DOI: 10.15439/2024F7601
ISSN 2300-5963 ACSIS, Vol. 39

Dashboard User Interface (UI) Implementation for Remote Critical Infrastructure Inspection by using UAV/Satellite in times of Pandemic

Romaio Bratskas
0000-0002-4337-5828
Skyld Security and Defence ltd, Alkaiou 23, 2404, Nicosia, Cyprus
Email: rb@skyld.com.cy

Dr. Dimitrios Papachristos
0000-0002-3453-0022
Skyld Security and Defence ltd, Alkaiou 23, 2404, Nicosia, Cyprus
Email: d.papachristos@skyld.com.cy

Dr. Petros Savvidis
0009-0000-0444-2314
Skyld Security and Defence ltd, Alkaiou 23, 2404, Nicosia, Cyprus
Email:p.savvidis@skyld.com.cy

Dr. George Leventakis
Dept. Shipping, Trade & Transport, Univ. of Aegean, Korai St. 2A, 82132, Chios, Greece
Email: gleventakis@aegean.gr

Enea Qerama
Skyld Security and Defence ltd, Alkaiou 23, 2404, Nicosia, Cyprus
Email: e.qerama@skyld.com.cy

George Dahrouje
Skyld Security and Defence ltd, Alkaiou 23, 2404, Nicosia, Cyprus
Email: d.dahrouje@skyld.com.cy

***Abstract*—In times of pandemic, many activities of the society, economy are minimized due to the risk of transmission. In particular, in the period of Covid-19, with the implementation of the Lockdown, many infrastructure monitoring and maintenance activities were suspended to prevent the spread of the virus. In essence, pandemics of highly contagious viruses may directly impact the critical infrastructure monitoring sector. More specific, in the context of monitoring critical infrastructure through satellites and UAVs, data processing involves extracting valuable insights, detecting potential threats, and assessing the overall condition of the infrastructure. This processed information is then used to make informed decisions regarding maintenance, security measures, and response strategies to mitigate risks and safeguard the critical assets. In this paper, we present a user interface dashboard dedicated to inspecting the critical infrastructure events captured from UAV or satellite. The design and architecture of the Dashboard User Interface its primary goal continues to be delivering real-time images to users, showcasing areas/components/points of failure in critical infrastructure, including damaged components, structural issues, corrosion, vegetation obstruction etc.**

***Index Terms*—remote inspection, critical infrastructures, pandemic era, User Interface dashboard**

I. Introduction

CURRENTLY, remote inspections are not only considered simply as a backup in case of major disruption (i.e. pandemic), but as a strategic and sustainable solution for the future. Critical Infrastructures (CIs) across industries have realized the benefits of harnessing new technology to streamline inspection processes, improve accessibility, and enhance overall operational efficiency. The SUNRISE platform aims to provide users with real-time information on critical infrastructure by analyzing image and video feeds. This information includes details on damaged components, structural issues, corrosion, and obstructing vegetation. The tool utilizes AI-assisted components to process data from satellite and UAV feeds and present it in the user interface. In addition, the use of UAVs is important and more convenient than the use of satellites thanks to the technological evolution of UAVs (communications, propulsion/navigation, swarms etc.) providing high reliability in data recording [1][2][3][4].

In this context, the user interface (UI) tool of the SUNRISE platform for remote infrastructure inspection provides users with real-time images of areas, components, or points of failure in critical infrastructure, such as damaged components, structural issues, corrosion, obstructing vegetation, and more.

In this paper, we present a user interface dashboard tool dedicated to inspecting the critical infrastructure events captured from UAV or satellite and an architectural paradigm including back-end design. UI design is a complicated process that requires detailed analysis of operators' performance and preference. Furthermore, developments in technology require an understanding of "trust" aspects of interaction [5].

