Towards Adaptive Anomaly Detection and Root Cause Analysis by Automated Extraction of Knowledge from Risk Analyses

Bram Steenwinckel, Pieter Heyvaert, Dieter De Paepe, Olivier Janssens, Sander Vanden Hautte, Anastasia Dimou, Filip De Turck, Sofie Van Hoecke, and Femke Ongenae

Ghent University - imec, IDLab, Ghent, Belgium
bram.steenwinckel@ugent.be

Abstract. Sensors inside internet-connected devices analyse the environment and monitor possible unwanted behaviour. Current risk analysis tools, such as Fault Tree Analysis (FTA) and Failure Mode and Effect Analysis (FMEA), provide prior information on these malfunctions. Many people are involved in this risk analyses process, resulting in disambiguations and incompleteness. Ontologies could resolve this issue by providing a uniform structure for the failures and their causes. However, domain experts are not always ontology experts, resulting in a lot of human effort to keep the ontologies up to date. In this paper, automated mappings from the FMEA data to a domain-specific ontology and the generation of rules from a constructed FTA were researched to annotate and reason on sensor observations semantically. The approach is demonstrated with a use case to investigate the possible failures and causes of reduced passenger comfort levels inside a train.

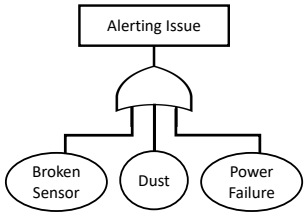
Keywords: Anomaly detection · Root Cause Analysis · Risk Analysis · Semantics · Ontology development · Sensor data · IoT

1 Introduction

Sensor monitoring systems are transforming industry, with game-changing applications in, e.g., transportation [5] and healthcare [17]. These systems can yield valuable insights into a company's physical assets and the interaction of these assets with their environment. However, sensors have limited added value without data analysis [19]. More and more, new methodologies are defined to specify the correct functioning of the system based on these sensor observations. Common methodologies for observing unwanted system behavior with this data are Anomaly Detection (AD) and Root Cause Analysis (RCA). AD is the identification process of events or observations, which do not adhere to the expected pattern or other items inside a dataset [17]. RCA guides the problem solver to deduce and understand the real causes of the anomalies [16]. Interest in AD & RCA will continue to grow as more relevant data is generated and tools become widely accessible that can handle data from diverse operating environments.

Component	Function	Failure Mode	Failure Effect	Failure Cause	Control Method
Fire detector	Detects the fire	Unwanted evacuation	Falsely generated alarm	Broken sensor	None
Smoke sensor	Sense the smoke level in the environment	Values too high	Broken sensor	Dust accumulation	Clean system

(a)



(b)

Fig. 1: Example of FMEA (a) and FTA (b)

However, domain-specific knowledge needs to be leveraged to clearly define the unwanted behavior and its causes inside these tools as sensor, or system behavior in general, varies wildly between application domains. This knowledge is often provided by domain experts by using risk analysis, which define all the possible failures and their (observable) effects on the system. Failure Mode and Effects Analysis (FMEA) [2] and Fault Tree Analysis (FTA) [8] provide templates to easily provide such analyses. As shown in Figure 1 (a), FMEA captures, on multiple levels of the system, the potential failures that can occur to the components and their underlying causes and effects. FTA analyses the undesired states of a system using Boolean logic. This combination of low-level events, leading to system failures, can be visualised using a tree, as exemplified in Figure 1 (b).

Constructing these FMEA and FTA documents is a time-consuming process when applied thoroughly. A large number of experts are involved, who each have expertise on other parts of the system and interpret different parts of the risk analysis differently. Ambiguities, inconsistencies and duplicates are, therefore, quite common. This reduces the advantages of these risk analysis and makes it difficult for non-experts to interpret these document. Sharing, however, a common understanding about the structure of the system and contextual knowledge amongst the experts could help in separating the domain knowledge from the operational knowledge about the (mal)functioning of the system. Ontologies and accompanying inference rules have proven their worth in providing a common knowledge representation about a domain [20]. However, most system experts are not familiar with ontology design, which makes these approaches difficult to implement. Semantic Web experts are required to constantly maintain and update these ontologies and rules with the domain-specific knowledge.

However, enabling domain experts to generate ontologies and rules based on the domain knowledge captured in the FMEA and FTA documents will lower the barrier to use them in existing data analysis methodologies. The generated ontologies and rules themselves can be used to annotate and reason upon incoming sensor observations, to eventually provide some basic tools for preliminary semantic-based AD and RCA.

