applied sciences

MDPI

Article

Railway Cloud: Management and Orchestration Functionality Designed as Microservices

Ivaylo Atanasov [1], Evelina Pencheva [2,*], Ventsislav Trifonov [2] and Kiril Kassev [1]

[1] Faculty of Telecommunications, Technical University of Sofia, 1756 Sofia, Bulgaria; iia@tu-sofia.bg (I.A.); kmk@tu-sofia.bg (K.K.)
[2] Faculty of Telecommunications and Electrical Equipment in Transport, Todor Kableshkov University of Transport, 1574 Sofia, Bulgaria; vencislavtrifonov@gmail.com
* Correspondence: evelina.nik.pencheva@gmail.com

Abstract: The cloudification and virtualisation of railway functions have the potential to improve railway operation efficiency, reliability, safety, and security, as well as to enhance passenger experience by offering innovative services. This paper considers issues related to the management and orchestration of railway clouds that host cloudified railway functions. A microservices-based approach to the design of railway cloud management and orchestration functionality is proposed. The basic railway cloud concepts were defined, and functionality related to the basic orchestration of the railway cloud and deployments is analysed in order to derive the requirements of platform resources and workload management. This functionality is further designed in the form of microservices, meaning that they could possibly be used in orchestration applications to enable improvements in scalability, fault isolation, and data security. The design of microservices follows the principles of the Representational State of Transfer (REST) application programming interface (API) as a set of interlinked resources. Resources related to railway cloud orchestration are identified with their associated data, relationships to other resources, and applicable methods. The resources' methods are used in railway applications to implement the required orchestration functionality and to maintain the state of railway cloud orchestration processes. To verify the synthesised microservices, the common orchestration application logic and microservices' logic were modelled, and it was proved that the orchestration processes, which run concurrently, expose equivalent behaviour. The proposed approach was validated using a simulation, aiming to evaluate injected latency as a key performance indicator for the reliability and safety of railway operations. Additionally, some safety and security issues related to railway cloud management and orchestration are considered.

Keywords: railways; cloud computing; management and orchestration; microservices; concurrent process modeling; key performance indicators

check for updates

Citation: Atanasov, I.; Pencheva, E.; Trifonov, V.; Kassev, K. Railway Cloud: Management and Orchestration Functionality Designed as Microservices. *Appl. Sci.* **2024**, *14*, 2368. https://doi.org/10.3390/app14062368

Academic Editors: Borna Abramović and Piotr Gorzelańczyk

Received: 10 February 2024
Revised: 7 March 2024
Accepted: 7 March 2024
Published: 11 March 2024

1. Introduction

Digitalisation is shaping the transformation of railway transport, and it is essential for building sustainable economies and lifecycles. Mission critical communications and cloud technologies are accelerating the railway journey into the realms of digitalisation and sustainability. Fifth-generation (5G) communications, artificial intelligence (AI), cloud computing, and virtualisation technologies empower railways with new levels of connectivity, safety, reliability, and sustainability.

Using a 5G cloud native railway infrastructure and virtualisation of railway functions enables improvements in operation efficiency, lower costs, and the fast deployment of innovative applications. The provision of ubiquitous and uninterrupted connectivity using 5G ultra-reliable low-latency communications and the support of the Future Railway Mobile Communication System (FRMCS), defined by the International Union of Telecommunications, is expected to increase the level of automatic train operation, enabling a good level of security for passengers and staff, and improving service punctuality and the passenger experience [1–5].

AI methods have huge potential to solve practical problems in the railway industry related to train automation, the maintenance of track and catenary systems, the monitoring of civil structures, etc. [6–9]. There is an increasing number of contributions in the literature applying AI methods in railway operations, such as using renewable energy and smart grid systems [10], digital twins for smart railways [11,12], automatic train operation [13,14], trackside predictive maintenance operations [15–17], the dynamic forecasting of train dwell time [18], transport system modelling [19], etc.

Cloud computing could accelerate the digital journey of railways by gathering operational information, such as train speed and location and track and meteorological conditions, and by sending control information, which is critical to life, in real time to the onboard computer in the train cab. This could ensure safer railway control systems and stable operations. An analytic cloud-based system combined with real-time monitoring can reduce unexpected operation interruptions, improve the performance of railway traffic management, and contribute to more reliable, safe, and passenger-oriented services [20,21]. By shifting railway applications into a cloud paradigm, huge sets of diverse operational data reflecting physical asset states could be stored, processed, and analysed in real time to identify anomalies, potential damages, or even hazardous conditions. With its computing power, cloud-based railway platforms provide scalable and flexible solutions for the processing of colossal and complicated datasets related to the monitoring of train movements, maintenance, and traffic forecasting. With advances in computing technologies such as multi-core processors, the embedded systems in railways could be virtualised. Railway functions that could be virtualised include train signalling and control, monitoring and surveillance, diagnostics and maintenance, passenger information systems, ticketing, and communication systems. Virtualisation provides a flexible solution for all types of railway applications, enabling upgrades with minimal effort [22,23]. Virtualisation, with its modular software architecture, could contribute to improvements in the safety of mission critical functions. Different tasks can be executed independently of each other, having access to their own partition and own sets of resources, enabling the assignment of different trust levels and the application of appropriate security policies. The real-time requirements of safety-critical functions can be met by the assignment of the necessary computing resources.

In the authors' previous research, a cloud-based railway control system architecture, which enables the use of AI and function virtualisation, was proposed [24,25]. The main advantages of this proposed architecture are its modular structure and the decoupling of time-sensitive control functions from time-tolerant control functions.

In this paper, more details of the railway cloud platform are discussed, and cloud management functionality is considered. Also, an approach to design microservices for railway cloud orchestration is presented.

The remainder of this paper is organised as follows. The next section presents works related to railway cloud architecture, management, and applications and highlights the novelty of this paper. Section 3 briefly describes the proposed cloud-based railway control system architecture, with an emphasis on the railway cloud platform and its management functionality. Section 4 is devoted to explaining railway cloud management use cases, which are the basis for synthesising the functionality of management microservices. Section 5 presents the proposed approach for the design of railway cloud orchestration microservices, including the verification and validation of microservices. Some issues related to safety and security in cloudified railway systems are discussed in Section 6. Finally, the conclusion summarises the benefits of this approach and points out the inherent disadvantages of microservices that cloud be mitigated through proper planning and management.

2. Related Works

Research related to railways' cloudification can be classified into three areas: the employment of edge clouds related to the optimisation of offloading real-time control onto edge clouds; cloud computing platform's architectures concerning resilient and secured frameworks embedding intelligent workflows and virtualisation technologies; and cloud-based railway applications for automatic train supervision, track monitoring and surveillance, fault diagnostics, and ticketing systems.

Real-time monitoring in railways provides accurate support in terms of fault management, enabling the generation of valuable and actionable insights, and edge computing is a good method used to perform analytics close to data sources. In [26], the authors propose a method for train predictive control, which offloads computations related to real-time control onto edge services, while a deep learning-based algorithm for trajectory prediction is executed on a centralised server. In [27], an approach to optimise railway logistics based on mobile edge cloud is proposed. The authors of [28] exploit the benefits of multi-access edge computing to offload train tasks to edge servers using blockchain technology for authenticity. An edge computing-based solution for the problem of the processing of mass data, gathered from intelligent railway station equipment, is presented in [29].

Moving the tasks of railway operation and maintenance to the cloud sets strict requirements in terms of reliability, safety, and security. The resilience requirements of the cloud for its use in critical railway systems are considered in [30]. The capabilities of the cloud to deploy virtualised real-time, safety-critical functions to commercial off-the-shelf products are studied in [31]. To cope with problems related to diverse logistic information across borders, in [32], the authors propose a hybrid cloud with three layers of services. A cloud-based architecture of an intelligent operation and maintenance system in railways is presented in [33], with focus on the respective workflows. In [34], the authors study the capabilities of building cloud computing platforms for railways with open-source products and propose a high-level security architecture for the cloud. In [35], the authors present the idea of cloud-based supervisory control systems for urban railway transport and identify the related functionality, which could be synthesised as microservices. However, the authors do not provide any details on the design of microservices and do not discuss cloud management aspects. A study on security architectures and key technologies for railway clouds is provided in [36].

Most of the existing research focused on railway clouds is devoted to specific cloud-based railways applications [37]. These applications cover automatic train supervision [38], virtual coupling [39], track surveillance and monitoring [40,41], fault detection in railway infrastructure components [42], predictive maintenance [43], smart ticketing systems [44,45], train movements prediction [46], operation and maintenance of rail power supply systems [47], and security-related issues [48,49].

Table 1 summarizes some of the recent works concerning problems related to the aspects of the railway cloud.

Table 1. Summary of works concerning problems related to the railways cloud aspects.

Area	References	Considered Problems
Railway edge clouds	[26–29]	Optimization of offloading real-time control on edge clouds
Cloud computing platform architectures	[30–36]	Resilient and secure frameworks embedding AI and intelligent workflows, virtualization technologies
Cloud-based railway applications	[37–49]	Automatic train supervision, track monitoring, fault diagnostics, track surveillance, secure communications, ticketing systems

The management of industrial clouds comes with a set of problems related to data security and privacy, performance and reliability, cloud migration, interoperability, optimization of management costs, and required skills and knowledge.

The management of industrial clouds comes with a set of problems related to data security and privacy, performance and reliability, cloud migration, interoperability, optimisation of management costs, required skills, and knowledge.

In [50], the authors provide a study on industrial multi-cloud management platforms, analysing the difficulties and challenges of building such architectures. In [51], innovative approaches to resource allocation and performance optimisation techniques in a multi-cloud environment are explored. A blockchain-based multi-cloud infrastructure that unifies the management of clouds provided by different vendors is presented in [52]. In [53], the principles of cloud management required for optimal cloud resource allocation and the migration of virtual machines are stressed. In [54], the authors provide a survey on cloud resource management by comparing different cloud computing algorithms. Problems related to the balance between cloud performance and power consumption and existing management approaches are discussed in [55]. In [56], the author proposes a hybrid approach to secure cloud resource allocation based on machine learning and task scheduling. A cloud resource management strategy is discussed in [57], which is an auction mechanism and improves traditional resource allocation strategies. The problems related to data recovery and backup in cloud computing are discussed in [58]. In [59], the authors present a review of the research in cloud computing security, point out threats and challenges, and propose ideas for resolving cloud data integrity. Based on the analysis of existing works on cloud computing security, in [60], the authors identify associated challenges and propose some recommendations. In [61], a layered security performance management model is studied. In [62], the authors propose a method for the estimation of access to cloud data and a new security access control scheme.

Table 2 summarises the latest research on cloud management.

Table 2. Summary of the recent research on cloud management.

Cloud Management Aspects	Reference Numbers	Considered Problems
Multi-cloud platform management	[50–52]	Architectures, performance optimization, unified management
Cloud resource allocation	[53–58]	Principles, algorithms, performance, data recovery and backup
Cloud security	[59–62]	Threats, challenges, recommendations, access control methods

The variety of works conducted on railway edge clouds and cloudified railway applications do not concern the interfaces between clouds and the rest part of the railway control system. They focus on control functions that could be cloudified and on the benefits of user proximity. The mechanisms of interaction between edge rail applications and managed railway objects are not refined. Works related to cloud computing platform's architectures consider resilient and secured frameworks that make it possible to embed intelligent and virtualised functions which stress reliability, safety, and security requirements. Railway cloud management, being a unified, open, and multi-vendor solution, seems to be an open research question.