II. Background

Critical infrastructure (CIs) is a critical component of the economy in all countries with technological development [6]. Critical infrastructure, as defined by [7], refers to systems that include industries, organisations with distribution & transportation capabilities that provide a continuous flow of essential services vital to the defence and economic security of society. Typically, these systems are also called lifeline systems.

This work was supported by SUNRISE H2020 (EU Project)

Critical infrastructure incorporates various systems such as electric power systems, telecommunications, water treatment and supply, natural gas supply, transportation systems, and healthcare systems [8]. The term critical infrastructure is defined in Section 1016(e) of the USA Patriot Act of 2001 as those “systems and goods, both physical and virtual, so vital to the nation that their malfunctioning or destruction would produce a debilitating impact on the security of citizens, on the economic security of the nation, on national public health and on any combination of the above” [9].

The CIs framework is constantly changing to reflect current concerns and respond to new issues, notably security and resilience [10]. At the same time, the quantity and variety of CIs have increased dramatically in recent decades [11]. Moreover, their protection from the various shocks and stresses, and ensuring their continuous functioning have become a basic priority [12]. In addition, reducing damages on CIs is critical in societal well-being and achieving resilience and sustainability [13].

One of the great challenges is to make CIs truly sustainable from all points of view: economic, social, environmental. CIs plays a key role in directly and indirectly impacting progress on achieving sustainable development goals (SDGs). The recent pandemic crisis showed that the future is uncertain for humanity. Therefore, societies should prepare themselves to address the weaknesses that make us vulnerable to such important risks. The sustainability, reliability and security of CS are issues of huge importance with international resonance that require a new approach. Therefore, several techniques and methods (based in science & technology) have been developed to ensure CIs, taking into consideration different issues such as human error, maintenance policies, energy, water supply, healthcare protection and emergency transportation in case of disaster [14]. CIs are, therefore, those material resources, services, IT systems, networks and infrastructure assets that, if damaged or destroyed, would cause serious repercussions on the crucial functions of society, including the supply chain, health, security and the economic or social well-being of the state and the population [15].

That's why the use of technologies such as UXVs (i.e. UAV, UGV) are a new approach to inspecting critical infrastructure. For example, the use of UAVs has significant potential to increase efficiency, reduce O&M costs, improve operational safety as well as minimize production downtimes in the wind energy sector 16]. Essentially, UAVs are a useful data collection tool for capturing large volumes of data, such as high-resolution images and other information in a relatively short period of time. This information can be used to monitor progress or changes in a critical infrastructure for its sustainability and safety [4].

III. Tool Architecture

A. High Level Design

The Tool consists of the following parts (Fig.1): the Web Application-Dashboard UI (including a and b), the MQTT Bus and the Backend Coordinator (including a Data Base and Reporting subsystem). As main functionalities, the user interface (UI) tools and dashboards furnish a dynamic web platform, empowering end-users to have a complete engagement with all inspection infrastructures’ components of SUNRISE system. This module incorporates contemporary tools and presents a map where during inspection, constantly refreshed with real-time data on inspection point. This Tool enhances the inspection process, enabling end-users to accomplish a full range of inspect activities, receive event-driven messages, that means anomalies detection with use of AI algorithms-based methods.

More specific, the selected web application architecture is described through the numbered bus lines as follows:

1. Incoming messages/events from UAV/Satellite systems are received. All this data is routed through an MQTT bus system. Within this system, the data is systematically queued, ensuring a sequential flow.
2. The Backend Coordinator processes all incoming messages/events. It retrieves the data at the front of the MQTT queue.
3. All incoming messages/events are internally stored in the Backend Inventory (MongoDB server).
4. The Backend Coordinator sends live or historical data to the Dashboard UI for visualization and responds to historical data requests from the Dashboard UI.
5. The Dashboard UI communicates with the Google Maps infrastructure to render maps, markers, points of interest, and heat maps, among other elements.
6. The Backend Coordinator sends requests to the Reporting Subsystem in order to compile the requested data and then receives the results.
7. The Reporting Subsystem and the Backend Inventory communicate with each other in order to process the requests, and subsequently transmits the results to the Backend.
8. The Dashboard UI obtains an Access Token from the Identity Server to access backend APIs. Access to the UI is exclusively granted to authorized users, with authentication and authorization handled by a dedicated Authentication/Authorization unit, responsible for controlling user access and logging into the application.
9. Additional public services can offer crucial meteorological data, weather forecasts, maritime information, alerts, and more for visualization within the Dashboard UI.

