Uncertainty based Fault Type Identification for Fault Knowledge Base Generation in System of Systems

Seungchyul Shin, Sangwon Hyun, Yong-jun Shin, Jiyoung Song, Doo-Hwan Bae
School of Computing
Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Republic of Korea
{scshin, swhyun, yjshin, jysong, bae}@se.kaist.ac.kr

2021 16th International Conference of System of Systems Engineering (SoSE) | 978-1-6654-4454-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/SOSE52739.2021.9497475

***Abstract*—A System of Systems (SoS) is a large-scale complex system composed of Constituent Systems (CSs) that interact organically to achieve SoS level goals that cannot be achieved by individual CSs. Various uncertainties may arise in SoS in addition to the uncertainties of each CS. Uncertainty in SoS may cause it to fail in various situations. To debug these failures efficiently, a high-quality fault knowledge base is needed for the SoS. However, existing studies assume that (1) there are sufficient data when creating an initial fault knowledge base and that (2) various uncertainties related to the characteristics of the SoS cannot be considered. Therefore, this study proposes an approach to create a fault knowledge base in SoS that considers uncertainty and then determines the quality of fault data so as to build a high-quality fault knowledge base for SoS. The proposed approach categorizes faults based on the nature of uncertainty and the manifestation location, making it possible to find and add fault types that are currently not considered. Through a case study of an Advanced Driver Assistance System (ADAS), we followed the process domain experts use to create a fault knowledge base. Extracting and classifying knowledge from more than 9,000 fault data entries revealed that only 7 out of 10 fault types were observed. The proposed approach could successfully find unknown faults that were not in the generated fault knowledge base.**

***Index Terms*—System of systems, uncertainty based classification, fault knowledge base**

I. Introduction

A System of Systems (SoS) is a large complex system composed of constituent systems (CSs) that interact with each other to achieve SoS-level goals that cannot be achieved by individual CSs [1]. Each CS have operational and managerial independence, and their complex interactions make emerging behaviors of SoS. A lack of knowledge about the behavior of an SoS often manifests as unexpected emergent behavior of the SoS or unexpected behavior of the CSs. Therefore, various uncertainties can be manifested in the SoS, resulting in various faults that are difficult to be expected in the individual CSs. Faults are unpermitted deviations of at least one system characteristic from the standard condition [2]. From the perspective of the SoS, faults are manifestations of uncertainties that cause failures of the system. Therefore, an SoS can suffer from various faults arising from uncertainties.

Fault diagnosis technique helps to guarantee the safety and reliability of the system, and fault knowledge plays an important role in the fault diagnosis process [3]. Fault diagnosis technique can efficiently debug faults in systems, including an SoS. To effectively diagnose faults in SoS, a fault knowledge base for an SoS that considers various faults from uncertainties in an SoS is necessary. The fault knowledge base can be used to classify the types of faults and to provide possible solutions for debugging faults. If the fault knowledge base cannot consider various types of faults, it can encounter faults that are not in the fault knowledge base, which are called unknown faults. When a fault diagnosis technique detects unknown faults, every process from understanding to extraction of knowledge has to be re-performed. To avoid these processes and lower the cost of debugging, an initial fault knowledge base should cover information about various types of faults which makes the fault knowledge base to be a high-quality.

The generation of an initial fault knowledge base has been studied in various fields [3]–[11]. Various techniques have been proposed to generate initial fault knowledge bases with the highest possible quality. Generally, initial fault knowledge bases have been generated through either manual or semi-manual collection. Manual collection of fault knowledge bases is done by domain experts. Domain experts manually gather initial failure data from both historical data and literature surveys. Then, they extract fault knowledge from the collected data and communicate with other domain experts to generate a high-quality fault knowledge base. Because this process is entirely manual, it suffers from two major limitations. First, because every process is performed manually, considerable human effort and cost are required. Second, the generated fault knowledge base can contain expert biases due to the lack of generality and poor handling of uncertainties [2]. Thus, recent studies have aimed to automate partial procedures using various techniques to reduce human effort and expert biases.