The existing research related to industrial clouds focuses on multi-cloud interoperability and cloud resource management issues. The proposed cloud management architectures provide high-level views without elaborating on management functionality, generic cloud management services, management functions, and numerous elements. Furthermore, the interworking between cloud services management and controlled components, as a complex of interfaces and procedures, is left without discussion or in-depth typical use cases. Interworking implies an increase in complexity; however, this must be achieved without compromising the vital key performance indicators.

This literature review shows that there is a lack of unified open cloud management platforms that meet the strict requirements of railway operators for ultra-high reliability, safety, and security.

In this paper, novel railway cloud functions and a unified management interface are proposed. The presented cloud-based railway control architecture enables a railway operator to both use clouds from different providers and manage them uniformly. The suggested cloud management functionality is synthesised as services for the orchestration of railway cloud life cycle processes based on open interfaces. Open interfaces expose improved capacity, high reliability, and better service provisioning in comparison with any proprietary ones.

The intelligent railway control system architecture presented in [24] aims to enhance resiliency, scalability, automation, interoperability, innovation fostering, unified management, and enhanced security. The architecture is cloud-based and enables hardware and software disaggregation, eliminates the need for proprietary hardware, and mitigates the vendor-dependency risks. The cloudification of railway functions fosters scalability, enabling the deployment of flexible and modular components. The architecture allows for embedding intelligence in railway operation and management, incorporating machine learning (ML) framework, and providing an environment to run both non-real-time and near-real-time AI applications. It has the potential to improve interoperability as railway operators can manage different components based on open application programming interfaces (APIs), which also reduces complexity and fosters innovation. The focus in [24] is on the disaggregation of control functionality related to the automation of railway functions and the open interface between time-tolerant and time-sensitive control.

In [24], the intelligent control functionality is elaborated on, and a microservices-based approach to the design of AI/ML lifecycle procedures is proposed. Microservices' functionality includes the discovery of the capabilities of the ML model and inference host, ML model training, selection, deployment, inference, performance monitoring, retraining updates, and termination.

In this paper, the focus is on railway cloud as an important component of an intelligent railway control system architecture and its interface towards achieving the management, automation, and orchestration of a platform. This paper's novelty lies in our proposed approach to design railway cloud management and orchestration functionality as microservices. Our research findings will hopefully enable railway operators to develop their own Mobility as a Service (MaaS) applications that offer customised mobility solutions tailored to the user's individual needs [63].

The research methodology includes an analysis of basic railway cloud orchestration use cases, the identification of railway cloud management functionality, the synthesis of railway cloud orchestration microservices, the verification of microservices, and the evaluation of microservice performance indicators, which are all critical for creating railway operations that are reliable and safe. The analysis of railway cloud orchestration includes cases, with the goal of deriving the requirements of railway cloud management functionality. Based on these requirements, the basic functionality required for railway cloud provisioning is identified and synthesised as services. As microservice design follows the Representational State Transfer (REST) principles, the basic service resources are identified and organised in tree structures, the methods that allow for resource manipulation are determined, and data types and structures are modelled. The verification of microservices is conducted by modelling the common logics of orchestration applications and microservices, which must be synchronised. The last step of this research methodology aims to demonstrate the benefits of the proposed approach for fulfilling strong railway requirements by evaluating latency, as one of the key performance indicators, which is injected by the microservices' application programming interfaces.

3. Railway Cloud Basic Concepts, Management, and Orchestration Services

A railway cloud platform may be regarded as a cloud computing platform that meets the requirements to host the relevant cloudified railway functions and supporting software components. It is required to expose the appropriate management and orchestration functions through service-based interfaces. A railway operator may employ clouds from different providers, but they need to manage them in a uniform way. The railway cloud platform hardware is composed of computing, storage, and networking resources grouped in one centralised or more distributed railway cloud resource pool. The railway cloud platform software, including virtualisation software, services, and management tools, is decoupled from the hardware and exposes open API, which improves flexibility and mitigates dependency on a single vendor.

Railway cloud resources are organised in resource pools. A railway cloud node, or simply a node, is a logical or physical server with assigned cloud resources and an operating system. Several nodes form a group. One or more groups form railway cloud clusters with the aim of achieving the required availability. Within a group, the nodes are interconnected in a group network. Within a railway cloud cluster, communication is based on a cluster network, which unites all group networks included in it. One or more railway cloud clusters can be installed at a given location (site). A united network exists for the site clusters. For each site, there is a gateway from one or more network providers to enable communication across the whole railway cloud. A specific role is assigned to each node. Some of the nodes may be dedicated to cloud management functions, whereas other nodes may be exposed to functions that manage the resources assigned to the cloudified functions, and there may be some nodes that host the cloudified railway functions.

The Railway Management Automation and Orchestration (RMAO) platform, as an integral part of the intelligent railway control system, is responsible for the management and orchestration of resources and services. It is responsible for service provisioning and lifecycle management, resource orchestration, and service assurance. The management interface between the RMAO platform and the railway cloud platform (RCP) needs to support two types of services:

- Cloud resource management (CRM) services which are responsible for the allocation and delivery of cloud resources and resource management software, i.e., services that orchestrate railway cloud lifecycle processes, e.g., cloud deployment services, cloud infrastructure catalogue services, railway cloud monitoring services, railway cloud provisioning services, etc.
- Cloudified function management (CFM) services which manage the lifecycles of the cloudified railway functions deployed on the railway cloud.

Figure 1 shows the basic concepts and services related to the railway cloud.

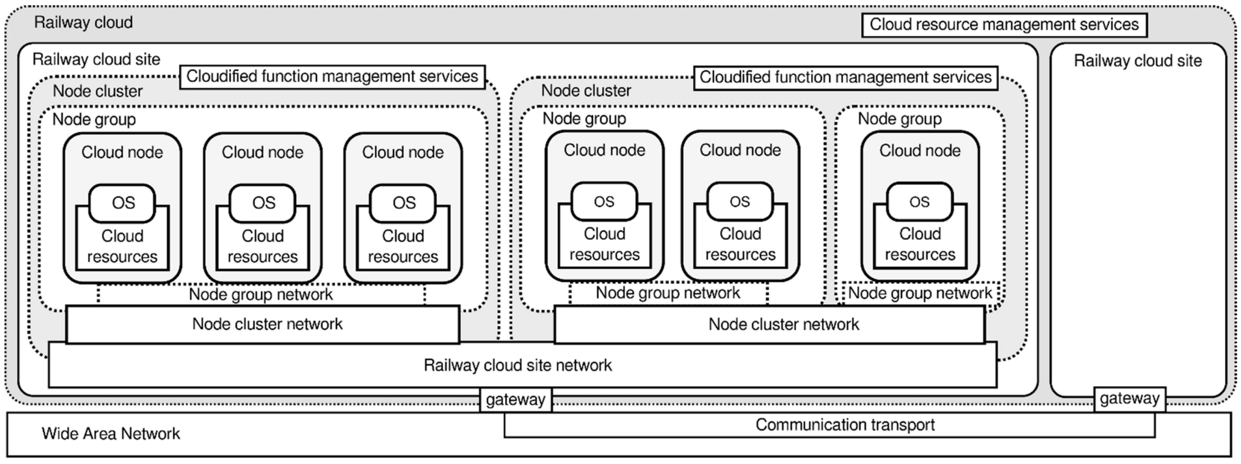

Figure 1. High-level railway cloud concepts and services.

The basic orchestration of the railway cloud is related to the railway cloud's provisioning. The provisioning of the railway cloud includes the railway cloud's deployment and the integration of cloud services within the railway infrastructure. It assumes that the basic infrastructure (e.g., physical resources) is built and installed and that secured connectivity between the components of the intelligent railway control systems exists. The focus is on the deployment of railway cloud services and resource allocation.

The following use cases represent well-known steps in the basic orchestration of the railway cloud, as show in Figure 2:

- *Railway cloud pre-deployment processing*. The RMAO creates a record associated with the new railway cloud in its catalog.
- *Railway cloud platform software installation*. The RMAO initializes the railway cloud, enabling communication with the RCP.
- *Railway cloud registration and initialization*. The RMAO initializes and registers the railway cloud and prepares it for installing the cloudified railway functions.
- *Railway cloud post-deployment hardware infrastructure scaling*. The RMAO powers on and initializes additional railway cloud nodes for an existing railway cloud.
- *Railway cloud platforms software update*. The RMAO upgrades or downgrades the platform software of an existing railway cloud.
- *Functional status query and update*. The RMAO queries and updates of the functional status of the railway cloud resources.
- *Update of CRM software*. The RMAO updates the railway cloud's CRM service software.
- *Update of CFM software*. The RMAO updates the railway cloud's CFM service software.

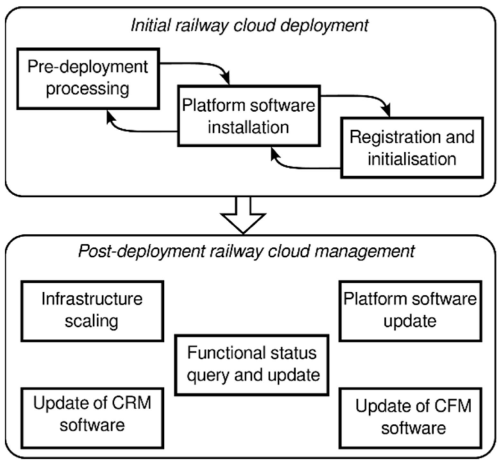
Initial railway cloud deployment
Pre-deployment processing
Platform software installation
Registration and initialisation
Post-deployment railway cloud management
Infrastructure scaling
Platform software update
Functional status query and update
Update of CRM software
Update of CFM software

Figure 2. Basic railway cloud's orchestration use cases.

4. Railway Cloud Orchestration Use Cases

A railway operator may have multiple edge railway clouds to face the requirements of real-time functionality, and in this case, the railway clouds' deployment must be performed by the RMAO platform. Railway cloud instantiation and deployment are triggered by the RMAO platform and include pre-deployment processing, platform software installation, registration, and activation.

Railway cloud pre-deployment processing is required to configure the RMAO platform to add a railway cloud into its catalogue prior to the activation and registration of the railway cloud. The process is initiated by the cloud manager, and the RMAO platform generates a unique identification of the new cloud instance and creates a catalogue with the railway cloud ID. Then, the new railway cloud nodes are installed, and connectivity is setup by the personnel; afterwards, the RMAO is notified and provided with information to connect to the built railway cloud, including the target node and identity of the basic software to be loaded onto the railway cloud. The RMAO uploads the railway cloud's basic software on the first railway cloud node, which in turn raises the railway cloud to additional railway cloud nodes. The CRM services in the RCP load the railway cloud's basic software onto other nodes, activate CFM services, and create the software catalogue.

Figure 3 shows the high-level interaction between the RMAO platform and the RCP during the pre-processing of the railway cloud and installation of the platform software.

