Networked Smart Objects: Moving Data Processing Closer to the Source

Alessandra Rizzardi[1(✉)], Daniele Miorandi[2], Sabrina Sicari[1], Cinzia Cappiello[3], and Alberto Coen-Porisini[1]

[1] Universita' degli Studi dell'Insubria, Via Mazzini 5, 21100 Varese, Italy
{alessandra.rizzardi,sabrina.sicari,alberto.coenporisini}@uninsubria.it
[2] CREATE-NET, Via alla Cascata 56/D, 38123 Trento, Italy
daniele.miorandi@create-net.org
[3] Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy
cinzia.cappiello@polimi.it

Abstract. The satisfaction of security and data quality requirements plays a fundamental role in the Internet of Things (IoT) scenario. Such a dynamic environment requires the adoption of heterogeneous technologies to provide customized services in various application domains and both security threats and data quality issues need to be addresses in order to guarantee an effective and efficient data management. In this paper, a lightweight and cross-domain prototype of distributed architecture for IoT is presented and evaluated by means of open data provided by different sources. We show how users can access different types of data by changing security and quality requirements.

Keywords: Internet of Things · Security · Data quality · Middleware · Prototype

1 Introduction

The term "Internet of Things" (IoT) [1] is becoming widely used for broadly defining a future in which objects equipped with sensing and actuation capabilities get connected to a global networked infrastructure. An IoT system can be depicted as a collection of smart devices which interact on a collaborative basis to fulfill a common goal, acquiring data from and acting upon the environment they are in. Security, privacy and data quality represent critical requirements, which can hinder the large scale adoption of IoT services. Furthermore, IoT deployments are characterized by a large heterogeneity in terms of adopted technologies; some deployments may include a large number of devices, leading to scalability issues to be faced. The most crucial challenge in building an IoT system lies in the lack of common and standardised software framework. To fill this gap, the adoption of Service Oriented Architectures (SOA) and Service-oriented Communications technologies [2,3] in IoT is shared by the majority of scientific community, but the state of the art is mostly limited to starting research activities[1]. Furthermore, several middleware layers are employed to enforce the integration and the security of devices and data within the same information network [4–6]. Several related scenarios have been encompassed in related EU projects, such as: FP7 COMPOSE (Collaborative Open Market to Place Objects at your Service), iCORE, IoT.EST (Internet of Things Environment for Service Creation and Testing), Ebbits, uTRUSTit, Butler[2].

Besides security and privacy, also data quality has to be addressed. In [7], authors claim the need of control over data sources to ensure their validity, information accuracy and credibility. Accuracy, timeliness and the trustworthiness are instead the dimensions considered in [8]: anomaly detection techniques are widely employed to remove noises and inaccurate data in order to improve data quality. Besides temporal aspects (i.e., currency) and data validity, another important dimension is availability [9]. Authors defined new metrics for the cited quality dimensions in the IoT environment and evaluate the quality of the real world data available on an open IoT platform called Cosm. They have shown that data quality problems are frequent and they should be solved or at least users should be aware of the poor quality of the used data sources.

In order to address these issues, we propose a flexible and distributed IoT architecture based on the idea of bringing processing, security and data qualification closer to the actual data sources. In a previous work [10], we presented a security-and quality-aware system architecture for IoT, based on the concept of the Networked Smart objects (NOS), which are computationally powerful smart nodes aiming to create a distributed storage in order to handle the data acquired from large-scale IoT deployments. In order to ease the development of applications and the management of the system, a middleware has been designed and prototyped. It includes features for users and applications to dynamically specify the minimal level of security and data quality suitable for their own purpose. The distributed architecture automates the deployment of adequate filters for ensuring that only qualified data are passed over to the actual service. The prototype is evaluated connecting real-time open data services to a simple visualization dashboard. The main innovative contribution lies in the implementation of a highly modular and lightweight architecture for the NOS, able to (i) provide proper interfaces with heterogeneous data sources (ii) automatically evaluate the security and quality of the received data, and (iii) provide standardized interfaces and data models for applications/services to access qualified IoT information, where raw data is enriched with metadata specifying their security and quality levels.

