Journal of KONBiN 2022
Volume 52, Issue 4
DOI 10.2478/jok-2022-0045

Tomasz MUCHOWSKI[1], Adam SZELEZIŃSKI[1], Lech MURAWSKI[1], Adam MUC[1], Marcin KLUCZYK[2]
[1] *Gdynia Maritime University (Uniwersytet Morski w Gdyni)*
[2] *Polish Naval Academy (Akademia Marynarki Wojennej w Gdyni)*

THE PROCESS OF ACQUIRING, COLLECTING, PROCESSING AND ARCHIVING DATA FOR THE SHM SYSTEM DESIGNED TO IDENTIFY DEFECTS IN THIN-WALLED STRUCTURES
Proces pozyskiwania, gromadzenia, przetwarzania i archiwizacji danych dla systemu SHM przeznaczonego do rozpoznawania wad struktur cienkościennych

Abstract: *The article shows the results of the preparatory steps taken to create the artificial intelligence used in the automatic recognition of defects in ship thin-walled structures. The above steps are used to create a university private cloud and a computer system maintaining a dataset of vibration signal samples. In the article, a prototype of the private cloud was designed and developed, a model of the vibration sample was prepared, and a microservice was designed aimed at sharing the obtained data. The article demonstrates the results of the completed development.*
Keywords: Vibrations, defects detection, SHM, neural network, database, private clouds

Streszczenie: *W artykule przedstawiono wyniki przeprowadzonych działań przygotowawczych do stworzenia sztucznej inteligencji wykorzystywanej w automatycznym rozpoznawaniu defektów okrętowych konstrukcji cienkościennych. Przeprowadzone kroki służą stworzeniu uczelnianej chmury prywatnej oraz systemu informatycznego utrzymującego zbiór danych próbek sygnałów drganiowych. W ramach artykułu zaprojektowano oraz stworzono prototyp chmury prywatnej, przygotowano model próbki drganiowej oraz zaprojektowano mikroserwis służący udostępnianiu uzyskanych danych. Artykuł przedstawia wyniki wykonanej pracy.*
Słowa kluczowe: drgania, wykrywanie defektów, SHM, sieć neuronowa, bazy danych, chmury prywatne

1. Introduction

Harsh weather conditions and deterioration resulting from the continuous operation of transportation vessels used at sea and in the air lead to damage. The resulting structural defects can result in malfunctions, which carry enormous risks for the transported passengers and crew. It is necessary to conduct periodic tests of the quality of transport vessels and detect the resulting defects. To eliminate hazards, periodic NDT (Non-Destructive Testing) tests are carried out [9, 10], but unfortunately, periodic testing is not sufficient. For this reason, the topic of SHM (Structural Health Monitoring), which allows for continuous monitoring of the structure's integrity, is increasingly being addressed.

Structural Health Monitoring, as a technique involving continuous automatic observation and analysis of the structure, is based on continuous recording of measurement samples in real time. The advantage of SHM over the traditional form of NDT is that it is not carried out periodically, so the possibility of failure between tests is excluded. This guarantees the safe operation of machinery, equipment, vehicles and materials in terms of failure due to undetected damage. Implementation of the SHM system requires analysis of all samples taken by the system to detect emerging changes and the occurrence of new damage [4]. One of the most challenging tasks of the SHM system is the automatic and reliable analysis of the incoming measurement data. This analysis aims to detect, locate, quantify damage, and estimate the time for the safe operation of the device. The use of neural networks as a form of processing and analyzing the recorded vibration samples will allow the learning process of the algorithm for defect recognition, and, through this, the creation of an artificial intelligence system.

Creating artificial intelligence that reliably analyzes recorded data requires an appropriate learning process. By learning artificial intelligence, it is necessary to provide information about possible defects and measurement data indicating their occurrence [2, 5]. Therefore, extensive amount of measurement data is needed. It transpires that collecting and analyzing them is an expensive activity beyond the capabilities of a single educational facility.

The authors decided to prepare a computer infrastructure carrying out SHM using a neural network. The work on the infrastructure was divided into several stages. The first step is to design a dataset and implement a management system to collect the recorded information. The collected measurement data are provided from different measurement software. This indicates that the structure of the data varies and forces the need to abandon relational databases in favour of non-relational databases. It is also necessary to prepare a separate sample model for each software. The article is based on measurement software from m+p International.