Nonetheless, there exists no approach that fully automates the process of fault knowledge base generation [12]. Existing studies tried to automate each procedure to reduce the effort of the expert [5]–[7]. However, there exists a primary limitation in previous studies; they assume to have high-quality of fault data. These studies mainly aimed to generate a fault knowledge base with given data and to remove redundant data from the fault knowledge base. Thus, they are highly dependent on the quality of the collected fault data, which makes a fault knowledge base vulnerable with insufficient fault data given.

To overcome aforementioned limitations, we propose a

semi-automated fault knowledge base generation approach for an SoS by using uncertainty classification to check the quality of fault data. In the case study, our approach generated fault knowledge base for ADAS followed by finding and defining several unknown faults.

The remainder of this paper is organized as follows. Chapter 2 introduces the background of uncertainty classification. Chapter 3 describes the related works of fault knowledge base generation. Chapter 4 describes the proposed approach. Chapter 5 describes a case study applying the proposed approach on the ADAS system. Finally, Chapter 6 presents the conclusions of this study.

II. Background

A. Uncertainty Classification

Uncertainty has been studied in SoS and Self-Adaptive Systems (SAS) which are similar to the SoS. Uncertainty classification in SoS can be classified based on manifestation location, which is a fair and unconditional index [13]. Classification of uncertainties based on the manifestation location in SoS with short descriptions is in Table I.

In SASs, there exists many criteria for classifying the uncertainty factors that can be observed when modeling a system. The uncertainty classification is performed according to the location where the uncertainty is expressed (Location), the level of the uncertainty itself (Level), the nature of the uncertainty (Nature) [14], or according to the source level of uncertainty in software development levels (Level) [15]. In the nature of the uncertainty, uncertainty can be classified into two types. Each classification and its description is depicted in Table I.

TABLE I: Uncertainty classification based on manifestation location and nature

Manifestation Location Type	**Description**
Contextual	Uncertainty found in which CSs the SoS is composed of
Structural	Uncertainty found in the structure of CSs constituting the SoS
Parametric	Uncertainty found inside the CS
Managerial	Uncertainty found in the manager of SoS and the manager of each CS
Runtime	Uncertainty found in the runtime environment
Nature Type	**Description**
Epistemic	Uncertainty due to the lack of knowledge
Aleatory	Uncertainty due to inherent variability or randomness

III. Related Work

A. Methods of Fault Knowledge Base Generation

Studies have developed systematic methods to extract knowledge and generate knowledge base in various system fields. The fault knowledge extracted from data can play an important role in the failure analysis of systems. Thus, various studies on systematic approaches or frameworks to generate a fault knowledge base focused on how to generate a high-quality fault knowledge base, and how to form a fault knowledge base.

Liu [10] extracted two types of knowledge to generate a fault knowledge base. Ontology model was built to generate an instance knowledge base and a reasoning engine was built to generate a rule knowledge base. By first collecting a fault knowledge base in a target domain, the ontology-based knowledge base is constructed and then, rule-based semantic reasoning is performed to enrich the ontology knowledge and rule knowledge. Fei et al. [11] proposed a fault identification method to generate a fault knowledge base. First, they conducted a principal component analysis of the data, and selected the highest matching rate fault into fault knowledge base. Then they performed fault diagnosis using a least squares support vector regression algorithm based on particle swarm optimization. After diagnosis, matching rates in the fault knowledge base were updated to produce a higher-quality fault knowledge base. Yuijong et al. [3] proposed a general model of a fault knowledge base that can well present and manage knowledge. Faults are understood with aspects of the fault object, sensitive characteristic parameters, distributed fault expression, and risk assessment. By using each of these aspects as one orthogonal dimension axis, fault knowledge is extracted and a multidimensional attribute fault knowledge base is generated. However, these studies have a primary limitation. They assumed to have enough types of faults, which makes the generated fault knowledge base highly dependent on the quality of collected fault data. Thus, a classification of faults to check the quality of fault data is required to ensure the high-quality fault knowledge base.

