2022 International Young Engineers Forum (YEF-ECE) | 978-1-6654-6731-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/YEF-ECE55092.2022.9849899

Extending the Synoptics of Things (SoT) Framework to Manage ISoS Technology Landscapes

Bruno Serras
ISEL - Instituto Superior de Engenharia de Lisboa
IPL – Instituto Politécnico de Lisboa
POLITEC&ID
Lisbon, Portugal
A43538@alunos.isel.pt

Carlos Gonçalves
ISEL - Instituto Superior de Engenharia de Lisboa
IPL – Instituto Politécnico de Lisboa
POLITEC&ID
Lisbon, Portugal
carlos.goncalves@isel.pt

Tiago Dias
ISEL - Instituto Superior de Engenharia de Lisboa
IPL – Instituto Politécnico de Lisboa
POLITEC&ID
INESC-ID
Lisbon, Portugal
tiago.dias@isel.pt

A. Luís Osório
ISEL - Instituto Superior de Engenharia de Lisboa
IPL – Instituto Politécnico de Lisboa
POLITEC&ID
Lisbon, Portugal
aosorio@deetc.isel.ipl.pt

***Abstract*—Managing and monitoring the software and hardware artifacts of an industrial organization are fundamental efforts that can often be challenging endeavors to achieve, especially when such technological landscapes are composed of multiple heterogeneous systems. Usually, Internet of Things (IoT) devices are provided by different suppliers and may use different protocols and interfaces. Thus, the integration of these devices results in complex development and maintenance cycles. The Synoptics of Things (SoT) framework can address these problems, in conjunction with the Informatics System of Systems (ISoS) platform by promoting an open market competitive technology landscape for organizations. The purpose of the research presented in this paper was to extend the SoT framework in order to manage and monitor the different elements of an ISoS-enabled organization, namely the *ISystems* (*Informatics Systems*), *CES* (*Cooperation Enabled Services*), and *Services*. We argue that the SoT framework can be essential in a supervisory control and data acquisition (SCADA) system in today's modern web, by adopting the concept of Web Components as a standard to enable the development of custom and reusable components. We present and discuss such issues in the context of the HORUS system, an informatic system responsible for payment enforcement in fueling stations. In this system, several devices must work in coordination, such as video cameras and video recorders, which are fundamental for the retrieving of license plate images and, therefore, be continually monitored to ensure the correct functioning of the HORUS system.**

***Index Terms*—Internet of Things, Supervisory control and data acquisition systems, Web Components, Synoptics**

I. Introduction

Systems supervision plays a critical role in modern organizations. Managing and monitoring such systems of systems can be a challenging task, since these technological landscapes are typically composed of several products provided by different manufacturers, each one with its proprietary protocols and Application Programming Interface (API). This approach compromises the replacement or upgrading of existing artifacts, such as sensors and actuators, and has proven to be an obstacle to sustainable innovation [1]. Also, different administrators can be responsible for the supervision of distinct parts of a system, which represents another problematic challenge to address.

Supervisory Control and Data Acquisition systems, or SCADA systems, usually implement proprietary solutions, that are used to manage and monitor industrial processes. These systems are comprised of computers and peripheral devices such as Programmable Logic Controllers (PLC) and Remote Terminal Units (RTU) [2], used to transmit telemetry data and change the state of the objects. A Master Terminal Unit (MTU) is directly connected to Graphical User Interfaces (GUI), such as Synoptics, and receives data from RTU and PLC devices. A Synoptic presents an overview of the processes, with diagrams that can show the state of the system and alarm displays. Fig. 1 presents a common architecture of a SCADA system.

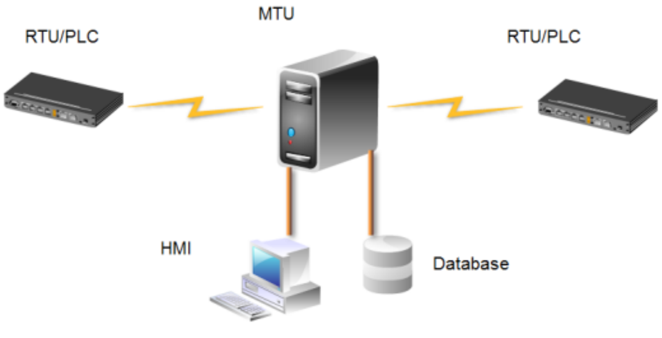


Fig. 1. Common architecture of a SCADA system [2].

