Model-based Interoperability IoT Hub for the Supervision of Smart Gas Distribution Networks

Ahmed Ahmed, Mathias Kleiner, and Lionel Roucoules

Abstract—Industrial monitoring environments have evolved from single monolithic systems to widely distributed heterogeneous systems. These include the Internet of Things, Industrial IoT, Cyber- Physical Systems and Enterprise Application. One of the key challenges is the integration of heterogeneous systems and data exchange interoperability. In the industrial smart gas project in which this work takes place, current standard-based or middleware solutions are not sufficient to handle these issues and often require specific ad-hoc developments. This paper proposes a generic, modular, and extensible interoperability architecture based on modeling principles. We provide a free software implementation and illustrate the approach on industrial usecases. Some criteria are then proposed for a first qualitative evaluation.

Index Terms—interoperability, data aggregation, smart systems, model-based engineering, Industrial Internet of Things (IIoT).

1 Introduction

During the past years, there has been a continuous growth and rapid evolution in the IT infrastructure of industrial/enterprise environment. It has evolved from containing single monolithic systems to widely distributed heterogeneous systems. This includes the Internet of Things (IoT) systems, Industrial IoT systems [1], Cyber-Physical Systems (CPS) [2] and Enterprise Application (EA). This integration and the collaboration between these systems results in a large complex system called System of Systems (SOS) [3]. The emerging smart environments, such as Smart Grids, Smart Gas network, Smart Cities, Smart Home, Future Industry 4.0, etc. are examples of SOS [4], [5], [6].

As we are moving towards larger complex systems where millions of devices, applications and systems need to be integrated, the requirement for an inexpensive and rapid integration solutions is an essential need [7]. Thus, this work proposes the study and the development of a generic, modular, and extensible interoperability architecture that is based on modeling principles. It aims to ease the system integration and promote interoperable data exchange between the heterogeneous systems in the industrial and smart environments. In this paper, it is applied in the context of a smart gas network.

This paper is organized as follows: The next subsections introduce the various systems that constitute the smart gas environment, a running example and the problematic tackled by this paper. The second section is a survey of existing related work. The third section presents the conceptual and technical architecture proposal. This will be followed by some use cases in the fourth section. An evaluation will be presented in the fifth section. Finally, the sixth section concludes with some insights on ongoing and future work.

- *A. Ahmed is with Arts et Métiers ParisTech, France and Sogeti-HT Cagemeini, France.*
  E-mail: ahmed.ahmed@capgemini.com
- *M. Kleiner and L. Roucoules are with Arts et Métiers ParisTech, France.*
  E-mail: {mathias.kleiner,lionel.roucoules}@ensam.eu

Manuscript received April 00, 0000; revised August 26, 0000.

1.1 Context

This work takes place in a French national project[1] that manages a real-time smart gas distribution network. In the current network specifically, and in the industrial environment generally, many distributed systems are used to perform useful operation independently and to increase the enterprise competitiveness [7]. An example of such systems in the smart gas network is shown in figure 1. It includes various Supervisory Control and Data Acquisition (SCADA) architectures [8], that is a vertically integrated system, for representing the vertical data exchange between equipment and Human Machine Interface. In a given environment, various SCADA systems may co-exist from different vendors, each with its own standard for accessing, manipulating, and representing data [7] such as OPCUA [9], MQTT[2], radio transmission based network like Sigfox[3], etc. Each vertical oriented closed system is able to work separately and independently.

Furthermore, other enterprise applications (EA), simply referred by systems, coexist such as Computerized Maintenance Management System (CMMS), Enterprise Resource Planning (ERP), Geographic Information System (GIS), decision-support systems, logistic systems, and others. They are provided by different vendors, using different standards, programming language and protocols [7].

The heterogeneity of these systems, due to different communication mechanisms, data format, and data semantics, makes the interoperable exchange of data very challenging. However, the current architecture, mainly based on the use of a single standard (OPC Unified Architecture - OPCUA [9]), relies on system-specific ad hoc developments whenever new systems have to be integrated. This article proposes an interoperable integration platform for the global supervision system that is an extension to the work introduced in [10].

1. Gontrand FUI project: https://www.gontrand.net/
2. http://mqtt.org/
3. http://www.sigfox.com/

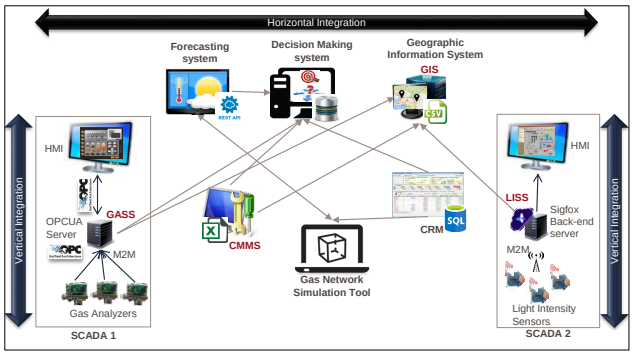

Fig. 1. Example of distributed systems used in the smart gas network.

