Generic Architecture for Edge Computing Based on SPF for Military HADR Operations

Manas Pradhan
Information Technology for Command and Control
Fraunhofer Institute for Communication,
Information Processing and Ergonomics
Wachtberg, Germany
Email: manas.pradhan@fkie.fraunhofer.de

Filippo Poltronieri, Mauro Tortonesi
Distributed Systems Research Group
University of Ferrara
Ferrara, Italy
Email: {filippo.poltronieri,mauro.tortonesi}@unife.it

***Abstract*—Internet-of-things (IoT) devices have led to ubiquitous, remote and autonomous computing at the edge of the networks. These devices offload sensing, actuation and processing tasks away from the core of the network. The concept of Smart Cities tries to leverage Edge Computing based on IoT technologies for remote and distributed computing. Sieve, Process and Forward (SPF) is a Value-of-Information (VoI) based Fog as a Service (FaaS) solution for dynamic IoT applications in Smart City scenarios. The military has been looking to utilize the SPF platform for Edge Computing to assist in Human Assistance and Disaster Recovery (HADR) operations. A recent NATO IST 147 RTG demonstration proved the validity of SPF, but also highlighted the need of extending the current architecture to support specific use-case scenarios for HADR systems. This paper tries to propose a generic architecture based on SPF to enable interoperability between military C2 (Command and Control) and core computing systems to support future HADR operations in Smart City environments.**

***Index Terms*—Humanitarian Assistance and Disaster Recovery (HADR), Internet-of-Things (IoT), Smart Cities, Fog Computing, Edge Computing.**

I. Introduction

The idea of IoT refers to the pervasive and ubiquitous presence of things connected to the network of things such as micro-computers, micro-processors, Radio-Frequency Identification (RFID) tags, sensors, actuators, mobile phones, etc via the Internet or internet [1]. These things can interact and collaborate with each other to reach common goals such as computing, sensing and actuation. Such a connected environment of independent and intelligent things allows for new interactions among IoT devices and humans, and enables the realization of concepts of smart cities i.e. infrastructures, and services that enhance the quality of life [2]. According to Cisco, an estimated 50 billion devices will connect to the Internet by 2020 [3]. These IoT devices would generate 500 zettabytes per year in data and that number is expected to grow exponentially [4]. This post-cloud era asks for computing at the edge of the network so that the IoT applications satisfy real-time computation needs by computing tasks and filtering required data for the core. Edge computing allows for addressing needs of IoT applications such as latency, mobile devices' limited battery life, bandwidth costs, security, and privacy.

978-1-5386-4980-0/19/$31.00 ©2019 IEEE

The concept of Smart Cities tries to leverage modern ICT technologies to provide services to its citizens. IoT assets in Smart Cities enable the city administrations to remotely monitor, manage and control city activities and services, and create new insights and actionable information from massive streams of real-time data [5] [6]. These IoT devices are remotely deployed on buildings, streets, people's houses, industrial installations etc. inside a city. They gather a plethora of information and actuate resources based on demands. Sending all the data to a central server or cloud in raw form puts a severe strain on resources such as bandwidth usage, processing and analysis resources of the core components of the city's ICT, security [7] implications of data transmission and interoperability concerns of the data transmitted from multiple sources. In order to circumvent these issues, solutions such as Edge Computing, Fog Computing, Multi-access Edge Computing (MEC), mobile cloud computing etc. have come into the picture which offload the computation tasks away from the core of the city's ICT systems [8]. This would allow for real-time analysis of the data collected at the edge, filter the data required to reported to the core ICT and allow quicker responses for the operational demands.

Now, with the growth of cities, the tangible resources in the city have increased by many fold. The human population concentrates in the cities now rather than rural or sub-urban areas. This upsurge of population creates a huge demand on city administrations to manage people and its resources [9]. Smart Cities with their ICT and IoT systems try to solve this issue. The natural disasters due to global warming as well as man-made situations like fatal industrial accidents, terrorist attacks etc. call for cities to be prepared for large-scale Humanitarian Assistance and Disaster Recovery (HADR) operations [10]. These HADR operations would require large involvement of technology assets along with the human assets.