The Informatics System of Systems (ISoS) platform [3], a contribution to the Model-Driven Open Systems Engineering (MDEOS), promotes an open market competitive technology landscape for organizations. One of the main goals of the ISoS platform is to make it possible to seamlessly replace system elements of an informatics system, or the whole system, with an equivalent technology artifact from a different manufacturer. The adoption of this platform in an organization establishes clear boundaries of responsibility, often not evident due to the relationships between the informatics engineering teams and the business managers of the organization.

Fig. 2 depicts the ISoS model, which is based on three elements: i) the *ISystem*, as an autonomous computational responsibility; ii) the *CES* (*Cooperation Enabled Services*), as an atomic component of an *ISystem* that is composed of *Services* and is part of the strategy to attain partial substitutability; and iii) the *Service*, as the functional operating element that can be either a software artifact or a cyber-physical element, e.g., actuators and sensors. A special *ISystem* named *ISystem$_0$* is responsible for managing the whole ISoS technology system, acting as a directory service for metadata of the ISoS elements that exist within an organization. To access the *Services*, lookup operations are performed using the *ISystem$_0$*. The ISoS reference implementation is currently based on the Apache Zookeeper [4].

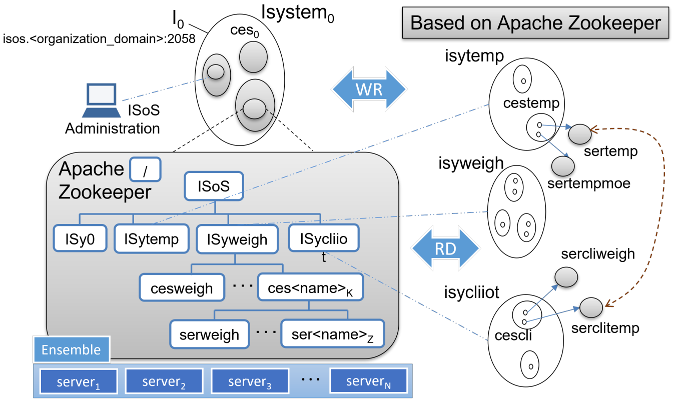


Fig. 2. The ISoS reference implementation strategy.

The Synoptics of Things (SoT) framework [5] enables the use of web browsers to create and configure synoptics as a fundamental element of SCADA systems [6]. The SoT framework potentially applies to various scenarios, such as power generation and consumption or water network distribution. SCADA systems allow operators to manage and monitor industrial processes, offering them several functionalities such as alarm handling and report generation. The interaction between a SCADA system and an operator is called Human-Machine Interface (HMI), in which synoptics take an important role. Synoptics are displayed on a monitor screen and can be interacted by a mouse click or by touch, displaying an overview of an industrial process and providing global visibility of the managed process. Typically, SCADA interfaces are native proprietary applications of an organization. The SoT framework aims to provide an open platform for the agile and efficient development of synoptics. In [7], we present a case study of a silos' operator managing a maritime terminal where *"the existing legacy management and supervision solutions are neither integrated nor properly prepared to cope with collaborative processes"*. We further argue that extending the SoT framework to navigate and introspect through ISoS elements in a synoptic can significantly promote a clear supervising boundary in an organization's technological landscape. The approach allows the integration of heterogeneous software and hardware artifacts from multiple suppliers.

In this paper, we present the HORUS project [8]–[10] as a case study and discuss how the SoT framework can help to manage and monitor a multi-supplier technological landscape in an ISoS-enabled environment. HORUS is an informatics system to control post-payments in fueling station's forecourts, whose goal is to prevent refueling by drivers that once left without payment. Accordingly, several closed-circuit television (CCTV) devices are used by the HORUS system to capture the license plates of the vehicles. A corresponding action is necessary to correct a situation resulting from a failure of any such devices, typically under the responsibility of a provider company of surveillance. Hence, the importance in establishing clear responsibility boundaries in the management of the HORUS system.