1.2 Running Example

This example will be used throughout the paper to illustrate the proposal on a simplified scenario of information exchange in a smart gas network supervision environment. It is composed of a subset of systems that were shown previously in figure 1. Throughout this paper, the system providing the information and the system using it is referred as data producer (DP) and data consumer (DC), respectively. It is composed of three DPs: A **Gas Analyzer SCADA System (GASS)** that monitors the quality and quantity of gas at certain points of the gas network on real-time. It exposes the data using OPCUA protocol; A **Light Intensity Sensing system** (LISS) (DP) that provides light intensity information on real time. The information is available through a (Rest) web service from a Sigfox Cloud back-end server. Sigfox is a low-cost communication solution that encodes the data using simple 12 bytes data chunks; A **CMMS** that provides maintenance and geolocation information for the analyzers and the light sensors. Their data is available as a shared tabular file (Excel format). Furthermore, this example includes a **Geographic Information System (GIS)** (DC). It displays the GASS (e.g. CH4) and LISS (e.g. Light Intensity) information on their appropriate location on the map in real time. Additionally, operational rules are required on the sensors data before it can be presented to the GIS. Data must be provided through a shared tabular file (CSV format).

1.3 Issues

The information exchange between the heterogeneous systems in the running example impose the need for an interoperability solution such as adhoc and middleware solutions. Thus, a functional decomposition of such solutions leads us to the following main issues: **1. Different communication mechanisms** are used by the different systems in order to provide or consume data (here OPCUA, Rest web service, file sharing); **2. Different data formats** are used by the systems (here OPCUA Data Model, plain bytes, tabular CSV and Excel files); **3. Different data semantics** coexist. Indeed, each system has its own semantics in producing or interpreting data. These can range from simple naming (for instance, here the geolocation information uses different coordinates system) to more complex structural differences; **4. Operational/business rules** may be required in order to process input data before it can be presented to the consumer (here, the GIS system requires an additional aggregated value);**5. Various interaction paradigms** are used by the communication protocols for the dissemination of data such as request/reply and publish/subscribe [11]. The first three issues require to address interoperability at the technical level, syntactic level and semantic level, respectively [12]. The fourth issue focuses on domain knowledge by applying business rules. The last issue also belongs to the technical interoperability for data dissemination. It should be noted that these differences can be independent from each other. For instance, multiple systems can share a communication mechanism while using different data formats and/or semantics.

2 Related Work

Smart gas platforms are a relatively recent application area which led to various ad-hoc developments. However interoperability solutions can rely on level-specific studies and on the various proposals made for similar architectures (such as the broader IoT scope). The first part introduces the existing work that focuses on one of the three interoperability level, then the second part presents more general solutions that can be considered in our context.

2.1 Level-specific interoperability

2.1.1 Communication Interoperability

Numerous standards, protocol and mechanisms exist for device and application communication such as OPCUA [9], PROFIBUS, Modbus, MQTT [13], COAP [14], radio transmission for Sigfox network, Restful web service [15], and so on. Heterogeneous environments often need a combination of these mechanisms for the systems to communicate with each other.

2.1.2 Syntactic Interoperability

When interacting with external systems, various interchange format and standards are used such as Extensible Markup Language (XML), JavaScript Object Notation (JSON), Comma-Separated Values (CSV), databases, etc. The XML Metadata Interchange (XMI) [16] format is an OMG standard for exchanging metadata information via XML. It is widely used for the serialization of Metaobject Facility (MOF) metamodels. Heterogeneous environments must interact with a variety of these formats.

2.1.3 Semantic Interoperability

Many approaches are proposed to deal with semantic interoperability. One way is creating domain-specific data models such as the Common Information Model (IEC 61970, 61968 and 62325) [17] that defines the components of the electrical power systems and their relationships, Gas Distribution Model (GDM) [4] for representing the components of gas network, etc. Another way relies on domain ontologies [18]. Ontologies are *formal explicit specifications of a shared conceptualization* [19]. Again, ISO-15926 also provides an ontology for oil and gas industry. [20] is an ontology for manufacturing system engineering ontology. These domain

4. http://gtigpsresearch.blogspot.com/p/gas-distribution-model.html

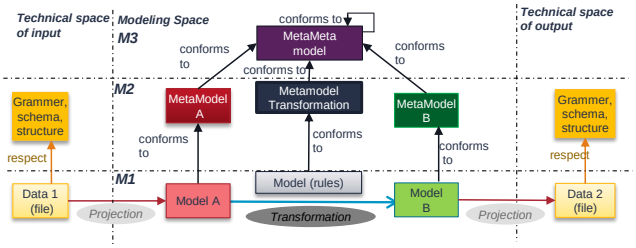

Fig. 2. MBE projection and transformation concepts

ontologies and data model standardization have brought semantic convergence for information exchange, however, the lack of an environment complying to the common agreements and data model is still the main encumbrance for interoperability. OPC UA [9] in addition to its communication mechanisms, provides a generic way to create data models. Some commercial tools like Kepware [5] propose ad-hoc adapters to interoperate with no-opcua systems. Work, involving data model mapping, has been done in the electricity domain for the mapping of CIM to OPCUA [21]. Another generic data model is Open Data Format (O-DF) which aims at providing a unified semantic at a domain agnostic abstract level [22].

