Monitoring and controlling energy distribution: Implementation of a distribution management system based on Common Information Model

CrossMark

L. Fiaschetti [a,c,*], M. Antunez [b,c], E. Trapani [c], L. Valenzuela [b,c], A. Rubiales [a,c], M. Risso [b,c], G. Boroni [b,c]

[a] Comisión de Investigaciones Científicas de la Provincia de Buenos Aires (CIC-PBA), Buenos Aires, Argentina
[b] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
[c] Pladema Institute, Gral. Pinto 339, 7000 Tandil, Argentina

ABSTRACT

The management of renewable and distributed energy resources is changing the way in which electric distribution systems are being operated. Many companies commonly create several custom system solutions to manage power distribution. The functions provided by these systems are network visualization, state estimation, system control and data acquisition, among others. Despite their relevance, many midsize or small distribution companies have problems to install one of these solutions because they are either very expensive, incompatible, or limited by difficulties in data exchange. As a possible solution to this problem, this paper presents the development of a Distribution Management System (DMS) based on open source technology. This system combines four main components: OpenDSS-framework, ActiveMQ-broker, applications for visualizing and editing electrical network, and Common Information Model (CIM). The tool is part of Tenergia R&D project [1] and has a collection of applications designed to monitor and control the distribution networks located in Argentina.

© 2017 Elsevier Ltd. All rights reserved.

1. Introduction

There is a significant interest for the electrical industry to develop systems supported by computer-aided engineering [1]. In order to attend decision making and maintenance activities, these systems should provide several functions: Network Connectivity Analysis (NCA), Distribution System State Estimation (DSSE), Load Flow Applications (LFA), Volt/Var Control (VVC), Fault Management & System Restoration (FMSR), Distribution Load Forecasting (DLF), among others [2–4].

A Distribution Management System (DMS) is a set of software components which monitor and control the network with the aim of optimizing the behavior of an electrical smart grid, and therefore enabling utility companies to provide power in a more reliable, efficient and safer way [5–7]. Moreover, it improves outage reduction, outage time minimization, maintenance of acceptable voltage levels and line loss reduction [8]. Many research and

commercial tools such as DIgSILENT PowerFactory [9], ETAP [10], Schneider [11] are available.

Several works [12,13] present the benefits of installing a DMS, both for the end users and the distribution company. The process of instantiating a DMS requires specific software, communication infrastructure and hardware for the automation of different network components. In [14] it is shown that the software represents 20% of the total value for automating the network.

The implementation of such systems can be achieved developing solutions that could use existing open source components, and incorporating monitoring and communication network functions. In addition, the cost factor is interesting for the proposal. According to reference, for an investment of 28 million dollars in the first 3 years, the capital was recovered after 5 years, and the profit reached 51% of the invested capital after 10 years [14]. Although the values are interesting for any company, only a few distributors can afford such an investment. Considering that open source components allow cost reduction, the architecture is adapted to the needs of the region.

Moreover, this kind of systems have to be supplemented with appropriate message passing and synchronization capabilities in a prescribed format. For that reason, standards are being used to facilitate the data exchange between applications within a company, as well as with other companies. These two standards are the IEC 61970–301 and the IEC 61968–11, which correspond to

the Common Information Model (CIM) and Distribution extensions [15]. Although CIM is primarily designed for node-breaker representation, with certain extensions it can be leveraged to achieve a standard interoperable bus-branch model representation and exchange format. The power system network can be modeled using CIM with as many abstraction levels as required. In [16] three models (physical, topological, and dynamic) to represent power network information on the developed DMS are described.

If more accessible analysis tools are needed, flexible and reliable open source software packages should be used [17]. In this context, OpenDSS is a comprehensive open source tool for electric power distribution systems developed by the Electric Power Research Institute (EPRI) [18] to model and simulate the electrical behavior of the distribution network. Another characteristic of OpenDSS is that it includes a COM interface to communicate with other applications, providing flexibility and modifiability [19,20].

