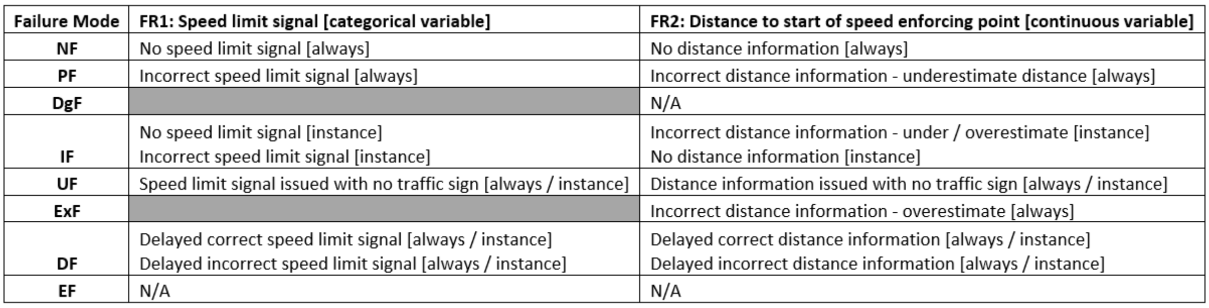


Figure 6. Function analysis of TSR system



Failure Mode	FR1: Speed limit signal [categorical variable]	FR2: Distance to start of speed enforcing point [continuous variable]
NF	No speed limit signal [always]	No distance information [always]
PF	Incorrect speed limit signal [always]	Incorrect distance information - underestimate distance [always]
DgF		N/A
IF	No speed limit signal [instance] Incorrect speed limit signal [instance]	Incorrect distance information - under / overestimate [instance] No distance information [instance]
UF	Speed limit signal issued with no traffic sign [always / instance]	Distance information issued with no traffic sign [always / instance]
ExF		Incorrect distance information - overestimate [always]
DF	Delayed correct speed limit signal [always / instance] Delayed incorrect speed limit signal [always / instance]	Delayed correct distance information [always / instance] Delayed incorrect distance information [always / instance]
EF	N/A	N/A

Figure 7. Function failure modes analysis for the TSR system

The categories of failure modes in the proposed taxonomy have worked well; e.g. we can have an exceeding function for the distance as a continuous variable, even though there is no exceeding function for the SL signal (e.g. a correct signal could be provided with overestimated distance). The processing time is not an output as such, and therefore it is an indicator variable for the delayed / early function category. Based on these failure modes, a feature concept level FMEA can be carried out. The next step

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

in the analysis is to identify the effects of each of the failure modes; failure modes with effects that have a potential safety- or legal-critical consequence (which is potentially the case with most of the failure modes identified in Figure 7), would be marked as critical characteristics, and traced throughout the design and development of the feature. Function failure modes of the decomposed system (lower level in Figure 6) will be identified as potential causes for top level, with verification methods defined as current controls. This should extend to validation of the impact of noise factors (Figure 1) to evaluate the robustness of the system through robust design verification and testing.

5. Discussion and conclusions

This paper makes a contribution by proposing an extension to the AIAG&VDA (2019) function failure modes taxonomy, to make it suitable for the analysis of smart systems and features, with embedded AI. In contrast with other domain specific failure modes taxonomies discussed in this paper, which are based on the phenomenology of the failure for both electro-mechanical and software systems, the functional basis of the AIAG&VDA (2019) is important because it supports the *logical analysis* of the system, thus underpinning early stages of design and development, within the systems engineering framework.

The theoretical basis for the proposed extension stems from the detailed consideration, from a mathematical perspective, of the type of variables involved the specification of the function. From a methodological point of view, this enables a more precise definition of taxonomical categories of function failure modes described in the AIAG&VDA (2019) standard, thus providing a richer prescriptive guidance to the FMEA practitioners to both facilitate the analysis and increase the consistency of the generated FMEAs.

In relation to other reference taxonomies, the main benefit of the proposed taxonomy is that it facilitates logical level analysis early in the feature development process, facilitating the collaboration of engineers belonging to different disciplinary backgrounds to work together with a common reference taxonomy and language, and maintains solution neutrality of the analysis.