The connection between the Backend Coordinator and the MQTT system is bidirectional. If any data needs to be transmitted from the application outward, the Backend Coordinator places it in MQTT, within the corresponding queue.

B. Backend Coordinator

The proposed Tool incorporates the following services:

- *MongoDbService.* This background hosting service is responsible for writing all events received from the corresponding Detection Services to the "UAV" and "SAT" collections.

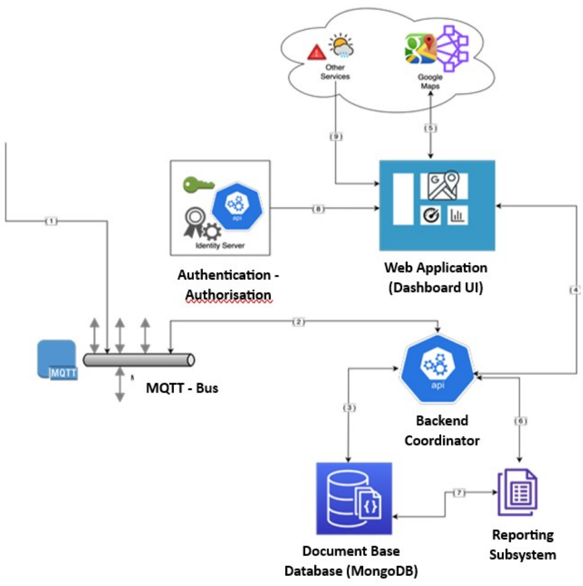
Other Services
Google Maps
Identity Server
Authentication - Authorisation
Web Application (Dashboard UI)
MQTT
MQTT - Bus
Backend Coordinator
Document Base Database (MongoDB)
Reporting Subsystem

Fig 1. Tool’s Architecture Diagram

- *MqttSubscriberService.* This background hosting service is responsible for listening to the MQTT topics and forwarding these messages to the Dashboard UI using WebSocket communication for real-time event presentation. It utilizes the MongoDbService to store these messages in the Document-Based Database (MongoDB) Backend Inventory.
- *ReportingService.* This background hosting service is tasked with generating filters based on criteria selected by end users on the Dashboard. It listens to user requests from the Dashboard for searching and reporting purposes. Upon receiving these requests, it executes queries on the Document-Based Database (MongoDB) Backend Inventory through the MongoDbService of the Backend Coordinator Service. Subsequently, it returns the results to the Dashboard Web App for further visualization and exporting functionalities.

C. Authentication Procedure

Authentication, Authorisation and Audit Logging component, is responsible for intelligently controlling access to UI tools system functions and interfaces (both GUI and REST-API), enforcing policies, and keeping an audit trail of events happening. Based on assigned roles, authenticated users are able to access different UI system functions and interfaces. The audit logging mechanism log several types of information that the system generates during normal execution, such as data changes and actions/commands invoked by the end-users. Structuring the UI web application to support a security token service (Authentication, Authorization and Audit Logging component) leads to the architecture and protocols shown in Figure 2.