2.1.4 Model-based engineering (MBE)

In the last years, this field of software engineering has studied how different data formats and semantics can be related and technically handled in a generic architecture. Although it originates from software modeling and generation techniques, it has since been successfully applied to represent and exchange information from the physical world [23].

MBE naturally promotes separation of concerns. It uses "models" as a unifying concept to represent information [24]. The community distinguishes three levels of models: (terminal) model, metamodel, and metametamodel. A **model** (M1) is a representation of a system and a way to vehiculate information (M0) that is described through a modeling language called **metamodel**. This **metamodel** (M2) is itself described by a meta-language i.e. the metametamodel (M3). The OMG proposes a standard 4-layers metametamodel called Meta Object Facility (MOF) to implement these models as shown in figure 2.

MBE provides many techniques for model operations that can be used to achieve interoperability in our context. At the syntactic level, *projections* allow to convert data formats from/to specific technical spaces (TS) to/from the modeling environment. These can be implemented manually or facilitated by using XML or grammar-based tools [25]. At the semantic level, *transformations* [26] allow to map data from one metamodel semantics to another. Such metamodels may either be domain-specific, based on standards, or provide custom unified semantics. Some attempts have been made in order to use models to promote interoperability for enterprise software integration [27] and SOS [28]. They focus mainly on defining interoperability models and the generation of code.

5. https://www.kepware.com

2.2 Multi Level Interoperability

2.2.1 Standards

Many organization tackle the interoperability issue by using domain-specific standards. These standards may handle one or more interoperability levels. In our context, we may consider ISA-95 for Enterprise-Control System Integration [29], MIMOSA [6] and PRODML [7] for the oil and gas interoperability solutions, ISO-15926 for data modeling and data integration including oil and gas industry [30], etc.

2.2.2 Architectures and Middlewares

All the interoperability aspects discussed previously must be applied in an architectural solution so that all the aspects are glued together and complete each another. Therefore, some work-groups and organization steer their effort to decompose the interoperability issues by defining a reference model in architectural approaches, such as Grid Wise Architectural Council (GWAC) [30], the Smart Grid Architecture Model for electricity domain [4] and Reference Architecture Model for Industry 4.0 (RAMI 4.0) [31]. These architectures are divided into several abstract layers representing interoperability categories for business, policies, functions, information, and communication, etc.

Service Oriented Architecture is a conceptual-level architectural model that relies on services for integrating systems. It is used by the"IoT Architectural Reference Model (ARM)" [32] that provides a common structure and guidelines for the Internet of things. Arrowheam [33] is also a SOA-based Framework for the interoperability and integration aimed at enabling IoT, Industrial IoT and system of systems. Middleware solutions helped to facilitate the design and development of concrete SOA solutions mostly referred by "Service oriented middleware" (SOM) [34]. SOCRADES Integration Architecture for IoT is one of the SOM solutions that allows the integration between things (devices) and applications using web services [35].

IoT hub-based solutions have been created to achieve a degree of interoperability through dedicated middlewares. Some of these hubs, such as ThingsSpeak and Xively, are based on web technologies and protocols, therefore referred as Web of things interoperability solutions.

2.3 Discussion

Level-specific interoperability solutions handles the communication, syntactic and semantic interoperability separately, therefore, they are not sufficient in our context. Indeed, more generic solutions that bind all the interoperability levels are required. Generic solutions involves imposing standards such as [29], [30] MIMOSA, PRODML for gas and oil industry. However, authors in [36] concluded that there is no unified standard for system integration in the oil and gas industry. Another works in [37] also showed the limitations of these standards for smart gas networks. The National Institute of Standards and Technology (NIST), that introduces an interoperability framework for Smart Grid, concluded that hundred of standards are required which themselves must be interoperable, to achieve end-to-end

6. http://www.mimosa.org/
7. http://www.energistics.org/production/prodml-standards

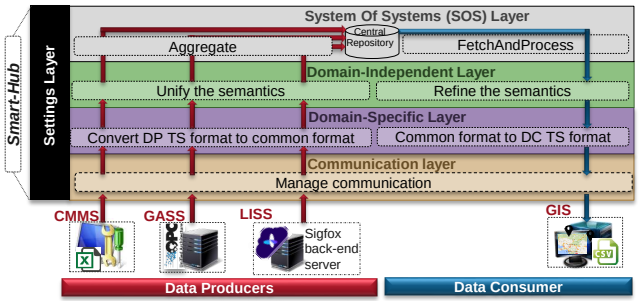

Fig. 3. Conceptual Smart-hub Architecture

interoperability [38]. Additionally, standards have to be reworked (and thus systems adapted) whenever new technical or semantic issues need to be taken into account. To sum up, in such heterogeneous environment, it is unpractical to impose and comply with a single standard [39]. However, when applicable, support of standards greatly reduces the need to develop specific components.

The model driven efforts to handle interoperability [27], [28] focus on semantics and code generation for integration purposes, which do not take in consideration all levels of interoperability. In the architectures and reference models given by [4], [30], [31], the communication and information interoperability were defined by domain-specific standards. In a heterogeneous environment, prior guaranty that all systems comply with the same agreement of standard is not granted. The SOA architecture presented in [32] is very abstract and provides an approach for modeling an IoT system. However, it does not provide an approach for the technical implementation of the system under study. Works in [33], [34], [35], ThingsSpeak [8], Xively [9] provided technical solutions, however, there are applicable to service based/web-based systems with specific syntax and semantics. However, in our context, there are many non service-based systems (database applications, file based application, etc.) and different syntax and semantics need to be managed. Therefore, specific adapters must be developed to integrate these systems which are expensive in terms of cost and time.