In this paper, we propose an approach to automatically generate the required ontologies and inference rules from the aforementioned risk analysis outcomes. The entries from the FMEA table will be used to create a domain specific ontology. The fault trees will be used to derive rules to clarify if a certain sensor observation leads to a failure or not. The automation of this approach removes the need for the involvement of ontology and rule experts in the risk analysis process while enabling maintainable, semantic-based AD and RCA. As such, the domain experts can focus on their primary task, i.e., applying their domain knowledge to accurately capture the unwanted behaviour of a system and its causes.

The remainder of the paper is structured as follows. Section 2 situates our approach with respect to the related work. The designed approach is discussed in detail in Section 3, while Section 4 details the application of the approach on a real-life use case, i.e., investigating the possible failures and causes of reduced passenger comfort levels inside a train. Section 5 highlights the most important accomplishments and discusses the directions for future work.

2 Related work

As previously mentioned, ontology-based risk analysis methods have been proposed. Dittmann et al. [7] describe a process to capture the results of a FMEA in an ontology, instead of in a document, and highlighted the (dis)advantages. Rehman et al. [15] and Zhou et al. [21] designed high-level ontologies to model the main concepts of a FMEA and their relationships. The first applied it to model the results of a FMEA in the automotive domain. The second used it to capture the FMEA of wind turbines and developed a reasoning framework to perform intelligent fault diagnosis using the designed ontology capturing the domain-specific concepts. Both papers showed how an ontology can be used to easily trace the relationships between failures and their corresponding causes, making it easier to interpret the risk analysis. Ontologies to automatically link the observations made within a particular system to anomalies or faults that can occur, have also been proposed [13]. Although high-level concepts have been defined to model irregularities and link them to system components and effects, an ontology expert is required to model all the domain-specific anomalies that can occur and how they link to the sensor observations. None of the proposed ontologies are publicly available, hindering re-use. Moreover, all the approaches propose to replace the existing methodologies with a process in which the results of a FMEA are directly captured in an ontology. This requires extensive knowledge about ontology design from the system experts as the current FMEA are performed using standard spreadsheet tools.

FTA has the advantage to be a more rigorous approach due to the step-by-step reasoning. Contrary to FMEA, FTA is a graphical method and already identifies the interrelations between concepts. As a result, FTA is more interpretable than FMEA as the latter forces the analyst to decompose the system [14]. In an effort to automate the construction of the FTA trees, Venceslau et al. [18] de-

fined an ontology to model the system components and failures and constructed a technique to automatically generate the FTA tree from the constructed ontology. The use of the ontology solves the issue of inconsistencies and ambiguities between FTA trees due to the lack of a common knowledge representation and the automatic generation of the tree ensures human understanding of the result. However, it again requires ontology design knowledge from the system experts.

While an ontology can capture the various concepts occurring within a domain and their intricate relationships, additional expressivity is required to derive that a fault has occurred out of the combination of various system observations. Rule languages, such as RuleML [3] and SWRL [11] can define inference rules, which are used inside a semantic reasoner to derive logical consequences. Recently, techniques have been designed to extract SWRL rules from text using NLP [9] or mine Semantic Web Association Rules from RDF data (SWARM) [1]. However, there are, to our knowledge, currently no techniques which allow the automated extraction of rules from risk analyses.

It can be concluded that currently no approaches exist that allow system experts to use their traditional risk analysis methodologies, i.e. FMEA tables and FTA trees, while still providing methods to automatically extract unambiguous and consistent ontologies and rules from them in a user-friendly manner.

3 User-friendly approach to extract knowledge from risk analyses

Both AD and RCA analysing tools require the semantic annotation of the sensor observations to operate. Defining rules which detect the failures based on the incoming sensor observations, in combination with a domain-specific ontology, enables the detection of irregularities and the derivation of their cause. The generated ontologies and rules are the building blocks to determine unwanted behavior. They can be incorporated in a knowledge-based monitoring system to continuously identify anomalies and their causes. For example, they can be integrated in MASSIF, a data-driven platform for the semantic annotation of and reasoning on internet-connected data, allowing complex decision-making processing [4]. When new sensor observations are generated by the system, MASSIF semantically annotates them using the domain-specific ontologies. MASSIF then uses a semantic reasoner to process the generated SWRL rules and links defined in the ontologies to determine whether failures are occurring and what their possible causes are. As such, the sensed data can be combined on the fly with background knowledge, resulting in enhanced and adaptive context-aware AD and RCA applications.

To realize these rules and ontologies in a user-friendly manner, we propose an approach to automatically extract them from FMEA and FTA documents and trees, as visualized in Figure 2.