This implementation enhances security by ensuring secure authentication, fine-grained authorization, token-based security, secure communication, client credential management, token revocation, and adherence to standardization and best practices. In addition, the Token-based security is a crucial aspect, as it relies on access tokens for secure communication between the Dashboard UI client and the Backend Coordinator backend service. These tokens are short-lived and cryptographically signed, minimizing the risk of unauthorized access and data breaches. Access tokens serve as temporary credentials, granting access to protected resources for a limited duration. Upon expiration, the Dashboard UI client obtains new tokens through the OAuth authentication process, reducing the window of vulnerability and enhancing security. Secure authentication is achieved through the use of OAuth, which replaces the direct sharing of sensitive credentials like usernames and passwords with the issuance of access tokens. These tokens serve as proof of authentication and are included in subsequent API requests to the Backend Coordinator backend service, ensuring secure access to protected resources. The Identity Server issues access tokens during the OAuth authentication process, providing a secure and efficient way to authenticate users.

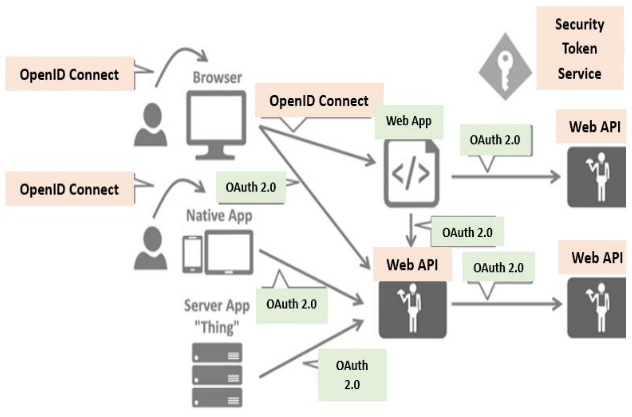

Fig 2. Security Token Architecture and Protocols

D. Two-Factor Authenticator of Dashboard UI

To enhance the security access of the Dashboard UI, a Two-Factor Authentication feature using Google Authenticator has been integrated with the existing Identity Server OAuth service. Two-factor authentication (2FA) on web applications works as follows: The user first logs in with their username and password, which represents the first authentication factor - something they know. After successfully entering the username and password, the web application then prompts the user to provide a second form of authentication, such as a one-time code sent to their registered mobile device. This one-time code represents the second authentication factor - something the user has. The user receives the one-time code, typically via SMS or a mobile app, and enters it into the web application to complete the login process. Once the user provides the correct one-time code, they are granted access to the web application.

E. WebSocket of Dashboard UI

In this tool, real-time data presentation between the Backend Coordinator and the Dashboard UI is facilitated through SignalR, a library that implements the WebSocket protocol. This implementation prioritizes security to safeguard sensitive information exchanged in real-time.

IV. MQTT Integration

Within the Dashboard UI, an MQTT architecture is utilized with a central MQTT broker facilitating a publish-subscribe mechanism for data ingestion from three distinct sources: UAV platform, satellite, and potentially legacy systems. The integration involves the Satellite Component, which is part of the Backend Coordinator system, interacting with the Dashboard UI to facilitate the prediction process based on user-defined areas of interest (Fig.3).

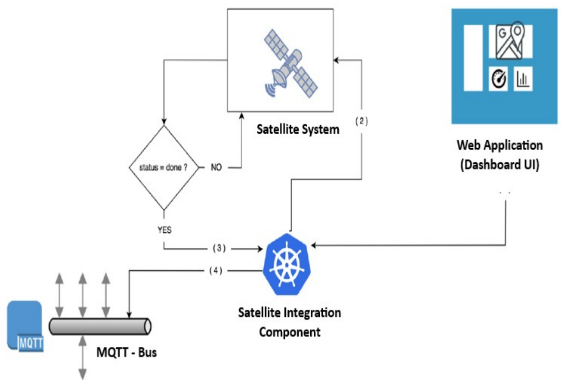

Fig 3. Satellite Component Diagram

As far as, the integration which involves the UAV Component, contains a addition to the Dashboard's internal infrastructure, collaborating with the UAV Platform and UAV Detection System to enhance the detection process (Fig.4).

