Ukko: Resilient DRES management for Ancillary Services using 5G service orchestration

Charalampos Rotsos*, Angelos K. Marnerides†, Abubakr Magzoub*, Anish Jindal‡,
Paul McCherry*, Martin Bor*, John Vidler*, and David Hutchison*
* School of Computing & Communications, Lancaster University, UK
{c.rotsos, a.magzoub1, p.mccherry, m.bor, j.vidler, d.hutchison}@lancaster.ac.uk
† School of Computing Science, University of Glasgow, UK
angelos.marnerides@glasgow.ac.uk
‡ School of Computer Science & Electronic Engineering, University of Essex, UK
a.jindal@essex.ac.uk

Abstract—The modern Smart Grid (SG) requires the adequate exploitation of Ancillary Services (AS) in order to dynamically support evolving energy demands, ensure grid stability and optimize energy trading transactions. Distributed Renewable Energy Sources (DRES) are considered as the most promising avenue to underpin the dynamic composition of AS. Nonetheless, the inherent legacy, distributed and resource-constrained properties of DRES deployments trigger a plethora of challenges with respect to the underlying end-to-end (E2E) data communication performance with direct implications on AS orchestration. Hence, assuring resilient and scalable E2E AS provisioning is a highly challenging task requiring advanced networking mechanisms. This work goes beyond architectural, policy-level and theoretical suggestions of how 5G can be utilized in the SG context and provide a proof-of-concept, system for DRES management. We introduce Ukko; an open-source, 5G network service design facilitating the programmable, "in-network" orchestration of DRES management, supporting real-time AS application requirements. Experiments conducted over a UK-wide 5G testbed using real use-case scenarios demonstrate that the proposed solution can assure scalable and resilient DRES management at the likely occurrence of data communication challenges.

I. Introduction

The key behind SG grid deployments and the envisaged Internet of Energy (IoE) as expressed by key organizations (e.g., ENTSO-E[1]) lies with the decentralization of the conventional power grid. Ultimately, the aforementioned aim is mapped with the smooth integration of Distributed Renewable Energy Sources (DRES) such as wind-farm deployments, solar power plants and tidal installations. By contrast with the setup of conventional grids that rely purely on centralized fossil fuel-based power generation and transmission networks, DRES are the greener alternative. Apart from their low carbon footprint and adhering to environmental policies, DRES' inherent properties facilitate a basis for the integration of Ancillary Services (AS) (e.g., voltage/frequency regulation, inertial response, reactive power) such as to dynamically serve grid demands in the context of peak-load operation, energy trading and grid balancing.

The rapid adoption of AS by Distribution System Operators (DSOs) and Transmission Control Operators (TSO) via new grid application domains (e.g., Virtual Power Plants) has raised the bar on the underlying DRES data communication requirements. A number of AS have strict latency and liveness requirements on the data exchange of DRES with centralized and sometimes cloud-based optimization processes such as to prevent the complete grid system from collapsing [1]. Increased network latency and jitter can result in a ripple effect on the grid system and may eventually lead to blackouts or brownouts [2]. Hence, DRES integration poses an important real-time control and management challenge on the communication layer. Therefore, the effective use of DRES requires scalable and continuous end-to-end monitoring underpinned by adequate network instrumentation and management mechanisms.

This paper argues that in order to deliver scalable, timely and resilient monitoring capabilities for AS, utility networks require better integration with the 5G infrastructure. Specifically, we present Ukko; an open-source network service design supporting resilient and scalable delivery of monitoring information from DRES deployments to cloud-based storage services. The service exploits the programmability and orchestration capabilities of modern 5G infrastructures and provides an open-source service descriptor using the ETSI NFV MANO data model, supported by production 5G service orchestrators. To quantify the benefits of 5G technologies on monitoring, we deployed the Ukko service over an experimental UK-wide 5G testbed supported by the Initiate project and we demonstrate how Ukko can ensure continuous and timely response to major outage scenarios. Specifically, we highlight the following contributions entailed with the proposed Ukko service:

- An end-to-end (E2E) 5G service design allowing resilient monitoring of DRES system and timely management of AS. Ukko is backwards-compatible

[1]ENTSO-E view on IoE: https://www.entsoe.eu/events/2019/03/19/internet-of-energy/

978-1-7281-6127-3/20/$31.00 ©2020 IEEE

with existing application stacks and does not require end-host modification.
- An open-source straw-man service implementation using off-the-self software and standardized service description data models. The Ukko implementation is compatible with existing commercial and experimental 5G network orchestration platforms.
- AS monitoring and measurement services relying on DRES deployments can be restored in less than 2 minutes with negligible data losses at the face of severe communication failures.