IoT assets particularly in this case can be largely used for providing real-time Situational Awareness (SA) by sensing, computation and actuation since they operate closest to the ground. They are built for the purpose of running on restricted resources such as power which allows them to operate in

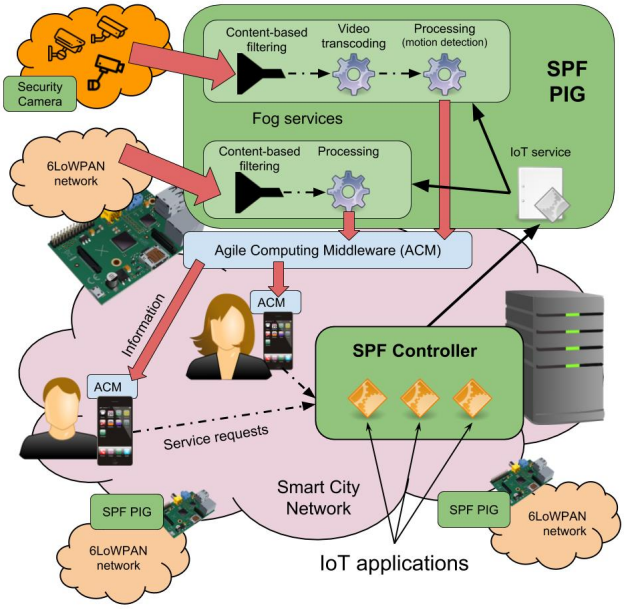

Fig. 1. SPF in a Smart City Scenario

adverse conditions. Thus deploying Edge Computing solutions on these edge (IoT) assets would inherit the advantages of IoT devices and further their usage in restricted conditions by reporting filtered out, analyzed and useful data required for HADR operations.

The NATO Informations Systems Technology (IST) 147 Research Task Group (RTG) was formed for investigating the military applications of Internet-of-Things (IoT). The group is looking towards using SPF Edge Computing platform for deployment in HADR operations. A recent demonstration in Warsaw proved the validity of SPF, but also highlighted the need of extending the current architecture to support specific use-case scenarios. This paper tries to propose a generic architecture to the existing SPF platform to extend its functionality and its implementation scope. The new architecture also tries to address some of the drawbacks of the existing SPF implementation related to resource discovery and management. The new architecture aims at ready deployment by the military as well as the civilian counterparts which in future could be used for HADR operations.

II. Sieve Process and Forward (SPF)

SPF is a Fog as a Service (FaaS) platform [11] that allocates part of the IoT information processing at the edge of the network, thus exploiting the proximity of sources and users. SPF is based on the Adaptive, Information-centric and Value-based (AIV) information model that adopts filtered and value-based information processing policies of the IoT data [12]. According to the AIV, SPF ranks each information object according to its Value of Information (VoI) [13] in order to prioritize the dissemination of critical information at the edge of the network.

Fig. 1 illustrates the two main components of the SPF architecture: the Controller and the Programmable Internet Gateways (PIGs). The SPF Controller acts as interface between the users and the different PIGs that can be located at the Edge or in the Cloud. The SPF Controller is responsible to dispatch the requests for services, received from the users, to the available PIGs, which will execute the required services.

SPF applications are available to users, which can request the execution of a service to the SPF Controller over the available communication networks. When the SPF Controller receives a request for a service, it analyses the information contained in the request and then selects a PIG, which is the component responsible for executing the users' requests and delivering back the responses.

Within the SPF architecture, management functionalities are provided by the SPF Controller, which is responsible for deploying the information processing and dissemination functions required by the registered applications. Using the management functionalities, the SPF Controller can also reprograms the PIGs in the case a new service is required using a proprietary programming interface.

The Information Processing and Dissemination functions are instead provided by Proactive Dissemination Service (DSPro) [13], which leverages the set of filtering and communications functions implemented by the software platform, according to the commands received by the SPF Controller. According to the SPF design, PIGs can be deployed directly on the gateway nodes at the edge of the network or on dedicated hardware placed in the gateway nodes' proximity.

Finally, the SPF architecture identifies different roles for the stakeholders: administrators which are responsible deploying, running, and operating the SPF components, service providers that develop and deploy IoT applications, and the users of the SPF applications.

III. Integrating SPF within a Military C2 Infrastructure

Considering the role of stakeholders envisioned for the SPF applications, the military C2 Infrastructure is one of the possible users of the SPF platform. Especially for HADR operations, where the NATO IST-147 group is looking towards applying the IoT technologies, SPF is considered as a possible candidate for Edge Computing solutions. For HADR operations in Smart City environments, the SPF solution can be used to gain hybrid SA i.e. from the IoT assets deployed by Smart Cities as well as by the military at the edge [14], [15].

