# A Proposal for Smarter Railway Maintenance

1st João T. Fernandes
*Center for Informatics and Systems*
*Department of Informatics Engineering*
*University of Coimbra*
Coimbra, Portugal
joaonf@dei.uc.pt

2nd Marilia Curado
*Center for Informatics and Systems*
*Department of Informatics Engineering*
*University of Coimbra*
Coimbra, Portugal
marilia@dei.uc.pt

3rd Fernando Boavida
*Center for Informatics and Systems*
*Department of Informatics Engineering*
*University of Coimbra*
Coimbra, Portugal
boavida@dei.uc.pt

***Abstract*—In contrast to many other technological areas that benefit from open and flexible tools and platforms, existing smart railway maintenance systems typically rely on closed, tailor-made, non-standard solutions that are a major obstacle to service integration, software components reuse, compatibility between different system modules potentially from different vendors, resource orchestration, and overall programmability. In this paper we propose an evolution of current smart railway maintenance systems, which can be the basis for all of the mentioned features. We then explore the proposed concept in a proof-of-concept implementation over a cloud environment and obtain some preliminary results that point to the adequateness of the proposal.**

***Index Terms*—Smart Railway Maintenance, Context-Aware Internet-of-Things, Railway Communications, Software-Defined Railway Monitorization**

## I. Introduction

Over the last century, rail transport has evolved from steam, low-velocity trains to electric, high-speed, and high-tech trains. In parallel with trains, which carry more people and freight, infrastructures also evolved to provide safety, comfort, and robustness. Nowadays, trains can be used for traveling between cities and in urban areas, for delivering freights between two or more different points, or even for connecting different countries. Railway systems are now crucial to modern society, with a general, sustained, overall increase in passengers and freight, and, consequently, added pressure on both trains and infrastructures not to decrease the quality of service. In this scenario, continuous railway systems operation with stringent quality requirements turns inspection and maintenance into critical activities, for which procedures, tools, and technologies must be developed and put in place. For this, both vehicles and infrastructure monitoring and maintenance activities are unavoidable and should be continuously performed, taking advantage of all technological advances for simultaneously delivering the needed functionality and reducing costs. In this respect, technologies such as Internet of Things (IoT) and Industry 4.0, Big Data, and, last but not least, 5G [1] are coming into play and must continue to be explored to their fullest potential, leading to what is now envisioned as Smart Railway Maintenance (SRM). Nevertheless, one very important aspect that is lacking in all of the existing railway maintenance systems is flexibility. In general, the construction of railway maintenance systems uses a vertical approach, in which every piece of the system is specially built for the specific application at hand. This is a paradigm similar to the one existing in computer systems and networking systems several decades ago.

In this context, the main goals of the present paper are to provide answers to the following questions:

- How can we make railway maintenance more flexible, exploring the advantages of programmability, virtualization, modularization, and scalability?
- Can components of flexible SRM systems reside in the Cloud and operate normally without significant impact on communication and event triggering capability?

The main contributions of this paper are the following:

- We propose a novel SRM architecture that potentiates programmability, service integration, dynamic scaling, orchestration, virtualization, and openness of railway maintenance;
- We provide an insight on a proof-of-concept (PoC) implementation of such architecture, through some preliminary experiments that address the behavior of key components of the proposed SRM architecture.

The paper is organized as follows. Section II briefly identifies relevant work in the area of smart railway maintenance. In Section III, we describe the proposed SRM architecture. Experimental results arising from a PoC implementation and respective discussion are presented in Section IV. We provide concluding remarks and guidelines for further research in Section V.

## II. Related Work

Current work on Smart Railway Maintenance explores technologies such as 5G communications, edge and cloud computing, Internet-of-Things (IoT), and Big Data analysis.