A first part of this automation approach uses declarative mapping rules to map FMEA documents on domain-specific ontologies describing the components and their associated anomalies, causes and system effects. Second, predefined

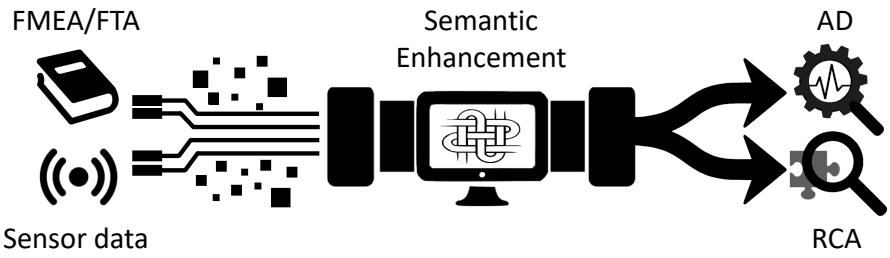


Fig. 2: Overview of the approach to combine knowledge with sensor data for AD and RCA

translation scripts are used to extract the inferences rules from the FTA trees. It is important to state that both the mapping rules and scripts are generic and can be re-used for every new FMEA table and FTA tree. Only when the structure or template of the documents change, additional mappings or changes to the scripts will have to be provided. Different methodologies to easily provide these changes with a minimum amount of human effort or knowledge about ontologies and inference rules are also provide to perform these changes. Both are part of the semantic enhancement visualised in Figure 2. A more detailed overview of the combination of thses two approaches is given in Figure **??** and will be discussed in the following subsections.

To ease the explanation of the different steps, a running example based on a smart fire detector will be used in this section. The end goal is to semantically map the observations from this fire detector to possible failures and give further tools the possibility to derive possible causes. A part of the FMEA is visualized in Figure 1 (a) and it describes the possible failures of the available smoke sensor. A false alarm (failure effect) could be generated when dust accumulates in the device (failure cause), as it hinders the sensor from observing the environment correctly.

3.1 Ontology mapping approach

The data inside the FMEA tables can be used to define a domain-specific ontology, describing the links between several components of the system and their possible failures and related causes. The automated process transforming these tables to an ontology is visualised in the top part of Figure 3. The FMEA tables, represetned as CSV files, will be used for the mapper as input. The mapper itself will use rules to convert the data inside the table to ontology concepts with predefined links between them. Both the description of the rules and the links between the concepts in the FMEA tables will be explained in this section.

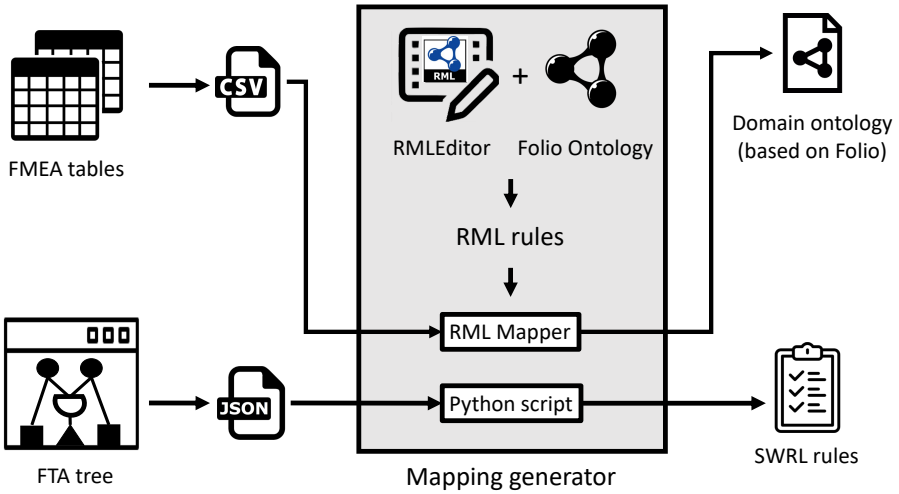


Fig. 3: Overview of the approach to extract knowledge from risk analyses

Folio ontology

Before we can specificy the methodology to extract knowledge from FMEA & FTA documents, a definition of the common concepts within the system risk analysis domain should be given. Therefore, we developed an ontology, called Folio[1], which captures all application-independent concepts that occur within FMEA, FTA and anomaly detection methods. It is based on the aforementioned ontologies constructed by Zhou, et al. [21] and Pardo, et al.[13]. There are several concepts inside the FMEA template similar in the anomaly domain. The effects and causes of an anomaly can be related to the failure causes and effects, while both have detection methods and a degree of severity. Combining the concepts of both of them enables the derivation of the possible anomaly causes with the available knowledge inside the FMEA worksheets.