Furthermore, being able to combine applications is important if a robust messaging scheme is required. Most messaging applications are implemented as a message broker which connects producers with consumers, where producers send messages and consumers process them [21]. There are many message systems that can be used to communicate information within different parts of the systems, such as RabbitMQ, ActiveMQ, OpenMQ [22]. The ActiveMQ broker is chosen because it is faster on message reception. This is an essential requirement in a system where several devices are constantly sending information [22].

The electrical companies have installed substantial monitoring equipment that regularly sends relevant information of the electrical network. In order to save this information, a variety of NoSQL and SQL databases [23] are studied. In this sense, the NoSQL databases show more benefits when the analysis of large volumes of data [24] is required, which is very important in the distribution sector. Thus, Apache Cassandra [25] is selected due to present a good performance in all the tests. Moreover, this is an open source database, which is one of the requirements defined by this development.

An EMS architecture is shown in [26]. Although it is useful in our proposal, it adopts only a static CIM to represent and update information. This feature degrades real-time data management performance. Besides, data acquisition and device control are not represented. Additionally, in [27] it is shown that software OpenDSS provides many useful algorithms for the distribution sector. However, it cannot support real-time execution.

Finally, current systems are very expensive and lack features such as oriented services, modularity, common standards and flexibility. Another drawback worth mentioning is the dependence on providers. Thus, the main purpose of this work is to design and implement a DMS architecture based on open source technology which provides the requirements mentioned above in contrast with existing monolithic structures. Moreover, the combination of OpenDSS, ActiveMQ, CIM and NoSQL databases on the proposed architecture allows the incorporation of new algorithms and tools dynamically in contrast with the closed and private existing systems.

The following section shows the motivation for using the mentioned applications and the requirements for their interconnection. The proposed architecture is demonstrated in Section 3 and described in Section 4 considering the most important implementation aspects. Section 5 depicts several tests which demonstrate the feasibility of using the system over a real distribution network. Finally, conclusions are presented in Section 7.

2. Motivation and requirements

The main purpose of this article is to contribute with solutions applied to the distribution grid. Therefore the design and imple-

mentation of the DMS is presented, with characteristics such as flexibility, extensibility and evolution, opposed to the monolithic architectures currently on the market. Instead of developing an independent solution, technologies that are well tested are combined and effectively used in the industry. Specifically, this DMS is developed by connecting several well-known technologies with a collection of applications designed to monitor and control distribution networks located in Argentina.

The analysis tools need to have a clear representation to synchronize real-world and simulation behavior. OpenDSS is a flexible and reliable open source tool to model and simulate the electrical behavior of the distribution network.

One important requirement of the current tools is the integration of power flow modeling and simulation environments. A mechanism to separate the application logic from the simulation execution is another important requirement. In order to combine different applications, it is essential to use a standard model with several levels of abstraction so as to export and import a variety of tools. The chosen data model should enable the generation of specific input models with a coherent transformation.

The main motivations for using CIM to model the network are the availability of tools and libraries to support this format and the high degree of compatibility with other industry standards. These facts facilitate the application integration of various distributed systems. Thus, CIM standards is used to facilitate the exchange of data between applications within a company and the exchange of data between different companies.

In turn, these developments are encouraged by research-funding entities to research on strategic issues in science, technology and innovation policy. Thus, they support new ideas and strategies to develop products and services. In this regard, there is an opportunity to implement a DMS that provides the specific functionality to the companies taking into account their local needs. The developed tool is being applied in Tenergia R&D project[1]. It is intended that the resulting tool will be generically applied to regional studies in certain selected regions of Argentina and, consequently, be available to a larger community.

3. Implementation

As mentioned above, most DMS are based on monolithic architectures. At present, the new challenges in the distribution sector need new integrated tools to help and improve the operation of the electrical system. Thus, it is necessary to incorporate quality attributes such as flexibility, adaptability and interoperability without losing performance and security. An important decision is to use ActiveMQ as a message broker to communicate information between different parts of the system. The broker pattern allows to decouple the operation of the subsystems, and it manages communication between the modules. In addition, to make a DMS safer, security tactics are implemented based on sessions and encryption. Besides, this architectural pattern allows the implementation of several availability mechanisms which make the system more fault tolerant. Moreover, new technologies in data bases are used to save all the obtained information. Mainly, the NoSQL databases are optimized to use data mining and machine learning algorithms which are very useful for the proposed system.