Another important argument for the proposed taxonomy is that it supports the integration of the FMEA analysis with MBSE representations, and the automated function failure mode reasoning for intelligent MBSE-assisted FMEA generation (Korsunovs et al, 2022). Specifically, within a detailed MBSE interface model the specification will include data type associated with the port attributes. Our proposed taxonomy thus facilitates function failure mode reasoning integrated with the MBSE.

The case study considered for the preliminary validation of the proposed taxonomy has provided a good illustration of several practical features and complexities that need to be handled in the analysis, as well as demonstration of the applicability and value of the proposed taxonomy or the logical level analysis of systems with embedded AI components. This includes the fact that functions will commonly have multiple metrics associated with their achievement, illustrated in the case study with the TSR function output involving multiple variables, which have been analysed together. The case study also showed that the function outcome focus of the proposed function failure mode taxonomy works well with the logical analysis across physical and ML/AI embedded components.

The impact of using the FMEA for systems including embedded AI is that it contributes to establishing trust in the feature, as the behaviour linking function logic and failure modes includes the AI components, and thus robustness tests planned (based on the FMEA) to verify of the whole system will also cover the behaviour of all components.

References

AIAG and VDA (2019), *FMEA Handbook: design FMEA, process FMEA, supplemental FMEA for monitoring et system response*, Southfild, MI.

Aslansefat K, Kabir S, Abdullatif ARA et al (2021) Toward Improving Confidence in Autonomous Vehicle Software: A Study on Traffic Sign Recognition Systems. Computer. 54(8): 66-76. https://dx.doi.org/10.1109/MC.2021.3075054.

Atif M, Ceccarelli A, Zoppi T, Gharib, M and Bondavalli A. (2022), "Robust Traffic Sign Recognition Against Camera Failures", In IEEE Jrnl Intelligent Transp Syst, 3:709-722, https://dx.doi.org/10.1109/OJITS.2022.3213183.

Avizienis, A, Laprie, J C, Randell, B and Landwehr C. (2004), "Basic concepts and taxonomy of dependable and secure computing", In*: IEEE Trans Dep Secure Comp*, 1:1:11-33, https://dx.doi.org/10.1109/TDSC.2004.2.

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press

Barwise, G. (1977) Studies in Logic and the Foundations of Mathematics, Elsevier, Volume 90, 1977, Pages 5-46, https://doi.org/10.1016/S0049-237X(08)71097-8.

Blache, K M and Shrivastava, A B. (1994), "Defining failure of manufacturing machinery and equipment", *Proceedings Annual Reliability and Maintainability Symposium*; 69-75.

Blischke, W R, Murthy, D N P. (2000), *Reliability Modeling, Prediction, and Optimization*, John Wiley & Sons.

BusinessInsider (2023) Report available from https://www.businessinsider.com/volkswagen-cruise-control-accelerates-owners-car-without-warning-report-2023-1?r=US&IR=T, accessed 28 October 2023.

Clausing, D and Frey, D (2004), Failure Modes And Two Types Of Robustness, INCOSE International Symposium, 14: 489-501. https://doi.org/10.1002/j.2334-5837.2004.tb00511.x

Chu, T-L, Yuea, M, Postma, W. (2011), "A Summary of Taxonomies of Digital System Failure Modes", PSAM 2011 Helsinki. https://www.nrc.gov/docs/ML1206/ML120680552.pdf. Accessed: 15/09/2023.

Collins, J.A. (1993), *Failure of Materials in Mechanical Design: Analysis, Prediction, Prevention*, 2nd ed. Wiley Interscience; 1993.

Eifler, T., Campean, F., Husung, S., Schleich, B. (2023) 'Perspectives on Robust Design – An Overview of Challenges and Research Areas Across Industry Fields', in Proceedings of the International Conference on Engineering Design (ICED23), Bordeaux, France, 24-28 July 2023. https://dx.doi.org/10.1017/pds.2023.289

Farag W. (2019) Traffic signs classification by deep learning for advanced driving assistance systems," Intell. Decis. Technol., vol. 13, no. 3, pp. 305-314, https://dx.doi.org/10.3233/IDT-180064.