There exist studies that classified faults when generating fault knowledge base. Xi et al. [4] dealt with metro vehicle equipment to form a high-quality knowledge base that could be used for fault diagnosis. Within the original knowledge, they conducted a fault factor analysis using fault classification based on the reasons and performance of a fault. Ya-Jin et al. [9] proposed a framework and rules to generate a fault knowledge base in reciprocating compressors. Because the target domain is complex, the representation and extraction of knowledge are difficult. Thus they used a fault tree analysis (FTA) to extract knowledge and generate a fault knowledge base. However, these studies only classifies known faults, which are already in fault data. Thus, there is a limitation that generated fault knowledge base cannot consider unknown faults, which are not in fault data. Thus, an approach to generate a fault knowledge base which can check the types of faults and include unknown faults based on the classification is required. In this study, we classify faults based on uncertainty.

IV. Approach

We propose an approach to generate a fault knowledge base that considers uncertainties in SoS. The proposed approach can check the quality of faults based on the uncertainty manifestation location and nature, and generate a fault knowledge base using extracted information from fault data.

A. Overview of Approach

The proposed approach is divided into three steps as shown in Fig. 1: fault data collection, fault information extraction, and fault knowledge base generation. The fault data collection step uses a crawler to collect failure cases in domain fields and gather fault data from failure cases using keyword-based extraction. From these failure cases, the fault keywords given by domain experts are used to extract raw fault data from failure cases. The fault information extraction step extracts fault information from the gathered raw fault data and then classifies faults based on the related uncertainty types. The fault knowledge base generation step converts classified fault information into a knowledge base with the manual injection of faults that can reinforce the fault knowledge base. With the classification of collected faults, a domain expert can check the distribution of the collected fault types, and manually inject deficient types of faults into the fault knowledge base.

B. Fault Data Acquisition

In the proposed approach, a crawling method is used to collect failure cases and keyword based extraction is used to extract raw fault data.

1) Raw Failure Data Crawling: To use a crawling method, a wordsack composed of two types of keywords is required; domain target keyword and failure related keyword. Domain target keywords are keywords that are related to the target domain that experts try to collect the failure data. Failure related keywords are keywords to find the failure cases out of various data in the target domain. When the expert inputs domain target keywords and failure related keywords, an wordsack is generated with combination of domain target keywords and failure related keywords.

Crawling requires also requires the URLs to search. When desired URLs to search are given from domain experts as inputs, the crawler stores each search result. Failure data is crawled based on three types of sources to collect data: articles, reports and papers. When news URLs such as CNN, BBC that domain experts wants to collect failure cases is given, the proposed approach collects article information from given news URL. By using the search results in URLs, the crawler extracts paragraphs that include failure related keywords from search results, and stores them into CSV file. Further, papers are searched in Google Scholar. Because we cannot always download these papers in PDF format without permission or payment, only the paper titles are saved in the CSV file. Finally, the crawler searches given specific URLs or uses the Google Search engine for reports and stores them into a CSV file.

2) Fault Keyword based Extraction: Fault data are gathered from collected failure cases through fault keyword based extraction. The proposed approach assists domain experts to easily organize failure cases and gather faults. When domain experts input fault related keywords, the proposed approach gathers keyword contained sentences from failure cases. From the CSV files that contain article paragraphs containing failure cases will extract sentences that includes fault keywords. From the CSV files that includes paper titles, domain expert should check each paper. From CSV files that contain reports extracts a cell that contains fault related keywords.

C. Fault Information Extraction

The gathered faults from failure cases are raw fault descriptions. Understandings of faults can classify faults to reduce redundant faults and add deficient fault types to generate a high-quality fault knowledge base. The detailed process for fault information extraction and uncertainty classification that can classify faults is described in the following subsections.

1) Uncertainty Type Classification: Based on two uncertainty classification criteria, uncertainty factors can be classified into ten categories since two categorization criteria, nature of uncertainty and manifestation location of uncertainty, are orthogonal. The generated uncertainty classification is used to classify faults in SoS.

2) Fault Classification based on Uncertainty Type: Based on the uncertainty classification, domain experts select classification keywords that can interpret faults in each category. With the classification keywords as inputs, the gathered fault data are classified into each category. Thus, the fault knowledge is formed with the nature type and manifestation location type.