The remaining of this paper is organized as follows. Section II revisits the SoT framework in more detail and introduces the extensions that enable the support for ISoS environments. Section III presents the HORUS project and discusses the use of the SoT framework in this study case. Finally, Section IV draws some conclusions and presents guidelines for further research.

II. Revisiting the Synoptics of Things Framework

The SoT framework aims to develop synoptics in web interfaces by using the Web Components standard [11], allowing for custom and reusable components.

In the first approach, we prioritized the supervision of IoT devices without considering the organization of such cyber-physical devices in the company's technological landscape. This research extends the SoT framework to manage ISoS-enabled organizations, thus achieving clear responsibility boundaries.

Subsection II-A revisits the base SoT framework with its primary elements and essential concepts such as *Widgets-IoT* and alarm handling, one of the most critical features in SCADA systems. Next, subsection II-B presents important concepts about the implementation of the framework. Finally, subsection II-C extends the base model to support the creation and configuration of graphical ISoS widgets.

A. The SoT Base Framework

The SoT framework uses the concept of widgets to support the interaction with the supervising operator. We call these widgets *Widgets-IoT*.

A *Widget-IoT* is an abstraction of a hardware or a software element, e.g., a video camera device or a license plate recognition software. The *Widget-IoT* comprises common properties such as size, color and image. Specializations of *Widget-IoT* might contain properties for accessing a cyber-physical device.

Widgets-IoT are standard Web Components, which also makes it possible to use them in other contexts than the SoT framework. Since they are standard Web Components, *Widgets-IoT* must have a context menu that is opened when double-clicking the widget. This menu can contain configuration options and monitoring information of the associated hardware or software system element. *Widgets-IoT* can also aggregate other Web Components to accomplish interactivity, e.g., a video camera widget may have a video player that happens to be a web component.

Also, a *Synoptic of Things* aggregates *Widgets-IoT* under a common relationship, where such widgets can be dragged and dropped.

A simplified Systems Modeling Language (SysML) block definition diagram of the base model of the SoT framework is shown in Fig. 3. SysML extends the Unified Modeling Language (UML) by supporting specification, analysis, verification and validation of hardware and software components [12].

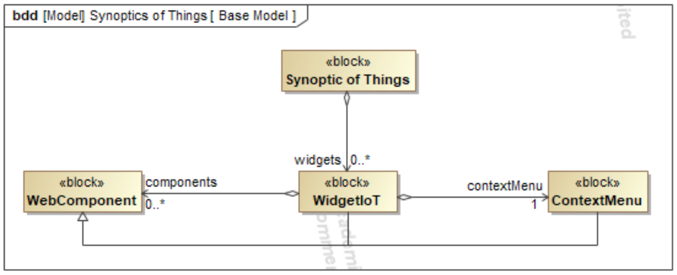
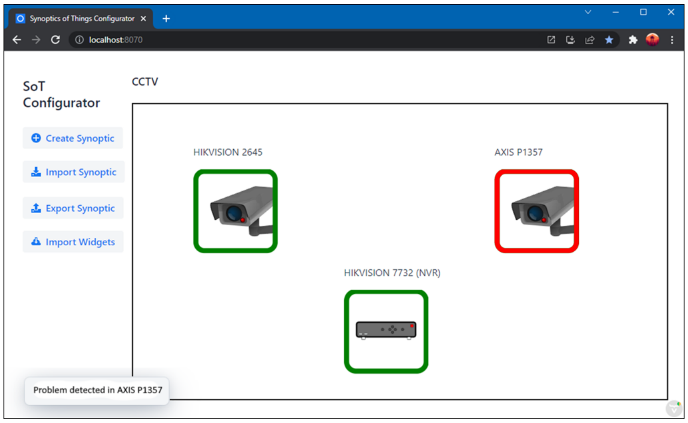


Fig. 3. SysML block definition diagram of the base SoT model.