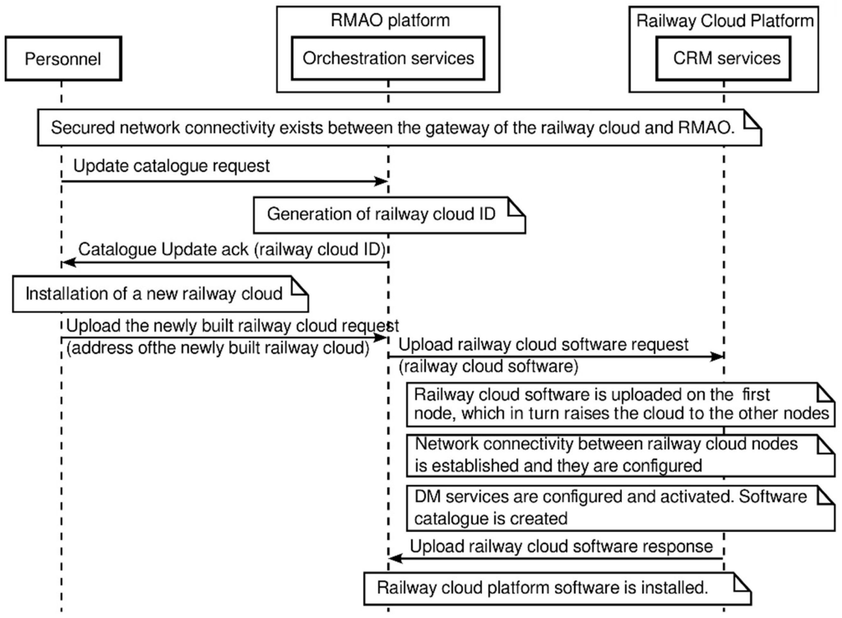
RMAO platform
Railway Cloud Platform
Personnel
Orchestration services
CRM services
Secured network connectivity exists between the gateway of the railway cloud and RMAO.
Update catalogue request
Generation of railway cloud ID
Catalogue Update ack (railway cloud ID)
Installation of a new railway cloud
Upload the newly built railway cloud request (address ofthe newly built railway cloud)
Upload railway cloud software request (railway cloud software)
Railway cloud software is uploaded on the first node, which in turn raises the cloud to the other nodes
Network connectivity between railway cloud nodes is established and they are configured
DM services are configured and activated. Software catalogue is created
Upload railway cloud software response
Railway cloud platform software is installed.

Figure 3. High-level flow of railway cloud platform pre-processing and installation.

The registration and initialisation of the railway cloud include the update of the CRM services' software catalogue, if required, configuration of the railway cloud, discovery of new CFM services and their capabilities, and registration of them in the railway cloud catalogue.

The update of the railway cloud catalogue is related to the renovation of information about the railway cloud type and its capabilities, e.g., resource configuration, capacity, allocations, applications, and availability. It is triggered upon an initial railway cloud deployment or later when changes are required. The scaling of the hardware infrastructure of the railway cloud is related to resource scale-up or -down and it is triggered, e.g., when a decision is made to install and register a new railway cloud node. In this case, the railway cloud catalogue update procedure is used.

The railway cloud platform's software update must ensure service continuity. As a precondition, the cloudified railway functions, deployed on the affected edge railway clouds, need to be migrated to other railway clouds. Within the update process, CRM services download and validate the software, perform a pre-check procedure to determine whether the software is acceptable or not, update the software's images, perform a restart, if required, and update the software catalogue. In case of software update failure, a roll-back procedure to the working software version takes place.

The update of CRM and CFM services' software includes software download, pre-checking, modification, and testing. To provide more details on upgrading or downgrading the software of the CFM services, this use case is elaborated as follows and illustrated in Figure 4.

The CFM services' software update process is triggered by the railway cloud manager, who requests the RMAO platform to update the software version within a railway cloud. To synchronise CFM services' software, the RMAO platform queries the CRM services in the RCP for its current software catalogue, and if the software catalogue is not the required one, the RMAO platform provides information about the software version. If the CRM services obtain the requested software version, then the RMAO platform is notified that the software version is available; otherwise, the RMAO platform is notified that the software update is unsuccessful. The deployment of the CFM services' software images begins with pre-checking, which is requested by the RMAO platform. If the pre-check procedure shows that the requested software version is acceptable, the RMAO platform is notified and, in turn, it instructs the CRM services to install the new CFM services' software. The CRM services allocate the necessary resources, install the requested software version, and perform specific testing. In the case of a successful software update, the CRM services update their software catalogue and notify the RMAO. Any failure during the deployment of the CFM services' software is reported to the RMAO. The software update process terminates with a notification to the railway cloud manager about the result.

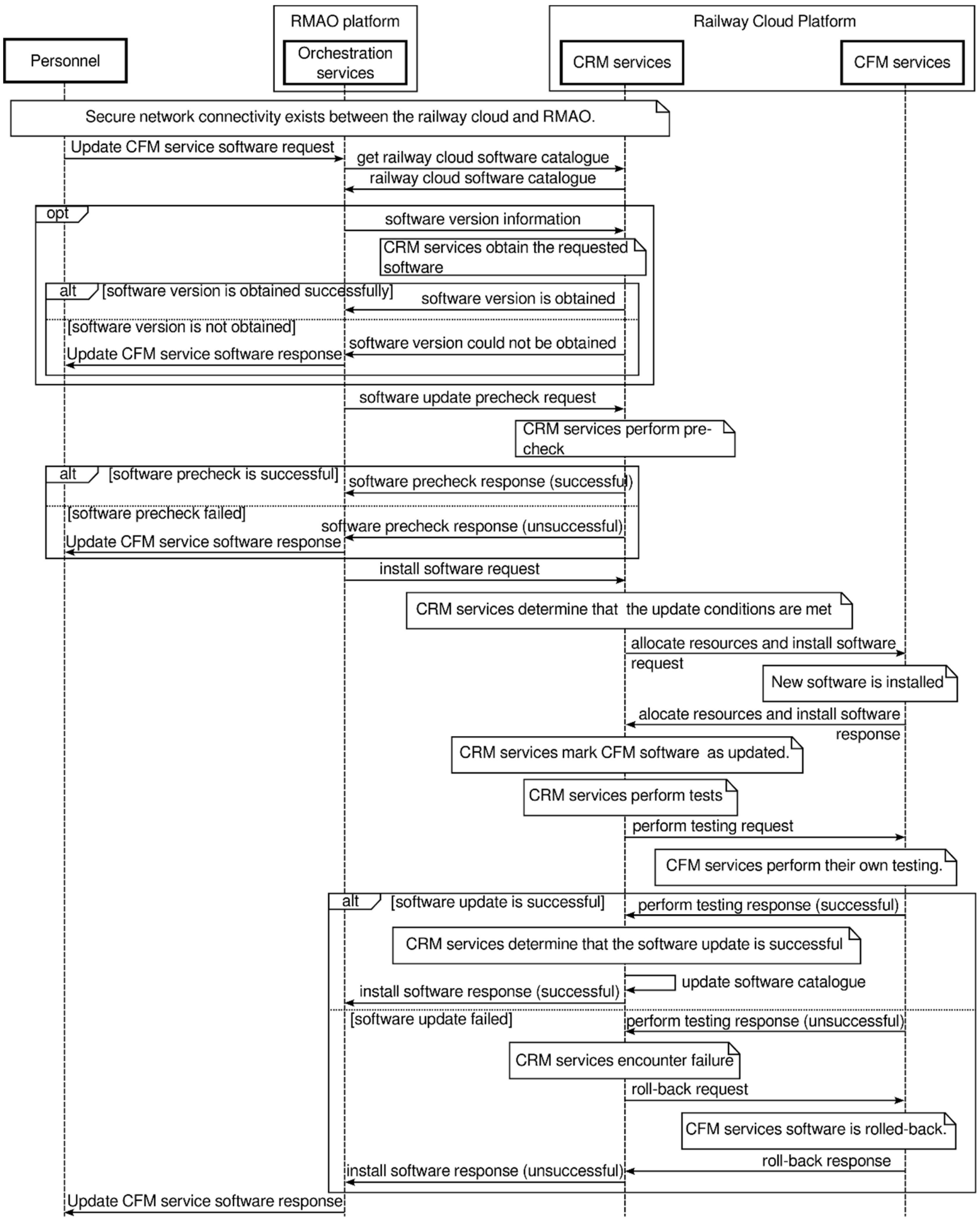
RMAO platform
Railway Cloud Platform
Personnel
Orchestration services
CRM services
CFM services
Secure network connectivity exists between the railway cloud and RMAO.
Update CFM service software request
get railway cloud software catalogue
railway cloud software catalogue
opt
software version information
CRM services obtain the requested software
alt [software version is obtained successfully]
software version is obtained
[software version is not obtained]
software version could not be obtained
Update CFM service software response
software update precheck request
CRM services perform pre-check
alt [software precheck is successful]
software precheck response (successful)
[software precheck failed]
software precheck response (unsuccessful)
Update CFM service software response
install software request
CRM services determine that the update conditions are met
allocate resources and install software request
New software is installed
alocate resources and install software response
CRM services mark CFM software as updated.
CRM services perform tests
perform testing request
CFM services perform their own testing.
alt [software update is successful]
perform testing response (successful)
CRM services determine that the software update is successful
update software catalogue
install software response (successful)
[software update failed]
perform testing response (unsuccessful)
CRM services encounter failure
roll-back request
CFM services software is rolled-back.
roll-back response
install software response (unsuccessful)
Update CFM service software response

Figure 4. Flow of CFM services' software update.

Based on the analysis of basic use cases of railway cloud orchestration, requirements could be derived. As preconditions, the railway cloud hardware must be installed, pre-configured and ready, and a stable connectivity must exist. The basic requirements related to the orchestration of the railway cloud are as follows:

- The RMAO needs to assign a unique identifier to a newly deployed railway cloud that will be used for handling the cloud registration process.
- The RMAO needs to load the railway cloud platform software on the first node.
- The CRM services need to load the basic railway cloud software on the other nodes to configure and activate the CFM services and to create/refresh the software catalogue.
- The RMAO needs to be able to query the CRM services about the software catalogue to configure the CFM services, if required, and to register the CFM services in its catalogue.
- The RMAO should be able to update information in the railway cloud catalogue (e.g., resource configurations, capacity, utilisation, etc.) or to add information about a newly installed node.
- The RMAO should be able to scale the railway cloud hardware infrastructure.
- The RMAO should be able to update the railway cloud platform software by triggering the software download and validation procedure.
- The CRM services need to be able to perform pre-checking of the downloaded software to migrate the cloudified railway functions deployed on the affected nodes to other nodes, to update the software images, and to perform a restart if required.
- The RMAO needs to be able to query and update the railway cloud functional status.
- The RMAO needs to be able to update the software of the CRM services and the software of the CFM services, including the triggering of the requested software version's download, pre-checking, installing, and catalogue update.
- The RMAO needs to notify the railway cloud operator about the result of the orchestration process.

The requirements for the orchestration of the cloudified railway functions include the following capabilities:

- Instantiation of the cloudified railway functions on the cloud.
- Scaling of the cloudified railway functions.
- Software update of the cloudified railway functions.
- Termination of the cloudified railway functions.

In addition to the considered basic use cases, orchestration includes functionality for railway cloud and cloudified function healing, alarm management, performance management, network connection management, provisioning, and resource optimisation.

The next section presents an approach to design the railway cloud's orchestration functionality as microservices. The focus is on microservices related to the software update of CRM and CFM services.

5. Microservices for Railway Cloud Orchestration

The identified railway cloud orchestration functionality can be designed as RESTful microservices. Microservices enable the design of a modular application, which allows for rapid and reliable application development. Microservices can be deployed independently; they communicate using APIs and are loosely coupled. The well-known benefits of the design of microservices include resilience, reusability, technological independence, and easy deployment.

RESTful microservices are designed following the principles of the REST client–server architectural style. RESTful services expose uniform APIs that feature scalability, updatability, and performance. In REST, functionality and data are regarded as resources, which are accessible through Uniform Resource Identifiers (URIs), which are usually web addresses. Every content can be treated as a resource. A resource is represented by its type, associated data (called resource attributes), relationships with other resources, and a set of applicable methods. Resources are manipulated through their representations, where the clients and server exchange resource representation using HTTP methods (POST to create a new resource, GET to retrieve resource information, PUT or PATCH to update a resource, and DELETE to remove a resource). Both data modelling techniques and object-oriented design are used to identify resources based on the analysis of railway cloud orchestration use cases.