Rikhotso et al. [2], Vijaykumar et al. [3], Feng et al. [4], and Gan et al. [5] proposed visual inspection of tracks and components to detect superficial defects. The main idea of these studies was to collect images (using cameras) to assess the information of certain parts of the infrastructure (e.g., rails or sleepers). Regarding railway vehicles, Lu et al. [6], Ulianov et al. [7], and Liu et al. [8] developed systems to monitor and get information about the train. By using a variety of techniques, especially with image capturing, the authors were

able to capture defects and anomalies on components of the vehicles. One option that is currently being explored is the use of railway vehicles to assess information on the infrastructure or the vehicle itself. Lederman et al. [9] and [10], Bocciolone et al. [11], Pau et al. [12], and Goodman et al. [13] all provided systems and insights on how a vehicle can be used to collect information from the infrastructure, to detect defects either in the tracks or vehicles.

All the previous cases were focused on anomaly detection, which is the core objective of Smart Railway Maintenance. Nonetheless, systems such as these can generate large amounts of data, that must be somehow transmitted over the network. Thus, work in the communication area, with emphasis on 5G, are especially useful and important to improve the current state of railway maintenance systems. For example, Jamaly et al. [14] and Talvitie et al. [15] studied how the 5G technology can have a positive impact on modern railway systems. He et al. [16] studied millimeter-Wave (mmWave) communication and how to use it to improve train-to-infrastructure (T2I) communication in the current smart rail mobility era. Concerning high-speed trains (HST), Liu et al. [17] and Wu et al. [18] used 5G communication technologies, especially mmWave, to enhance railway communications.

Despite the fact that existing as well as proposed systems are getting technologically more advanced, they have some common limiting features, namely the fact that they mostly consist of vertical solutions that are closed (i.e., tailored for specific equipment/systems), technology-specific, and not modular. This prevents integration and reuse of services, modules, and components, and is an obstacle to the benefits of abstraction, virtualization, and orchestration of large numbers of devices/resources.

## III. Proposed SRM Architecture

The proposed architecture explores benefits that come from software-defined approaches, similarly to what is now being used in Software-Defined Networking (SDN). With this approach we intend to allow for some key features, which include:

- Virtualization – ability to deal with monitoring and maintenance resources independently of their physical details;
- Programmability – ability to change the monitoring behavior of the system on the fly;
- Performance – ability to optimize the use of resources (physical resources can be shared by several monitoring applications);
- Service integration – modular approach allows code and service reuse, for ease of development;
- Openness – compatibility between different system modules, potentially from different vendors;
- Orchestration – ability to manage large numbers of devices, with full visibility over them;
- Dynamic scaling – ability to scale the system according to the application needs, through resource virtualization and cloud operation;
- Automation – ability to automate parts of the system monitoring application, leading to better performance and lower operation costs.

The architecture, presented in Figure 1, considers three planes, namely: (1) Data Plane, dealing with data collection from trains and/or infrastructure; (2) Control Plane, where the bulk of control decisions and processing occurs; and (3) Application Plane, that deals with global SRM management decisions, as well as with different SRM applications, potentially from different SRM entities. The proposed architecture establishes two important interfaces between the referred planes: the southbound interface, that connects the Data and Control Planes, and the northbound interface, that connects the Control and Application Planes. It should be noted that this architecture allows for the decoupling of the various planes, i.e., it allows for modules belonging to different planes to reside in different systems, possibly at diverse locations. Moreover, as modules in different planes interact through well-defined interfaces, this means that these interfaces can potentially be standardized and the modules can be developed by different manufacturers.