B. Implementation of the SoT Framework

Developing widgets for the SoT framework can be achieved by using open-source libraries such as Lit[1] and Polymer[2], both from Google. The current implementation of the SoT framework is based on Lit, as it is a simple yet powerful library.

Fig. 4 shows an example of a synoptic created with the SoT framework containing two video camera devices (labeled as HIKVISION 2645 and AXIS P1357) and one video recorder device (labeled as HIKVISION 7732 NVR). By viewing this synoptic, the operator can become aware that the AXIS P1357 device has a critical problem, hence its surrounding red color. Meanwhile, the other devices are green colored, meaning no problem was detected. The yellow color can notify a problem that is not critical but requires attention. Notifications appear in the bottom left corner of the synoptic when changes in devices' status are detected.



Problem detected in AXIS P1357

Fig. 4. Synoptic panel created using the SoT framework where several CCTV devices are shown.

Multiple tools are candidates for continuous monitoring of the several elements that comprise an informatics system [13]. Such monitoring tools provide programming interfaces to retrieve events and data alarms. Typically, the configuration of the monitoring environment and data retrieval uses a REST (Representational State Transfer) API. The functionalities more commonly provided by monitoring tools are event generation, alarming, notifications, report generation, performance measurement, service assurance, and thresholding. OpenNMS[3], Nagios[4], Zabbix[5] and Prometheus[6] are examples of such monitoring tools, to name a few.

In the SoT framework we have been using the open-source monitoring platform OpenNMS to retrieve alarm data from the devices. We chose OpenNMS due to its development under the open-source model. However, other monitoring tools are eligible to work with the SoT framework, provided that the specific interface is developed - a concept for future research. Such tools often display the regional status of monitored devices and systems on a map, as shown in Fig. 5, where each color represents the alarm type.

Since OpenNMS does not directly access the devices, a software component had to be developed to aggregate the devices data and export it to OpenNMS. The Simple Network Management Protocol (SNMP) provides functionalities to perform read and write operations on IP networks. Typically, SNMP has been used for monitoring network devices such as routers and switches, but it can be applied in virtually every device if it offers an SNMP interface or uses the concept of an SNMP agent. Fig. 6 depicts a simplified architecture of the interaction between the SoT framework and CCTV devices to achieve monitoring. An SNMP agent collects information from the devices (1) and exports data using the mentioned protocol under a Management Information Base (MIB). OpenNMS polls data from the SNMP agent (2) and generates events and

[1] www.lit.dev
[2] www.github.com/Polymer/polymer

[3] www.opennms.com
[4] www.nagios.org
[5] www.zabbix.com
[6] www.prometheus.io

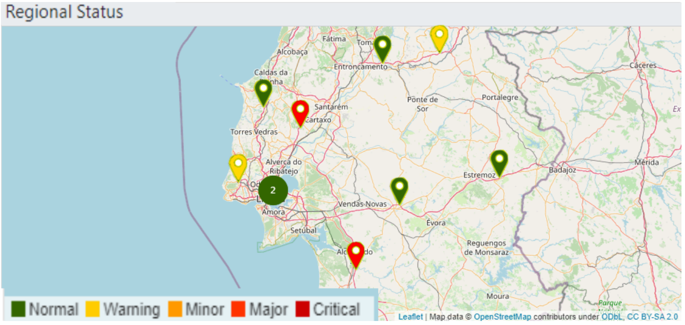


Fig. 5. Regional status of monitored devices in OpenNMS, e.g., video cameras in the forecourts of fueling stations. The green color represents normal status, while the yellow color represents a warning. The shades of red color represent a major or critical problem.

alarms depending on the configured thresholding. For example, an alarm must be thrown if the resolution of a video camera is lower than 1920x1080. Finally, the SoT framework performs HTTP (Hypertext Transfer Protocol) calls using the REST API of the OpenNMS platform (3) and processes the data to show notifications on the synoptic and change the status color of the monitored elements.