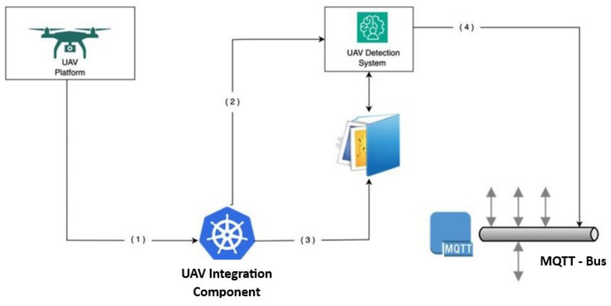

Fig 4. UAV Integration Component Diagram

V. Deployment

The Eclipse Mosquitto MQTT broker and the MongoDB database are key components of the overall system architecture, providing the necessary messaging and data storage capabilities to support the various subsystems and the Dashboard UI. Specifically:

- *MQTT Bus Service:* this involves installing the Eclipse Mosquitto MQTT broker, version 5 or 3.1.1, on the cloud platform, configured to listen on default ports (1883). The initial setup of the broker includes creating two main topics: "UAV" for use by the UAV detection subsystem and "SAT" for use by the Satellite Component. Both subsystems publish the results of their detection processes to these topics.
- *Document-Based Database (MongoDB) for Backend Inventory:* this entails installing the latest version of MongoDB on the cloud platform, configured to listen on default ports (27017). The initial setup of the database includes a database named "Sunrise" with two collections: "UAV" for storing all detection events from the UAV detection subsystem and "SAT" for storing all detection events from the Satellite Component. These events are later utilized for reporting and historical data visualization on the Dashboard UI.

In addition, in Tool's design, a comprehensive suite of security features has been integrated to enhance the integrity, confidentiality, and reliability of our MQTT communication protocol. These measures collectively fortify the security posture of our system, ensuring data protection and secure communication channels. The next figure shown the integration diagram with MQTT Broker of this tool:

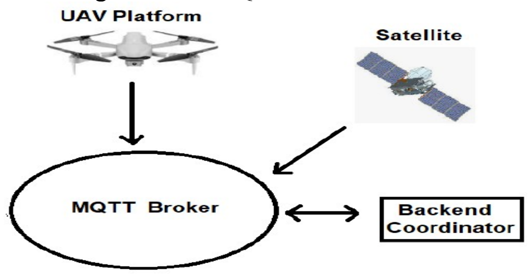

Fig 5. Integration Diagram

Finally, the Integration Process described as follow: The Publishers establish a connection to the MQTT broker and authenticate themselves using credentials. Once connected, publishers can start sending their messages. The MQTT broker receives these Messages containing data and topic information and temporarily stores them. Subsequently, the MQTT broker routes the messages to subscriber based on subscriber's interest. On data integration the subscriber receives the messages that containing relevant data and proceeds as needed. Integration can involve storing data in databases, triggering actions, generating notifications, or updating visualizations.

On Data validation in an MQTT system, a crucial step is to ensure that the incoming data from publishers is accurate, consistent, and conforms to the expected format. This helps prevent erroneous data from being distributed to subscribers and ensures the overall integrity of the system. Data validation in an MQTT system is based on several parameters which are Payload Format Validation, Subject Validation, Data Range and Constraints, Message Size Validation, Protocol Validation, Quality of Service (QoS) and Identification, Flag Validation Preservation, Security Checks and Error Handling.

VI. Dashboard Mockups

The SUNIRISE login page presents a secure gateway to access the platform. Users input their credentials – a unique combination of username and password – to verify their identity. The page's design is intuitive, with fields for entering credentials prominently displayed. Once verified, users gain authorized entry, unlocking the platform's features and personalized /CI related content. In case of forgotten credentials, the page also provides options for password recovery or account assistance. After granting access to the SUNRISE platform, the user navigates to the first page with the GIS Map and relevant layers of the infrastructure of their responsibility. On the right side a list of all the events that have been detected is presented with some fundamental information regarding each specific event.