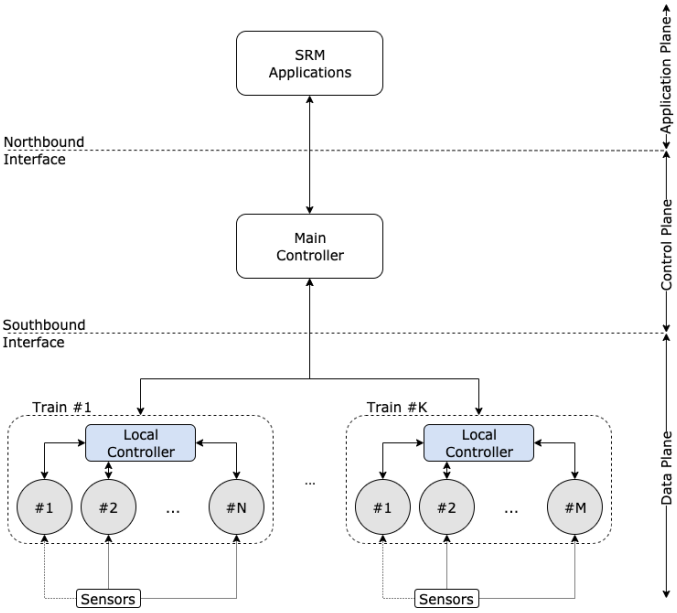



Fig. 1: Proposed SRM architecture.

Figure 2 provides an example of some interactions between entities belonging to different planes. In this example, the Control Plane (comprising a server, a database, and a processing module) receives and processes the data, with the possibility of sending it to an external processing module, which, then, uses and stores information in the database. Finally, the Application Plane acts as an interface to applications that want to access the data. This plane can also receive triggered events from the Control Plane.

The purpose and scope of local controllers (Data Plane) and main controllers (Control Plane) are quite different. While the former deal with local data gathering from multiple

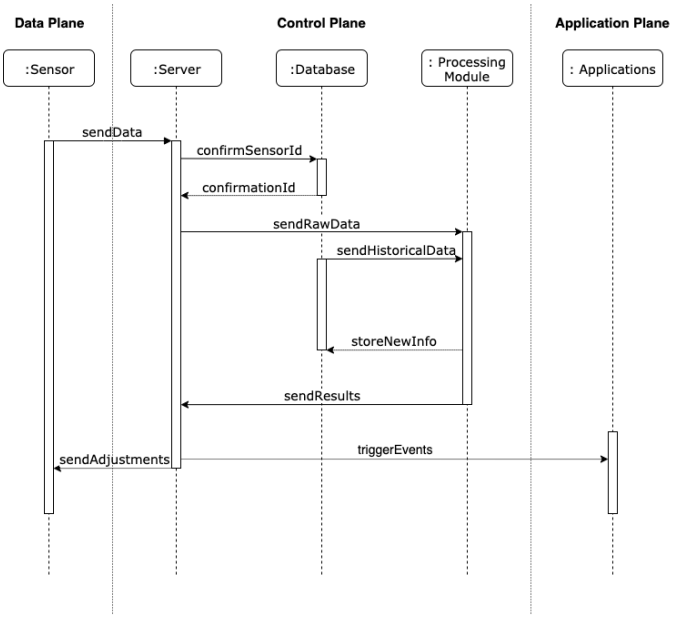



Fig. 2: Example of sequence diagram for data exchanges between the various planes.

sensors, possibly with some simple processing and/or temporary storage, the latter can perform extensive processing of data from multiple local controllers, and typically consist of unconstrained machines residing in the Cloud. Examples of interactions sent from local controllers to main controllers are data messages containing sensor readings with timestamps, or update messages informing the main controller of changes in the sensors, sampling rate, or communication rate. In the opposite direction, the main controller can instruct a local controller to perform specific sensor readings with specific sensing rates, obtain a list of available sensors, or consult its status. In the case of the northbound interface, applications can interact with a main controller in order to establish high-level control objectives (e.g., perform readings on all trains passing a given railway section in a given period). Unlike the southbound interface, this interface does not need to have a specific semantic. Instead, it can work over an API (e.g., RESTful API) allowing quick implementations of third-party applications.

JSON encoding is used to exchange information between each plane. At the time of writing, several types of messages have been defined, but the set is not limited and may grow as new functionality is developed. As a way of example, Table I provides information on the parameters contained in a message carrying sensor readings from a temperature sensor. Each local controller gets a unique ID (provided by the main controller) at registration time. Moreover, each sensor is identified by a sensor ID unique within the scope of the respective local controller.