The `Anomaly` class defined inside Figure 4 and the directly connected classes include all the possible anomaly information. These classes were adapted to ensure applicability in a context of detecting anomalies for internet-connected devices and can determine the irregularities in streaming data. The Semantic Sensor Network (SSN) ontology[2] describes sensors and their observations for a diverse range of devices and is included in this upper ontology. The SSN architecture includes a lightweight, but self-contained core ontology called SOSA (Sensor, Observation, Sample, and Actuator) for its elementary classes and properties. With their different scope and different degrees of axiomatization, SSN and SOSA can support a wide range of applications and use cases. By using SSN & SOSA, the Folio ontology can describe the sensor's observations that are the

[1] Folio ontology: https://github.com/IBCNServices/Folio-Ontology/blob/master/Folio.owl
[2] SSN ontology: https://www.w3.org/TR/vocab-ssn/

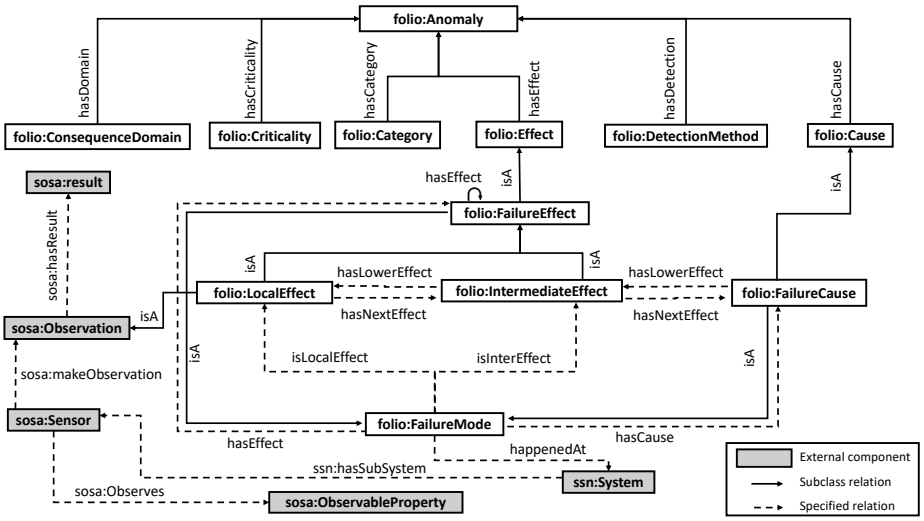


Fig. 4: The Folio ontology.

basis for analyzing the system behavior. Relationships were defined in Folio to correlate the SSN classes with possible failures and effects.

The FMEA concepts from Zhou, et al. were extended and related to the anomaly class inside the Folio ontology. The `FailureEffect` and `FailureCause` classes are subclasses of the anomaly `Effect` and `Cause` classes.

Relations between causes and effects are needed to describe the corresponding connections between multiple components. A `FailureCause` defines a concept with no further `hasNextEffect` relations. An `IntermediateEffect` concept will describe the influence of an intermediate component that is affected by, but not causing, the detected problem. The whole detection flow can have multiple `Intermediate Effect`s. The `LocalEffect` refers to the first detected effect onto the system. A `LocalEffect` will mostly be related to a faulty sensor observation itself, describing the current state of the device or system component. For the fire detector example, the accumulation of the dust will be defined as a `FailureCause`. The malfunctioning of the sensors are `IntermediateEffect`s and they could even have multiple causes. A `LocalEffect` could be a value too high observation, indicating something is wrong with the system.

Domain knowledge transformation

The previously described upper ontology Folio can now be used to form a more domain specific ontology, using the data from the FMEA tables. As such, anomaly knowledge can be extracted from the FMEA itself, and the causes of these anomalies can be derived by following the created semantic links. The mapping approach from these table entries to Folio concepts is visualised in the top part of Figure 3 and consists of three different steps. First, the tables

itself must be transformed into a computer-readable format in order to process the information. Second, rules must be generated to map the column and header information from the FMEA table to existing concepts inside the upper ontology. At last a mapping procedure is require to transform all the rows inside the formatted table to domain specific concepts, eventually creating the domain specific ontology.