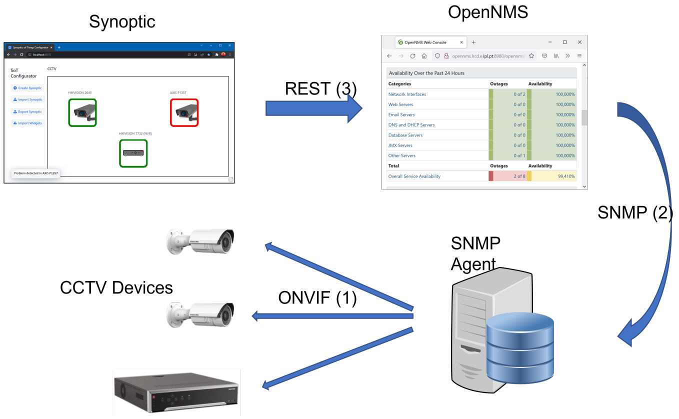


Fig. 6. Interaction between software elements and devices to achieve monitoring.

The Open Network Video Interface Forum[7] (ONVIF) acts as a standard interface for IP-based physical security products to access the CCTV devices. The retrieval of information is done using the Simple Object Access Protocol (SOAP) that is based on HTTP and Extensible Markup Language (XML) standards.

C. Extending the SoT Framework to support ISoS

The SoT framework has been extended to provide the creation and configuration of ISoS-specific *Widgets-IoT* to be efficiently used to manage and monitor an ISoS technological landscape. A child component of *Widget-IoT* named *ISoSWidget-IoT* was developed to act as a generic ISoS element and provide access to the *ISystem$_0$* of an organization in order to perform the necessary lookup operations. For example, a practical use case of a lookup operation in the context of the HORUS project is accessing a video camera device, modeled as a *Service* in the ISoS model. The *ISystem$_0$* contains all the metadata associated to every element of the organization, including their entry points. After retrieving the entry point to an element, e.g., a video camera device, the authentication process begins and, if successful, the widget can provide access to it.

The SysML block definition diagram that depicts the extended SoT framework is presented in Fig. 7. For every ISoS element there is a specialization of *ISoSWidgetIoT* that inherits its specific functionality: *ISystemWidgetIoT* is associated with an *ISystem*, *CESWidgetIoT* is associated with a *CES*, and a *ServiceWidgetIoT* is associated with a *Service*. A *ServiceWidgetIoT* acts as the base definition for all the widgets related to cyber-physical devices and software elements, e.g., the video camera *Widget-IoT* is a specialization of *ServiceWidgetIoT*. Before the extension to support ISoS elements, a video camera *Widget-IoT* would directly extend from the generic *WidgetIoT*.

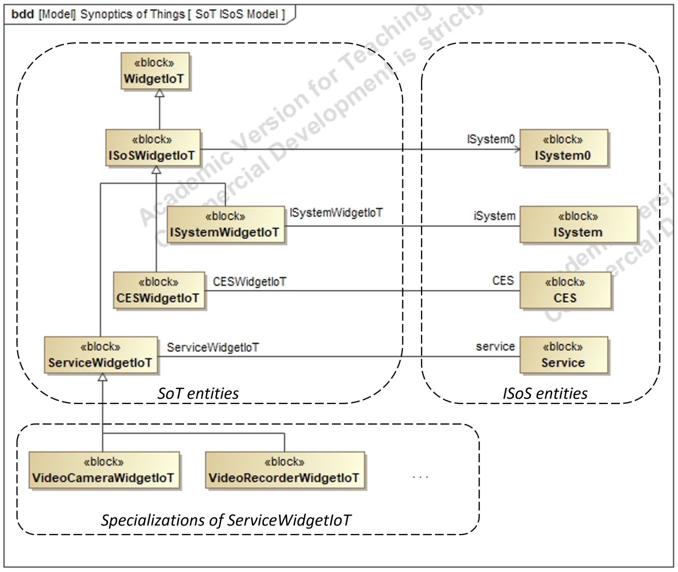


Fig. 7. SysML block definition diagram of the several components that support the ISoS platform and its associations. On the left are represented SoT entities (widgets), where as on the right are represented ISoS modeling elements.