With the possibility of having multiple local controllers scattered around various locations (e.g., multiple railway lines, sidings), with ability to collect data on a variety of aspects

TABLE I: Mandatory parameters for sending a message from the local controller to the main controller.

| Parameter | Key | Sub-keys | Example |
|---|---|---|---|
| Identification | lc_id | - | 75b6aa28-0c31-321a-afc7-b832eb5ae042 |
| Sensor ID | sensor_id | - | ['p_temp_1'] |
| Sensor Name | sensor_name | - | ['Temperature'] |
| Unit | unit | - | ['Celsius', 'C'] |
| Payload | payload | value<br>timestamp | {'value': 30, 'timestamp': 1576005605} |

(e.g., engine's temperature, train's velocity), the information needs to be organized into different topics. This information may be read by Central Controllers at any given time, using appropriate status request messages. It is up to the Central Controllers to keep track of the capabilities of each local controller and of which information is being gathered from each local controller and for what purpose.

Whenever information arrives at the Control Plane, the main controller will process it according to the requirements set by the Application Plane. The gathered data/results can be compared with historical data to determine trends, detect significant events (e.g., an anomaly, a malfunctioning or inoperative sensor), and trigger further actions if needed. Actions may include changes to the sampling rate and/or communication rate, collecting data from other sensors, or triggering alarms, among others.

## IV. Experimental Results

A proof-of-concept implementation of the proposed architecture was done, in order to support some initial assessment of the proposal. The first two sets of experiments targeted the behavior of a local controller and a main controller, according to the scenario represented in Figure 3.

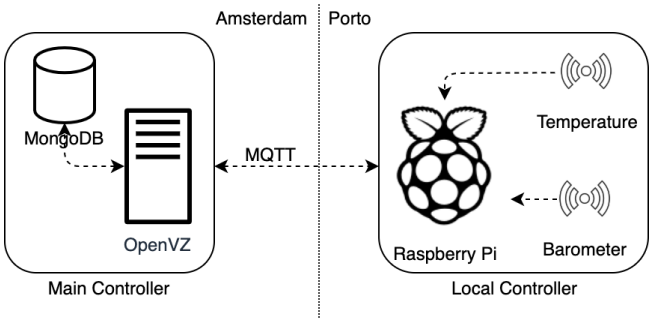



Fig. 3: Test scenario.

In the test scenario, communication was performed using the Message Queuing Telemetry Transport (MQTT) protocol, with the local controller located in Porto, Portugal, and the Main controller in Amsterdam, Netherlands. The main controller was a container-based virtualization (OpenVZ) server, running Ubuntu 16.04 and with a shared 1x3.5GhZ core, 256 Mb DDR3 RAM and 30 GB of SSD storage. A MongoDB database and MQTT Mosquitto broker were used to store and manage data, respectively. For the local controller, a Raspberry Pi 2 Model B, with a 900MHz quad-core ARM Cortex-A7, 1Gb of Ram and an 8Gb Micro SD Card, with NOOBS

version 3.2.1. The used sensor was the SunFounder BMP180, which contains a barometer and a temperature sensor, with an accuracy of 0.12hPa and 0.5C, respectively. According to the documentation[1], it can go up to 128 samples per second, although this highly depends on the used software/library. In both systems, Python scripts were written, making use of Eclipse-Paho and Pymongo to send/receive information from the MQTT broker and database, respectively, which decreased the number of samples per second, reaching a maximum of 10, for the temperature sensor, and 5, for the pressure sensor.

### A. Local Controller Measurements

The first set of experiments had the objective of assessing the behavior of the local controller while it was collecting and sending data to the remote main controller over the Internet. In these experiments, one sensor was used, with sampling rates of 1, 2, 5, and 10 samples per second. A message with the collected data was sent to the main controller after collecting 50 temperature readings. A total of 20 runs were performed for each sampling rate.