In the envisioned scenario, the various sensing and reporting platforms from the ICT systems of the Smart City and the military gather SA data about various events and incidents in a disaster or emergency situation. These ICT systems may gather data from various sensors, cameras, human-source intelligence, Non-Government Organizations (NGOs) and first-emergency responders to help citizens across the city. They might get and report specific inputs which might of interest for further analysis and operations to help citizens or secure resources in the emergency.

Based on the data or reports gathered form the ICT systems, further analysis might be performed to find out what action needs to be performed correspondingly. These might be according to the operational needs as set for the disaster recovery situation where a specific set of actions need to performed to assist citizens in that situation. For example, an incident might be reported by a team located on ground of a severe fire in a building and based on the severity of fire, the city's ICT systems might want to direct the nearest located fire trucks to the rescue area to save time and increase efficiency of the rescue operation. After getting the analyzed data from the ICT systems, the administrators (Smart City) or commanders (Military) might want to initiate an action to work on the received inputs or intelligence. These can be autonomously done through the C2 applications to initiate an action. For example, whenever the C2 application knows about a fire in a building, based on the camera resources available at the edge, it would trigger the cameras at the vicinity of the incident to send in the video streams to monitor the situation. Also, manually a resource can be chosen to take some action. For example, the map being displayed by the C2 UI can be used to pin-point camera resources on the ground and activate them to report the video streams. These C2 applications can be running on the central command's ICT systems or on a Mobile Tactical Operations Center (MTOC) based on Federated Missions Networking (FMN) architecture.

The demonstration at the International Conference on Military Communications and Information Systems (ICMCIS 2018), Warsaw, Poland, 22-23 May, 2018 by the IST-147 group, showed the possible implementations and use-cases for military applications of IoT and military C2 systems [14]. These demonstrations proved that SPF is particularly well suited as on edge computing platform showing use-cases for running face recognition and object detection algorithms [16] for video captured at the edge and streamed back to the C2 end. However, in order to highlight its potential within military environments, the Warsaw demo identified the need to extend the SPF architecture to support better integration for military systems and processes.

From the management perspective, there is the need to develop low interfaces with external C2 applications such as Android Tactical Assault Kit (ATAK) [14]. The current implementation of SPF components i.e. Controller, and PIGs is based on proprietary interfaces that could be improved to increase the possible use-case scenarios. These components right now are not abstract in nature and are designed (code-wise) to carry out specific tasks. As a result, whenever a new interaction or operation involving any of these components is required then an entire new functionality has to be added to the existing workflow. Also, future additions and configuration of specific modules would require a lot of rework and thus a lot of redesign of the existing implementation that would satisfy that particular use-cases and might not be applicable for future use-cases.

At the other end, the components in charge of executing SPF services (pipelines and services) right now communicate using a proprietary protocol based on unicast UDP, and disseminate information using the ACM DSPro middleware. In order to better highlight the SPF potentials for military applications, these interfaces should be compatible with IoT-specific along with other legacy protocols. With this regard, the NATO IST-147 RTG identified the Message Query Telemetry Transport (MQTT) as a protocol for message and telemetry transfer within the IoT domain for military operations [15]. Fig. 2 shows the MQTT topic format and a sample JSON message being exchanged between the military systems as demonstrated in the Warsaw demo for NATO IST 147 RTG. The interfaces between components and information producers/consumers should adhere to these MQTT message formats for systems to exchange data between themselves, initiate requests and responses, and trigger specific strategies and pipelines at run-time.

Finally, from the execution model perspective there is the need to consider additional methods for running services within a PIG. The services on PIGs are invoked through the processing strategies and pipelines running on them. In fact, the Warsaw's demo highlighted the need to consider different services, such as the transcoding of video formats, using COTS components. At the same time, we can envision the opportunity to allow the execution of components based on industry standard protocols, such as OSGi.

IV. Extended Architecture for SPF

To address these issues, we extended SPF to make interactions between the components open and extensible for various use-case scenarios or implementations, to re-utilize certain components, and to reduce the coupling and dependencies between the components by adopting common format for data exchange and interaction based on the existing military C2 platform architecture and ICT systems.