In this section, an analysis of open source components and information standards integrated into this project is presented.

3.1. Applications for visualizing and editing electrical networks

The proposed system has an operation diagram representation (schematic and geographic) which allows the user to create, modify and operate the power network. To implement the schematic

interface, several open source editing tools such as Jgraph [28], CircuitSimulator [29], JHotDraw [30], JDiagram [31] are studied. A variety of features related to customizable edition (component insertion, drag and drop, stencils, copy and paste, etc.) and license are analyzed to select the most adaptive tool.

As a result, Jgraph, which is developed in Java through the Swing and AWT libraries, is the selected framework. In fact, there are articles where these libraries are proposed for similar developments [32]. It has a license BSD 3-Clause and provides functionality for visualization and interaction through nodes, including features such as templates XML, which support the creation of new figures. This is useful for creating new electrical elements which correspond to different standards. Several visual features can be developed with this framework, such as mechanisms of connection between components under the CIM model.

Open source tools are assessed to implement the geographic representation or GIS (Geographic Information System). These tools must allow electrical network visualization and edition, object tracking (electric crews) and alarm visualization, among others. The tools under evaluation are OpenLayers [33], GeoTools [34], OpenGeoSuite [35], Udig [33], gvSIG [33] and Open JUMP [33].

The assessment considers the requirements mentioned above, as well as quality attributes such as flexibility, adaptability, efficiency, usability, maintainability and portability. Therefore, OpenLayers library is selected as the best option. This tool is developed in JavaScript and allows web or embedded Java development.

Finally, to complete the GIS development, it is necessary to choose a free access map. Consequently, Open-StreetMap [36] is selected. It provides free and editable maps of the entire world, maps which are built by volunteers with an open content license (Fig. 1).

3.2. OpenDSS-framework

OpenDSS is an open source software package developed by EPRI [27,37], which can be used to simulate distribution networks. It contains detailed models of several network components, for example lines, loads, generators, monitors, etc. The program basically supports analysis commonly performed for utility distribution systems including distributed generation analysis. An important property of OpenDSS is that it is very expandable and, thus, can be easily modified [27]. OpenDSS is used in Distribution Planning and Analysis, Analysis of Distributed Generation Interconnections, Harmonics and Interharmonics Analysis and Neutral-to-earth Voltage Simulations, among others [38].

The connection between OpenDSS and the DMS can be seen in Fig. 2. OpenDSS counts with a COM interface to interchange data with other applications; this interface has a set of methods and properties which can be used through many programming languages. There are several frameworks which can implement these connections. In this case, a variety of libraries for JAVA languages are tested, and JACOB Project is chosen due to the flexibility and reliability of its results. This library is an open-source project which uses JNI (Java Native Interface) to make native calls into the COM and Win32 libraries [39].

Communication between the proposed DMS and OpenDSS is implemented through the Builder and Adapter design patterns [40]. First, electrical network modeled into the CIM is accessed to build the OpenDSS script. Then, the JACOB library is used to compile the script and get the results of different algorithms provided by OpenDSS. Finally, the obtained results are adapted to an internal format to be processed and displayed.

In Section 5, load flow analysis is performed for the IEEE nodes test feeder [41] and compared with the available results to verify the correctness of the circuit modeled in the proposed DMS.