Figure 4 presents the obtained results, for the various sampling rates, concerning the total time to send the collected 50 sensor readings, the CPU usage, and the RAM usage, measured in the local controller. As explained, the highest sampling rate was 10 due to limitations on the sensor. The figure shows the average of the 20 runs, along with the standard deviation.

In order to determine if there was any significant difference between the results obtained for each of the four sampling rates, an analysis of variance (ANOVA) test was performed. We considered a single factor, with an alternative right-tail probability (alpha) of 0.05 with the following hypothesis:

- Null Hypothesis ($H_0$) - the average of the groups is the same.
- Alternative Hypothesis ($H_1$) - the average is not the same for all the groups.

For the case of the time to send (Figure 4a), the ANOVA test returned an F-test of 1.58, which was less than the critical F value (which is 2.866). This indicates that $H_0$ is accepted and, thus, there are no significant differences between the groups. Regarding CPU usage (Figure 4b), the same statistical test was performed, and it returned 0.71, which is also lower than the critical value, thus accepting $H_0$. Finally, concerning RAM usage, the result of the F-test is 8.14, which is greater than the critical value. This indicates that $H_0$ is rejected and, thus, $H_1$ is accepted. Although all the tests were performed under the same conditions, there are operating system processes that cannot be stopped, which can occupy memory on the device. Moreover, it is difficult to control how RAM is used in these devices, especially when they automatically free the memory. By looking at Figure 4, there is a difference between the test with 1 sample per second and the other 3 tests (in what concerns both the average and the standard deviation). This is due to the fact that, during the test, the device freed memory, resulting in a decrease from around 77% to 58%. However, by looking at the values in Figures 4a and 4b, it is possible observe that the higher variance on the RAM usage did not affect the performance of the local controller.

[1]https://cdn-shop.adafruit.com/datasheets/BST-BMP180-DS000-09.pdf - last consulted on January 9th, 2020

### B. Main Controller Event Triggering

Having received data from a local controller, the main controller must process in order to extract some information and/or trigger actions. The second set of tests had the objective of demonstrating the ability of the main controller to continuously analyze received data and trigger events based on it. At this point in time, we were not concerned with what to do once an event was triggered, as this is highly dependent on the policies and objectives defined at the Application Plane.

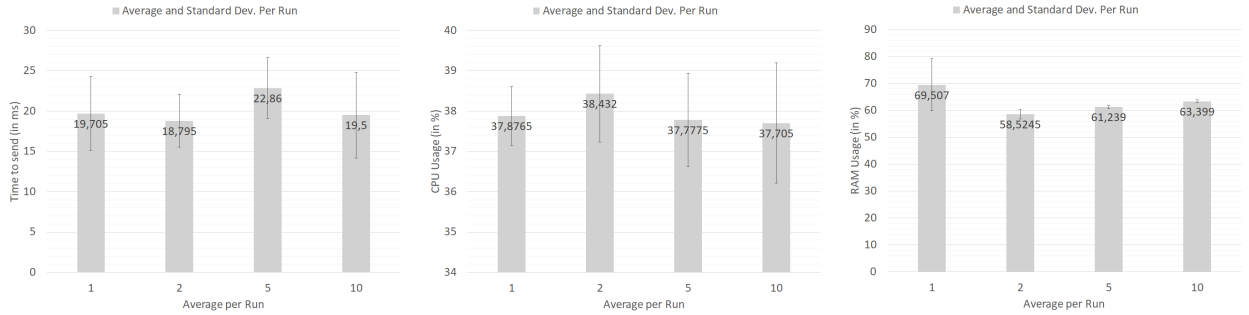
As in the previous case, in this set of experiments the temperature sensor was considered, with 50 values per run and a total of 20 runs. We considered a sampling rate of 1 sample per second only, as this was enough to demonstrate the operation of the event triggering functionality. For the analysis, the average temperature of a certain run ($T_r$) was compared with the average temperature of the previous run ($T_{r-1}$) and, if the absolute value of the difference exceeded 0.5 a trigger was generated, either positive or negative, depending on the sign of the difference. Figure 5 presents the measured temperature and the generated triggers, showing that the system operated as expected.