By clicking the marker on the map or an event on the Events List on the right side, the annotated images from the inspected infrastructure are shown on a pop-up window in the main screen of the GIS (Fig.6). This pop-up window also presents the fundamental info such as event type, time stamp, location, source of inspection etc.

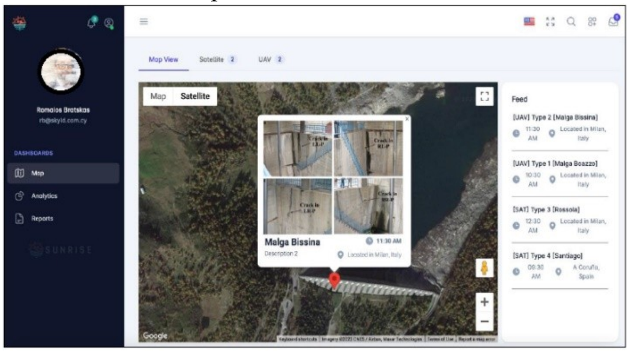

Fig 6. Event Presentation

Finally, for example, we can have the list of the events by source category (Satellite – UAV)(Fig.7).

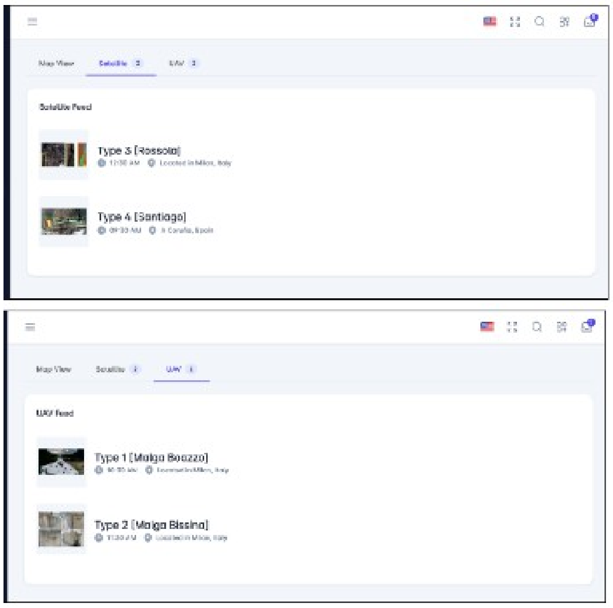

Fig 7. List of the events by source category (Satellite, UAV)

VII. Conclusion

The primary goal of the user interface (UI) for remote infrastructure inspection is to provide users with real-time images of critical infrastructure, highlighting areas, components, or points of failure such as damaged parts, structural issues, corrosion, and obstructing vegetation. This is achieved by analyzing image and video feeds and presenting outputs from AI-assisted components into the UI. The design and implementation of the user interface as well as the inspection functionalities provided that required by SUNRISE project, have been based on the following components:

- Two inspection data sources which are a UAV platform and satellite imagery. Both systems through AI technologies can detect the types of problems CI operators are concerned about.
- The interconnection and exploitation of any legacy systems which may at the CI.
- The visualization of the imports in the form of lists of events. These imports are data that have already been annotated by the corresponding inspection tool.
- The reporting services where statistics about the inspections as well as accessing historical information will be provided – if existing.

Acknowledgment

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This research was funded by SUNRISE H2020, which has received funding from the European Union’s Horizon Europe research and innovation programme under grant agreement no. 101073821.