All the previous works impose a particular standard, solution or technology to promote a holistic and a non-extensible interoperability solution. However, to the authors knowledge, there is no existing solution that promotes the interoperability by extending its different levels and combining them as a single solution. Thus, these existing holistic solutions can be reused as a single unit but cannot promote the re-usability at a specific interoperability level. This latter is one of our requirements to facilitate the adaption of the interoperability solutions to external system changes which will result in improving the development cost and time.

In this papers, the authors attempt to enable the interoperability between heterogeneous systems through a generic, conceptual, technically implementable, **modular** and **extensible** architecture that promotes **re-usability** at each interoperability level, easily **adapt** to external system changes and **does not impose** a specific standard.

8. https://thingspeak.com/
9. https://www.xively.com

3 Smart-hub

In the first part of this section, the authors present the conceptual architecture of the approach which promotes modularity by relying on a deep separation of concerns. Then the technical implementation, which facilitates extensibility by relying on unified model-based techniques, is described.

3.1 Conceptual Architecture

Figure 3 introduces a 5-layers architecture dubbed Smart-hub. It acts as a hub between data producers (DPs) and data consumers (DCs) heterogeneous systems.

- **Communication layer:** This layer provides a "Catalog Repository" for a set of data connectors to interface, establish communication and exchange data with the external systems either directly or via the system's proxy/gateway.
- **Domain-Specific Layer:** This layer manages various formats by converting the data provided by DPs from their technical space format to a common format. Similarly, it converts the data from the common format to the data consumers technical space format. This process of conversion between formats relies on a data repository (Common Format Repository) that includes the common format description of the systems and a functional repository (Conversion Rules) that includes the format conversion procedures.
- **Domain-Independent layer:** This layer addresses the different semantics of data. It includes a functional repository (Transformation Rule) that manages the conversion rules to unify the data provided by the domain-specific layer to common semantics and to refine the common-semantics to domain-specific data. It also includes a repository to hold these intermediate data.
- **System of Systems Layer:** This layer includes the Central repository to hold the unified data. It aggregates the data provided by the domain-independent layer, and a functional repository (Data Processing Rules) holds rules that allow for fetching/processing the data required by DCs.
- **Settings Layer** This layer manages the configuration data for connected systems.

An Orchestration layer (not presented) handles the data dissemination from DPs and to DCs according to different interaction patterns. However, it is out of the scope of this paper. Simply, you can consider that the Smart-hub requests and publishes the data from DPs and to DCs, respectively.

3.2 Architecture implementation

The implementation of the architecture heavily relies on the use of the model-based technologies previously discussed. (Meta)models are used for modeling the exchanged data, the unified semantics and the Smart-hub configuration itself. This has several advantages. First, abstract modeling eases common understanding between domain experts and developers. Second, syntactic and semantic interoperability are achieved using the same paradigm which, coupled with efficient model-based software tools, allows to ease developments and further extensions. The authors have chosen the Eclipse Modeling Framework (EMF)[10] to support the implementation.

10. https://eclipse.org/modeling/emf/

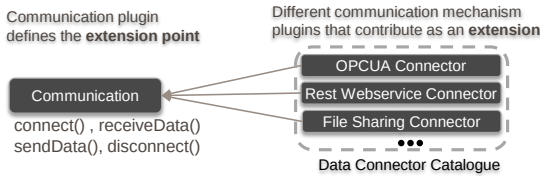

Fig. 4. Communication Layer.

Additionally, thanks to the conceptual separation of concerns that permits the creation of loose coupling components. Loose coupling eases the integration and addition of components.

3.2.1 Communication Layer

The communication layer must provide the following functionality: establish a communication with the external system; disconnect from the external system; receive data from the data producer; send data to the data consumer. The different communication mechanisms (OPCUA client connector, file sharing, Rest Webservice connector) are created as identified extension plug-ins that implement and contribute to these functionalities. These components for our running example are shown in Figure 4.

3.2.2 Domain-Specific Layer

This layer manages the projection of data formats from the technical space of the connected system to the modeling environment (*ProjectToModel*) and vice-versa (*ProjectFromModel*). For each format, a plug-in contributes and implements a domain-specific projection rule.

Here model-based technologies greatly reduce the development times. First, XML or grammar-based data formats can be projected by software tools using simple rules [25]. Second, a domain-specific metamodel can be shared by different systems. Either because it is a generic format (such as the OPCUA data model), or because the developer can group systems that provide the same domain-specific data. After the data is received via OPCUA connector, it is handled by this layer to generate a model conforming to the generic OPCUA metamodel.