## V. Conclusion and Future Work

In this paper we proposed a new smart railway maintenance architecture that builds on concepts similar to the ones that underlie software-defined networking. With this approach, we open the way to developing SRM systems that are programmable, scalable, and open, and that are able to orchestrate large numbers of devices, independently of their physical details, in an efficient way. The proposal was subsequently subject to prototyping, with the objective of assessing not only its feasibility, but also some of its characteristics. To this effect, results on the performance of a local controller when communicating with a remote main controller over a cloud environment, and on the ability for triggering events, were obtained, showing that the approach is both feasible and effective. Naturally, these were only preliminary results that will be complemented in the near-to-mid-term future with functionally rich implementations, exploring failure prediction techniques and real-time anomaly detection. Moreover, future work will continue to explore the proposed approach by further specifying the southbound and northbound interfaces, and by detailing the general architectures of local controllers and main controllers.

## Acknowledgment

This work was carried out in the scope of the MobiWise project: From mobile sensing to mobility advising (P2020 SAICTPAC/0011/2015), co-financed by COMPETE 2020, Portugal 2020 - POCI, European Union's ERDF.




(a) Time to send results. (b) CPU usage results. (c) Ram usage results.

Fig. 4: Results obtained in the local controller, for 1, 2, 5 and 10 samples per second, for a) time to send, b) CPU usage, and c) RAM usage.

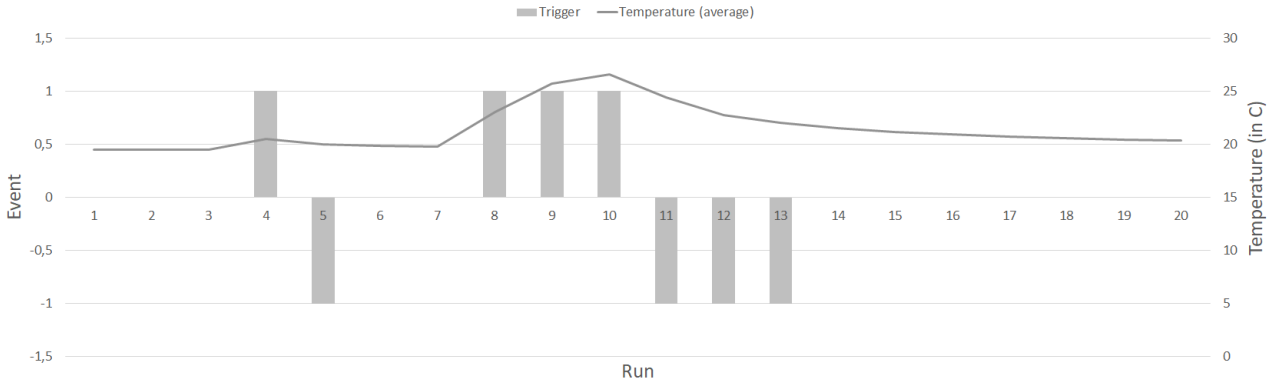



Fig. 5: Temperature-based event triggering at the main controller.

## REFERENCES


[1] A. Nordrum and K. Clark, "Everything you need to know about 5g," Jan. 2017. [Online]. Available: https://spectrum.ieee.org/video/telecom/wireless/everything-you-need-to-know-about-5g

[2] V. Rikhotso, N. Steyn, and Y. Hamam, "3D rail modelling and measurement for rail profile condition assessment," in *2017 IEEE AFRICON: Science, Technology and Innovation for Africa, AFRICON 2017*. IEEE, Sep. 2017, pp. 1522–1527. [Online]. Available: http://ieeexplore.ieee.org/document/8095708/

[3] V. R. Vijaykumar and S. Sangamithirai, "Rail Defect Detection using Gabor filters with Texture Analysis," *International Conference on Signal Processing, Communication and Networking*, vol. 3, pp. 1–6, Mar. 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7219838/