Interacting with *Widgets-IoT* by double-clicking them opens a context menu containing processed information that was retrieved from the *ISystem$_0$*. The context menu can also be used to explore the ISoS elements hierarchy. For example, double-clicking an *ISystem* opens a context menu to access its associated *CES*. Besides, the context menu shows a general status of the child elements of an ISoS element. In the case of the *Service* element, which is the leaf node of the hierarchy, the context menu displays its properties and a button to access it. Fig. 8 depicts this interaction in a synoptic of a generic ISoS landscape. The user can navigate through the different *CES* and *Services* and even access them, e.g., a database

[7] www.onvif.org

management interface.

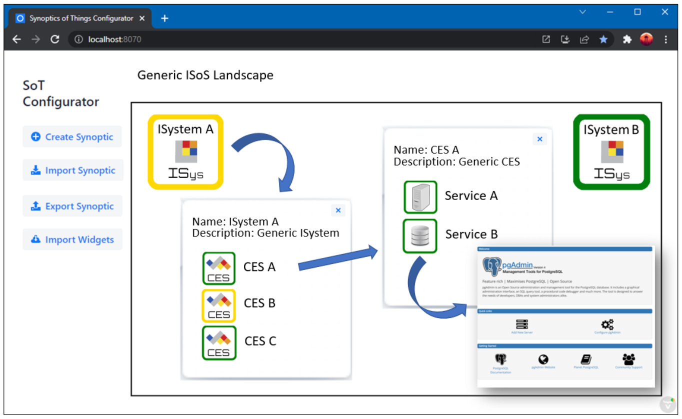


Fig. 8. Synoptic of a generic ISoS landscape.

III. The HORUS Project Case Study

This research paper uses the HORUS project as a case study to showcase the extended SoT framework in a ISoS-enabled environment. The following subsections are organized as follows. Subsection III-A introduces the base scenario of the HORUS project, while subsection III-B presents the use of the SoT framework in the HORUS case study.

A. The HORUS Project Base Scenario

The main goal of the HORUS project is to manage the post payment system in a gas station forecourt to avoid 'making off without payment' incidents. In this system, an automatic license plate recognition (ALPR) software captures the identification of the vehicles that enter the forecourt. The point-of-sale (POS) system interacts with a HORUS service to assess past payment incidents when fueling. A warning message is displayed to the POS operator if such events were recorded, as depicted in Fig. 9.

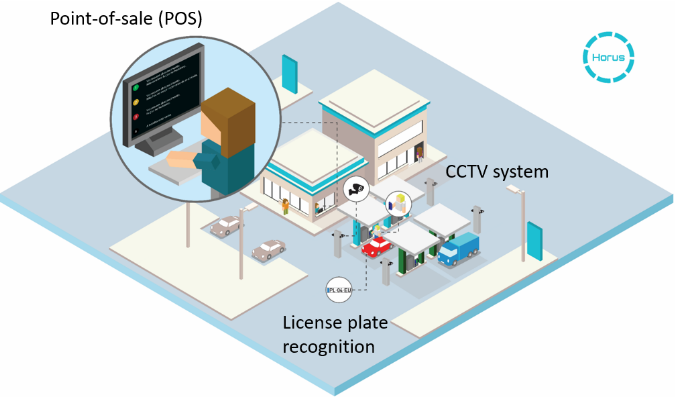


Fig. 9. Forecourt of a fueling station in the context of the HORUS project.

Using the ISoS platform, the HORUS landscape can be represented using three *ISystems*, as shown in Fig. 10: i) the HORUS *ISystem*, with the payment enforcement responsibility; ii) the POS *ISystem*, with the payment operation responsibility; and iii) the CCTV *ISystem*, with the surveillance responsibility. In addition, each *ISystem* is monitored by a corresponding *ISystem-M* (the *M* standing for monitoring). For example, the CCTV *ISystem* is monitored by the CCTV-M *ISystem*, which comprehends a *CES* with two *Services* to model the OpenNMS platform and the SNMP agent.