The remainder of this paper is structured as follows: Section II discusses related work whereas Section III elaborates on communication requirements related to DRES management and AS provisioning. Section IV introduces the Ukko design principles followed by the Ukko system architecture in Section V. The Ukko service evaluation is presented in Section VI whereas Section VII summarizes and concludes this paper.

II. Related Work

The management and control of SGs through 5G-enabled infrastructures has been the topic of discussion within several :wstudies in the last decade. Cankaya et al. [3] highlight the benefits of Software Defined Networking (SDN) and Network Function Virtualization (NFV) on ensuring high reliability, availability and resilience of the various SG applications. With a similar mindset, the work in [4] proposed a reliable backup method using NFV nodes. However, none of the two studies looked explicitly at the instrumentation of measurement and monitoring of DRES deployments for provisioning of ancillary Services (AS). The latter was the point of interest in [5] in which SGs act as a use-case for 5G technologies to optimize the performance of 5G while providing smart energy as a service.

Within the context of DRES management, Reddy et al. [6] note that the integration of DRES deployments in SGs is crucial and advanced 5G-based data communication architectures are essential. In parallel, the work in [7], emphasizes the need for adequate DRES control and management in order to serve real-time SG services. Moreover, the work in [8] highlights challenges for AS provisioning and how 5G technologies can potentially resolve them. Nonetheless, there was no tangible system implementation or performance evaluation but rather discussion on design choices. Hassan et al. [9] perform a theoretical assessment for AS support and demonstrating how energy can be optimized through 5G-enabled cellular services. However, there was minimal reference and no assessment on the system-wide view related to DRES management and control over the communication infrastructure. On the other hand, the work in [10] investigated the applicability of SDN technologies in the context of AS provisioning via Plug-in-Electric Vehicles (PEVs). Despite the intriguing outputs from that study, the focus was tailored on PEVs, thus lacking generality with respect to system trade-offs for generic DRES deployments as we focus herein. Similarly, the work in [11] focused on how 5G can enable Vehicle-To-Grid (V2G) networks to facilitate AS. Due to the explicit focus on V2G there was lack of focus on the back-end performance analysis with respect to the underlying instrumentation and measurement mechanisms underpinning real-time DRES management for AS provisioning as aimed in this paper.

III. DRES Management & AS Provisioning

Monitoring and measurement requirements of DRES management overlap with many real-time AS provisioning requirements. Given the various specifications and the variety of application-level requirements derived from AS provisioning, monitoring and measurement functions need to be resilient, secure, available and reliable [8]. Typically, heterogeneous DRES deployments in modern SGs face several resilience, security and scalability challenges that consequently affect the E2E availability and reliability of any management function underpinning AS provisioning [12]. There exists a variety of reasons for such challenges to emerge ranging from the diversity on standards up to the range of machinery and software with minimal embedded security residing in such deployments [7], [8], [12]. In parallel, E2E network QoS is affected inherently by the need of large SGs to rely on multiple Internet providers for connectivity, since multi-domain resource control is limited in existing Internet protocols.

Current trends on E2E AS provisioning rely heavily on cloud services. Hence, cloud-based web interfaces are the primary access gateway to monitoring data. Nonetheless, current solutions consist of monolithic cloud-based solutions where complex data processing is performed at the cloud service with minimal exploitation of the resources offered by the network infrastructure [8], [12]. AS provisioning is decomposed into various stages and their envisaged real-time operations cannot fully depend on conventional cloud-based approaches. The continuous integration of DRES deployments will only create bottlenecks, if traditional E2E communication mechanisms are employed, regardless of any advanced data optimization techniques employed on the cloud side [8]. Hence, requirements for "in-network" processing are essential such as to enhance the aspect of scalability and strengthen the domains of availability, reliability and resilience.