D. Fault Knowledge Base Generation

After classifying the fault data based on the uncertainty, to strengthen the fault knowledge base, domain experts can inject faults. Then every fault knowledge are recorded into fault knowledge base with fault classification types and descriptions as attributes. This fault knowledge base can make the management and analysis of fault knowledge easier.

1) Manual Injection of Faults: Using the classification results, domain experts can find which types of faults are not or less considered in fault data. Deficient types of faults are unknown faults to the fault knowledge base. Thus, to consider unknown faults, domain experts should identify deficient fault types.

Using the combination of classification keywords based on two criteria, the proposed approach can generate unknown fault related keywords. Since domain experts can know the nature type and manifestation location type of unknown fault, they can generate a fault based on each type related keywords.

2) Fault Knowledge Base Generation: Fault knowledge is injected into the fault knowledge base with a fixed form with attributes. One fault knowledge gets into each row with columns representing attributes. Each attributes are as follows; ID, Nature, Manifestation Location, Description. Each attributes are filled with knowledge made from fault information extraction step. The types of each attribute and description are as follows.

- **ID/Int:** the number of ID of each fault knowledge
- **Nature/String:** the nature type of each fault knowledge
- **Manifestation Location/String:** the manifestation location type of each fault knowledge
- **Description/String:** the fault description sentence of each fault knowledge

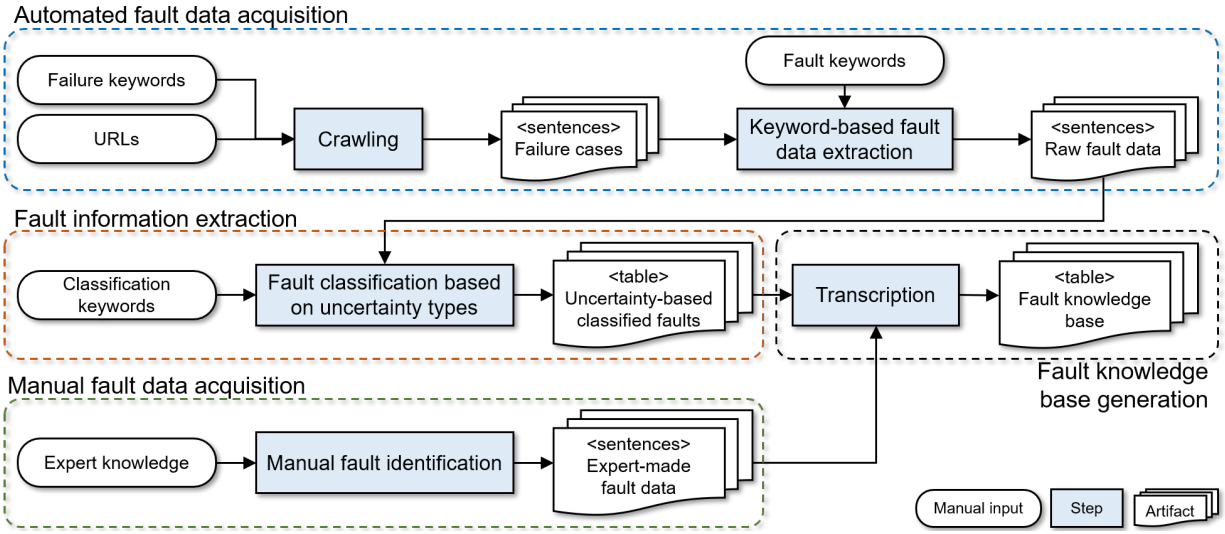

Fig. 1: Approach overview

V. Case Study

A. Experiment Setting and Data

We proposed a fault knowledge base generation approach to reduce the effort of the expert, consider various uncertainties in SoS, and enable to check the quality of fault data. To check whether the proposed approach can achieve intended goals, following research questions (RQs) are selected. Since semi-automating techniques such as crawling and keyword extraction are obvious to reduce the effort of the expert, they are not compared in detail with manual expert approaches. The main RQs in this study are as follows.

- RQ1: Can the proposed classification reasonably cover all the known fault data?
- RQ2: Can the proposed classification identify unknown faults?