References

[1] Jaroshaw, M. (2022). Location Accuracy of a Ground Station based on RSS in the Rice Channel. Proceedings of the 17th Conference on Computer Science and Intelligence Systems, pp. 577-80, dx.doi.org/10.15439/2022F15
[2] Buczyński, H.Pisarczyk, P. and Cabaj, K. (2022). Resource Par titioning in Phoenix-RTOS for Critical and Noncritical Software for UAV systems. Proceedings of the 17th Conference on Computer Science and Intelligence Systems, pp.605-9, http://dx.doi.org/10.15439/2022F163
[3] Danilchenko, K. and M. Segal (2021). An Efficient Connected Swarm Deployment via Deep Learning. Proceedings of the 16th Conference on Computer Science and Intelligence Systems, M. Ganzha, L. Maciaszek, M. Paprzycki, D. Ślęzak (eds). ACSIS, Vol. 25, pages 1–7 (2021).http://dx.doi.org/10.15439/2021F001
[4] Adam, T. and F, Babic (2021). UAV Mission Definition and Implementation for Visual Inspection. Proceedings of the 16th Conference on Computer Science and Intelligence Systems, M. Ganzha, L. Maciaszek, M. Paprzycki, D. Ślęzak (eds). ACSIS, Vol. 25, pages 343–346 (2021) http://dx.doi.org/10.15439/2021F24
[5] A. Dillon, (2006). User Interface Design. DOI: 10.1002/0470018860.s00054
[6] A. Panda, J.N. Ramos, Making Critical Infrastructure Resilient: Ensuring Continuity of Service Policy and Regulations in Europe and Central Asia, 2020. www. undrr.org.

[7] W. Clinton, Presidential Decision Directive 63; The White House: Washington, DC, USA, 1998. Available online: fas.org/irp/ offdocs/pdd/pdd-63.htm (accessed on 29 April 2024).

[8] National Infrastructure Advisory Council (US); T. Noonan; E. Archuleta. The National Infrastructure Advisory Council's Final Report and Recommendations on the Insider Threat to Critical Infrastructures; DHS/NIAC: Washington, DC, USA, 2009.

[9] L. Coppolino, L. S. D'Antonio, S.V. Giuliano, G. Mazzeo and L. Romano. A framework for Seveso-compliant cyber-physical security testing in sensitive industrial plants. Comput. Ind. 2022, 136, 103589. https://doi.org/10.1016/j.compind.2021 .103589.

[10] Monstadt, and M. Schmidt, Urban resilience in the making? The governance of critical infrastructures in German cities 56 (11) (2019) 2353–2371, https://doi.org/10.1177/ 0042098018808483

[11] G.P. Cimellaro, P. Crupi, H.U. Kim, and A. Agrawal, Modeling interdependencies of critical infrastructures after hurricane Sandy, Int. J. Disaster Risk Reduc. 38 (2019), 101191, https://doi.org/10.1016/J.IJDRR.2019.101191.

[12] B. Rathnayaka, C. Siriwardana, D.J. Robert, P. Amaratunga, and S. Sujeeva. Improving the resilience of critical infrastructure: Evidence-based insights from a systematic literature review. International Journal of Disaster Risk Reduction, 2022, h ttps://doi.org/10.1016/j.ijdrr.2022.103123.

[13] OECD, Good Governance for Critical Infrastructure Resilience, OECD, 2019, https://doi.org/10.1787/02F0E5A0-EN.

[14] F. De Felice, I. Baffo, and A. Petrillo. Critical Infrastructures Overview: Past, Present and Future. Sustainability 2022, 14, 2233. https:// doi.org/10.3390/su14042233.

[15] C. Berger, P. Eichhammer, H.P. Reiser, J. Domaschka, F.J. Hauck, and G. Habiger. A Survey on Resilience in the IoT: Taxonomy, Classification, and Discussion of Resilience Mechanisms. ACM Comput. Surv. 2022, 54, 147. https://doi.org/10.1145/3462513.

[16] M. Barnes, K. Brown, J. Carmona, D. Cevasco, M. Collu, C. Crabtree, W. Crowther, S. Djurovic, D. Flynn, P.R. Green, P.R. et al. Technology Drivers in Windfarm Asset Management. Home Offshore. 2018. Available online: https://doi.org/10.17861/20180718 (accessed on 15 May 2024).