Fig. 3 shows the extended architecture for utilizing the SPF Edge computing platform for a HADR scenario involving military and civilian ICT systems and assets. The MTOC system houses the IoT Applications for C2 and resource management of IoT resources. These applications and resources employ the SPF Controller for application management and

<Org_Id >/<Country_id>/<ownership_type>/<device_class>/<device_type>/<message_type>/<data_type>/"JSON_msg"

MQTT Topic: NATO/GER/Private/Requestor/C2/Command/CameraFeed/"JSON_msg"
JSON Payload: {"Obj_id": GerC2, "lat": 52.245621, "UTC": "2018-02-01 10:17:30.227125", "lon": 21.012371, "value": HDFeed}

Fig. 2. MQTT Topic Format and Sample JSON Message

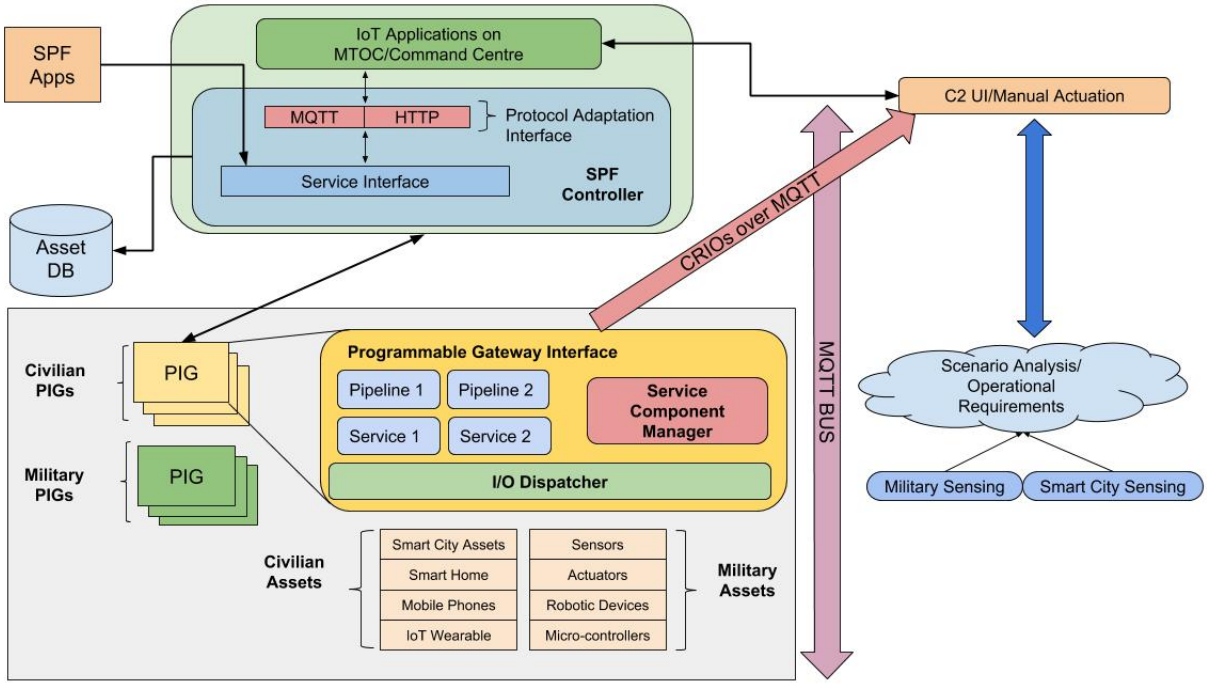

Fig. 3. Extended SPF Architecture

coordination. In this architecture, the SPF Controller issues commands and get responses based on commands received from the C2 application. In order to receive commands, the MTOC application would use the Protocol Adaption Interface (PAI), to communicate with the SPF Controller. PAI is a protocol adapter i.e. it can employ any kind of protocol as suited for the use-case, such as MQTT and HTTP as illustrated in Fig. 3. The Warsaw demo demonstrated that MQTT well suits the needs of military applications and it can be used as data messaging protocol. By adopting MQTT, PAI can thus receive the JSON payloads from the C2 applications included in the MQTT messages and interpret the request type. While, at the other end, the SPF Controller makes use of the PAI to communicate with the MTOC application.