2 Architecture

In the IoT system we identified two main entities: (i) the nodes, which are the sources of the data and can be represented by heterogeneous devices (RFID,

[1] www.hcilab.org/projects/perci/index.htm, cs.adelaide.edu.au/peertrack.

[2] iot-at-work.eu, iot-icore.eu, ict-iotest.eu/iotest, www.ebbits-project.eu, www.utrustit.eu, www.iot-butler.eu/.

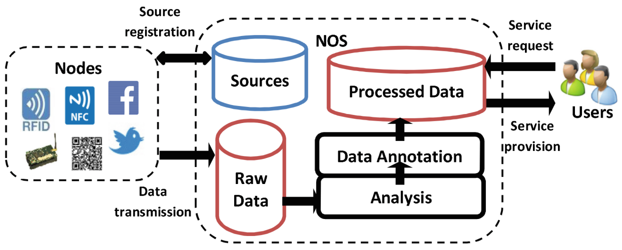


Fig. 1. System architecture

NFC, WSN, actuators, social networks); (ii) the users, who interact with the IoT system by requesting some information by means of services. In order to handle such a huge amount of data closer to the sources and better satisfy the users service requests (in terms of quality and reliability), the NOS layer is introduced (Fig. 1) [10]. NOSs are networked smart nodes without strict constraints in terms of energy and computational capabilities. They include self-organizing features and can be deployed where and when needed, in a distributed manner; through their interface with enterprise platforms and IoT enabling technologies, they can be used to enrich software platforms, making them able to interact with the physical world following the adopted rules. NOSs collect the data provided by different sources, which can be registered and no registered. Registered sources may specify an encryption scheme for their interactions with NOSs. Since the received data are of different types and formats, NOSs initially put them in the storage unit named *Raw Data* and elaborate them according to the two-phases named *Analysis* and *Data Annotation*, in order to obtain a uniform representation.

Firstly, the data are analyzed (*Analysis* phase) in terms of security and data quality. As regards security [11], if the data source is a registered one, the information access requires the authentication of the source and the decryption of the data; therefore, the system preserves indeed complete knowledge, including encryption scheme, used key etc. Each encryption schema (ECC, RSA, PKI) has associated a score in the range [0:10], established by the system administrator, who, according to the required security levels, defines the score assignment rules. Then, a score is assigned to: authentication, confidentiality, integrity and privacy. For registered sources: (i) *confidentiality* and *privacy* are evaluated assigning a score to the encryption schema (based on the robustness of the used encryption technique and of the adopted key distribution schema); (ii) *integrity* is checked and a score is given (0 for a violated and 2 for a unviolated data); (iii) *authentication* is set to 1. For non registered sources confidentiality, privacy and authentication are set to 0, while integrity is set to 1. Note that some registered sources may use neither authentication credentials nor encryption. Further, for unregistered/unknown sources no information may be available, therefore the scores have associated a low value. As regards data quality assessment, a score in the range [0:10] is assigned to timeliness, completeness, accuracy and precision levels [12,13]. *Timeliness* is conceived as the temporal validity of data and is calculated on the basis of the freshness (currency) of data and on the frequency of changes (volatiliy), as in Eq. 1.