[4] H. Feng, Z. Jiang, F. Xie, P. Yang, J. Shi, and L. Chen, "Automatic fastener classification and defect detection in vision-based railway inspection systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 4, pp. 877–888, Apr. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6630109/

[5] J. Gan, J. Wang, H. Yu, Q. Li, and Z. Shi, "Online Rail Surface Inspection Utilizing Spatial Consistency and Continuity," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–11, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8354938/

[6] S. Lu, Z. Liu, and Y. Shen, "Automatic Fault Detection of Multiple Targets in Railway Maintenance Based on Time-Scale Normalization," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 4, pp. 849–865, Apr. 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8268578/

[7] C. Ulianov, F. J. Franklin, P. Hyde, and R. Shaltout, "Experimental investigation of wheel-track interaction characteristics using test track bogie dynamics measurements," in *Proceedings of 2016 IEEE International Wheelset Congress, IWC 2016*. IEEE, Nov. 2017, pp. 117–121. [Online]. Available: http://ieeexplore.ieee.org/document/8068379/

[8] Z. Liu, L. Wang, C. Li, and Z. Han, "A High-Precision Loose Strands Diagnosis Approach for Isoelectric Line in High-Speed Railway," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1067–1077, Mar. 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8114270/

[9] G. Lederman, S. Chen, J. Garrett, J. Kovačević, H. Y. Noh, and J. Bielak, "Track-monitoring from the dynamic response of an operational train," *Mechanical Systems and Signal Processing*, vol. 87, pp. 1–16, Mar. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888327016302230

[10] G. Lederman, S. Chen, J. H. Garrett, J. Kovačević, H. Y. Noh, and J. Bielak, "A data fusion approach for track monitoring from multiple in-service trains," *Mechanical Systems and Signal Processing*, vol. 95, pp. 363–379, Oct. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888327017301516

[11] M. Bocciolone, A. Caprioli, A. Cigada, and A. Collina, "A measurement system for quick rail inspection and effective track maintenance strategy," *Mechanical Systems and Signal Processing*, vol. 21, no. 3, pp. 1242–1254, Apr. 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888327006000434

[12] M. Pau and B. Leban, "Experimental detection of wheel-rail contact irregularities," *Proceedings of the 7th World Congress on Railway Research (WCRR)*, pp. 4–8, 2006. [Online]. Available: http://www.railway-research.org/IMG/pdf/133.pdf

[13] D. L. Goodman, J. Hofmeister, and R. Wagoner, "Advanced diagnostics and anomaly detection for railroad safety applications: Using a wireless, IoT-enabled measurement system," in *AUTOTESTCON (Proceedings)*, vol. 2015-Decem. IEEE, Nov. 2015, pp. 273–279. [Online]. Available: http://ieeexplore.ieee.org/document/7356502/

[14] N. Jamaly, S. Mauron, R. Merz, A. Schumacher, and D. Wenger, "Delivering Gigabit Capacities to Passenger Trains: Tales from an

Operator on the Road to 5G," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 18–23, sep 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8847220/

[15] J. Talvitie, T. Levanen, M. Koivisto, T. Ihalainen, K. Pajukoski, and M. Valkama, "Positioning and Location-Aware Communications for Modern Railways with 5G New Radio," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 24–30, sep 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8847221https://ieeexplore.ieee.org/document/8847221/

[16] D. He, B. Ai, C. Briso-Rodriguez, and Z. Zhong, "Train-to-Infrastructure Channel Modeling and Simulation in MmWave Band," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 44–49, sep 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8847225/

[17] Y. Liu, C.-X. Wang, and J. Huang, "Recent Developments and Future Challenges in Channel Measurements and Models for 5G and Beyond High-Speed Train Communication Systems," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 50–56, sep 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8847226/

[18] K. Wu, W. Ni, T. Su, R. P. Liu, and Y. J. Guo, "Recent Breakthroughs on Angle-of-Arrival Estimation for Millimeter-Wave High-Speed Railway Communication," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 57–63, sep 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8847227/