The SPF Controller illustrated in Fig. 3 has two main components: the above described PAI and the Service Interfaces (SI). According the workflow of this architecture, PAI listens to receive inputs from the IoT application to execute a job/service over the HTTP and MQTT protocol. Instead, SI acts as an interaction-counterpart for the SPF controller and the SPF based applications. In the proposed architecture, the SI would interact with the PAI by receiving commands and initiating responses. The SPF controller would use the SI commands to initiate its internal functionality of triggering the edge resources or the remote PIGs deployed at the edge. The PAI will translate any commands received by non-SPF applications into a SI specific command.

In detail, this SPF's operation would involve to observe the implemented PAI to receive commands from MTOC application, to translate the requests from the MTOC into specific or use-case based requests, and to apply the application configuration dynamically at run-time instead of maintaining an application specific configuration file. Instead, the SI would be responsible for locating and triggering the use-case based PIGs based on the requests.

The remote PIG running at the edge has a list of hardware resources which it can interact with such as cameras, sensors, actuators etc. The PIG is responsible for triggering specific services on the deployed resources to execute use-case specific tasks. For instance, a request from the SPF Controller can ask the PIG to deliver HD Camera stream from its available camera to monitor a building under fire. The PIG can instantiate a service that sends back byte streams of camera feed back to the SPF Controller. In addition, it can also employ security specific mechanisms to handle military and civilian data streams separately. The triggering of services is based on Strategy-Pattern where a specific instance of a service on the PIG can be instantiated based on the request from the C2 application and its interpretation by the SPF controller.

Based on the service and processing strategies, the Service Component Manager (SCM) decides which pipelines and/or services have to be executed/invoked on the PIG. The SCM has a list of available pipelines for the PIG, and thus triggers specific pipelines and/or services to execute specific tasks and return results. Typically, in the original architecture of SPF, pipelines and services were dedicated to the processing

of all content of the same type. However, in this extended architecture, pipelines and services for the Civilian assets and Military assets are differentiated and initiated as separate, since military assets are deployed by the military and are presumed to be more reliable and trustworthy than the civilian assets for which the owner could not be verified in HADR operations.

PIGs can collect data from assets using two different methodologies depending on the type and the capabilities of those assets. First, PIG can periodically polling data from assets in case they expose a REST fashion API, such as traffic cameras located on the street. On the other hand, we also specify that PIGs can collect data by means of the MQTT Bus illustrated in Fig. 3, which listens for data sent by assets on pre-defined and configurable MQTT topics.

Apart from sending back the results to the SPF controller running at the MTOC level, the PIG can also send data directly to the C2 application, based on the specific inputs from the MTOC IoT applications. As illustrated in Fig. 3 these results/packets (Consumer Ready Information Objects in Fig. 3) are delivered as JSON payloads over MQTT. The PIG also employs a proprietary UDP connector which can be used for sending out UDP packets instead of using MQTT in specific use-cases such as delivering video streams. More in detail, remote PIG's operations would involve: to receive and parse commands from the SPF Controller, to configure dynamically the PIG components at run-time instead of using static configuration files, to invoke user-case Service and Processing Strategies based on the request's parameters, and to locate the appropriate pipeline to execute the service on.

Based on the civilian or military edge system at which the PIG is deployed, different and specific pipelines can be deployed for the specific assets. For example, in a HADR scenario that involves the use of civilian assets, video streams provided by cameras deployed on the streets can be used as input data for face recognition pipelines running on PIGs. In this case, the video stream will be elaborated and analyzed to provide to the SPF Controller and the requesting C2 application the results of the face recognition algorithm. As illustrated in Fig. 3 other examples of assets can be sensors, actuators, and IoT Wearable devices located across the Smart City scenario.

V. SPF: Applications Deployment on PIG

Another architectural change would involve to make the SPF platform capable of supporting dynamic services instantiation on the remote PIGs at run-time. In the original SPF architecture, a SPF Controller can configure and manage multiple remote PIGs, also by activating applications at run-time. This architectural change will let the SPF Controller also to upload new applications on PIGs and to schedule their execution and activation without re-configuring or restarting the PIGs. Furthermore, as illustrated in Fig. 4 to extend the range of possible services deployable on PIGs we envisioned there different types of services running on the SPF platform: generic (Unix) processes, SPF specific applications, and OSGi bundles.