Design processes for future DRES management schemes will have to conform with 5G technologies where network programmability is a core element. By exploiting technologies such as Network Function Virtualization (NFV) it is feasible to develop adaptive management schemes where "in-network" data processing can facilitate the basis for adequate and real-time resource allocation. Thus, ensuring that network resources are managed under a scalable fashion and AS provisioning applications are available even during the occurrence of communication failures.

Also, "in-network" data processing will allow SGs to seamlessly roll-out new capabilities for their communication infrastructure by updating intermediate network functions and with no need for end-node firmware upgrades.

IV. Ukko Design Principles

The term 5G describe the re-design effort of mobile networks to improve scalability, resource control and programmability and support new industrial applications and verticals. In addition to new physical layer protocols, 5G standards explore how new networking paradigms, like NFV and SDN, can improve service delivery. An example of the 5G architecture is depicted in Figure 1.

At the physical layer, new control abstractions improve resource virtualization and service isolation via slicing. In parallel, 5G technologies use "in-network" computation via Virtual Network Functions (VNF) to improve end-to-end network service performance.

In parallel, to simplify the network control and management, network services follow a service-oriented design approach. Similar to cloud microservices, 5G services encode their architectures into deployment templates using data models, like the ETSI NFV Management and Orchestration (MANO) data model [13]. Service descriptors describe the configuration and connectivity requirements of network functions and allow the orchestrator to automate service management.

5G infrastructures can greatly improve the service delivery for a wide range of applications. Nonetheless, in order to improve application performance without sacrificing efficiency, they require an iterative design process. A service must identify the individual KPIs of an application and identify VNFs that meet these requirements.

Ukko exploits the rich slicing capabilities of 5G networks to deliver E2E connectivity with resource guarantees. Furthermore, in order to secure AS monitoring from cyber-physical threats and infrastructure failures, we employ VNFs to assist the information delivery and improve resilience. Ukko exploits the north-bound interface of the 5G orchestration layer to simplify the management and offload the monitoring of the E2E service to the underlying management subsystems. Finally, AS stakeholders and DRES owners can easily interact with the Ukko service and define their explicit AS and DRES management service requirements using the NFV MANO data model [13].

V. Ukko System Architecture

Ukko is a network service architecture, designed to improve the resiliency and security in the delivery of monitoring information from DRES deployments to cloud storage using 5G technologies. The design of the service aims to address two challenges. Firstly, the service aims to improve the monitoring scalability of DRES deployments and further allow efficient resource utilization by rapidly adapting to varying workload. Secondly, the service is designed to allow backwards compatibility with existing utility devices without any firmware change. Ukko is an open network service, using of-the-self network functions and standardized service description data models.

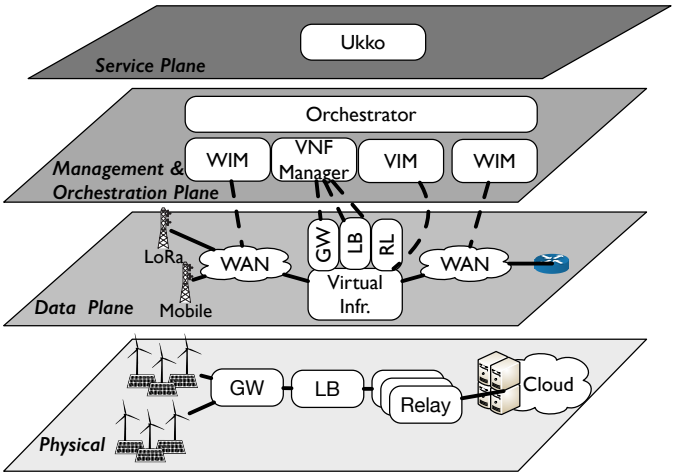

Fig. 1. Overview of the Ukko DRES Management Service framework for 5G-enabled SmartGrids.

The architecture of the Ukko service is depicted in Figure 1. The service model considers a scenario where varying DRES deployments, like smart meters, transformers and wind turbines, are mapped as data aggregators and transmit monitoring information to a central cloud storage service. Each aggregator utilizes a local hardware interface to collect monitoring information from a metering device and uses a 5G interface (NewRadio, mmWave, LoRA) to transmit them to the central cloud service via a 5G operator and using the TCP/IP protocol stack. Monitoring information can use an application layer protocol, like HTTP/HTTPS, Modbus/TCP or IEC 61850, to encapsulate the monitoring data. The available monitoring information on the cloud storage can be accessed by individual TSO/DSO/AS providers using a technology/vendor-specific cloud web interface. Ukko exploits the inherent programmability of the 5G infrastructure and enables in-network data processing for E2E scalability and resilience functions and in parallel encrypts data related to the storage service.