$$Timeliness = \max\left(1 - \frac{Currency}{Volatility}, 0\right), \tag{1}$$

where currency can be defined as the interval from the time when the value was sampled to the time instant at which data are sent to NOS. Volatility is a static information which indicates the amount of time units (seconds) during which data remain valid; it is usually associated with the type of phenomena that the system monitors and depends on the change frequency. *Completeness* is calculated as the ratio between the numbers of collected values and the number of expected values. *Accuracy* is usually defined as the degree of conformity of a measured quantity to its actual (i.e., true) value; it is also related to *precision*, which is the degree to which further measurement or calculations show the same or similar results. Formally, accuracy can be defined as the error expressed by the difference between the mean of the measurements v_n and a reference value v_{ref}. The measure is considered accurate if such difference is smaller than the acceptable measurement error ϵ_{acc}. Precision is often characterized in terms of the standard deviation of the measured values: the smaller the standard deviation, the higher the precision. For this reason, a measure can be considered precise if the reciprocal of variance is smaller than ϵ_{prec}. Note that in case of streaming data such metrics have to be calculated periodically on data samples in order to detect quality changes over time and also to gather information about the trustworthiness of the source. The choice to provide a score for each security and data quality requirement makes such a solution flexible for smart integration in different application scenarios. Finally, *Data Annotation* phase represents the information obtained from *Analysis* phase according to a specified format [10] and includes also a semantic description of the data content: the data are annotated with a set of metadata (a score for each security and quality level). The annotated data are stored in the storage unit named *Processed Data*.

3 Prototype

In this section we describe a lightweight data management and service middleware platform consisting of distributed NOSs. The proposed prototype is based on the architecture described in Sect. 2 and its main features are:

- *Data acquisition and analysis:* the middleware is modelled in order to manage registered and no registered sources, characterized by different security communication schemas and providing diverse quality levels for their data.
- *Unified data representation:* the middleware addresses the need of an uniform representation for the data by providing a format for annotating data once they are processed by the NOS layer.

– *Standardised design:* in order to deal with heterogeneous services, the middleware consists of a set of lightweight modules and interfaces working in a non-blocking manner through distributed NOS devices, aiming to perform data analysis, discovery, and query.
– *Reconfigurability:* since the existing IoT deployments are conceived for very specific applications and thus hardly reconfigurable, the middleware supports dynamic reconfiguration capabilities and can be remote orchestrated through internet/intranet settings, which are based on open standards.

More in details, the prototypical implementation of NOS has been realized by means of *Node.JS* platform[3] and *MongoDB*[4] for storage management. The code is released openly[5]. Modules interact among themselves through *RESTful* services. They can be distinguished in: node interfaces, processing modules, and service interfaces. The node interfaces manage the sources registrations, receive the data from the external sources, and insert them in the storage of *Raw Data*. The processing modules are in charge of performing *Analysis* and *Data Annotation* phases. Finally, the service interfaces publish the *RESTful* services and web pages to which users or external applications can interact at any time.

Note that, a great advantage of such an implementation is that it is possible to add new modules or duplicate the existing ones since they are able to work in a parallel (non-blocking) manner or to define new functionalities or removing the active ones. The exploitation of *MongoDB*, which is a no-relational database (i.e., NoSQL), consent to the data model to evolve dynamically; in fact, data are handled as document in *JSON* format, which is a lightweight data-interchange format and is both easy for humans to read and write and easy for machines to parse and generate. Hence, such an implementation is independent both from the data model and the application domain. In a production setting the usage of *MongoDB* would be more likely replaced by a message queueing system providing durability guarantees (e.g., Apache Kafka[6]) and a stream processing framework (e.g., Apache Storm[7]) implementing the analysis and data annotation tasks.

4 Application Case Study

In order to verify the behavior of the presented middleware platform, a set of open data are exploited as sources for running the system. These are obtained in real time from six sensors at the meteorological station of the city of Campodenno (Trentino, Italy). The measures, referred to temperature, humidity, wind, energy consumption, air quality, are retrieved in *JSON* format, which is compliant with *MongoDB*, through *GET* requests. Furthermore, also the data in the storages are stored in *JSON* format.

[3] nodejs.org/.
[4] www.mongodb.org/.
[5] bitbucket.org/alessandrarizzardi/nos.
[6] kafka.apache.org/.
[7] storm.apache.org/.