5.1. Design of Microservices

The main entities in railway cloud orchestration are related to the railway cloud catalogue. The railway cloud catalogue contains information about the cloud physical and logical resources and cloudified railway functions deployed on the cloud. The RMAO can retrieve information about the assigned cloud resources and their uses. The catalogue information is manipulated by RMAO provisioning requests for cloud resource assignment, e.g., in the case of faults or maintenance operations. The catalogue information is exchanged between the orchestration services in the RMAO and CRM services in the RCP. When resources are needed in railway cloud clusters, the CRM services can add a resource to the respective cloud resource pools, allocate resources to the nodes or the site networks, and notify the RMAO. The RMAO can discover railway cloud clusters, a cluster network, and nodes.

The railway operator may have clouds based on different virtual cluster technologies and those provided by different vendors. Each railway cloud cluster provides CFM services and exposes an interface towards the RMAO, which enables the lifecycle management of the cloudified railway functions. The CRM software catalogue contains information about different CRM services, devoted to, e.g., railway cloud software platform installation, railway cloud registration and initialisation, railway cloud catalogue update, cloud platform software update, railway cloud functional status query and update, CRM services' software update, and CFM services' software update.

Therefore, the relevant resources related to basic railway cloud orchestration represent information about the railway cloud platform, platform software, and CRM and CFM services' software. The considered use case illustrated in Figure 3 exemplifies interface functionality for the software update of CFM services; more details on the CRM services' software catalogue are described.

In addition to the resources representing the CRM services' software, the software catalogue contains a resource that represents the list of available CFM service packages, which could be based on different virtual cluster technologies. This resource is a container of resources, each of which represents a package of CFM services. The resource representing a specific CFM service package contains a sub-resource representing the list of available software versions of this package. A specific software version of the CFM service package is also represented by a resource. The resource representing a specific software version contains a resource representing the list of CFM services, and the individual CFM service in a specific CFM package version is also presented as a resource. A resource, representing all active subscriptions for notifications about changes in the state of a specific resource, could be attached to the resource. The resource representing all active subscriptions is a collection of individual subscriptions.

The resources identified so far are organised in a tree structure, where all the resource URIs have a root, namely the URI of the list of CFM service packages and could be published in a service registry. Table 3 summarises the names, URIs, and applicable HTTP methods to the resources related to the software update of CFM services. For simplicity, the table does not include subscription-related resources.

The applicable HTTP methods to the resource representing all active subscriptions related to the state of other resources are GET, which retrieves the list of all subscriptions, and POST, which creates a new subscription. The resource representing an individual subscription could be manipulated by applying the GET method, which retrieves the subscription information, the PUT method, which modifies the subscription, and the DELETE method, which removes the subscription.

Table 3. CFM services' software update: resources and methods summary.

Resource Name	Resource URI	HTTP Method	Meaning
CFM service packages	.../CFMservicePackages	GET	Retrieves information about the list of available CFM service packages.
A specific CFM service package	.../CFMservicePackages/{CFMservicePackageID}	GET	Retrieves information about specific CFM service packages.
All software versions of the specific CFM service package	.../CFMservicePackages/{CFMservicePackageID}/SWversions	GET	Retrieves the list of software versions of the CFM service package.
		POST	Creates a new software version of the CFM service package.
An individual software version	.../CFMservicePackages/{CFMservicePackageID}/SWversions/{SWversionID}	GET	Retrieves information about specific software versions of the CFM service package.
		PUT	Updates information about the software version.
		DELETE	Deletes the software version.
All CFM services in the software version	.../CFMservicePackages/{CFMservicePackageID}/SWversions/{SWversionID}/CFMservices	GET	Retrieves the list of all CFM services in the software version.
An individual CFM service	.../CFMservicePackages/{CFMservicePackageID}/SWversions/{SWversionID}/CFMservices/{CFMserviceID}	GET	Retrieves information about specific CFM services.
		PUT	Updates the CFM service.
		DELETE	Deletes the CFM service.

The data model defines the data structures that are used in resource representation. The resource attributes are metadata that provide information about properties associated with the resource.

Figure 5 shows the structure of the resource representing an individual CFM service software, which is described by its attributes and sub-resource. In addition to common attributes such as access rights, creation time, and last modification time, the attributes of the resource, representing CFM service software, include the CFM service name, software version, service software URI, and the software action status, which indicates the status of the action (i.e., progress indicator or final state). The software action resource is a sub-resource, which is described by attributes that allow the new software to be downloaded, pre-checked, installed, tested by CRM services, tested by CFM services, activated, or deactivated. The execution of an action on the CFM service software is triggered by applying the HTTP PUT method to the corresponding software action's attribute.

The logic of orchestration applications and the microservices' logic manipulate the state of resources by applying the respective HTTP methods on them.

5.2. *Verification of Microservices*

The orchestration application logic must maintain the state of the software update process. It triggers the transitions in the model by applying the supported HTTP methods on the resources related to the software update service. On the other hand, the respective CRM microservice in the RCP also needs to have a view of the software update process, which must be synchronised with the view of the orchestration application.

To illustrate the practicability of the proposed microservices-based approach to the design of railway cloud orchestration services, models representing the state of the CFM services' software process are developed. The models reflect the views on the process state of the orchestration application and of the respective CRM microservice.

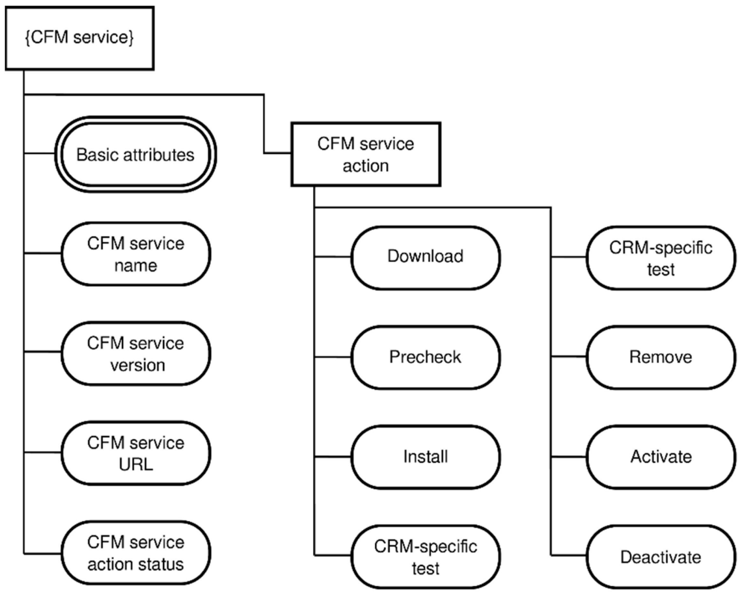

Figure 5. Associated data and elements of the resources representing CFM service software.

Figure 6 shows the CFM services' software update model supported by the orchestration application.

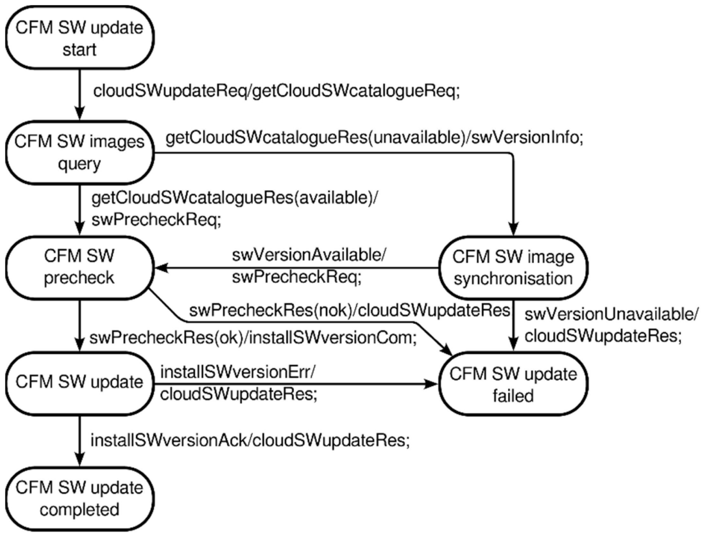
CFM SW update start
cloudSWupdateReq/getCloudSWcatalogueReq;
CFM SW images query
getCloudSWcatalogueRes(unavailable)/swVersionInfo;
getCloudSWcatalogueRes(available)/ swPrecheckReq;
CFM SW precheck
swVersionAvailable/ swPrecheckReq;
CFM SW image synchronisation
swPrecheckRes(nok)/cloudSWupdateRes
swVersionUnavailable/ cloudSWupdateRes;
swPrecheckRes(ok)/installSWversionCom;
CFM SW update
installSWversionErr/ cloudSWupdateRes;
CFM SW update failed
installSWversionAck/cloudSWupdateRes;
CFM SW update completed

Figure 6. CFM services' software update model is supported by the orchestration application.

The CFMswUpdateStart state is the initial state of the CFM software update process. A request to update triggers the CFM software image binaries' synchronisation. In the CFMswImagesQuery state, the application queries the software catalogue. In the CFMswImageSynchronisation state, the requested software version images are not in the catalogue, and information about the software version is sent to the CRM microservice. In the CFMswPrecheck state, a pre-check procedure of the software version is performed. In the CFMsoftwareUpdate state, the new CFM services' software version is updated. In the CFMswUpdateCompleted state, the update process has been completed successfully. Any failures in the update process result in transition to the CFMswUpdateFailed state.

Figure 7 shows the CFM services' software update model supported by the inebreak CRM microservice.

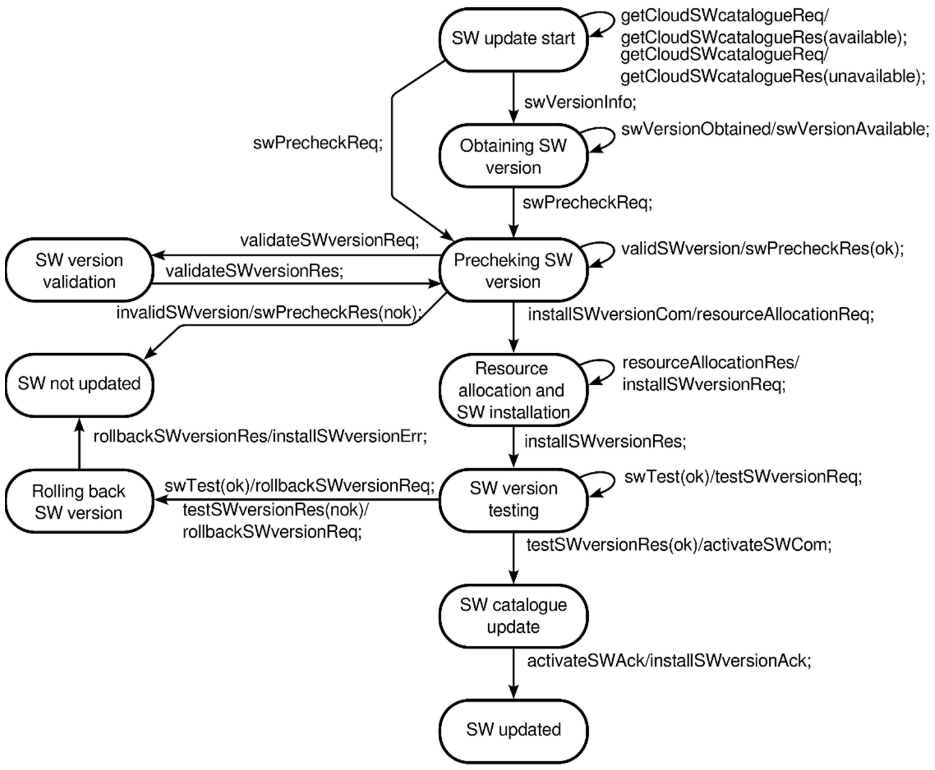
SW update start
getCloudSWcatalogueReq/
getCloudSWcatalogueRes(available);
getCloudSWcatalogueReq/
getCloudSWcatalogueRes(unavailable);
swVersionInfo;
swPrecheckReq;
Obtaining SW version
swVersionObtained/swVersionAvailable;
swPrecheckReq;
validateSWversionReq;
validateSWversionRes;
SW version validation
Precheking SW version
validSWversion/swPrecheckRes(ok);
invalidSWversion/swPrecheckRes(nok);
installSWversionCom/resourceAllocationReq;
SW not updated
Resource allocation and SW installation
resourceAllocationRes/
installSWversionReq;
rollbackSWversionRes/installSWversionErr;
installSWversionRes;
Rolling back SW version
swTest(ok)/rollbackSWversionReq;
testSWversionRes(nok)/
rollbackSWversionReq;
SW version testing
swTest(ok)/testSWversionReq;
testSWversionRes(ok)/activateSWCom;
SW catalogue update
activateSWAck/installSWversionAck;
SW updated

Figure 7. CFM services' software update model is supported by the CRM microservice.