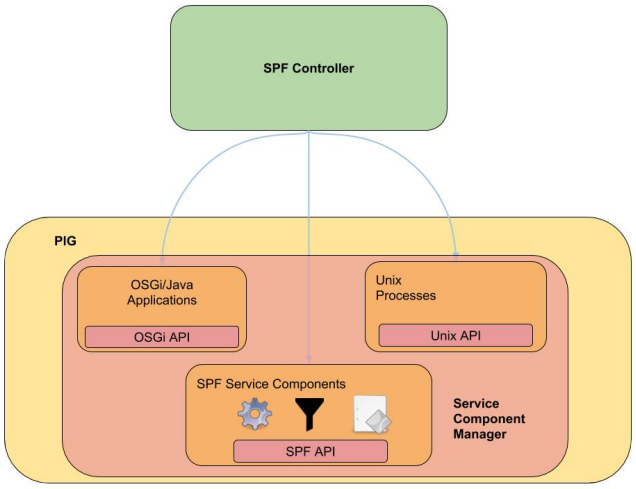

Fig. 4. SPF: PIG applications

First, SPF service components are algorithms and tasks specifically designed as integrated services for SPF. These applications are written as part of PIG's software or could be standalone Ruby classes and represent the applications part of the legacy SPF architecture.

Furthermore, in the SPF architecture, a PIG is supposed to be executed on a Unix-like operating systems, and thus enabling the execution of architecture-specific applications such as video decoder and transcoder, elaboration tools, and so on. Each application is supposed to be pre-installed on PIGs and it needs to be trusted and known otherwise, the execution of an untrusted application could compromise the security of the whole system.

Finally, we believe in the adoption of the OSGi [17] [18] specification[1] to support the execution, the composition, and the management of bundles (Java based applications) at run-time. In particular, adopting OSGi enables the possibility to efficient manage applications through containerization techniques, which will let new bundles to be created, uploaded, and activated on request without performing any cold or warm restarts of the other applications running on the same remote gateway.

We believe that extending the SPF platform with the possibility of running multiple types of applications would enhance its utility and facilitate its integration within civilian or military environments.

VI. Processing a HADR operation within the proposed architecture

To illustrate the capabilities provided by the extended architecture in an HADR scenario, let us consider the following use case. In the envisioned architecture, the SPF Controller is assumed to be running on the MTOC or to be a component of an IoT application running on the command vehicle, where the command center of the HADR operation is operating. The

[1]https://www.osgi.org/developer/architecture/

assumption is due to the fact that the Controller is expected to have larger resources for handling requests from service consumers and notably more stable (not susceptible to crashes due to lack of resources like memory, computing power etc.). The SPF Controller would also be associated with a database containing a registry or records of the PIG resources and services available, so would need larger computation resources as opposed to remote IoT assets/resources.

An example of a possible request for a HADR operation could be the request of an elaborated video feed from a camera in a certain location, e.g. the monitor the presence of moving objects after an earthquake. In this example, a C2 application or a user would request the elaborated video stream by sending a command to the MTOC application using a MQTT message containing a JSON payload with a list of parameters: the action to be performed (object detection on video feed), the location of the camera specified as latitude and longitude, the resolution of the requested video feed, and so on.

The request for service is elaborated when the MTOC running the IoT application receives the command and reads the JSON payload contained in the MQTT message. The MTOC sends the service invocation request along with other required parameters to the PAI of the SPF Controller using one of the supported protocol, in this case MQTT. The request is then received by the SPF Controller, which looks up in the associated Asset database in order to find information regarding PIGs and resources that need to be invoked to serve the received request. In particular, the SPF controller looks for a PIG capable to serve request based on multiple characteristics such as its location (being matched as requested by the C2 application) and the resources (assets, computational power, and so on) associated to that PIG.

When a PIG capable of serving the request is found, the SPF Controller communicates with the PIG by forwarding it the request using the SPF proprietary protocol. Then, the SCM schedule the request to start the object detection algorithm on the video stream on the correct pipeline. If the elaboration requires multiple capabilities such as the transcoding and decoding of the video stream, the SCM would coordinate the effort of more pipelines and/or services. Finally, the elaborated video stream can be sent over the network using the UDP connector or delivered to back to the SPF Controller and then to the C2 application, which requested it.

VII. Conclusion

The paper describes an extended architecture of an Edge Computing solution based on SPF for a military HADR scenario which can be extended for any third party usage such as Smart City ICT systems. The idea is to make the service interfaces abstract and extend them for a SOA approach. This would enable better cohesion between the interacting applications and allow for future application/service integration and scalability in operations. Also, a way to store and retrieve asset information and capabilities is described so that a specific asset can be used for a specific use-case based request in a HADR scenario.