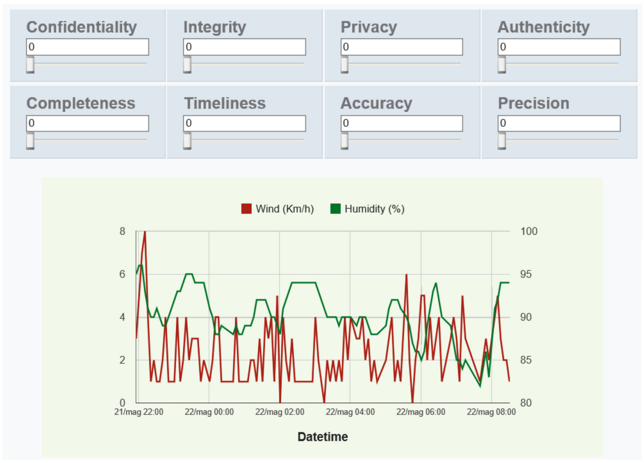
The experimental setup includes: a Raspberry Pi on which NOS platform is installed; a laptop, which emulates the behavior of several nodes which send the data; a set of user devices, which access the provided service. As just said, the Raspberry Pi executes NOS functionalities and communicates with the nodes running on the laptop. The data analyzed by NOS are accessible through a RESTful service and can be interpreted by various applications (web services, desktop applications, mobile applications). Users can connect to a public IP address by means of their computers, tablets or smartphones, in order to visualize on their browser the gathered data, dinamically filtering them on the basis of the required security and quality levels, in terms of confidentiality, integrity, privacy, authentication, completeness, timeliness, accuracy, and precision, respectively, through the sliders provided by the web service itself, as shown in Fig. 2. Since meteorological data are obtained from six different sources, we assign to them the security and quality scores through an analysis of the incoming data, as specified in Table 1.

Table 1. Source parameters

Parameters	Source 1	Source 2	Source 3	Source 4	Source 5	Source 6
Authentication	1	1	0	1	0	0
Security schema score	10	6	0	2	0	0
Privacy schema score	10	6	0	2	0	0
Timeliness	9	8	6	3	9	7
Completeness	9	10	8	6	7	10
Accuracy	9	7	5	6	7	10
Precision	9	8	4	5	8	9

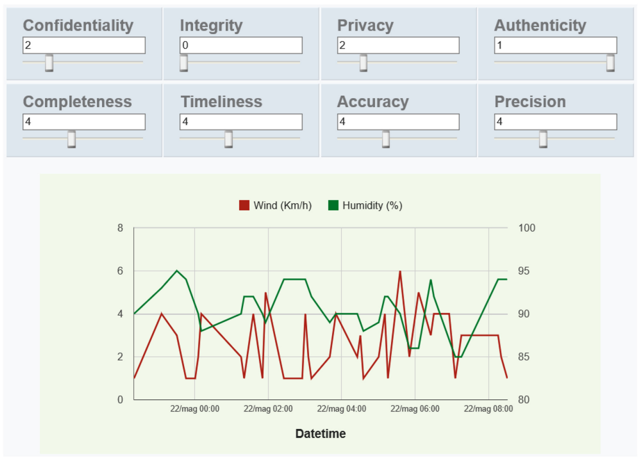
Figure 2(a) shows in the graph the data of wind speed (in Km/h) and humidity (in %) processed by NOS without any filters of security or quality; in fact all the sliders are set to 0, therefore the data provided by all the sources are displayed to the requesting user. Whereas, in Fig. 2(b) both filters on security and quality are applied: the data, shown in the graph, are obtained from authenticated sources (the parameter related to the authentication is set to 1), for which the robustness of the encryption algorithm has a score equal or higher than 2; while the data quality parameters have to be equal or higher than 4. We remark that, in this case, the web service provides less data with respect to Fig. 2(a).

We remark that this represents only an example of NOS application in a context which has to deal with data received in real time. Other possible applications are: the management of smart home and/or buildings from an energy consumption point of view; the analysis of business activities in real time; the retailing experiences; the search for restaurants or other places related to tourism experiences; all the domains which require an asynchronous data analysis in order to take strategic decisions. In fact, from an application standpoint, the acquisition



Confidentiality 0	Integrity 0	Privacy 0	Authenticity 0
Completeness 0	Timeliness 0	Accuracy 0	Precision 0

(a) `No_Filters`



Confidentiality 2	Integrity 0	Privacy 2	Authenticity 1
Completeness 4	Timeliness 4	Accuracy 4	Precision 4

(b) `Security_and_Quality_Filters`

Fig. 2. Results

and management of data in real time represent critical issues also regarding user satisfaction and market expansion of specific categories of services.