The SWupdateStart state is the initial state of the CFM services' software update from the CRM service point of view. The update process moves to the ObtainingSWversion state if software binaries synchronisation is needed, and in this state, the CRM microservice obtains the software version information. In the PrecheckingSWversion state, a preliminary internal check of the new CFM software version is performed. Pre-checking includes validation of the software version images in the SWversionValidation state. The transition to the ResourceAllocationAndSWinstallation state is triggered upon a command to install the new software version. In this state, the CRM microservice ensures that the update conditions are met, requests resource allocation and software version installation, then marks the CFM services' software as modified. The triggering of a testing procedure moves the software update process to the SWversionTesting state. In the SWcatalogueUpdate state, the CRM microservice updates its software catalogue and moves to the final SWupdated state. If the software test fails, the CRM service requests roll-back to the previously running CFM services' software version (the RollingBackSWversion state) and moves to the SWnotUpdated state. A transition to the SWnotUpdated state is also triggered when the pre-checking procedure fails.

To prove mathematically that both software update models are correct and synchronised, it is necessary to describe the models formally, for which the Labelled Transition System (LTS) concept is used. An LTS is a tuple (S, L, T, s_0), where S is a finite set of states, L is a finite set of labels that trigger the state transitions, T is a state transition relation, and s_0 is an initial state.

In the following definitions, the names of the states and labels are denoted by short notations in brackets.

Definition 1. *Let M^{app} = (S^{app}, L^{app}, T^{app}, $s^{app}{}_0$) be an LTS representing the model of the CFM services' software update process, supported by an orchestration application, where:*

S^{app} = {CFMswUpdateStart [$s^a{}_1$], CFMswImagesQuery [$s^a{}_2$], CFMswImageSynchronization [$s^a{}_3$], CFMswPrecheck [$s^a{}_4$], CFMsoftwareUpdate [$s^a{}_5$], CFMswUpdateCompleted [$s^a{}_6$], CFMswUpdateFailed [$s^a{}_7$]}

L^{app} = {cloudSWupdateReq [$l^a{}_1$], getCloudSWcatalogRes(available) [$l^a{}_2$], getCloudSWcatalogRes(unavailable) [$l^a{}_3$], swVersionAvailable [$l^a{}_4$], swVersionUnavailable [$l^a{}_5$], swPrecheckRes(ok) [$l^a{}_6$], swPrecheckRes(nok) [$l^a{}_7$], installSWversionAck [$l^a{}_8$], installSWversionErr [$l^a{}_9$]}

T^{app} = { ($s^a{}_1$ $l^a{}_1$ $s^a{}_2$), ($s^a{}_2$ $l^a{}_3$ $s^a{}_3$), ($s^a{}_3$ $l^a{}_4$ $s^a{}_4$), ($s^a{}_2$ $l^a{}_2$ $s^a{}_4$), ($s^a{}_4$ $l^a{}_6$ $s^a{}_5$), ($s^a{}_3$ $l^a{}_5$ $s^a{}_7$), ($s^a{}_4$ $l^a{}_7$ $s^a{}_7$), ($s^a{}_5$ $l^a{}_8$ $s^a{}_6$), ($s^a{}_5$ $l^a{}_9$ $s^a{}_7$)}

$s^{app}{}_0$ = CFMswUpdateStart [$s^a{}_1$].

Definition 2. *Let M^{crm} = (S^{crm}, L^{crm}, T^{crm}, $s^{crm}{}_0$) be an LTS representing the model of the CFM services' software update process, supported by the CRM microservice, where:*

S^{crm} = {SWupdateStart [$s^m{}_1$], ObtainingSWversion [$s^m{}_2$], PrecheckSWversion [$s^m{}_3$], SWversionValidation [$s^m{}_4$], ResourceAllocationAndSWinstallation [$s^m{}_5$], SWversionTesting [$s^m{}_6$], SWcatalogUpdate [$s^m{}_7$], SWupdated [$s^m{}_8$], RollingBackSWversion [$s^m{}_9$], SWnotUpdated [$s^m{}_{10}$])

L^{crm} = {getCloudSWcatalogReq [$l^m{}_1$], swVersionInfo [$l^m{}_2$], swVersionObtained [$l^m{}_3$], swPrecheckReq [$l^m{}_4$], validateSWversionReq [$l^m{}_5$], validateSWversionRes [$l^m{}_6$], validSWversion [$l^m{}_7$], invalidSWversion [$l^m{}_8$], installSWversionCom [$l^m{}_9$], resourceAllocationRes [$l^m{}_{10}$], installSWversionRes [$l^m{}_{11}$], swTest(ok) [$l^m{}_{12}$], testSWversionRes(ok) [$l^m{}_{13}$], activateSWAck [$l^m{}_{14}$], swTest(nok) [$l^m{}_{15}$], testSWversionRes(nok) [$l^m{}_{16}$], rollbackSWversionRes [$l^m{}_{17}$]}

T^{crm} = {($s^m{}_1$ $l^m{}_1$ $s^m{}_1$), ($s^m{}_1$ $l^m{}_2$ $s^m{}_2$), ($s^m{}_2$ $l^m{}_3$ $s^m{}_2$), ($s^m{}_1$ $l^m{}_4$ $s^m{}_3$), ($s^m{}_2$ $l^m{}_4$ $s^m{}_3$), ($s^m{}_3$ $l^m{}_5$ $s^m{}_4$), ($s^m{}_4$ $l^m{}_6$ $s^m{}_3$), ($s^m{}_3$ $l^m{}_7$ $s^m{}_3$), ($s^m{}_3$ $l^m{}_8$ $s^m{}_{10}$), ($s^m{}_3$ $l^m{}_9$ $s^m{}_5$), ($s^m{}_5$ $l^m{}_{10}$ $s^m{}_5$), ($s^m{}_5$ $l^m{}_{11}$ $s^m{}_6$), ($s^m{}_6$ $l^m{}_{12}$ $s^m{}_6$), ($s^m{}_6$ $l^m{}_{13}$ $s^m{}_7$), ($s^m{}_7$ $l^m{}_{14}$ $s^m{}_8$), ($s^m{}_6$ $l^m{}_{15}$ $s^m{}_9$), ($s^m{}_6$ $l^m{}_{16}$ $s^m{}_9$), ($s^m{}_9$ $l^m{}_{17}$ $s^m{}_{10}$)}

$s^{crm}{}_0$ = SWupdateStart [$s^m{}_1$].

The proof that both models support synchronised views on the state of the software update process is based on the fundamental notion of a weak bi-simulation. Bi-simulation uses structural conditions to characterise the equivalent (indistinguishable) observable behaviour of concurrent processes. It is a binary relationship between the states which associates the concurrent processes that behave in the same way [64,65]. Weak bi-simulation hides internal transitions which are not externally visible in the process behaviour. It is used for model verification and constraints satisfaction.

Proposition 1. *The LTSs representing the model of the CFM services' software update process, supported by an orchestration application and by the CRM microservice (namely, M^{app} and M^{crm}), expose equivalent behaviour.*

Proof. By R_1 = {($s^a{}_1$, $s^m{}_1$), ($s^a{}_4$, $s^m{}_3$), ($s^a{}_5$, $s^m{}_5$), ($s^a{}_6$, $s^m{}_8$), ($s^a{}_7$, $s^m{}_{10}$)} it is denoted a relationship between the states of M^{app} and M^{crm}. The following bijection function between transitions in the state pairs in R_1 is identified:

1. A software update is triggered, a CFM services' software image binaries synchronisation takes place, a pre-check of the new CFM services' software version is triggered: $\forall$ ($s^a{}_1$ $l^a{}_1$ $s^a{}_2$) $\wedge$ (($s^a{}_2$ $l^a{}_3$ $s^a{}_3$) $\wedge$ ($s^a{}_3$ $l^a{}_4$ $s^a{}_4$) $\vee$ ($s^a{}_2$ $l^a{}_2$ $s^a{}_4$)) $\exists$ ($s^m{}_1$ $l^m{}_1$ $s^m{}_1$) $\wedge$ (($s^m{}_1$ $l^m{}_2$ $s^m{}_2$) $\wedge$ ($s^m{}_2$ $l^m{}_3$ $s^m{}_2$) $\wedge$ ($s^m{}_2$ $l^m{}_4$ $s^m{}_3$) $\vee$ ($s^m{}_1$ $l^m{}_4$ $s^m{}_3$)).
2. The pre-check procedure is successful, and a software uploading is triggered: $\forall$ ($s^a{}_4$ $l^a{}_6$ $s^a{}_5$) $\exists$ ($s^m{}_3$ $l^m{}_5$ $s^m{}_4$) $\wedge$ ($s^m{}_4$ $l^m{}_6$ $s^m{}_3$) $\wedge$ ($s^m{}_3$ $l^m{}_7$ $s^m{}_3$) $\wedge$($s^m{}_3$ $l^m{}_9$ $s^m{}_5$).
3. The pre-check procedure is unsuccessful: $\forall$ ($s^a{}_4$ $l^a{}_7$ $s^a{}_7$) $\exists$ ($s^m{}_3$ $l^m{}_8$ $s^m{}_{10}$).
4. The CFM services' software upload, which includes resource allocation, software installation and testing, is successful: $\forall$ ($s^a{}_5$ $l^a{}_8$ $s^a{}_6$) $\exists$ ($s^m{}_5$ $l^m{}_{10}$ $s^m{}_5$) $\wedge$ ($s^m{}_5$ $l^m{}_{11}$ $s^m{}_6$) $\wedge$ ($s^m{}_6$ $l^m{}_{12}$ $s^m{}_6$) $\wedge$ ($s^m{}_6$ $l^m{}_{13}$ $s^m{}_7$) $\wedge$ ($s^m{}_7$ $l^m{}_{14}$ $s^m{}_8$).
5. The CFM services' software is uploaded, resources are allocated, and the software is installed, but the test procedures fail, and a rollback procedure is executed: $\forall$ ($s^a{}_5$ $l^a{}_9$ $s^a{}_7$) $\exists$ (($s^m{}_5$ $l^m{}_{11}$ $s^m{}_6$) $\wedge$ (($s^m{}_6$ $l^m{}_{15}$ $s^m{}_9$) $\vee$ ($s^m{}_6$ $l^m{}_{16}$ $s^m{}_9$)) $\wedge$ ($s^m{}_9$ $l^m{}_{17}$ $s^m{}_{10}$).