More specfic, FMEA are usually constructed using a spreadsheet program, transformable to CSV document used for further analysis. The different possible elements of each record in the FMEA are fixed and defined by the column headers of the provided FMEA templates. More concrete, every FMEA table defines minimal the failures, their effects and their causes of a system but variatons are possible and provide addtional information. Consequently, to enable the mapping of the FMEA on the Folio ontology, these column headers should be mapped on ontological concepts. To realize this, a mapping language was used, which enables the declarative definition of how to generate RDF from existing data sources through a set of rules. This approach is here preferred because mapping languages provide a reusable solution, while custom software and mapping scripts are limited to a specific use case or implementation [6]. Another advantage is the adaptive character of the mapping rules: when making changes in the representation of the data (for instance, the risk analysts switch to a more advanced Failure Mode, Effect and Criticality Analysis method), updating the mapping rules will suffice to incorporate this extra information in the domain-ontology. Our approach uses the RML mapping language [6].

We defined the RML rules following the guidelines of the Folio ontology via the RMLEditor [10], which offers a graphical user interface to aid users in defining rules. The high levels steps we followed are as follows: (i) a sample of an FMEA table was loaded in the RMLEditor, (ii) the rules were created by an ontology expert, (iii) the corresponding RDF triples were generated, (iv) if the triples are not as expected the rules are updated, and (v) the rules are exported[3]. Afterwards, the RMLMapper[4], a tool to execute RML rules, is used to generate the ontologies for all FMEA tables. The mappings ensure that for each cell in the FMEA table, a new class is created in the ontology, which is a subclass of the class on which the column is mapped according to the rules. For example, if we consider the 5th cell on the second row of Figure 1 (a), the RMLMapper will create a new concept `DustAccumulation` in the ontology, which has as superclass the `FailureCause` class.

As such, these mappings can be re-used to translate any FMEA table that is created according to the standard FMEA structure. Changing the information inside or adding new information regarding risks and failures to the FMEA documents do not affect the generation of the domain ontologies at all. Only if the template changes, e.g. If a new column is added, a new rule must be defined, which can easily be created by mapping this column to the Folio concept by using the RMLEditor. Due to the frequently used FMEA templates, this will

[3] RML rules: https://github.com/IBCNServices/Folio-Ontology/blob/master/mapping.rml.ttl
[4] RMLMapper: https://github.com/RMLio/RML-Mapper

not happen often. In our fire detector example this means that updating the FMEA documents, by adding adding an additional cause for the unwanted evacuation (for instance a broken test button), does not affect the domain ontology generation process. The outcome of our mapping approach is a domain ontology in OWL,. In our fire detector example this ontology will relate the failures of the temperature and smoke sensors to the general system effects, using the relationships of the upper Folio ontology.

3.2 Rule generation approach

While the previous part relates the domain specific information of failures and causes together, faulty sensor observations must be mapped to the correct failures before a semantic-based monitoring system, such as the one describe in Figure 2, can be operational. Rules are helpful in specifying the irregularities in the data through defining patterns or operational ranges. For example, a smoke detector can be defined as faulty, when it measures impossibly high values due to dust accumulation. This requires experts to adequately define the normal value ranges for these sensors. The Folio ontology and the previously explained FMEA mapping approach already allow to define the observations and their resulting values made by sensors. This section describes how rules can be extracted from the FTA trees to link these observations to possible faults that occur, by using the process visualized at the bottom of Figure 3. Similar to the FMEA mapping approach, three different steps can be defined. First, the trees themselves must be transformed into a computer-readable format in order to process the rules. Second, tree-agnostic knowledge must mapped to specific rule concepts to perform the translation between the tree representations and the rule definitions. At last a mapping procedure is require to transform all the information inside the formatted trees to domain specific rules.

Decision Fault Trees

While original fault trees describe the relationship between the components of the system, they usually do not allow to differentiate the observations from their possible failures. In the case of the fire detector, the link between the sensor observations and all the possible failures shows the interaction of the different system components, but does not capture the difference between the accumulation of dust or, for example, a broken sensor. A fault tree created from the FTA restricts the analysis to the relations between the components inside the system solely. Therefore, a combination of a decision tree, which is capable of modelling the decision from observations to failure with the possible consequences, together with the general FTA tree, is used here. This so-called decision fault tree (DFT) provides tests on the intermediate edges of the tree, visualising the basic rules for further analysis. Figure **??** gives an example of such decision tree, related to our fire detector example. When a certain smoke observation have a value greater than 50 ppm, the observation can be classified as a *ValuesTooHigh* failure.