3.3. ActiveMQ-broker

Communication between components which can be installed in different devices, is provided by ActiveMQ software. ActiveMQ is an open source message broker with a full Java Message Service (JMS) client. It supplies features such as communication with more than one client or server, security, computer clustering, virtual memory, cache and the possibility of having access to any database as a persistence provider. In addition, it employs message queues as a persistent communication mechanism among processing
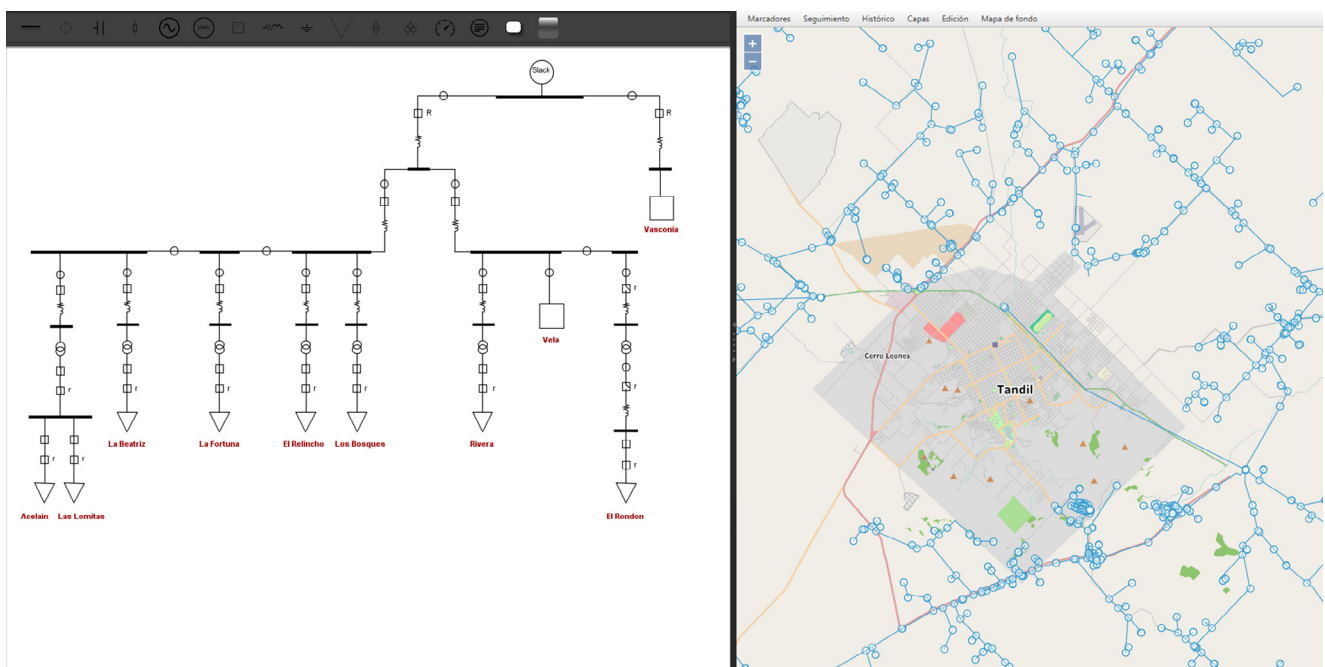

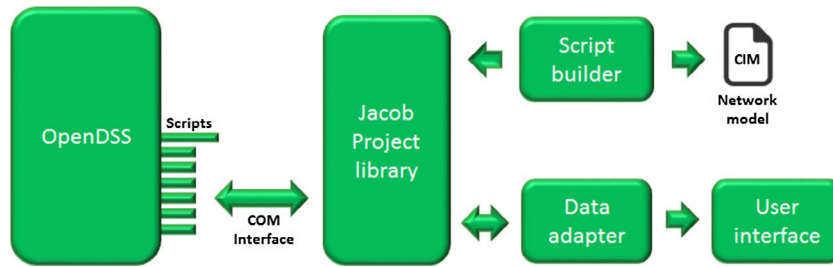**Fig. 1.** Schematic and geographic representation.