The identified bijection function between the transitions in the state pairs in R_1 proves that R_1 is a weak bi-simulation relationship and, therefore, the behaviour of both LTSs representing the model of the CFM services' software update process, supported by an orchestration application and by the CRM microservice, is equivalent. □

A respective CFM microservice also participates in the process of CFM services' software update. Figure 8 shows the model representing the software update process from the CFM service's point of view.

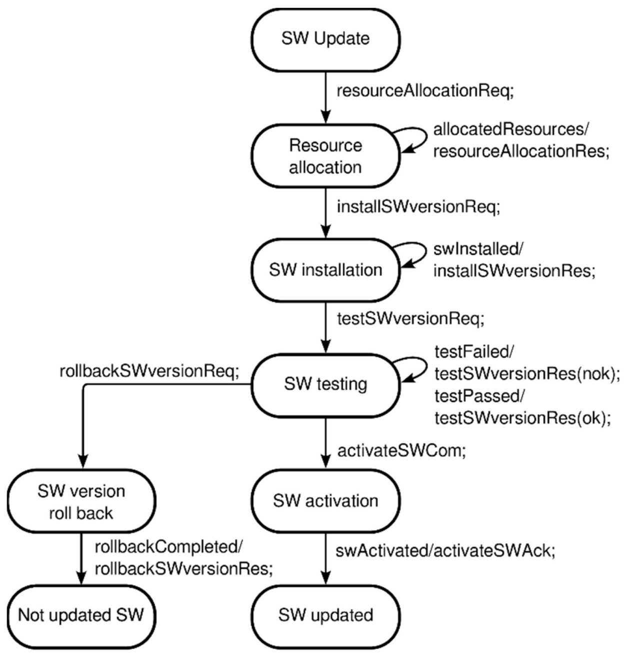
SW Update
resourceAllocationReq;
Resource allocation
allocatedResources/ resourceAllocationRes;
installSWversionReq;
SW installation
swInstalled/ installSWversionRes;
testSWversionReq;
SW testing
testFailed/ testSWversionRes(nok); testPassed/ testSWversionRes(ok);
rollbackSWversionReq;
activateSWCom;
SW version roll back
SW activation
rollbackCompleted/ rollbackSWversionRes;
swActivated/activateSWAck;
Not updated SW
SW updated

Figure 8. Model representing the software update process from the CFM service's point of view.

The SWupdate state is the initial state of the software update process. In the ResourceAllocation state, cloud resources are assigned. In the SWinstallation state, the new software version is installed. The testing of the newly installed SW version takes place in the SWtesting state; the new software version is activated in the SWactivation state, and in the UpdatedSW state, the CFM services' software is successfully updated. In the SWversionRollback state, the previously running SW version is rolled back, and the process moves to the NotUpdatedSW state.

Definition 3. *Let $M^{cfm} = (S^{cfm}, L^{cfm}, T^{cfm}, s^{cfm}{}_0)$ be an LTS representing the model of the CFM services' software update process, supported by the CFM microservice, where:*

S^{cfm} = {SWupdate [$s^n{}_1$], ResourceAllocation [$s^n{}_2$],SWinstallation [$s^n{}_3$], SWtesting [$s^n{}_4$], SWactivation [$s^n{}_5$], UpdateSW [$s^n{}_6$], SWversionRollback [$s^n{}_7$], UpdatedSWNot [$s^n{}_8$]}

L^{cfm} = {resourceAllocationReq [$l^n{}_1$], alocatedResources [$l^n{}_2$], installSWversionReq [$l^n{}_3$], swInstalled [$l^n{}_4$], testSWversionReq [$l^n{}_5$], testFailed [$l^n{}_6$], testPassed [$l^n{}_7$], activateSWCom [$l^n{}_8$], swActivated [$l^n{}_9$], rollbackSWversionReq [$l^n{}_{10}$], rollbackCompleted [$l^m{}_{11}$]}

T^{cfm} = {($s^n{}_1$ $l^n{}_1$ $s^n{}_2$), ($s^n{}_2$ $l^n{}_2$ $s^n{}_2$), ($s^n{}_2$ $l^n{}_3$ $s^n{}_3$), ($s^n{}_3$ $l^n{}_4$ $s^n{}_3$), ($s^n{}_3$ $l^n{}_5$ $s^n{}_4$), ($s^n{}_4$ $l^n{}_6$ $s^n{}_4$), ($s^n{}_4$ $l^n{}_7$ $s^n{}_4$), ($s^n{}_4$ $l^n{}_8$ $s^n{}_5$), ($s^n{}_5$ $l^n{}_9$ $s^n{}_6$), ($s^n{}_4$ $l^n{}_{10}$ $s^n{}_7$), ($s^n{}_7$ $l^n{}_{11}$ $s^n{}_8$)}

$s^{cfm}{}_0$ = SWupdate [$s^n{}_1$].

Proposition 2. *The LTSs representing the model of the CFM services' software update process, supported by the CRM microservice and the CFM microservice (namely, M^{crm} and M^{cfm}), expose equivalent behaviour.*

Proof. By $R_2 = \{(s^m{}_1, s^n{}_1), (s^m{}_5, s^n{}_2), (s^m{}_6, s^n{}_4), (s^m{}_8, s^n{}_6), (s^m{}_{10}, s^n{}_8)\}$ it is denoted a relationship between the states of M^{crm} and M^{cfm}. The following bijection function between transitions in the state pairs in R_2 is identified:

1. A software update is triggered, the new software version is available pre-checked, and a resource allocation is triggered: $\forall$ ($s^m{}_1$ $l^m{}_1$ $s^m{}_1$) $\wedge$ (($s^m{}_1$ $l^m{}_2$ $s^m{}_2$) $\wedge$ ($s^m{}_2$ $l^m{}_3$ $s^m{}_2$) $\wedge$ ($s^m{}_2$ $l^m{}_4$ $s^m{}_3$) $\vee$ ($s^m{}_1$ $l^m{}_4$ $s^m{}_3$)) $\wedge$ ($s^m{}_3$ $l^m{}_9$ $s^m{}_5$) $\exists$ ($s^n{}_1$ $l^n{}_1$ $s^n{}_2$) $\wedge$ ($s^n{}_2$ $l^n{}_2$ $s^n{}_2$).

2. The resource allocation is done, the new SW version is installed, and a testing procedure is triggered: $\forall$ (s^m_5 l^m_{10} s^m_5) $\wedge$ (s^m_5 l^m_{11} s^m_6) $\exists$ (s^n_2 l^n_3 s^n_3) $\wedge$ (s^n_3 l^n_4 s^n_3) $\wedge$ (s^n_3 l^n_5 s^n_4).
3. The new SW version successfully passes the testing procedure, and it is activated: $\forall$ (s^m_6 l^m_{12} s^m_6) $\wedge$ (s^m_6 l^m_{13} s^m_7) $\wedge$ (s^m_7 l^m_{14} s^m_8) $\exists$ (s^n_4 l^n_7 s^n_4) $\wedge$ (s^n_4 l^n_8 s^n_5) $\wedge$ (s^n_5 l^n_9 s^n_6).
4. The test of the new SW version fails, and a rollback to the previous SW version is done: $\forall$ (s^m_6 l^m_{16} s^m_9) $\wedge$ (s^m_9 l^m_{17} s^m_{10}) $\exists$ (s^n_4 l^n_7 s^n_4) $\wedge$ (s^n_4 l^n_{10} s^n_7) $\wedge$ (s^n_7 l^n_{11} s^n_8).

Therefore, R_2 indicates a weak bi-simulation relationship, and M^{crm} and M^{cfm} expose equivalent behaviour. $\square$

5.3. Validation of Microservices

Key performance indicators (KPIs) are essential metrics measuring the smooth operation of microservices and applications. Measuring KPIs and statistical performance analysis could reveal management operation improvement opportunities. In order to demonstrate the benefits of the proposed approach for fulfilling the strong mission critical requirements of railway operations and to validate the synthesised microservices, latency, injected by the microservices' API, is evaluated.

Latency is referred to as one of the functional metrics related to the performance of microservices. It is, in general, time-variable and could be measured in a specified time interval and described by a profile over time. The concept of latency is wide, but in this paper, latency is defined as the time taken for an HTTP request, generated by a railway cloud orchestration application to get to the server that hosts microservices, and the time taken by the respective HTTP response to travel back.

The RESTful approach is based on the well-known client–server pattern and considering that the requirement about the cross-platform property of the implementation must be kept, the programming tool Java is used for the numerical simulation. Both sides of the interface are deployed in a single group in order to avoid any influence of outer traffic. This makes the results as reference for a lower bound estimation of the latency with as modest resources allocated as four cores of the CPU and memory of four GBs. The client- and server-side are multi-threaded, but, purposefully, no scale-up is allowed. The results in terms of latency, shown in Figure 9, represent a frame of two thousand operations over the interface in a consecutive manner without any delays between.

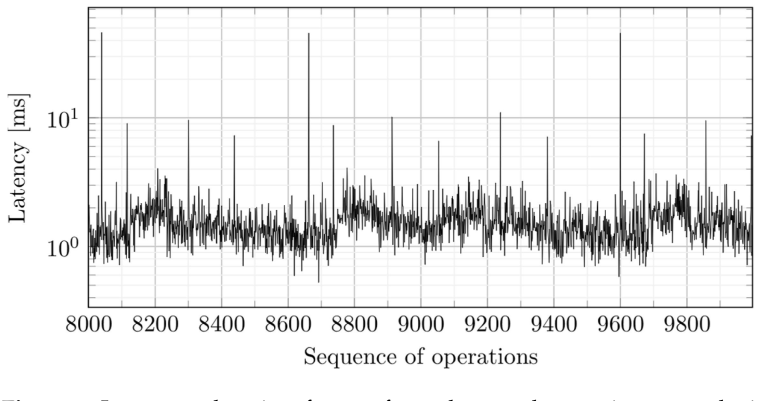

Figure 9. Latency values in a frame of two thousand operations over the interface between the RMAO and RCP.

The time frame is chosen well after the initial lazy instantiation of the objects. It might be observed that the mean of the latency is about 2 milliseconds, and the major part of the value is up to about 4 milliseconds. However, a minority of values exist that hit almost 50 milliseconds, and this is a clear signal that, if the load increases, one should expect further degradation of the indicator.

6. Railway Cloud Issues Regarding Safety and Security

The migration of railway control systems, real-time monitoring, and operation management to the cloud bring about challenges in terms of railway safety and security, and railway cloud management plays an important role in achieving the necessary requirements.

The aim of railway safety is to protect passengers, the environment, and railway assets by applying automatic protection functions. Railway safety requires understanding potential risks for the malfunctioning of equipment and software errors and to minimise their impact. The proposed railway cloud management system could assist in these processes. Usually, the proper automatic protection functions include the monitoring of system performance, the safe delivery of updates, and risk management. Functional railway safety requires performance assessment of the cloud and virtualised resources, identification of the risks for achievement of the required performance, risk mitigation by redundancy in the system design, and proving a performance achievement. For virtualised resources, it is important to provide operation continuity while minimising recovery time, which means that high cloud availability could be achieved by railway system resilience instead of individual equipment resilience. The required level of resilience can be achieved by distributed railway clouds and increasing the railway cluster's redundancy instances for on-time replacements. In the cloud, railway operators can continuously monitor workflows to discover faults and errors, as well as network traffic anomalies, or to balance the load. An important factor for railway system resilience is also the cloud archiving capabilities that facilitate recovery from failures.