Future work involves creating test-beds with integration of heterogeneous edge devices to test-out practicality and scalability of the proposed architecture. The existing architecture can be extended for addition of extra functionalities for edge specific operations both the PIG as well as the SPF levels. The concepts of workload distribution at run-time amongst the edge devices also has to be looked at.

References

[1] Atzori, Luigi, Antonio Iera, and Giacomo Morabito. "The internet of things: A survey." Computer networks 54.15 (2010): 2787-2805.

[2] Dastjerdi, Amir Vahid, and Rajkumar Buyya. "Fog computing: Helping the Internet of Things realize its potential." Computer 49.8 (2016): 112-116.

[3] Shi, Weisong, and Schahram Dustdar. "The promise of edge computing." Computer 49.5 (2016): 78-81.

[4] Liton, Melissa. “How Much Data Comes From IoT Devices?” Sumo-Logic, Sumologic, 2 Mar. 2018, www.sumologic.com/blog/machine-data-analytics/iot-devices-data-volume/.

[5] Kim, Tai-hoon, Carlos Ramos, and Sabah Mohammed. "Smart city and IoT." (2017): 159-162.

[6] Taleb, T., Dutta, S., Ksentini, A., Iqbal, M., Flinck, H. 6602641627;57190582612;55883979600;57202397683;6506821221; Mobile edge computing potential in making cities smarter (2017) IEEE Communications Magazine, 55 (3), art. no. 7876955, pp. 38-43.

[7] Fadele Ayotunde Alaba, Mazliza Othman, Ibrahim Abaker Targio Hashem, Faiz Alotaibi, Internet of Things security: A survey, Journal of Network and Computer Applications, Vol. 88, 2017, pp. 10-28, https://doi.org/10.1016/j.jnca.2017.04.002. (http://www.sciencedirect.com/science/article/pii/S1084804517301455)

[8] Hu, Yun Chao, et al. "Mobile edge computing—A key technology towards 5G." ETSI white paper 11.11 (2015): 1-16.

[9] Ahvenniemi, Hannele, et al. "What are the differences between sustainable and smart cities?." Cities 60 (2017): 234-245.

[10] Baer, Hans, and Merrill Singer. Global warming and the political ecology of health: Emerging crises and systemic solutions. Routledge, 2016.

[11] M. Tortonesi, M. Govoni, A. Morelli, G. Riberto, C. Stefanelli, N. Suri, Taming the IoT data deluge: An innovative information-centric service model for fog computing applications, Future Generation Computer Systems, 2018, https://doi.org/10.1016/j.future.2018.06.009.

[12] F. Poltronieri, C. Stefanelli, N. Suri and M. Tortonesi, "Phileas: A Simulation-based Approach for the Evaluation of Value-based Fog Services," 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Barcelona, 2018, pp. 1-6. doi: 10.1109/CAMAD.2018.8514969

[13] N. Suri, G. Benincasa, R. Lenzi, M. Tortonesi, C. Stefanelli and L. Sadler, "Exploring value-of-information-based approaches to support effective communications in tactical networks," in IEEE Communications Magazine, vol. 53, no. 10, pp. 39-45, October 2015.

[14] Johnsen, Frank T., et al. "Application of IoT in military operations in a smart city." 2018 International Conference on Military Communications and Information Systems (ICMCIS). IEEE, 2018.

[15] Pradhan, Manas, et al. “Toward an Architecture and Data Model to Enable Interoperability between Federated Mission Networks and IoT-Enabled Smart City Environments.” IEEE Communications Magazine, 16 Oct. 2018, pp. 163–169.

[16] Ben Taylor, Vicent Sanz Marco, Willy Wolff, Yehia Elkhatib, and Zheng Wang. 2018. Adaptive deep learning model selection on embedded systems. SIGPLAN Not. 53, 6 (June 2018), 31-43. DOI: https://doi.org/10.1145/3299710.3211336

[17] Jakub Floty´nski, Kamil Krysztofiak, and Daniel Wilusz, "Building Modular Middlewares for the Internet of Things with OSGi", The Future Internet, 2018, Springer.

[18] Vallati, C., Mingozzi, E., Tanganelli, G. et al. Wireless Pers Commun (2016) 87: 1071. https://doi.org/10.1007/s11277-015-2639-0