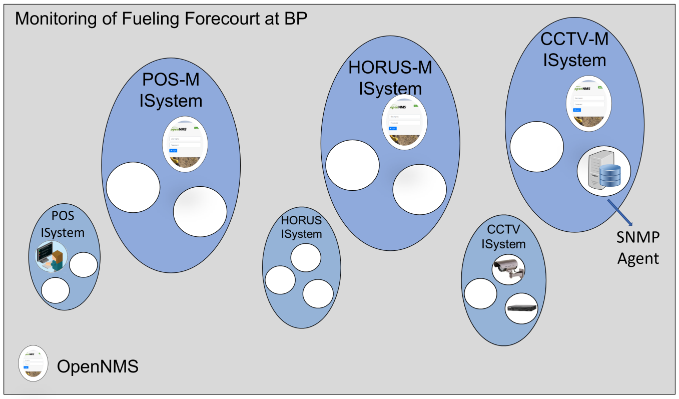


Fig. 10. Overview of the ISoS landscape of the HORUS project.

B. Using the SoT Framework for the HORUS Case

The SoT framework is composed of two interfaces: the *Widgets-IoT* Manager interface and the Synoptics Configurator interface. The former is used to create and edit instances of *Widgets-IoT*, while the latter is used to configure synoptics by displaying the widgets in a canvas using drag and drop functions. In both interfaces, the user can export the associated widgets and synoptics to an XML file and later import it to the framework to rebuild the interfaces.

As already mentioned, the HORUS landscape comprehends three particularly important *ISystems*. To manage and monitor these *ISystems* using the SoT framework, a user would start by creating one *Widget-IoT* for each *ISystem* in the *Widgets-IoT* Manager interface. The main properties of these widgets would be the path to access the corresponding *ISystem* in the *ISystem*$_0$, as well as the entry point of the *ISystem*$_0$ of the ISoS organization associated with the *ISystem*, since different *ISystems* can belong to distinct ISoS landscapes. The *ISystem*$_0$ would then be able to return all the necessary information to access each *ISystem*, including the paths for each *CES* and *Services*. Fig. 11 depicts a synoptic containing the three *ISystems* of the HORUS landscape.

Using this synoptic, the operator would have access to a general view of the status of each *ISystem* and could introspect the *CES* and *Services* displayed in a new synoptic. For example, clicking in the CCTV *ISystem* could open a synoptic containing a *CES* representing the set of video cameras and recorders devices available in the HORUS landscape. Then, clicking on that particular *CES* would display another synoptic comprising each device, as depicted in Fig. 4. At that point, the operator would be able to view information for each device

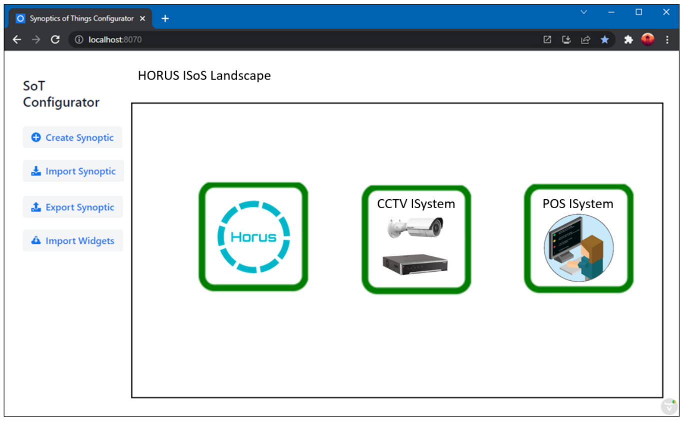


Fig. 11. Synoptic of the ISoS landscape for the HORUS project.

and even access the video stream. The alarming could work as follows: the status of each ISoS element would be constantly monitored at the element level and propagated up in the hierarchy to all the associated ISoS elements. For example, if a *Service* displays a warning, that event would also be propagated to the associated *CES* and *ISystem*.