1) Ukko Virtual Network Functions: As exhibited in Fig. 2, the Ukko service employs three main VNFs, each providing a specific management operation, while a service controller uses the orchestration North-bound API to monitor and manage any service faults. The secure gateway is the first layer of resilience and protection of the service against cyber threats. Specifically, the VNF inspects passively traffic using an Intrusion Detection System (IDS) to detect possible attacks, while a stateful firewall is deployed to control service access and mitigate scanning attacks. Valid packets are forwarded to a load balancer VNF, which allows seamless service adaptation to dynamic workload profiles. The load balancer exposes a single TCP port to all aggregators and HTTP requests

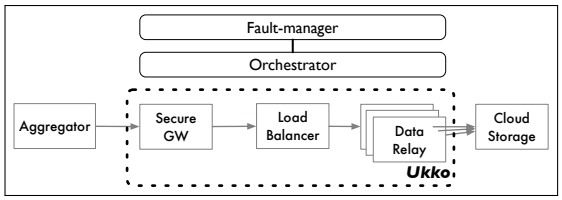

Fig. 2. Ukko implementation components. The service consists of three VNF types and passively processes and cache data between DRES data aggregators and the cloud storage service.

are transparently distributed to multiple backend servers. Although our implementation relies on the HAProxy implementation, any load-balancing service can be used to distribute HTTP requests between backend servers based on a load balancing algorithm that takes under consideration multiple parameters, including the load of individual servers and network manager preferences. In parallel, the lifecycle model of the load balancer VNF allows the orchestrator to dynamically change the backend server pool to ensure high service rate and resource efficiency. Finally, the load balancer checks the liveness of backend servers and removes from the server pool any misbehaving or offline servers.

Finally, Ukko uses a relay VNF to protect from data loss during network outages [14]. The relay VNF is responsible to monitor the connectivity towards the storage server and to cache aggregator monitoring data during downtime. Cached requests are re-transmitted once connectivity is restored.

Unlike traditional hardware-based network functions, VNFs are more prone to failures, due to the complexity entailed in modern software platforms. In order to gracefully manage VNF failures, 5G orchestrators offer an extensive north-bound interface to external service controllers, to implement custom service monitoring and recovery policies. Ukko uses a service controller to actively monitor the health and performance of individual VNFs. Since existing 5G orchestrators and service description models support the application of scaling policies, the Ukko controller is primarily responsible to provide fault-management capabilities and repair VNF failures. The implemented controller embeds in the service description a series of VNF health checks, which are offloaded to the orchestration layer of the infrastructure and upon a failure detection, the controller performs VM restarts.

2) Ukko prototype: Ukko is implemented as an open-source network service using open-source software [2]. In order to simplify the design of the service, we assume that the monitoring data transmission mechanism is built on InfluxDB [15], a time-series database with an HTTP interface commonly used to store monitoring data. This mechanism can be replaced with alternative application layer protocols, by adapting accordingly the software of the relay VNF. The specification of our network service uses the NFV-MANO data model [13]. In addition, the secure gateway interface implements the Snort IDS [3] and the Linux iptables firewall. The load balancer VNF relies on the HAProxy [16] software and uses bash scripts to manage backend servers. Finally, the relay VNF uses the InfluxDB-relay [17] Go package . We have successfully deployed our service specification using Open Source MANO (OSM v6) [18] and the OpenStack cloud management software. Finally, we have implemented a custom Python service controller integrated with the OSM north-bound interface and its alert mechanism.

The Ukko orchestration capabilities are not limited to compute resources and can equally control network resources. The NFV-MANO model defines primitives to model the service network requirements and enforce them through the network management layer. The MANO model considers two primary network management scenarios: intra-DC connectivity, typically implemented using an SDN controller, and intra-DC connectivity, modeled via a Wide-area Infrastructure Manager (WIM) abstraction. We have successfully run multi-site Ukko deployment scenarios with guaranteed bandwidth, using the DPB WIM [19].