2. Preparation of the required infrastructure

The infrastructure consists of a private cloud prototype, a database and microservices responsible for accepting data and placing it in the dataset.

The preparation of the private cloud was based on the prior work of one of the authors. The hardware part of the infrastructure consists of a router responsible for providing network communication capabilities, a NAS (Network Attached Storage) server storing collected data, and a cluster of computers maintaining microservices. The security layer was implemented using the Personalizable Local Area Network Cipher created by the Internet of Things and Embedded Systems Science Club at Gdynia Maritime University. It enabled to reduce the effort involved in creating the appropriate security features, such as providing encrypted communication between nodes in the cloud cluster by implementing mTLS using SSL certificates.

To create a cloud cluster, Kubernetes technology was used to act as an orchestrator of the implemented software processes and to equally distribute tasks among the nodes. The Docker containerization engine was installed on the cloud nodes to support containerized applications. This way, the created microservices became scalable and could be deployed on the nodes automatically without administrator intervention. This is an important feature, given the constant development of the infrastructure and the continuous implementation of new microservices responsible for receiving and processing information from various measurement software. The created hardware infrastructure is presented in Fig. 1.

The stability of the operation of private cloud nodes and the database management system is crucial to the reliability of the operation of the prepared computer system. A key aspect is using a suitable operating system to guarantee the uninterrupted operation of private cloud computing devices. Based on the literature review [1] and performed tests, it was decided to use operating systems based on the Linux kernel. Of the many Linux distributions, rolling-release distributions (such as Arch Linux [12]) were ruled out because they do not guarantee compatibility and proper cooperation of packages, as well as packages are updated to the latest versions, on which adequate stability tests are not conducted. There is a high probability of errors in their operation. Point-release distributions were chosen because they are released periodically after proper testing. Ubuntu is the most popular among such distributions, but since packages from the testing branch are placed in the repository of this distribution [11], it was decided to use the Debian distribution on which Ubuntu is based. Only packages from the stable branch are present in the Debian repository, and in addition, the institutions taking care of this distribution perform additional stability testing of packages delivered to the distribution's repository. Preparing an infrastructure based on the Debian distribution requires more work than for Ubuntu, due to the need to implement and configure security solutions such as apparmor oneself [8, 13]. The authors believe that the system’s stability is worth the additional time and effort.

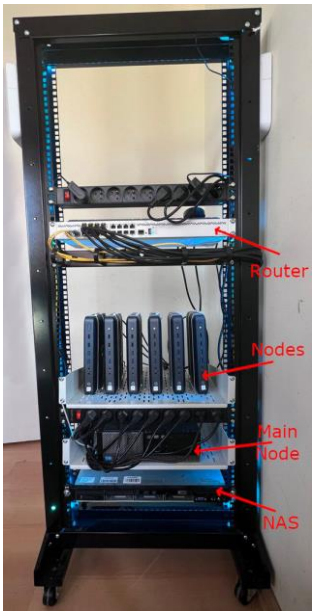

Fig. 1. Created hardware infrastructure

Document-based non-relational databases store information in documents, which are stored in collections. Documents do not have to have a specific structure, which means that a collection containing measurements of a particular object can store measurement results made with different software. This provides the flexibility required by the computer system. Based on the literature review [3, 6, 7] and tests conducted, it was decided to choose MongoDB as the database management system. This system is characterized by high performance and reliability, and since it was created using JavaScript, it provides high compatibility with object-oriented programming languages. The simplicity of implementing the communication layer between microservices and the database management system is a significant advantage, given the need to implement a large number of microservices processing and sending information to the database. An essential feature of measurement data collection is the ability to store static files in the database, thanks to the GridFS module.

MongoDB is also a cross-platform database management system and can, without having to compile the source code oneself, be installed on the Debian operating system. An important feature of MongoDB is the ability to prepare several instances of database management systems on different devices and then combine them into a replica set. In this way, if one of the servers fails, the remaining instances of the database management system have a full copy of the accumulated data and can be used as a replacement. An equally important feature is the ability to split a database between several servers through sharding. This allows faster access to data in case the database grows large. Both functions are

particularly important because the designed information infrastructure is subject to constant growth and is dedicated to storing large amounts of data [14].