3.2.3 Domain-Independent layer

This layer uses a single meta-model that provides unified semantics at an abstract and agnostic level. This agnostic meta-model called **Common Data Model (CDM)** is inspired from [9], [22], [40]. The main concepts of this metamodel are: an **Object** is an identified container of data variables and nested objects that are organized as a hierarchical tree; a **Data Variable** is a container of data that is characterized by a name, a data type and a value. It may include nested data variables and meta-variables; a **Meta Variable** is a container of data that provides additional meta-data information for a data variable; a **Value** concept is used to hold the value with an optional time stamp.

This layer achieves semantic interoperability by converting domain-specific models to this domain-independent model(unify) and vice versa (refine). For each domain-specific model, a plug-in contributes these functionalities.

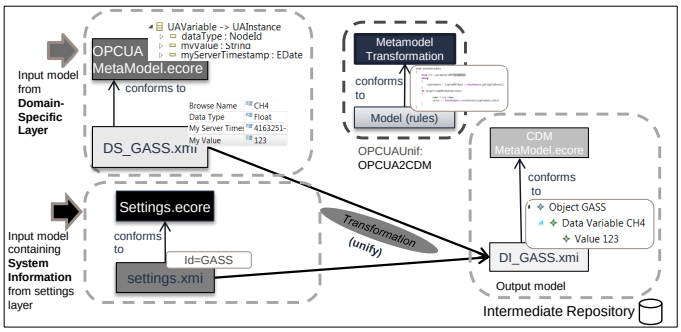

Fig. 5. Example of GASS Transformation for unifying the semantics

Here again, model-based technologies greatly ease the development. Our current transformations are mostly based on the ATL tool [41]. ATL uses simple rules that can be understood or written by domain experts. Figure 5 illustrates an example of transforming the DS_GASS model from the domain-Specific layer to a model conforming to the CDM metamodel. The Smart-hub settings are used here to assign the data of the corresponding system in the CDM.

3.2.4 System of Systems Layer

The **Central Repository** holds the unified data which also conforms to the CDM meta-model. Each system instance has a corresponding object in the repository. Two operations are supported in this layer:

- **Aggregate** "creates" or "updates" the repository with the system data from the domain-independent layer.This simple operation is not system-specific and thus implemented generically in the Smart-hub core.
- **SOSFetchAndProcess** "fetches" and "processes" the data from the Central Repository for further dispatching to a consumer system.

As previously discussed, data processing is a mandatory feature in our context. It involves the application of operational/business rules (arithmetic, structural, etc.) to the data when this is required by a specific consumer system. A plug-in contributes and implements these custom rules for each system that requires it. Again, model-based technologies such as ATL allow to facilitate the writing of these operational rules by domain experts. In our example, the aggregated data is processed through ATL rules and generates the model for the GIS system in the Intermediate Repository. This generated data then naturally follows the reverse process previously described: it is refined to the domain-specific layer and finally provided to the consumer system using its communication mechanism.

3.2.5 Settings Layer

We have have seen previously that the data go throw a number of levels for the data acquisition (DA) from DPs and the data generation (DG) for DCs. This layer manages the components to load for each smart-hub layer in order to establish interaction with the external systems.

In the smart gas environment, multiple systems coexist that share the same communication mechanism, data

format, and semantics. Therefore, the Smart-hub distinguishes two concepts: A **System-Type**, either a DataProducer (e.g. GasAnalyzerSystem) or a DataConsumer (e.g. GeolocalizationSystem), is a conceptual concept of a system sharing similar components. A **System-Instance**, either a DataProducerInstance (e.g.GASS, GASS1, GASS2, etc.) and DataConsumerInstance (e.g. GIS, GIS1, etc.), is the concrete system in the environment that uses the components identified by its system-type (GasAnalyzerSystem). These two concepts are distinguished via a meta model named "SettingRepository". Furthermore, a particular system-instance requires additional information such as authentication. These system-instance information are set via a metamodel extending the SettingRepository metamodel. This coupling of the two levels settings makes the Smart-hub modular due to its ability to add system instance information for specific environment without affecting the system-type repository.

4 Case Study

Figure 6 presents the application of the Smart-hub solution on the running example that was introduced in section 1.2. Firstly, a list of components is developed for the extension of the smart-hub layers. The components for the communication layer are: OPCUA, RestWebservice and FileSystem to interact with the OPCUA server(GAAS), the Rest-compliant systems (LISS) and the file-based systems (CMMS and GIS), respectively. The components for the Domain-Specific layer are: OPCUA for projecting OPCUA data to OPCUA model; LISSProjection for projecting the data provided by LISS system to LISS domain-specific model. ExcelProjection for projecting excel format to tabular model; CSVProjection for projecting CSV format to tabular model; The components for the Domain-Independent layer are: OPCUA to transform OPCUA model to the unified model; TabularTransformation to transform tabular data to the unified model; LISSUnification to transform LISS domain-specific model to the unified model. Likewise, a GISDP SOSFetchAndProcess component is developed to process the aggregate data in the central repository and generate the data required by the GIS system.

The data acquisition process (1-4) passes through a number of stages. Firstly the smart-hub acquires the data from the external system,then it unifies the syntax, after that it unifies the semantics and finally the data are aggregated in the central repository. The data generation process (5-9) is the reverse operation of the data acquisition where data are generated from the central repository in the unified semantics using the SOSFetchAndProcess operational rules. Then, the data are refined to the DC syntax and finally dispatched to the DC.