VI. Evaluation

The evaluation of the proposed Ukko solution depends on the deployment of Ukko over the UK-wide Initiate 5G testbed as explained in (§ VI-A). Ukko is discussed in terms of service scalability (§ VI-B) and resilience with respect to disruption tolerance under common failure scenarios (§ VI-C).

A. Experimental setup

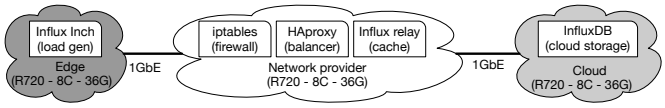

Fig. 3. Testbed topology. VNF images employ open-source software instances.

The UK-wide Initiate 5G testbed [4] interconnects multiple experimental 5G UK testbeds and allows experimenters to dynamically deploy network services and experiment with 5G compute and network programmability.

For our experimental scenarios, we allocate three servers (Dell R720, dual socket Intel Xeon E5-2630, 128GB RAM), in a topology depicted in Figure 3, in order to emulate the edge, core and cloud network layers. The servers are virtualized using KVM, and we configure a single-server OpenStack (Queens Series) configuration on each server. The edge server emulates the Ukko aggregator nodes. To generate high-rate data, we employ the

[2]https://github.com/DataPlaneBroker/Ukko

[3]Snort IDS: https://www.snort.org/

[4]http://initiate.ac.uk

InfluxDB inch software [20], an InfluxDB traffic generator, generating a steady-state stream of random data-series in fixed-size batches. The tool is multi-threaded and offers many traffic control parameters, including the number of emulated hosts, the total stream size and batch sizes.

The core server emulates a small-scale NFV infrastructure managed by a single service orchestrator (OSMv6) and hosts the Ukko service. The Ukko service specification uses the OSM policy module to generate HTTP alarms to the service controller when individual VNF instances fail and to implement a threshold-based scaling policy targeting individual VNF instances. In our experiments, we configure the policy to scale up the instances by one, if an NFV instance uses more than 70% of the CPU, whereas when the CPU utilization is below 30%, the VNF number is scaled down. Furthermore, the implemented policy uses a 5-minute cool-off period between scale operations in order to avoid oscillation. Scaling thresholds are user-defined and the suggested policy is used as a possible scenario. It is worth pointing out, that the relay VNF is the service bottleneck and was the target of all scaling operations. During operation, the gateway and the proxy VNFs did not experience more than 50% CPU utilization across all experimental scenarios. Finally, all VNF instances are built using the Ubuntu 18.04 cloud image, HAProxy v1.6.3 and the influxdb-relay v3.1.0. In addition, all VNF VM images use the Ubuntu 18.04 cloud image. Instances are connected using software bridges managed by the OpenStack neutron agent. The cloud server runs an InfluxDB server (v1.8) on an 8-core VM and emulates a cloud storage service.

B. Scalability assessment

In order to measure the scalability of the Ukko service we use the inch tool to generate a steady-state data stream conforming with data rates as observed in DRES deployments. We measure the impact of the scaling policy on the service throughput and the scaling latency.

Firstly, using the logging capabilities of the inch traffic generator, we measure the impact of the scaling policy on a typical deployment scenario. We configure the inch tool to generate a steady-state data stream, with a batch size of 1000 writes, using 32 client threads corresponding to DRES installations and write data to an InfluxDB server, relayed via our network services. During the experiment we collect the per-second throughput of the inch client, as well as the scaling command timestamps. Figure 4 presents the per-second service throughput of a single experimental run, as well the relative time of the scaling operations. Our analysis is restricted up to 5 replicas, as the server configuration experiences a noteworthy disk bottleneck and the performance gains were negligible. It is worth to point out that modern cloud storage services can easily scale to higher data-rates, by using the vast amount of datacenter resources.

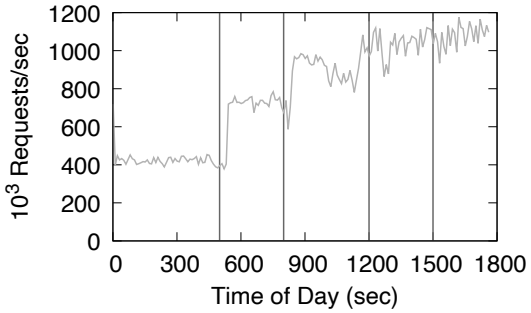

Fig. 4. The scaling policy of the Ukko service can detect and respond rapidly to communication data flooding events by scaling service monitoring resources every 300 seconds (vertical lines).