The database management system was installed on the NAS server of the private cloud prototype. In the current state of the computer infrastructure, the NAS function is performed by a single server. Data security is accomplished both through cyclic backups, but with the installation of more NAS servers and with the increase in the amount of accumulated measured data, it is planned to expand the database to more servers through sharding. Figure 2 shows a diagram of how replica set and sharding work.

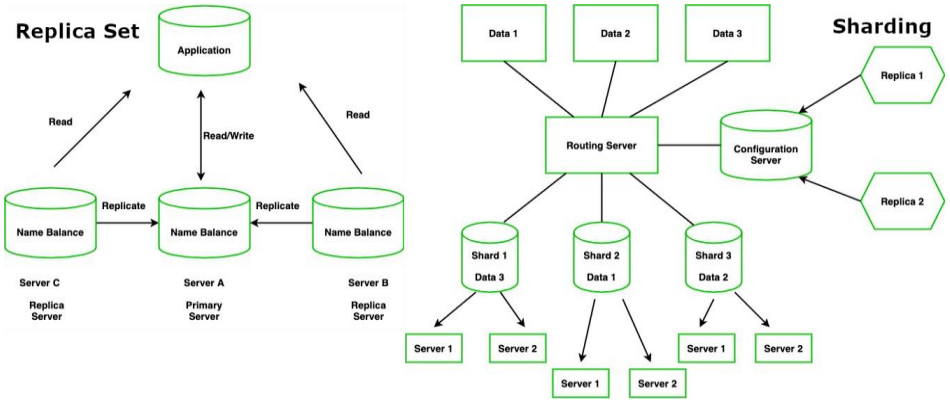

Fig. 2. Comparison of Replica Set and Sharding

The need for future sharding arises from the possibility that a single server may become unavailable if the demand for data transfer bandwidth is too high. Data transfer within one of the queries requiring significant data processing may lead to overloading of the server's processor and the buffer may exceed the operating memory possessed by a single server forcing an increase in the number of read and write operations of the hard disk. This has the effect of significantly reducing the performance of the database management system and the delays associated with providing the required information to the microservices. The use of sharding makes it possible to spread the traffic associated with data exchange over several servers, thereby reducing the number of read and write operations for which a single server is responsible. In addition, sharding's internal use of a replication mechanism ensures data availability if one of the servers fail. As the number of shards increases, the total capacity of the database also increases, which is crucial in collections that are dedicated to storing vast amounts of information.

3. Data composition

The composition of the document’s structure containing information about the vibration measurement made with the m+p software requires careful program analysis.

It is necessary to create the structure of the input data, which are the measurement presuppositions and environmental variables, and create the structure of the output data, which represents the measurement results.

A graphical interface is used when performing measurements with m+p VibRunner/VibEdit and m+p Analyzer software. Through the interface, measurement parameters and environmental variables are entered. The composition of the input data structure is based on creating a model of the input data. A fragment of the created input data structure is shown in Fig. 3 (the entire structure consists of several hundred lines of JSON code).


```
"input": {
    "slopes": [{
        "frequency": {
            "unit": "Hz",
            "value": 10
        },
        "acceleration": {
            "unit": "g",
            "value": 1.5
        }
    }],
    "units": {...},
    "measurement": {
        "number_of_lines": 16384
    },
    "throughput": {
        "sampling_rate": {
            "unit": "Hz",
            "value": 32768
        },
        "file_format": "sot",
        "start_with_self_test": false,
        "start_with_test_run": true,
        "stop_with_test_end": true,
        "pause_on_standby": true
    },
    "shaker_definition": {...},
    "specimen": {
        "name": "plyta",
        "weight": {
            "unit": "kg",
            "value": 4
        },
        "number": 2
    },
    "fixture": {...},
    "sweep_conditions": {
        "lower_frequency_limit": {...},
        "upper_frequency_limit": {...},
        "starting_frequency": {...},
        "sweep_direction": {...},
        "start_drive": {...},
        "startup_time": {...},
        "startup_tolerance": {...},
        "shutdown_time": {...}
    },
    "repetiton_parameters": {...},
    "channels": {...}
},
```