As seen in this simple example, the FileSystem connector was developed once but used twice. The same applies to the TabularTransformation, that was reused by more than one tabular based system. Now, systems compatible with the pre-existing components can be easily added to the current environment without any hard-coding as will be seen in the next section.

4.1 Extended running example

Now, the running example is extended by adding and modifying some systems. First, a number of DPs are added: another gas analyzer system (GASS1) that uses OPCUA; two pressure regulator stations (PRS) that monitors the gas pressure using OPCUA; a forecasting system using rest web service; Second, an OPCUA DC server is added that aggregates the data from all the DPs. Finally, the format of the CMMS system is changed from Excel to CSV.

Since GASS1, PRS56, and PRS78 are OPCUA-based system DPs, it is just required to reuse the pre-existing components. The rest web service component exists previously and can be re-used with the forecasting system at the communication level. However, since the Forecasting format and semantics does not exist previously, then it is required to create their domain-specific and domain-independent component. As indicated previously, the CMMS system changed its format from Excel to CSV format. Therefore, the Smart-hub is reconfigured to load the pre-existing CSV domain-specific component.

The OPCUA server DC (OPCUADC) must provide the full data in the central repository. Therefore, a SOSFetchAndProcess component is declared to fetch and process the central repository and make it available to the DC.

5 Evaluation

The Smart-hub has been tested by engineers on a number of real case studies in the smart gas network for the project Gontrand. The modularity of the architecture has led to focus and develop a number of loosely coupled components which tackle the different layers of interoperability. This has resulted in the development of a number of communication, domain-specific, domain-independent and operational/business rule components to test the functionality of the Smart-hub without imposing a particular standard. Then, the case studies in the project Gontrand have been subjected to several iterations for external systems changes to test the re-usability and extensibility feature, which was achieved successfully.

In order to make an evaluation of the proposal, the authors have created a comparison of the generic related works and the proposal based on some criteria as shown in Table 1. The first criterion verifies if the solution is generic in terms of application to a particular domain or if it is domain independent. The second criterion checks if the solution provides a deep separation of concerns to handle each level of interoperability separately and independently. The value of loosely coupled concerns is simplifying development, re-using functionalities and maintenance of each interoperability level separately. A solution such as OPCUA provides separate functionalities for the different level of interoperability, however, the layers are tightly coupled and dependent. The third criterion checks the solution capability to extend to new technologies and solutions at each level of interoperability, thus the ability of the solution to adapt to external systems changes (at any level of interoperability either communication, syntactic or semantic) without the need for complete replacement/update of an existing solution (e.g. changing the syntax of a system). Finally, the last criterion verifies whether a standard must be imposed for a particular solution or it can adapt to any standard (OPCUA, ISO15926).