To consult an experiment that can answer to RQs, experiment with advanced driver assistance system (ADAS) is held. An ADAS SoS is a system that assists driver to drive safely. An ADAS system is composed of various constituent systems that have operational independence. Since we propose uncertainty classification in SoS and fault classification based on the uncertainty that interprets each fault types, an experiment on fault data that can be classified based on the uncertainty classification is needed. Furthermore, the result of classification should show the quality of fault data. To show that proposed approach can classify faults well and show the quality of fault data, among various SoS, we have chosen the ADAS system which has the most plenty fault information we can assess. RQ1 aims to check whether the fault classification reasonably classifies all known fault data which are collected faults. Each classification results are compared in precision and recall using groundtruth as manually classified result. After we check whether the fault data are well classified based on uncertainty, we will see the classification coverage of fault data to see the quality of fault data. Having more classification coverage will show that the collected fault data have more types of faults which shows the quality of fault data. RQ2 aims to identify unknown faults will be held using classification keywords. From the fault classification result, there are deficient types of faults that are less or not considered. We check whether we can identify the types of unknown faults and generate a description that can express unknown faults based on their types. A combination of keywords that can generate a fault description is held. Further than checking the quality of fault data, the proposed approach can give inference to deficient fault types based on the classification results. The experiment settings to consult an experiment and each step in the proposed approach are described in the following subsection.

1) Fault Data Acquisition: To gather fault data, a wordsack containing domain target keywords and failure related keywords is required. Domain target keywords and failure related keywords selected to crawl failure data are shown in Table II. Total 45 combined keywords are filled in wordsack with combination of domain target keywords(9 keywords) and failure related keywords(5 keywords).

TABLE II: Keywords to collect data

Domain target keyword	Failure related keyword
advanced driver assistance system, ADAS, adaptive cruise control, ACC, automatic emergency braking, AEB, self driving, self-driving car, autonomous vehicle	failure, crash, disengagement, testing, test scenarios

Failure data are crawled in three manners. First, articles are collected from BBC, CNN news articles with web crawling using BeautifulSoup4 and Reqeusts in Python. The following URLs are manual inputs.

- **BBC** “https://www.bbc.co.uk/search?q=”
- **CNN** “https://edition.cnn.com/search?size=10&q=”

The news crawler searches combinations of domain target keywords and failure related keywords in the given URLs

by adding these combinations at the end of the URLs. From search results, the crawler searches through pages by adding '&page=' and an increasing number after the URLs. After the search result is given as HTML form from requests, Beautifulsoup parses the HTML. The crawler extracts every titles of each articles and with given href link in articles, crawler gets searched the connected link and crawls the extract keyword contained paragraphs from result articles. The search results are saved in a CSV files. Next, a literature survey is done with Google Scholar with crawler too. Since there are difficulties to save papers in PDF form without permission or payment, titles of the paper are saved in a CSV file. Finally, reports from practical testings and failures are collected from Google Search and expert desired URLs. For the ADAS system, a URL containing disengagement reports from the California Department of Motor Vehicles (DMV) is used. The DMV offers self driving car disengagement reports and crash reports as CSV files.

2) Fault Information Extraction and Fault Knowledge Base Generation: Failure keywords included sentences are then organized to determine the fault that causes the failure. First, faulty sentences are extracted based on fault related keywords which is given as input. For the ADAS system, 25 keywords are selected to express related faults. After extracting faulty sentences from the failure data in CSV files, each fault is classified based on the uncertainty classification using classification keywords as described below in Table III.

TABLE III: Keywords to classify faults

Classified types	Keywords
Epistemic	unexpected, undesired, unwanted, understand, knowledge
Aleatory	random
Contextual	performance, missing, correct, prediction, discrepancy, algorithm, accuracy
Structural	structure. network
Parametric	sensor noise, input, detection
Managerial	match, management, constraint
Runtime	circumstance, environment, human, weather

After the classification is done, unoccupied category of classification table are manually injected. Finally, the classified faults are inserted into the fault knowledge base with classified knowledge as attributes.

B. Results and Analysis

- RQ1: Can the proposed classification reasonably cover all the known fault data?