5 Conclusions

In this paper, a distributed IoT middleware, named NOS, has been designed and prototyped. Since security, privacy and data quality have been identified as critical requirements for the large scale adoption of IoT applications, NOS aims to create a distributed storage for handling IoT data along with their security and data qualification closer to the actual data sources. The analyzed data are then provided in real time to users, which are aware of the levels of security, privacy and quality of the data themselves. The prototype has been implemented with a highly modular and lightweight design, and NOS finally runs on a Raspberry Pi. Also an example of application case study is presented, in order to clarify the functionalities of NOS and its dynamic structure. In the future, we are planning to evaluate the robustness of the NOS architecture in a real IoT scenario, exploiting sensitive data and applying a new defined secure score algorithm.

References

1. Miorandi, D., Sicari, S., De Pellegrini, F., Chlamtac, I.: Survey internet of things: vision, applications and research challenges. Ad Hoc Netw. **10**(7), 1497–1516 (2012)
2. Papazoglou, M., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: state of the art and research challenges. Computer **40**(11), 38–45 (2007)
3. Yu, Q., Liu, X., Bouguettaya, A., Medjahed, B.: Deploying and managing web services: issues, solutions, and directions. VLDB J. **17**(3), 537–572 (2008)
4. Conzon, D., Bolognesi, T., Brizzi, P., Lotito, A., Tomasi, R., Spirito, M.: The virtus middleware: an XMPP based architecture for secure IoT communications. In: 2012 21st International Conference on Computer Communications and Networks, ICCCN 2012, Munich, Germany, pp. 1–6, July 2012
5. Gòmez-Goiri, A., Orduna, P., Diego, J., de Ipina, D.L.: Otsopack: lightweight semantic framework for interoperable ambient intelligence applications. Comput. Hum. Behav. **30**, 460–467 (2014)
6. Liu, C.H., Yang, B., Liu, T.: Efficient naming, addressing and profile services in internet-of-things sensory environments. Ad Hoc Netw. **18**, 85–101 (2013)
7. Guo, B., Zhang, D., Wang, Z., Yu, Z., Zhou, X.: Opportunistic IoT: exploring the harmonious interaction between human and the internet of things. J. Netw. Comput. Appl. **36**(6), 1531–1539 (2013)
8. Metzger, A., Chi, C.-H., Engel, Y., Marconi, A.: Research challenges on online service quality prediction for proactive adaptation. In: 2012 Workshop on European Software Services and Systems Research - Results and Challenges (S-Cube), pp. 51–57, June 2012
9. Li, F., Nastic, S., Dustdar, S.: Data quality observation in pervasive environments. In: Proceedings of the 2012 IEEE 15th International Conference on Computational Science and Engineering, pp. 602–609. IEEE Computer Society (2012)
10. Sicari, S., Cappiello, C., Pellegrini, D., Miorandi, D., Coen-Porisini, A.: A security- and quality-aware system architecture for internet of things. Inf. Syst. Front. **18**(4), 665–677 (2016)
11. Sicari, S., Rizzardi, A., Grieco, L.A., Coen-Porisini, A.: Security, privacy and trust in internet of things: the road ahead. Comput. Netw. **76**, 146–164 (2015)
12. Cappiello, C., Schreiber, F.A.: Quality- and energy-aware data compression by aggregation in WSN data streams. In: Proceedings of the 2009 IEEE International Conference on Pervasive Computing, Communications, pp. 1–6. IEEE Computer Society, Washington, DC, USA (2009)
13. Klein, A., Lehner, W.: Representing data quality in sensor data streaming environments. J. Data Inf. Qual. **1**(2), 10:1–10:28 (2009)