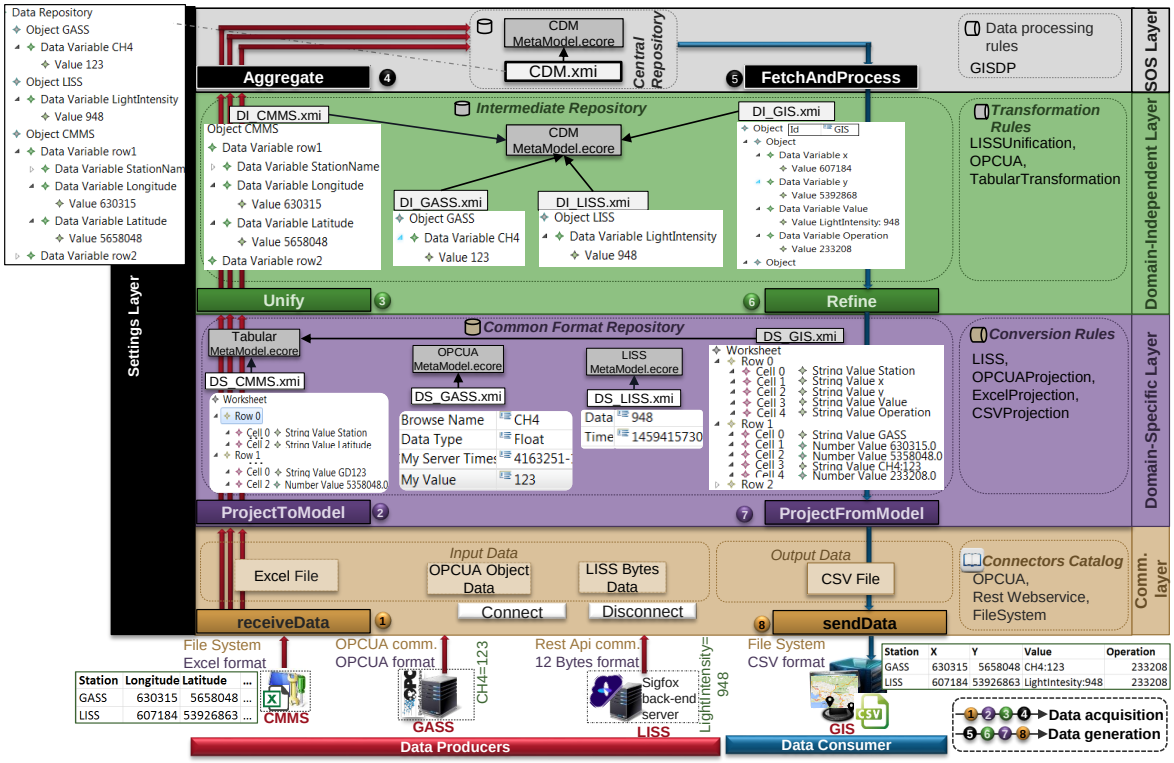

Fig. 6. The running example data exchange via the smart-hub

The smart-hub solution has been developed taking into consideration the requirement documents that states the previously defined criteria. One of the advantages of the Smart-hub is its support of reuse which has a great impact on reducing time-to-market, cost and improving quality [42]. The quantitative comparison of software reuse and other criteria such as performance, efficiency, etc. between the different solutions are reserved for the future work. The authors are aware that the smart-hub solution has some limitations. The smart-hub was able to integrate SOA solutions that acquire data from single service, however, it was not tested to integrate SOA solutions with multiple services. This is due to the fact that the smart-hub is data-oriented and not service oriented, i.e. the central pivot point for data exchange is the central data repository and not services. The smart-hub scalability to handle frequently changed and massive data has not been tested yet. However, the Smart-hub architecture provides a methodology that can be implemented using more efficient techniques.

TABLE 1
Evaluation with generic related work

		Generic	Modular	Extensible	Impose standard
Standards	ISA-95, MIMOSA, ISO15926	x	✓	x	✓
	OPCUA	✓	x	x	✓
Architectures and Middlewares	GWAC	x	x	x	x
	RAMI, IoT ARM, Hypercat	✓	x	x	x
Smart-hub		✓	✓	✓	x

6 Conclusion and Future Work

In this work, the authors have described a methodology which promotes the interoperability in order to achieve integration between systems in the smart gas distribution grid. This methodology has been based on multi-layer generic, modular and extensible architecture that relies on a deep separation of concerns and model-based software engineering techniques. The architecture has been developed and tested on a number of use cases from the Gontrand Project.

There are many perspectives for this work. First, ongoing work tackles the orchestration of data according to different interaction pattern. Second, we are investigating solutions to support built-in data storage and historization. Third, we plan to assess the scalability of the architecture with high frequency data exchanges. Fourth, we are building empirical data for creating a benchmark. Finally, we plan to test the architecture in similar environments such as Industry 4.0, smart grid, Industrial IoT, and all sorts of SoS where the integration and the collaboration between heterogeneous systems is required.

Acknowledgments

This work was carried out within the framework of a French national project (Gontrand) for the supervision of a smart gas grid. Support was given by Sogeti-HT Capgemini company, who funded this work. Finally, the authors would like to thank the gas distribution companies GRDF, REGAZ, and RéseauGDS for their collaboration.

References

[1] R. Buyya and A. V. Dastjerdi, *Internet of Things: Principles and Paradigms*. Morgan Kaufmann, 2016.

[2] R. Rajkumar *et al.*, "Cyber-physical systems: The next computing revolution," in *Design Automation Conference*, June 2010, pp. 731–736.

[3] M. Jamshidi, *Systems of systems engineering: principles and applications*. CRC press, 2008.

[4] CEN-CENELEC-ETSI and Smart Grid Coordination, "Smart grid reference architecture," Tech. Rep., nov 2012.

[5] K. Su *et al.*, "Smart city and the applications," in *International Conference on Electronics, Communications and Control (ICECC)*, Sept 2011, pp. 1028–1031.

[6] R. Drath and A. Horch, "Industrie 4.0: Hit or hype?" *IEEE Industrial Electronics Magazine*, vol. 8, no. 2, pp. 56–58, June 2014.

[7] W. He and L. D. Xu, "Integration of distributed enterprise applications: A survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 35–42, Feb 2014.

[8] P. Zhang, *Advanced Industrial Control Technology*. William Andrew, 2010.

[9] W. Mahnke *et al.*, *OPC Unified Architecture*. Springer Nature, 2009.

[10] A. Ahmed *et al.*, "Model-based interoperability solutions for the supervision of smart gas distribution networks," in *11th System of Systems Engineering Conference (SoSE)*, June 2016, pp. 1–5.

[11] G. Mhl *et al.*, *Distributed Event-Based Systems*. Springer, 2006.

[12] A. P. Sheth, *Changing Focus on Interoperability in Information Systems:From System, Syntax, Structure to Semantics*. Boston, MA: Springer US, 1999, pp. 5–29.

[13] A. Banks and R. Gupta, "Mqtt version 3.1.1," *OASIS standard*, 2014.

[14] C. Bormann *et al.*, "CoAP: An application protocol for billions of tiny internet nodes," *IEEE Internet Computing*, vol. 16, no. 2, pp. 62–67, mar 2012.

[15] L. Richardson and S. Ruby, *RESTful web services*. " O'Reilly Media, Inc.", 2008.

[16] M. Rys *et al.*, "XML metadata interchange," in *Encyclopedia of Database Systems*. Springer Nature, 2009, pp. 3597–3597.