TABLE I
Latency of the OSM platform to respond to user-generated scale requests (10 repeats).

Operation	mean latency (sec)	std
VM boot	42	3.7
Request service	67	8.3

As evidenced, the scaling policy achieves a significant improvement in service throughput, achieving almost a triple of the service throughput. This improvement is achieved via 4 scaling operations of the relay VNF instance, as this VNF is the main bottleneck in the beginning of the experiment. We would like to highlight that the reduced improvement of the scaling operations and the increasing variability of the service throughput is a result of resource limitations of the experimentation hardware. The cloud host experienced increased IO latency due to disk bandwidth bottleneck for more than 3 relay instances.

To further understand the impact of the scaling mechanism, we conduct 10 runs and record the time required to boot a new relay replica. Specifically, we record the latency to boot a new VM, via the OSM framework, as well as the time to process the first write request, using kernel logging information. Table I present the mean and standard deviation of the aforementioned measured latencies, using 10 experimental repetitions. It is evident that Ukko under the OSM orchestrator, supports sub-second scaling, while new replicas require approximately 20 seconds to boot all the required services and start serving use content. These results can be further improved with appropriate tuning of the relay VNF base Linux image, while new container-based OSM backends for the Kubernetes platform, will allow faster VNF boot times.

C. Resilience assessment

The resiliency of the Ukko service is assessed with respect to disruption tolerance. Thus, we conduct a micro-benchmark to characterize the incurred disruption during VNF failures. Specifically, an Ukko instance is bootstrapped with two relay VNF instances and the

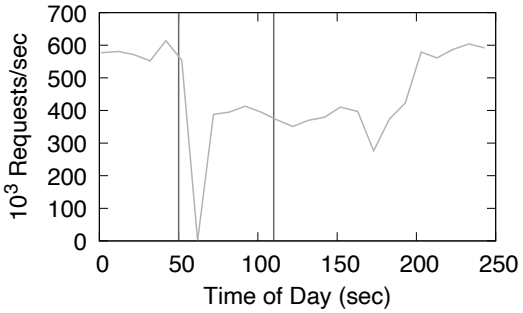

Fig. 5. Throughput of the Ukko service (relay cardinality=2) during a relay failure (t=40 sec). The OSM controller reports the failure to the fault-controller, which reboots the VNF (t=120 sec.) and the service recovers service performance (t=190).

throughput of the service using the measurement setup is obtained as described earlier (Section V). After 40 seconds of operation, one of the relay VNFs is turned-off in order to emulate a VNF failure. As already mentioned, the Ukko service controller implements an HTTP server to process VNF failure alarms and uses the OSM Rest-API to heal the network service.

The Ukko service throughput during the disruption scenario is presented in Figure 5. Worthy to mention is that since OSM uses an active VM status monitoring mechanism running every 30 seconds, some additional time to report the node failure to the controller is required. Consequently, the recovery process requires approximately 80 seconds to complete, while the throughput requires approximately another minute to increase the service throughput and saturate the new replica. Although the throughput is significantly reduced during the failure, there is negligible data loss (8 out of 112m points).

VII. Conclusion

Distributed Renewable Energy Sources (DRES) in the modern Smart Grid (SG) underpin fundamental processes related to SG stability as well as with energy trading market functions realized via Ancillary Services (AS). The composition of AS functionalities requires adequate real-time monitoring and measurement of diverse DRES with varying system and network characteristics. Currently, conventional cloud-based AS provisioning mechanisms are not able to cope with the resilience and scalability requirements invoked explicitly within end-to-end (E2E) DRES management affecting real-time AS provisioning. In this work we present and evaluate the applicability of Ukko; a programmable, open-source and 5G-enabled network service system prototype. We assess Ukko over a UK-wide experimental testbed and demonstrate via two use cases that its "in-network" service orchestration and management properties can facilitate scalable and resilient E2E DRES management. Arguably, the reported prototype can pave the path towards the smooth and practical integration of 5G capabilities over next generation SGs.