When the proposed approach was applied to the ADAS, the crawler generated 9,680 fault data entries. Each entry was classified using the classification keywords shown in Table III. Number of faults classified through each keywords is shown in Table IV. We can see that every faults were well classified orthogonal with two classification criteria by checking the total faults of each classification criteria. The number of each classification category in the classification result of the proposed approach is shown in Table V.

TABLE IV: Number of faults classified by each classification keywords

Classified types	# of classified faults	Total faults
Epistemic	8721	9680
Aleatory	959	
Contextual	4319	9680
Structural	12	
Parametric	2478	
Managerial	0	
Runtime	2871	

TABLE V: Fault classification for the ADAS scenario

	Contextual	Structural	Parametric	Managerial	Runtime
Epistemic	4,312	12	1,527	0	2,870
Aleatory	7	0	951	0	1

The collected 9,680 fault data entries had 595 distinct fault descriptions, and each 595 distinct fault descriptions were analyzed manually for fault classification. Using the manual fault classification result as the groundtruth, the proposed fault classification result is compared. The *true positives* counts the faults in each classification from result of proposed classification that are also classified same in the manually classified faults. The *false positives* counts the faults in each classification from result of proposed classification that are differently classified in the manually classified faults. The *false negatives* counts the faults in the manually classified faults that are differently classified in the proposed classification. The results of precision and recall are shown in Table VI and Table VII. The average precision was 0.989917, and the average recall was 0.770318.

TABLE VI: Precision of each classification

	contextual	structural	parametric	managerial	runtime
epistemic	0.939239	1	0.990177	X	1
aleatory	1	X	1	X	1

TABLE VII: Recall of each classification

	contextual	structural	parametric	managerial	runtime
epistemic	1	0.272727	1	X	0.930308
aleatory	0.189189	X	1	X	1

Before the manual injection of faults, the proposed approach could classify 7 out of 10 types of faults. Every faults were classified into the classification table successfully and we could find only 7 out of 10 types of faults were covered with the collected data. The coverage of classified types of faults can say the quality of the fault data, since having more various faults will make higher quality fault knowledge base. Therefore, we can say that the proposed approach can be used to check the quality of collect fault data by considering the coverage of classified types of faults.

- RQ2: Can the proposed classification identify unknown faults?

The proposed approach can identify unknown faults based on the classification result. Using the Table V, we can see

which types of faults are deficient. Thus, we can find the nature type and manifestation location type of faults that are less or not considered in the fault data. Among the 10 types of faults, three types were not considered in the given fault data: epistemic-managerial, aleatory-structural, and aleatory-managerial. Further, three types were less considered than other types of faults in the given fault data: epistemic-structural, aleatory-contextual, and aleatory-runtime. The proposed approach can then generate a combination of keywords based on the keywords listed in Table III. Based on keyword combinations such as 'unexpected constraint' as an epistemic-managerial type of fault generated from combination of keyword in Table III, domain experts can generate an unknown fault description using their own knowledge. For example, a fault with an epistemic nature and that manifests between managers is not considered. Using the combination of keywords in the epistemic nature and manifested in the manager, 'unexpected constraint' faults in the ADAS system can be found as 'Unexpected request failure due to the difference in constraints'.

C. Threats to Validity

Because the proposed classification table has classified only faults in the ADAS system, it is hard to guarantee that every faults in the SoS can be classified based on the classification table. Yet, no one can say that one has discovered uncertainty fully. The proposed approach can still find unknown faults out of given faults based on the classification table.

Manually classified faults might be biased and are not proper to use as a groundtruth oracle. Because the proposed approach is the first to classify faults based on an uncertainty classification table in consideration of SoS characteristics, no open source groundtruth is available for use. To avoid bias, manual classification was performed recursively, and to avoid abnormal classification, we have randomly selected fault data and classified and checked the classification results again and again.