[17] M. Uslar *et al.*, *The Common Information Model CIM: IEC 61968/61970 and 62325-A practical introduction to the CIM*. Springer Science & Business Media, 2012.

[18] T. Bittner *et al.*, "Ontology and semantic interoperability," *Large-scale 3D data integration: Challenges and Opportunities*, pp. 139–160, 2005.

[19] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, no. 2, pp. 199–220, Jun. 1993.

[20] H. K. Lin and J. A. Harding, "A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration," *Comput. Ind.*, vol. 58, no. 5, pp. 428–437, Jun. 2007.

[21] S. Rohjans *et al.*, "Uml-based modeling of opc ua address spaces for power systems," in *IEEE International Workshop on Inteligent Energy Systems (IWIES)*, Nov 2013, pp. 209–214.

[22] J. Robert *et al.*, "O-mi/o-df standards as interoperability enablers for industrial internet: A performance analysis," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, Oct 2016, pp. 4908–4915.

[23] K. Duddy and J. R. H. Steel, "Overview of the modelling of the physical world (motpw) workshop at models 2012," in *Proceedings of the Modelling of the Physical World Workshop*, ser. MOTPW '12. New York, NY, USA: ACM, 2012, pp. 1:1–1:2.

[24] J. Bézivin, "On the unification power of models," *Software & Systems Modeling*, vol. 4, no. 2, pp. 171–188, 2005.

[25] S. Efftinge and M. Völter, "oaw xtext: A framework for textual dsls," in *Workshop on Modeling Symposium at Eclipse Summit*, vol. 32, 2006, p. 118.

[26] K. Czarnecki and S. Helsen, "Feature-based survey of model transformation approaches," *IBM Syst. J.*, vol. 45, no. 3, pp. 621–645, Jul. 2006.

[27] J. P. Bourey *et al.*, "Report on Model Driven Interoperability," Apr. 2007, deliverable TG2.3.

[28] G. Tyson *et al.*, "A model-driven approach to interoperability and integration in systems of systems," in *Proc. of Workshop on Model-Based Software and Data Integration (MBSDI)*, 2011.

[29] S. Instrumentation and A. Society, *ANSI/ISA-95.00.01-2010 (IEC 62264-1 Mod) Enterprise-Control System Integration*, Std.

[30] The Gridwise and Architecture Council, *GridWise Interoperability Context-Setting Framework*, 2008.

[31] M. Hankel and B. Rexroth, "The reference architectural model industrie 4.0 (rami 4.0)," *ZVEI*, 2015.

[32] A. Bassi *et al.*, Eds., *Enabling Things to Talk*. Springer Nature, 2013.

[33] P. Varga *et al.*, "Making system of systems interoperable the core components of the arrowhead framework," *Journal of Network and Computer Applications*, vol. 81, no. C, pp. 85–95, Mar. 2017.

[34] J. Al-Jaroodi *et al.*, "Service oriented middleware: Trends and challenges," in *Seventh International Conference on Information Technology: New Generations*, April 2010, pp. 974–979.

[35] P. Spiess *et al.*, "SOA-based integration of the internet of things in enterprise services," in *2009 IEEE International Conference on Web Services*, July 2009, pp. 968–975.

[36] V. Veyber *et al.*, "Model-driven platform for oil and gas enterprise data integration," *International Journal of Computer Applications*, vol. 49, no. 5, 2012.

[37] ——, "Model driven approach for oil amp; gas information systems and applications integration," in *2010 6th Central and Eastern European Software Engineering Conference (CEE-SECR)*, Oct 2010, pp. 156–162.

[38] C. Greer *et al.*, "Nist framework and roadmap for smart grid interoperability standards, release 3.0," *Special Publication (NIST SP)-1108r3*, 2014.

[39] G. A. Lewis *et al.*, "Why standards are not enough to guarantee end-to-end interoperability," in *Seventh International Conference on Composition-Based Software Systems (ICCBSS)*, 2008.

[40] S. Nativi *et al.*, "Unidata's common data model mapping to the iso 19123 data model," *Earth Science Informatics*, vol. 1, no. 2, pp. 59–78, Sep 2008.

[41] F. Jouault *et al.*, "ATL: A model transformation tool," *Science of Computer Programming*, vol. 72, pp. 31–39, 2008.

[42] M. L. Griss, "Software reuse: From library to factory," *IBM Systems Journal*, vol. 32, no. 4, pp. 548–566, 1993.

Ahmed AHMED is a PhD student at Arts et Métier ParisTech and a research engineer with the company Sogeti-HT Capgemini, France. He obtained his master degree in complex information systems from the university of Western Brittany, Brest, France in 2014. His current research focuses on architectures, data modeling and interoperability using model driven engineering techniques.

Mathias Kleiner is an associate professor of computer science in Information and System Science Laboratory at Arts et Métier ParisTech, France. He obtained his PhD from University of Mditerrane, Marseille, France in 2007. His research interests include artificial intelligence, graphs, constraint programming and metamodeling, and the application of these technics to mechatronic products design.

Lionel Roucoules is currently professor at Arts et Métier ParisTech, France (www.ensam.eu). He develops his research in the Information and System Science Laboratory. The context of his research is integrated design and collaborative IT platform in a global PLM vision. His specific interest is product-process interface.