The aim of railway cloud security is to protect cloud-based infrastructure, applications, and data. The main railway cloud security benefits are as follows:

- Lower investments and reduced operational costs due to the security and compliance framework respected by cloud services providers.
- Improved reliability and availability based on the authorized access to data and centralized security control.
- Flexible cloud security scaling with evolving requirements.
- Improved protection against distributed denial of service via continuous attack monitoring and dispersing.

The main railway cloud components and data that must be protected include the following: software images repository, cloud catalogue information for hardware and software resources, cloudified railway functions information, software components and their associated data, private keys, certificates and credentials, railway cloud and cloudified railway function monitoring, etc.

The requirements of railway cloud security could be derived through the analysis of threats. For example, a railway cloud API attack could affect cloud deployments through a compromised virtualisation layer, which may result in information disclosure or tampering, or the denial of a service. Such unsecured cloud API and the lack of authorisation could impact availability, integrity, and confidentiality. Some of the common requirements to achieve cloud security can be found in published security guidelines and best practices. They include certificate management, cloud network segmentation and traffic filtering, identity authentication, access management, security configuration by cloud virtualisation level, secure lifecycle of the cloudified functions and applications, protection of software images, logging, monitoring, and alerting of threats, and security audit. In addition, secured access to encrypted storage, hardware, and network security protection must be provided.

7. Conclusions

The movement of the railway industry to the use of cloud technology brings with it many benefits that could improve the operation of railways and enhance passenger experience with travel-centric and innovative services. To provide high reliability, safety, and security, the railway cloud needs to ensure management and orchestration functionality.

In this paper, we proposed an approach to design railway cloud management and orchestration functionality as microservices. The main research contributions are summarised as follows:

- Identification of the key railway cloud concepts
- Recognition of the general principles of the railway cloud management and orchestration interface
- Description of some of the basic use cases related to the railway cloud orchestration
- Definition of requirements for the railway cloud orchestration
- Synthesis of cloud orchestration functionality as microservices by identification of resources, representing physical or logical objects in the railway cloud orchestration, their structure, applicable methods, and data types
- Verification of microservices by modelling the common orchestration logic of microservices and railway applications
- Validation of microservices by evaluation of latency injected by the microservices API as a key performance indicator for mission-critical applications.

Some issues related to the safety and security of the railway cloud are considered.

The main benefits of the proposed approach in comparison with any proprietary solutions are that it defines an open and extensible railway cloud management and orchestration interface that enables the addition of functions and information without changing the existing procedures. It also enables a railway operator to employ clouds from different vendors and to manage and orchestrate these clouds in a uniform way. The identified functionality and the proposed approach are independent of the specific RMAO platform and railway cloud platform implementations and enable a third party to create and integrate their orchestration applications using the interface.

Despite these benefits, which are inherent to microservices, potential drawbacks and challenges exist. The management of microservices is complex due to their number and interdependencies, as well as the increased use of computing and storage resources, which leads to decreases in performance. APIs inject latency due to network communications, which may cause network congestion. Some challenges in microservices-oriented design include ensuring service data consistency and integrity across different microservices, testing challenges, and security challenges, requiring the implementation of security measures across all microservices. These disadvantages could be mitigated through proper planning, design, and management.

Future works will aim to cover management functionality related to cloudified railway functions, including the description of related use cases and microservices' design, verification, and validation. Management functionality will encompass basic cloudified railway function orchestration use cases, the recovery use cases, fault use cases, performance management use cases, provisioning use cases, and resource optimisation use cases.

Author Contributions: I.A. contributed to the methodology. E.P. contributed to the conceptualization and writing—original draft preparation. V.T. contributed to the formal analysis and verification. K.K. contributed to writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: The authors thank the administrative and technical support of the Bulgarian National Science Fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Acknowledgments: The authors thank the administrative and technical support of the Bulgarian National Science Fund, grant number KP-06-H57/12.

Conflicts of Interest: The authors declare no conflicts of interest regarding the publication of this paper. The funders had no role in the design of this study; the collection, analysis, or interpretation of data; the writing of the manuscript; or the decision to publish the results.

References

1. Nikolopoulou, V.; Mandoc, D.; Bazizi, F.; Kloecker, M.; Tardif, S.; Holfeld, B.; Jornod, G.; Salhab, N.; Berbineau, M.; Gogos, S. 5GRAIL paves the way to the Future Railway Mobile Communication System Introduction. In Proceedings of the 2022 IEEE Future Networks World Forum (FNWF), Montreal, QC, Canada, 10–14 October 2022; pp. 53–57. [CrossRef]
2. Iacurto, C.; Catuogno, T.; Brizzi, A.; Pandolfi, L.; Miglietta, A.; Rokitansky, C.H.; Eschbacher, K.; Pellegrini, V.; Toptsidis, N. Capacity Study for a 5G Satellite System to support Railway FRMCS Critical service over Europe. In Proceedings of the 2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring), Helsinki, Finland, 19–22 June 2022; pp. 1–5. [CrossRef]
3. Tardif, S.; Salhab, N.; Nikolopoulou, V.; Kloecker, M.; Holfeld, B.; Bazizi, F.; Mandoc, D.; Berbineau, M.; Gogos, S. Experimental Trials for the Future Railway Mobile Communication System in 5GRail Project. In Proceedings of the 2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring), Florence, Italy, 20–23 June 2023; pp. 1–5. [CrossRef]
4. Bhagat, A.K.; Gandhi, J. A Comprehensive Analysis of 5G Security Core Technologies and Services: Conceptual Frameworks, Challenges, and Solutions. In Proceedings of the 2023 International Conference on Artificial Intelligence and Smart Communication (AISC), Greater Noida, India, 27–29 January 2023; pp. 1273–1281. [CrossRef]
5. Panda, I.; Ramanath, S. Railways Communication Beyond 5G: Opportunities and Challenges. In Proceedings of the 2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS), Bangalore, India, 3–8 January 2023; pp. 327–330. [CrossRef]
6. Phusakulkajorn, W.; Núñez, A.; Wang, H.; Jamshidi, A.; Zoeteman, A.; Ripke, B.; Dollevoet, R.; De Schutter, B.; Li, Z. Artificial intelligence in railway infrastructure: Current research, challenges, and future opportunities. *Intell. Transp. Infrastruct.* **2023**, *2*, liad016. [CrossRef]
7. Li, S.; Wang, J. Review of Network Anomaly Detection in the High-speed Railway Signal System Based on Artificial Intelligence. In Proceedings of the 2023 IEEE 3rd International Conference on Computer Communication and Artificial Intelligence (CCAI), Taiyuan, China, 26–28 May 2023; pp. 318–325. [CrossRef]
8. Zhang, J.; Zhang, J. Artificial Intelligence Applied on Traffic Planning and Management for Rail Transport: A Review and Perspective. *Discret. Dyn. Nat. Soc.* **2023**, *2023*, 1832501. [CrossRef]
9. Jevinger, Å.; Zhao, C.; Persson, J.A.; Davidsson, P. Artificial intelligence for improving public transport: A mapping study. *Public Transp.* **2023**, *16*, 99–158. [CrossRef]
10. Gore, P.; Dudhe, R.; Raina, R. A review of Renewable Energy & Smart Grid system applications in railways. In Proceedings of the 2023 Advances in Science and Engineering Technology International Conferences (ASET), Dubai, United Arab Emirates, 20–23 February 2023; pp. 1–6. [CrossRef]
11. De Donato, L.; Dirnfeld, R.; Somma, A.; De Benedictis, A.; Flammini, F.; Marrone, S.; Saman Azari, M.; Vittorini, V. Towards AI-assisted digital twins for smart railways: Preliminary guideline and reference architecture. *J. Reliab. Intell. Environ.* **2023**, *9*, 303–317. [CrossRef]
12. Wu, J.; Wang, X.; Dang, Y.; Lv, Z. Digital twins and artificial intelligence in transportation infrastructure: Classification, application, and future research directions. *Comput. Electr. Eng.* **2022**, *101*, 107983. [CrossRef]
13. Djordjević, B.; Fröidh, O.; Krmac, E. Determinants of autonomous train operation adoption in rail freight: Knowledge-based assessment with Delphi-ANP approach. *Soft Comput.* **2023**, *27*, 7051–7069. [CrossRef]
14. Jansson, E.; Olsson, N.O.; Fröidh, O. Challenges of replacing train drivers in driverless and unattended railway mainline systems—A Swedish case study on delay logs descriptions. *Transp. Res. Interdiscip. Perspect.* **2023**, *21*, 100875. [CrossRef]
15. Daniyan, I.; Mpofu, K.; Muvunzi, R.; Uchegbu, I.D. Implementation of Artificial intelligence for maintenance operation in the rail industry. *Procedia CIRP* **2022**, *109*, 449–453. [CrossRef]
16. Shim, J.; Koo, J.; Park, Y. A Methodology of Condition Monitoring System Utilizing Supervised and Semi-Supervised Learning in Railway. *Sensors* **2023**, *23*, 9075. [CrossRef]
17. Natali, C.D.; Mattila, J.; Kolu, A.; De Vito, P.; Gauttier, S.; Morata, M.; Garcia, M.; Caldwell, D. Smart tools for railway inspection and maintenance work, performance and safety improvement. *Transp. Res. Procedia* **2023**, *72*, 3070–3077. [CrossRef]
18. Pang, Z.; Wang, L.; Wang, S.; Li, L.; Peng, Q. Dynamic train dwell time forecasting: A hybrid approach to address the influence of passenger flow fluctuations. *Railw. Eng. Sci.* **2023**, *31*, 351–369. [CrossRef]
19. Birgillito, G.; Rindone, C.; Vitetta, A. Passenger Mobility in a Discontinuous Space: Modelling Access/Egress to Maritime Barrier in a Case Study. *J. Adv. Transp.* **2018**, *2018*, 6518329. [CrossRef]
20. Dekker, B.; Ton, B.; Meijer, J.; Bouali, N.; Linssen, J.; Ahmed, F. Point Cloud Analysis of Railway Infrastructure: A Systematic Literature Review. *IEEE Access* **2023**, *11*, 134355–134373. [CrossRef]
21. Li, P.; Xue, R.; Shao, S.; Zhu, Y.; Liu, Y. Current state and predicted technological trends in global railway intelligent digital transformation. *Railw. Sci.* **2023**, *2*, 397–412. [CrossRef]
22. Vanichchanunt, P.; Tanmalaporn, T.; Suthamvijit, C.; Noisri, S.; Wuttisittikulkij, L.; Pongyart, W.; Paripurana, S. Virtual Reality for Railway Signaling System Training. In Proceedings of the 2023 20th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Nakhon Phanom, Thailand, 9–12 May 2023; pp. 1–4. [CrossRef]