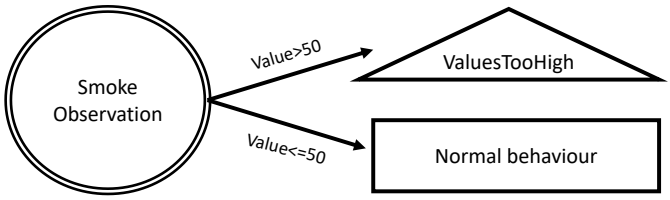


Fig. 5: example of a Decision Fault Tree.

A user interface was designed to built such DFTs, as shown in Figure 8. In this editor, descriptions of the observation and failure nodes can be given. These different node concepts should align with the concepts defined in the FMEA. Tests describing the relations between these observations and failures can be added or adapted. Several representations are possible for such DFTs. The user interface outputs JSON file to describe the nodes and the rule-specific edges.

Domain-specific decisions

The constructed DFTs can now be used as input to define the domain specific decisions, relating the observations to previously defined failures in the FMEA procedure. To translate the decision inside the tree to Rules, a rule generator script was designed in Python[5] to transform the JSON representation into SWRL rules. In a first step, the decision and nodes are gathered from the DFT inside a JSON format. Second, RDF syntax rule definitions are used as mock-ups for the SWRL rules. These definitions specify all the basic boolean operations, as well the logical operators ($<, \leq, ==, \geq, >$). The JSON DFTs are then provided as input to these definitions, resulting finally in specific SWRL rules. These SWRL rules can be attached to the FMEA RDF document or can be saved separately. To give an example, the generated SWRL rule specifying a `ValuesTooHigh` failure in the fire detector example of Figure 5 looks as follows:

```
SmokeObservation(?o) ^  hasResult(?o, ?result) ^
hasValue(?result, ?value) ^ swrlb:greaterThan(?value, 50)
 -> ValuesTooHigh(?o)
```

This rule describes the inference of a `ValuesTooHigh` failure when an observation is a `SmokeObservation` and the result of this observation is greater than 50.

Similar to the FMEA mapping procedure, the python mock-ups are defined once and are able to operate on all generated DFTs. Changing the information inside or adding new information regarding risks and failures to the DFTS do not affect the rule generation process. Only if the DFT components changes, e.g. If a new functional operators are added, a new mock-up must be defined.

[5] Script: https://github.com/IBCNServices/Folio-Ontology/blob/master/swrl_builder.py

4 Use case: Measuring Train Passenger Comfort

The growing requirements for quality of service put new challenges on the operation and development of trains and railway tracks. Therefore, research on the passenger comfort levels has reached high interest in the last decade [12]. As shown in Figure 6, train bogies are now equipped with accelerometers and gyroscope sensors, able to detect the shocks and damping effect of the train on the tracks. Multiple sensor observations of different train cars can be combined on a server to indicate the passenger comfort inside the train. Maintenance alerts are given to both the train or track staff to resolve the issues.

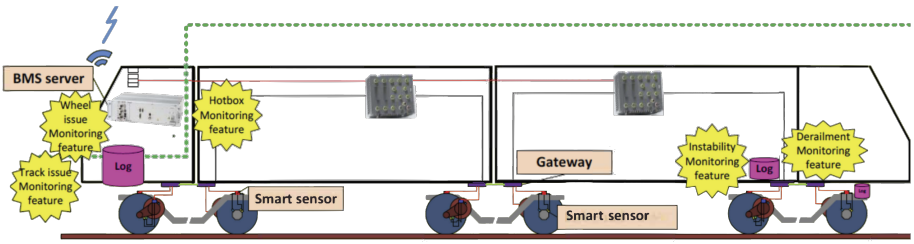


Fig. 6: Schematic overview of a train.

Component	Function	Failure Mode	Failure Effect	Failure Cause	Control Method	Containment Action
Passenger Comfort Unit	Detects the level of Comfort	False warning	Indicating impossible comfort level	Broken sensor	None	None
Accelerometer Sensor	Measures changes in gravitational acceleration	Values too high	Broken sensor	Degradation of the sensor	None	Replace Sensor
			Malfunctioning sensor	Rapid temperature changes	None	Calibrate sensor
Gyroscope Sensor	Measure the smoke level	Values too high	Broken sensor	Degradation of the sensor	None	Replace Sensor
			Malfunctioning sensor	Rapid temperature changes	None	Calibrate sensor

Fig. 7: Train passenger comfort FMEA example

The company installing these train monitoring units, i.e. Televic Rail, performed risk analyses. The resulting FMEA table, visualised in Figure 7, shows the possible failures of a disallowed comfort level that result in the effect of multiple falsely generated warnings for the train driver. Two possibilities are a broken or malfunctioning sensor. The FMEA table shows that the cause of the latter is varying outdoor temperatures while degradation causes the broken sensor. Replacing or recalibrating it could solve these issues.