IV. Conclusions and Further Research

This paper presents and discusses the extension of the Synoptics of Things framework to manage ISoS-enabled organizations, aiming to supervise informatics systems of systems comprised of heterogeneous informatics systems and devices using different communication protocols. The SoT framework adopts the Web Components standard to provide web interfaces for the development and configuration of synoptics for ISoS-enabled organizations, to develop custom and reusable components, which we call *Widgets-IoT*. A *Widget-IoT* can act either as a hardware or a software interface element for a user to interact with and to monitor its status. The paper discusses how the SoT framework could be used as a base to manage and monitor the collection of *ISystem*, *CES* and *Services* of an ISoS-enabled organization using the HORUS project as a case study. The open-source platform OpenNMS is used to retrieve alarm data from the cyber-physical and software elements, with the help of an SNMP agent that aggregates and exports information to OpenNMS. The ONVIF standard was adopted to retrieve the data from the CCTV devices used in the HORUS environment, since it provides standardized interfaces for the interoperability of IP-based cyber-physical security products.

For future research we want to study how to improve the integration in the SoT framework of other prominent monitoring tools such as Prometheus and Zabbix so that synoptics could aggregate information from several monitoring platforms.

Acknowledgment

This work had partial support from the SINCRO/ANSR research project, from the PT2020 project SITL-IoT promoted by the FORDESI company, and from BP Portugal through the research project HORUS.

References

[1] C. Gonçalves, A. L. Osório, L. M. Camarinha-Matos, T. Dias, and J. Tavares, "A Collaborative Cyber-Physical Microservices Platform – The SITL-IoT Case," in *Working Conference on Virtual Enterprises*, pp. 411–420, Springer, 2021.

[2] R. P. S. Lishev and A. Georgiev, "Laboratory SCADA systems – the state of art and the challenges," *Balkan Journal of Electrical and Computer Engineering*, vol. 3, no. 3, pp. 164–170, 2015.

[3] A. L. Osório, A. Belloum, H. Afsarmanesh, and L. M. Camarinha-Matos, "Agnostic Informatics System of Systems: The Open ISoS Services Framework," in *Collaboration in a Data-Rich World* (L. M. Camarinha-Matos, H. Afsarmanesh, and R. Fornasiero, eds.), pp. 407–420, Springer International Publishing, 2017.

[4] F. Junqueira and B. Reed, *ZooKeeper: distributed process coordination*. O'Reilly Media, Inc., 2013.

[5] B. Serras, C. Gonçalves, T. Dias, and A. L. Osório, "Synoptics of things (SoT): an open framework for the supervision of IoT devices," in *2021 International Young Engineers Forum (YEF-ECE)*, pp. 127–131, IEEE, 2021.

[6] A. Daneels and W. Salter, "What is SCADA?," *International Conference on Accelerator and Large Experimental Physics Control Systems*, 1999.

[7] A. L. Osório, L. M. Camarinha-Matos, T. Dias, and J. Tavares, "Adaptive integration of IoT with informatics systems for collaborative industry: the SITL-IoT case," in *Working Conference on Virtual Enterprises*, pp. 43–54, Springer, 2019.

[8] A. L. F. G. Osório, *Collaborative networks as open Informatics System of Systems (ISoS)*. PhD thesis, University of Amsterdam - Faculty of Science - Informatics Institute, Dec. 2020.

[9] P. Urze, A. L. Osório, H. Afsarmanesh, and L. M. Camarinha-Matos, "A Balanced Sociotechnical Framework for CollaborativeNetworks 4.0," in *Boosting Collaborative Networks 4.0*, pp. 485–498, Springer International Publishing, 2020.

[10] A. L. Osorio, "Towards vendor-agnostic IT-system of IT-systems with the CEDE platform," in *Working Conference on Virtual Enterprises*, pp. 494–505, Springer, 2016.

[11] T. Bui, "Web components: concept and implementation," *Bachelor's Thesis, Turku University of Applied Sciences*, 2019.

[12] M. Hause *et al.*, "The SysML modelling language," in *Fifteenth European Systems Engineering Conference*, vol. 9, pp. 1–12, 2006.

[13] Ł. Kufel, "Tools for distributed systems monitoring," *Foundations of Computing and Decision Sciences*, vol. 41, no. 4, pp. 237–260, 2016.