23. Ruscelli, A.L.; Fichera, S.; Paolucci, F.; Giorgetti, A.; Castoldi, P.; Cugini, F. Introducing Network Softwarization in Next-Generation Railway Control Systems. In Proceedings of the 2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), Cracow, Poland, 5–7 June 2019; pp. 1–7. [CrossRef]
24. Pencheva, E.; Atanasov, I.; Trifonov, V. Towards Intelligent, Programmable, and Open Railway Networks. *Appl. Sci.* **2022**, *12*, 4062. [CrossRef]
25. Atanasov, I.; Vatakov, V.; Pencheva, E. A Microservices-Based Approach to Designing an Intelligent Railway Control System Architecture. *Symmetry* **2023**, *15*, 1566. [CrossRef]
26. Li, Y.; Zhu, L. Collaborative Cloud and Edge Computing in 5G based Train Control Systems. In Proceedings of the GLOBECOM 2022—2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil, 4–8 December 2022; pp. 2542–2547. [CrossRef]
27. Zhang, X. Optimization design of railway logistics center layout based on mobile cloud edge computing. *PeerJ Comput. Sci.* **2023**, *9*, e1298. [CrossRef]
28. Tian, L.; Li, M.; Si, P.; Yang, R.; Sun, Y.; Wang, Z. Design and Optimization in MEC-Based Intelligent Rail System by Integration of Distributed Multi-Hop Communication and Blockchain. *Math. Probl. Eng.* **2023**, *2023*, 8858263. [CrossRef]
29. Haiyan, S.; Tianyun, S.; Jiaying, D.; Guoyuan, Y.; Qingbo, C. Research on Control Mode and Perception Model of Intelligent Railway Station Equipment Based on Edge Computing. In Proceedings of the 2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Chongqing, China, 18–20 June 2021; pp. 2085–2090. [CrossRef]
30. Aufderheide, H.; Brandes, N. Architecting for Resilience in the Cloud for Critical Railway Systems. AWS for Industries. Available online: https://aws.amazon.com/blogs/industries/architecting-for-resilience-in-the-cloud-for-critical-railway-systems-2 (accessed on 23 January 2024).
31. Gala, G.; Fohler, G.; Tummeltshammer, P.; Resch, S.; Hametner, R. RT-Cloud: Virtualization Technologies and Cloud Computing for Railway Use-Case. In Proceedings of the 2021 IEEE 24th International Symposium on Real-Time Distributed Computing (ISORC), Daegu, Republic of Korea, 1–3 June 2021; pp. 105–113. [CrossRef]
32. Huang, Q.; Yin, W.; An, J.; Zhou, Y. A China Railway Express-Based Model for Designing a Cross-Border Logistics Information Cloud Platform Scheme. *Appl. Sci.* **2020**, *10*, 4110. [CrossRef]
33. He, M.; Li, H.; Duan, Y. Research on Railway Intelligent Operation and Maintenance and Its System Architecture. In Proceedings of the 2019 6th International Conference on Dependable Systems and Their Applications (DSA), Harbin, China, 3–6 January 2020; pp. 488–490. [CrossRef]
34. Zhu, W. Research on Construction of Cloud Computing Platform for Railway Enterprises. In Proceedings of the 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), Dublin, Ireland, 16–18 October 2019; pp. 488–492. [CrossRef]
35. Yu, S.; Chang, H.; Wang, H. Design of Cloud Computing and Microservice-Based Urban Rail Transit Integrated Supervisory Control System Plus. *Urban Rail Transit.* **2020**, *6*, 187–204. [CrossRef]
36. Yu, S.D. Research on cloud computing in the key technologies of railway intelligent operation and maintenance sharing platform. *J. Phys. Conf. Ser.* **2021**, *1800*, 012010. [CrossRef]
37. Ren, S.; Li, Y. A review of high performance computing applications in high-speed rail systems. *High-Speed Railw.* **2023**, *1*, 92–96. [CrossRef]
38. Zhu, L.; Zhuang, Q.; Jiang, H.; Liang, H.; Gao, X.; Wang, W. Reliability-aware failure recovery for cloud computing based automatic train supervision systems in urban rail transit using deep reinforcement learning. *J. Cloud Comput.* **2023**, *12*, 147. [CrossRef]
39. Wang, Q.; Chai, M.; Wang, H.; Zhang, H.; Chai, J.; Lin, B. Cloud-Based Simulated Automated Testing Platform for Virtual Coupling System. In Proceedings of the 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; pp. 2738–2743. [CrossRef]
40. Rajan, T.K.; Karthick, S.; Nirmal, S.; Kumar, S.N.; Senthilmurugan, S. IoT Based Remote Surveillance For Animal Tracking Near Railway Tracks. In Proceedings of the 2023 International Conference on Networking and Communications (ICNWC), Chennai, India, 5–6 April 2023; pp. 1–7. [CrossRef]
41. Malekjafarian, A.; Sarrabezolles, C.A.; Khan, M.A.; Golpayegani, F. A Machine-Learning-Based Approach for Railway Track Monitoring Using Acceleration Measured on an In-Service Train. *Sensors* **2023**, *23*, 7568. [CrossRef] [PubMed]
42. Gosiewska, A.; Baran, Z.; Baran, M.; Rutkowski, T. Seeking a Sufficient Data Volume for Railway Infrastructure Component Detection with Computer Vision Models. *Sensors* **2023**, *23*, 7776. [CrossRef] [PubMed]
43. Ping, L.; Pu, W.; Jinzi, Z. Research on Global Optimization Scheduling Method of High-speed Railway Emergency Resource Based on Service Pool. In Proceedings of the 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications(AEECA), Dalian, China, 25–27 August 2020; pp. 100–108. [CrossRef]
44. Nawghare, S.; Somkunwar, R.K.; Shaikh, Z. Indian Railways Smart Ticketing Validation System with Improved Alert Approach. In Proceedings of the 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, 14–16 June 2023; pp. 1328–1332. [CrossRef]
45. Shan, X.; Zhang, Z.; Ning, F.; Li, S.; Dai, L. Application and realization of key technologies in China railway e-ticketing system. *Railw. Sci.* **2023**, *2*, 140–156. [CrossRef]

46. Tao, X.; Zhou, Y.; Mei, Y.; Fujita, H. Parallel simulation of high-speed trains controlled by radio block centers using Spark cloud. *Adv. Mech. Eng.* **2023**, *15*, 16878132231186258. [CrossRef]
47. Xu, L.; Pan, Z.; Liu, H.; Geng, S.; Wan, H. Intelligent Operation and Maintenance Platform for Power Supply System of Urban Rail Transit. In *Proceedings of the 5th International Conference on Electrical Engineering and Information Technologies for Rail Transportation (EITRT) 2021, Qingdao, China, 22–24 October 2021*; Jia, L., Qin, Y., Liang, J., Liu, Z., Diao, L., An, M., Eds.; Springer: Singapore, 2022; Volume 864, pp. 156–163. [CrossRef]
48. Zhang, Z.; Li, J.; Sun, Y.; Li, Y.; Song, H.; Dong, H. Blockchain-based Secure Key Management model for High-speed Railway. In Proceedings of the 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; pp. 1988–1993. [CrossRef]
49. Qlu, Y. Secure Mechanism of Intelligent Urban Railway Cloud Platform Based on Zero-trust Security Architecture. In Proceedings of the 6th International Conference on High Performance Compilation, Computing and Communications, Jilin, China, 23–25 June 2022; pp. 99–105. [CrossRef]
50. Xu, D.; Liu, F.; Chen, W.; He, F.; Tang, X.; Zhang, Y.; Wang, B. A review of research on multi-cloud management platforms. In Proceedings of the Computer Technology and Transportation ISCTT 2022, 7th International Conference on Information Science, Xishuangbanna, China, 27–29 May 2022; pp. 1–16.
51. Shevtekar, P.S.S.; Kulkarni, S.; Talwara, H. Cost-Effective Resource Allocation and Optimization Strategies for Multi-Cloud Environments. *Int. J. Res. Appl. Sci. Eng. Technol.* **2023**, *11*, 602–605. [CrossRef]
52. Kumar, P.; Garg, A.; Sharmila, R.; Parashar, H.; Khatri, V. SLA Framework and Management under Multi-Cloud Infrastructure. In Proceedings of the 2022 2nd International Conference on Innovative Sustainable Computational Technologies (CISCT), Dehradun, India, 23–24 December 2022; pp. 1–5. [CrossRef]
53. Toutova, N.V.; Erokhin, A.G.; Toutov, A.V.; Vanina, M.F.; Frolova, E.A. Principals of Optimal Cloud Resource Management. In Proceedings of the 2022 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED), Moscow, Russia, 10–11 November 2022; pp. 1–7. [CrossRef]
54. Manchanda, A.; Kaur, A.; Kaur, A. Cloud Computing: Resource Management, Categorization, Scalability and Taxonomy. In Proceedings of the 2023 2nd Edition of IEEE Delhi Section Flagship Conference (DELCON), Rajpura, India, 24–26 February 2023; pp. 1–5. [CrossRef]
55. Yezdani, R.; Quadri, S.M.K. Power and Performance Issues and Management Approaches in Cloud Computing. In Proceedings of the 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 16–17 December 2022; pp. 2112–2120. [CrossRef]
56. Zhou, H. A Novel Approach to Cloud Resource Management: Hybrid Machine Learning and Task Scheduling. *J. Grid Comput.* **2023**, *21*, 68. [CrossRef]
57. Chen, Q.; Wang, X.; Jiang, Z. Efficient Cloud Computing Resource Management Strategy Based on Auction Mechanism. In Proceedings of the 2023 24th Asia-Pacific Network Operations and Management Symposium (APNOMS), Sejong, Republic of Korea, 6–8 September 2023; pp. 286–289.
58. Hasan, M.Z.; Sarwar, N.; Alam, I.; Hussain, M.Z.; Siddiqui, A.A.; Irshad, A. Data Recovery and Backup management: A Cloud Computing Impact. In Proceedings of the 2023 IEEE International Conference on Emerging Trends in Engineering, Sciences and Technology (ICES&T), Bahawalpur, Pakistan, 9–11 January 2023; pp. 1–6. [CrossRef]
59. Kaur, M.; Kaimal, A.B. Analysis of Cloud Computing Security Challenges and Threats for Resolving Data Breach Issues. In Proceedings of the 2023 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 23–25 January 2023; pp. 1–6. [CrossRef]
60. Bhattacharyya, L.; Purohit, S.G.; Fatmawati, E.; Sunil, D.M.; Toktakynovna, Z.; Sriramakrishnan, G. Cloud Computing for Suitable Data Management and Security within Organisations. In Proceedings of the 2022 3rd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 20–22 October 2022; pp. 912–916. [CrossRef]
61. Sawhney, G.; Kaur, G.; Deorari, R. CSPM: A Secure Cloud Computing Performance Management Model. In Proceedings of the 2022 International Conference on Cyber Resilience (ICCR), Dubai, United Arab Emirates, 6–7 October 2022; pp. 1–5. [CrossRef]
62. Zhou, Q.; Ou, W.; Zheng, W. Research on Cloud Computing Security Technology under Computer Big Data Network. In Proceedings of the 2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 26–28 May 2023; pp. 1686–1691. [CrossRef]
63. Rindone, C. Sustainable Mobility as a Service: Supply Analysis and Test Cases. *Information* **2022**, *13*, 351. [CrossRef]
64. Van Der Schaft, A. Bisimulation of Dynamical Systems. In *Hybrid Systems: Computation and Control*; Goos, G., Hartmanis, J., Van Leeuwen, J., Alur, R., Pappas, G.J., Eds.; Springer: Berlin/Heidelberg, Germany, 2004; Volume 2993, pp. 555–569. [CrossRef]
65. Bian, G.; Abate, A. On the Relationship Between Bisimulation and Trace Equivalence in an Approximate Probabilistic Context. In *Foundations of Software Science and Computation Structures*; Esparza, J., Murawski, A.S., Eds.; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10203, pp. 321–337. [CrossRef]