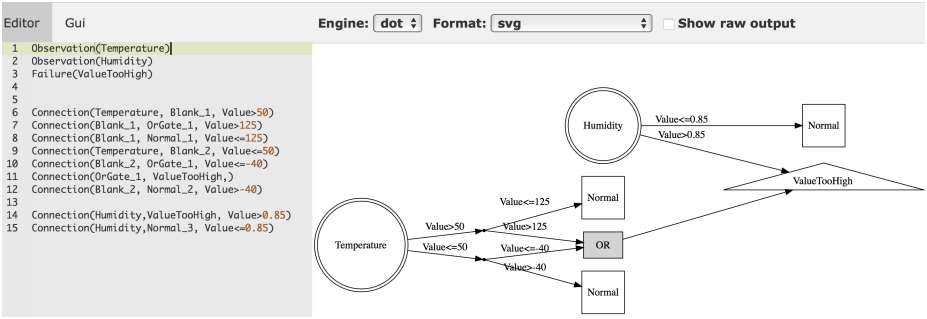


Fig. 8: Train sensors DFT example

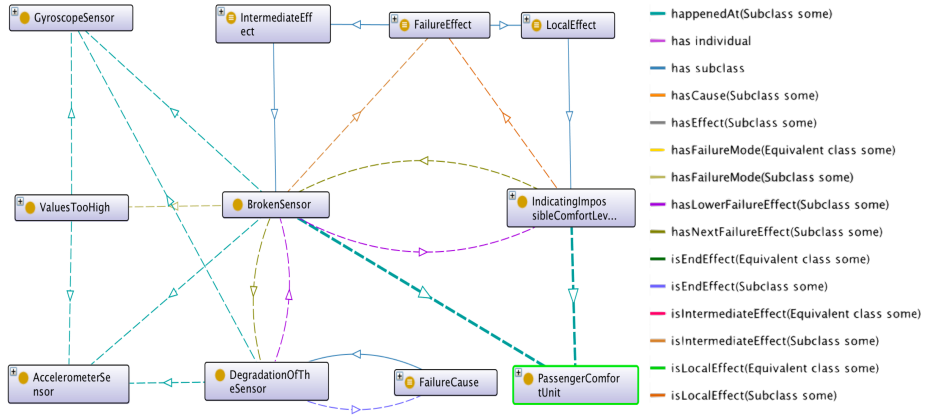


Fig. 9: Ontograf visualisation of the passenger comfort FMEA ontology

A DFT was also modeled by Televic in the designed web interface, as shown in Figure 8. This tree describes the relationship between the temperature observations of the accelerometer unit and the humidity observations of the gyroscope sensor unit with their possible failure modes. A `ValuesTooHigh` failure can occur when the temperature of the accelerometer has either a value higher than 125 degrees Celsius. The value range are this high because the accelerometer module operates on the wheel axles themselves and they are influenced by a lot of friction. A second failure can be derived when the temperature value is lower or equal than minus 40 degrees Celsius. At last, the same `ValuesTooHigh` failure can be used to indicate the humidity of the gyroscope has a value higher than 85%. All other observations are classified as normal in this simple use case.

The corresponding JSON file of the DFT and the CSV file of the table can be given as input to the mapping engine. The RML rules are here already predefined (same rules as defined in the fire detector example) and map the specific input fields to an RDF train-specific ontology. A schematic overview of the generated

ontology is given in Figure 9 and visualises the major concepts of Figure 7. The inferred rules of the DFT, given in Figure 8, are visualised in Listing 1.1. This listing describes three SWRL rules corresponding with the paths from the sensor observations to the single failure mode. When an accelerometer temperature observation reaches the reasoning engine, and its value is greater than to 125 degrees Celsius, the observation will be classified as a `ValuesTooHigh` failure, and further actions can be taken.

```
HumidityObservation(?o) ^
hasResult(?o, ?result) ^
swrlb:greaterThan(?Value, 0.85) ^
hasValue(?result, ?Value)
-> ValuesTooHigh(?o)

hasResult(?o, ?result) ^
swrlb:greaterThan(?Value, 50) ^
hasValue(?result, ?Value) ^
TemperatureObservation(?o) ^
swrlb:greaterThan(?Value, 125)
-> ValuesTooHigh(?o)

swrlb:lessThanOrEqual(?Value, -40) ^
hasResult(?o, ?result) ^
hasValue(?result, ?Value) ^
TemperatureObservation(?o) ^
swrlb:lessThanOrEqual(?Value, 50)
-> ValuesTooHigh(?o)
```