Acknowledgment

This work has received funding from the EU's Horizon 2020 RIA "EASY-RES" project under grant agreement No 764090 and the UK EPSRC TOUCAN (EP/L020009/1) and INITIATE (EP/P003974/1) projects.

References

[1] IEEE, "Ieee standard for scada and automation systems," IEEE Std C37.1-2007 (Revision of IEEE Std C37.1-1994), pp. 1–143, May 2008.
[2] R. Noskov, I. Petrović, H. Glavaš, D. Šljivac, and T. Barić, "Analysis of possibilities of alleviating repercussions caused by large disturbances in the power system," IET Conference Proceedings, pp. 31 (5 pp.)–31 (5 pp.)(1), January 2018.
[3] H. C. Cankaya, "Software defined networking and virtualization for smart grid," Transportation and Power Grid in Smart Cities: Communication Networks and Services, pp. 171–190, 2018.
[4] Y. Luo, Y. Luo, X. Ye, J. Lu, and S. Li, "Reliability-based and qos-aware service redundancy backup method in iot-based smart grid," in Artificial Intelligence and Security. Cham: Springer International Publishing, 2019, pp. 588–598.
[5] H. C. Leligou, T. Zahariadis, L. Sarakis, E. Tsampasis, A. Voulkidis, and T. E. Velivassaki, "Smart grid: a demanding use case for 5g technologies," in 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). IEEE, 2018, pp. 215–220.
[6] K. Reddy, M. Kumar, T. Mallick, H. Sharon, and S. Lokeswaran, "A review of integration, control, communication and metering (iccm) of renewable energy based smart grid," Renewable and Sustainable Energy Reviews, vol. 38, pp. 180–192, 2014.
[7] A. Jindal, A. K. Marnerides, A. Gouglidis, A. Mauthe, and D. Hutchison, "Communication standards for distributed renewable energy sources integration in future electricity distribution networks," in ICASSP 2019. IEEE, 2019, pp. 8390–8393.
[8] G. Bag, L. Thrybom, and P. Hovila, "Challenges and opportunities of 5g in power grids," CIRED - Open Access Proceedings Journal, vol. 2017, no. 1, pp. 2145–2148, 2017.
[9] H. Al Haj Hassan, D. Renga, M. Meo, and L. Nuaymi, "A novel energy model for renewable energy-enabled cellular networks providing ancillary services to the smart grid," IEEE Transactions on Green Communications and Networking, vol. 3, no. 2, pp. 381–396, 2019.
[10] N. Chen, M. Wang, N. Zhang, X. S. Shen, and D. Zhao, "Sdn-based framework for the pev integrated smart grid," IEEE Network, vol. 31, no. 2, pp. 14–21, 2017.
[11] M. Tao, K. Ota, and M. Dong, "Foud: Integrating fog and cloud for 5g-enabled v2g networks," IEEE Network, vol. 31, no. 2, pp. 8–13, 2017.
[12] A. Jindal, A. K. Marnerides, A. Scott, and D. Hutchison, "Identifying security challenges in renewable energy systems: A wind turbine case study," in ACM e-Energy 2019. New York, NY, USA: Association for Computing Machinery, 2019.
[13] ETSI NFV SIG, "Network Function Virtualization (NFV) Management and Orchestration, GS NFV-MAN 001 v0.8.1," 2014.
[14] F. Wang, Z. M. Mao, J. Wang, L. Gao, and R. Bush, "A measurement study on the impact of routing events on end-to-end internet path performance," SIGCOMM Comput. Commun. Rev., vol. 36, no. 4, p. 375–386, Aug. 2006.
[15] "InfluxDB," https://www.influxdata.com/.
[16] "HAProxy," http://www.haproxy.org/.
[17] "InfluxDB Relay," https://github.com/influxdata/influxdb-relay.
[18] "Open Source MANO," https://osm.etsi.org/.
[19] S. Simpson, A. Farshad, P. McCherry, A. Magzoub, W. Fantom, C. Rotsos, N. Race, and D. Hutchison, "Dataplane broker: Open wan control for multi-site service orchestration," in 2019 IEEE NFV-SDN, 2019, pp. 1–6.
[20] "Influx DB inch," https://github.com/influxdata/inch.