Hirtz J, Stone R B, McAdams, D A et al. (2002), "A functional basis for engineering design: Reconciling and evolving previous efforts", *Res Eng Design* 13, 65–82. https://doi.org/10.1007/s00163-001-0008-3

Hu T, Zhang H, Zhou, J. (2023), "Machine learning-based model for recognizing the failure modes of FRP-strengthened RC beams in flexure", *Case Studies in Construction Materials*. Volume 18. doi.org/10.1016/j.cscm.2023.e02076.

BS-ISO 21448 (2022) Road Vehicles: Safety of the Intended Functionality, ISBN 978 0 539 23095 6.

Koopman P and Wagner, M Challenges in Autonomous Vehicle Testing and Validation. SAE Int. J. Trans. Safety 4(1):15-24, 2016 https://doi.org/10.4271/2016-01-0128

Korsunovs A, Doikin A, Campean F, et al. Towards a Model-Based Systems Engineering Approach for Robotic Manufacturing Process Modelling with Automatic FMEA Generation. Proceedings of the Design Society. 2022;2:1905-1914. https://dx.doi.org/10.1017/pds.2022.193

Kumar, R S S, O`Brien, D, Albert, K, Viljoen, S and Snover, J. (2019), *Failure Modes in Machine Learning.* [online] Microsoft. https://arxiv.org/abs/1911.11034 (accessed: 13/10/2023)

McNelles, P, Zeng, Z C, Renganathan, G, Chirila, M, Lixuan Lu, L. (2017), "Failure mode taxonomy for assessing the reliability of Field Programmable Gate Array based Instrumentation and Control systems", *Annals of Nuclear Energy*, Volume 108, Pages 198-228. https://doi.org/10.1016/j.anucene.2017.04.033.

O'Halloran, B M, Stone, R B and Tumer, I Y. (2012), "A failure modes and mechanisms naming taxonomy", *Proceedings Annual Reliability and Maintainability Symposium*, pp. 1-6, 2012.

Scott, P, and Yampolskiy, R. (2020), "Classification Schemas for Artificial Intelligence Failures", *Delphi - Interdisciplinary Review of Emerging Technologies.* Volume 2, Issue 4. pp. 186-199 https://doi.org/10.21552/delphi/2019/4/8

Shafique, M et al. (2020), "Robust Machine Learning Systems: Challenges, Current Trends, Perspectives, and the Road Ahead". IEEE Design and Test, 37(2), pp. 30-57, https://doi.org/10.1109/MDAT.2020.2971217.

Stone, R B, and Wood, K L. (2000), "Development of a Functional Basis for Design", *ASME. J. Mech. Des.* 122(4): 359–370. https://doi.org/10.1115/1.1289637

Taguchi, G. (1986), Introduction to quality engineering – designing quality into products and processes, Asian Productivity Association. ISBN: 978-9283310846.

Thieme C A, Mosleh A, Utne I B, Hegde J. (2020), "Incorporating software failure in risk analysis—part 1: software functional failure mode classification", *Reliability Engineering & System Safety*, Volume 197. https://doi.org/10.1016/j.ress.2020.106803.

Tumer, I Y, Stone, R B and Bell, D G. (2003), "Requirements for a Failure Mode Taxonomy for Use in Conceptual Design", *In International Conference on Engineering Design*, Stockholm Sweden.

Vermaas, PE. (2009). The flexible meaning of function in engineering. In L. Leifer, P. Skogstad, M. Norell Bergendahl, & M. Grimheden (Eds.), Proc ICED 09, pp. 2.113-2.124. The Design Society.

Yildirim, U., Campean, F., and Williams, H. (2017) Function modeling using the system state flow diagram. AI-EDAM, 31:4:413-435. https://doi.org/10.1017/S0890060417000294

Yildirim U, Campean F, Korsunovs A, Doikin A. Flow heuristics for functional modelling in model-based systems engineering. Proceedings of the Design Society. 2023;3:1895-1904. https://dx.doi.org/10.1017/pds.2023.190

https://doi.org/10.1017/pds.2024.197 Published online by Cambridge University Press