VI. Conclusion

Diagnosing as many different faults is as possible is important since most SoSs are directly related to safety. To diagnose faults and test with various failure scenarios, a high-quality fault knowledge base is essential. This study thus proposes an approach for generating a fault knowledge base in SoS considering uncertainty, to generate a high-quality fault knowledge base by checking the quality of fault data and identify unknown faults. With a high-quality fault knowledge base, both modeling and testing an SoS can consider more faults that occur to the system. Therefore, a high-quality fault knowledge base built using the proposed method can build a more failure-safe SoS. As a future work, a case study on other SoS systems where various fault data and practical testing scenarios are available will be held. Also, instrumentalization of every process into one tool will be held.

Acknowledgment

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2021-2020-0-01795) and (No. 2015-0-00250, (SW Star Lab) Software RD for Model-based Analysis and Verification of Higher-order Large Complex System) supervised by the IITP(Institute of Information Communications Technology Planning Evaluation).

References

[1] John Boardman and Brian Sauser. System of systems-the meaning of of. In *2006 IEEE/SMC International Conference on System of Systems Engineering*, pages 6–pp. IEEE, 2006.

[2] Dubravko Miljković. Fault detection methods: A literature survey. In *2011 Proceedings of the 34th international convention MIPRO*, pages 750–755. IEEE, 2011.

[3] Yujiong Gu, Zhan Gao, Xiaolu Wang, Kun Yang, and Kunliang Chen. Research on the construction of fault knowledge base for power plant equipments. In *World Automation Congress 2012*, pages 1–5. IEEE, 2012.

[4] Xi Li, Xiaoning Zhu, Guoqiang Cai, and Limin Jia. A novel fault diagnosis expert system knowledge acquisition method of metro vehicle equipments. In *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, volume 5, pages 535–539. IEEE, 2010.

[5] Jakub Koperwas, Łukasz Skonieczny, Marek Kozłowski, Piotr Andruszkiewicz, Henryk Rybiński, and Wacław Struk. Intelligent information processing for building university knowledge base. *Journal of Intelligent Information Systems*, 48(1):141–163, 2017.

[6] Pero Subasic, Hongfeng Yin, and Xiao Lin. Building knowledge base through deep learning relation extraction and wikidata. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*, 2019.

[7] Se-Hyeon Jo, Hack-Jun Kim, So-Yeon Jin, and Woo-Sin Lee. A study on building knowledge base for intelligent battlefield awareness service. *Journal of the Korea Society of Computer and Information*, 25(4):11–17, 2020.

[8] Stefano Nativi, Mattia Santoro, Gregory Giuliani, and Paolo Mazzetti. Towards a knowledge base to support global change policy goals. *International Journal of Digital Earth*, 2019.

[9] Ya-Jin Liu and Qi Song. Construction of knowledge-base for fault diagnosis expert system of reciprocating compressors. In *MECHANICS AND MECHANICAL ENGINEERING: Proceedings of the 2015 International Conference (MME2015)*, pages 409–413. World Scientific, 2016.

[10] Bing Liu. Construction of ontology-based fault knowledge base of aerospace tt&c equipment. *International Core Journal of Engineering*, 6(1):224–230, 2020.

[11] Fei Chen, Zhongguang Fu, and Zhiling Yang. Research on intelligent fault identification technology of wind turbine supported by fault knowledge base. *AMSE Journals-AMSE IIETA publication-2017-Series: Modelling A*, 90(1):1–15, 2017.

[12] Ran Yan, Yang Jian, Li-Chao Hao, Xin-Yu Han, and Long-Li Tang. Research on automatic knowledge acquisition technology for software fault diagnosis. In *2019 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE)*, pages 901–907. IEEE, 2019.

[13] Seungchyul Shin, Sangwon Hyun, Yong jun Shin, Jiyoung Song, and Doo-Hwan Bae. Manifestation location-based classification of uncertainty factors considering characteristics of system-of-systems. *KTCP*, 26(10):451–457, 2020.

[14] Diego Perez-Palacin and Raffaela Mirandola. Uncertainties in the modeling of self-adaptive systems: A taxonomy and an example of availability evaluation. In *Proceedings of the 5th ACM/SPEC international conference on Performance engineering*, pages 3–14, 2014.

[15] Andres J Ramirez, Adam C Jensen, and Betty HC Cheng. A taxonomy of uncertainty for dynamically adaptive systems. In *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 99–108. IEEE, 2012.