Fig. 3. Fragment of the input structure of the sample model

The output of measurements made with m+p VibRunner and m+p Analyzer software is the measurement data from each sensor and the graphs generated by the software. From the point of view of the neural network, the collection of graphs in the database is an optional activity, but they are useful in the manual expert analysis. For this reason, the authors decided to include graphs in the created model. A fragment of the created output data structure is shown in Fig. 4. Graph data (GridFS file) and input data have been omitted for readability reasons.


```
"output": {
    "run": 2,
    "time": "05-16-22 09:33",
    "x": {
        "unit": "Hz",
        "data": [10, 10.0039, 10.0078, 10.0117, 10.0156]
    },
    "reference": {
        "unit": "g",
        "data": [1.5, 1.5, 1.5, 1.5, 1.5]
    },
    "channel": {
        "unit": "g",
        "name": "Channel 3",
        "type": "M - Filtered",
        "data": [0.0150448, 0.0151514, 0.0152579, 0.0153645, 0.015471]
    },
    "phase": {
        "unit": "deg",
        "data": ["-73.3", "-73.3", "-73.3", "-73.3", "-73.3"]
    },
    "graph": "..."
}
```


Fig. 4. Fragment of the output structure of the sample model

The microservice responsible for receiving measurement data processes the transmitted data and stores it in the database. The data is stored unaltered. The authors concluded that the processing of the data into a unified measurement model should take place immediately before the data is transferred to the neural network. In this way, no information is omitted. It should also be noted that individual measurement programs are equipped with closed-source algorithms for processing measurement data. Transforming the data into a unified model before analyzing the measurement results can lead to errors. All microservices created for the information infrastructure were made using the Python programming language.

4. Conclusions

The created model of vibration signal samples is one of the first stages of creating an computer infrastructure that enables SHM using neural networks. However, it is a crucial stage, as it forms the basis for further work on the infrastructure. The model presented in the article is based on software from m+p. However, the authors are working on models for

other software (including software from Bruel&Kjaer). A unified model is also being developed, as well as a model for analyzing measurement results and damage classification. It is currently undergoing validation by specialists in the field.

The ongoing work is currently closed internally at Gdynia Maritime University, but once the microservices responsible for sharing and receiving data from external facilities are created, which is the next stage of the work, an API will be made available to enable cooperation with external entities.

5. References

1. Boras M., Balen J., Vdovjak K.: Performance Evaluation of Linux Operating Systems. 2020 International Conference on Smart Systems and Technologies 2020.
2. Dreyfus G.: Neural Networks: Methodology and Applications. Springer 2005.
3. Faraj A., Rashid B., Shareef T.: Comparative study of relational and non- relations database performances using Oracle and MongoDB systems. International Journal Of Computer Engineering & Technology Vol. 5. 2014.
4. Gorinevsky D., Gordon G., Kumar A., Chang F.: Integrated SHM System for Commercial Aircraft Applications. 5th International Workshop On Structural Health Monitoring at Stanford 2005.
5. Grossi E., Buscema M.: Introduction to artificial neural networks. European Journal of Gastroenterology & Hepatology. Vol 19. 2008.
6. Gunjal B.: Database System: Concepts and Design. Proceedings of 24th IASLIC-SIG-2003. 2003.
7. Győrödi C., Gyorodi R., Sotoc R.: A Comparative Study of Relational and Non-Relational Database Models in a Web- Based Application. International Journal of Advanced Computer Science and Applications. Vol 6. 2015.
8. Kumar V.: Debian: A Linux based operating system for all purposes. Design and development of an information network for affiliating universities in Kerala 2019.
9. Szeleziński A., Gesella G., Murawski L.: Przegląd metod diagnostyki i monitoringu połączeń spawanych w konstrukcjach transportu morskiego, Logistyka 3/2015.
10. Uhl T.: Współczesne metody monitorowania i diagnozowania konstrukcji, Gliwice 2010.
11. https://help.ubuntu.com/community/Repositories/Ubuntu
12. https://wiki.archlinux.org/title/Arch_Linux
13. https://www.debian.org/doc/
14. https://www.mongodb.com/docs/manual/sharding/