Listing 1.1: SWRL rules derived from the DFT in Figure 8

5 Conclusion and Future work

In this paper, research is proposed to enable the automatic knowledge extraction out of risk analyses into domain-specific ontologies and accompanying inference rules. Mappings were provided to incorporate the knowledge inside FMEA documents into domain-specific ontology. An upper ontology was used to define relate the main concepts, making the methods operational for several, different applications. Inference rules were extracted from DFTs, which were able to express the link between sensor observations and defined failures. Both methods allow system experts to use the risk analysis methodologies and tools they are used too to build a domain specific ontology with accompayning rules, without the additional need for ontology experts. These ontologies and accompanying rules ensures that a common vocabulary and consistency check is maintained and can be used to enable on the fly detection of anomalies and their causes through semantic reasoning. It enables the system experts to focus on the risk analysis task, instead of on a knowledge modelling task for which they do not have the adequate ontology design expertise. Future research can now use the designed ontologies, together with accompanying rules to derive or reason on the possible causes inferred from the failures. Additionally, the DFT editor itself can extended with consistency checks to ensure improved rule generation.

Acknowledgment: This research is part of the imec ICON project Dyversify, co-funded by imec, VLAIO, Renson Ventilation NV, Televic Rail & Cumul.io.

References

1. Barati, M., et al.: Swarm: approach for mining association rules from semantic web data. In: Conference on Artificial Intelligence. pp. 30–43. Springer (2016)
2. Ben-Daya, M.: Failure mode and effect analysis. In: Handbook of maintenance management and engineering, pp. 75–90. Springer (2009)
3. Boley, H., et al.: Design rationale of ruleml: A markup language for semantic web rules. In: Proceedings on Semantic Web Working. pp. 381–401. CEUR-WS (2001)
4. Bonte, P., et al.: The massif platform: a modular and semantic platform for the development of flexible iot services. Knowledge and Information Systems (2017)
5. Camossi, E., et al.: Semantic-based Anomalous Pattern Discovery in Moving Object Trajectories. CoRR **abs/1305.1** (2013)
6. Dimou, A., et al.: Rml: A generic language for integrated rdf mappings of heterogeneous data. In: LDOW (2014)
7. Dittmann, L., et al.: Performing fmea using ontologies. In: 18th International Workshop on Qualitative Reasoning. Evanston USA. pp. 209–216 (2004)
8. Ericson, C.A.: Fault tree analysis. Hazard analysis techniques for system safety pp. 183–221 (2005)
9. Hassanpour, S., et al.: Framework for the automatic extraction of rules from online text. In: Workshop on Rules and Rule Markup Languages. Springer (2011)
10. Heyvaert, P., et al.: Rmleditor: a graph-based mapping editor for linked data mappings. In: International Semantic Web Conference. pp. 709–723. Springer (2016)
11. Horrocks, I., et al.: Swrl: A semantic web rule language combining owl and ruleml. W3C Member submission **21**, 79 (2004)
12. Karimpanal, T.G., Gadhia, H.M., Sukumar, R., Cabibihan, J.: Sensing discomfort of standing passengers in public rail transportation systems using a smart phone. CoRR (2017)
13. Pardo, E., et al.: A framework for anomaly diagnosis in smart homes based on ontology. Procedia Computer Science **83** (2016)
14. Peeters, J., et al.: Improving failure analysis efficiency by combining fta and fmea in a recursive manner. Reliability engineering & system safety **172**, 36–44 (2018)
15. Rehman, Z., Kifor, C.V.: An ontology to support semantic management of fmea knowledge. International Journal of Computers, Communications & Control (2016)
16. Solé, M., et al.: Survey on Models and Techniques for Root-Cause Analysis. Clinical Orthopaedics and Related Research (CoRR) (2017)
17. Souiden, I., et al.: A survey on outlier detection in the context of stream mining. In: Advances in Intelligent Systems and Computing (2017)
18. Venceslau, A., et al.: Ontology for computer-aided fault tree synthesis. In: Emerging Technology and Factory Automation (ETFA), 2014 IEEE. pp. 1–4. IEEE (2014)
19. YE: Big data: Changing the way businesses compete and operate (2014)
20. Ye, J., et al.: Semantic web technologies in pervasive computing. Pervasive and Mobile Computing pp. 1–25 (2015)
21. Zhou, A., et al.: A research on intelligent fault diagnosis of wind turbines based on ontology and fmeca. Advanced Engineering Informatics **29**(1), 115–125 (2015)