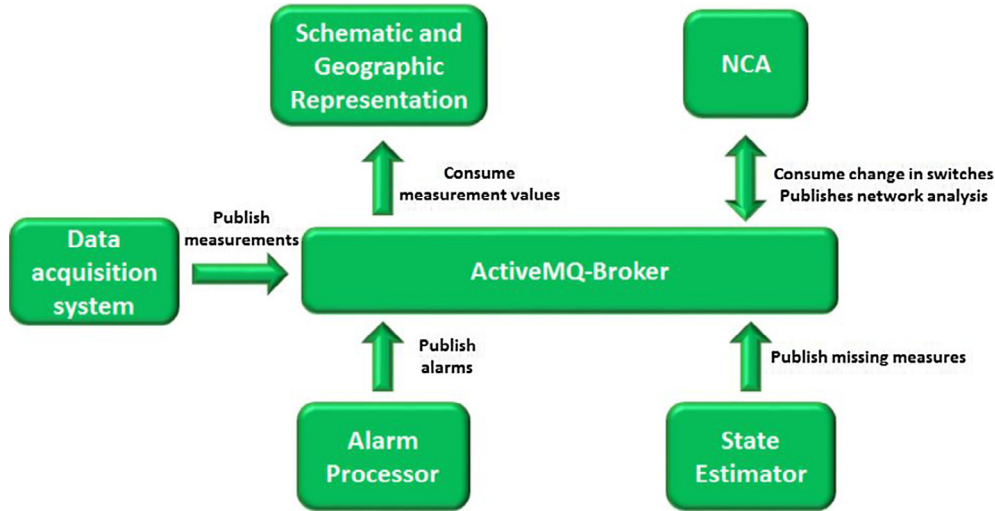**Fig. 2.** Communication between DMS and openDSS.

Fig. 3. Interaction of the applications with the activeMQ-Broker.

nodes. Message queuing facilitates data sharing and provides a publish/subscribe system which can be applied to the components in order to communicate with other components [42]. Finally, ActiveMQ supplies a fail over protocol so as to maintain high availability [43].

Functionality is based on topic management. Several topics are created, and every module of the system is subscribed to them. The topics are associated with different types of events generated in the distribution networks. Each event contains relevant information about some parts of the system. Hence, each part only needs to be subscribed to the topics that contain the necessary information. Therefore, a modular system is implemented where each module can work independently from the others.

As shown in Fig. 3, the system is divided into different applications.Each application communicates with the others through topics created in ActiveMQ-Broker. The applications not only consume the information of the topics, but also produce new information, which is sent to other topics. The applications that interact with ActiveMQ-Broker are Alarm Processor, State Estimator, Data Acquisition System (DAS), Schematic and Geographic Representation and Network Connectivity Analysis (NCA). For example, the modules responsible for visualizing will be subscribed to the new-measurements topic. When the Data Acquisition System publishes a new measurement on this topic, the ActiveMQ sends the new information, and the corresponding modules are informed in real time about the changes in the measurement values.

The authentication system supplied by ActiveMQ is used in the project [44]. Thus, a user name and password have to be employed for each application (producer and consumer) in order to log into the topic. Moreover, Apache Camel is used to encrypt all the messages [45]. The PGP Data Formatter [46] is implemented to encrypt every message the applications send to the topics. The PGP Data Formatter has two keys, a public and a private one. The public key is necessary for the producer applications to encrypt the messages. On the other hand, the private one allows the consumer applications to decrypt the messages.

3.4. CIM

CIM is developed by EPRI as a unique information model for electric power management systems to obtain interoperability between internal modules in the system and export data to external tools. This model is defined under the series of IEC 61970 and IEC 61968 norms corresponding to data transmission and distribution, respectively. In addition, it is described by a UML (Unified Modeling Language) class diagram [47].

As it was mentioned in [16] the power system network can be modeled using CIM at different levels of abstraction presenting the respective level of detail. Thus, a modular division between static model and dynamic model is implemented to address the problem of real-time model exchange. The former enables the representation of the network topology and the connections among them. The latter defines the state of the network measurements. Each application will use each model type when required. Fig. 4 shows the main classes of these models. Consequently, the static model is the input of all algorithms presented into the DMS proposed and the dynamic model provides measurement values in real-time.

The exchange of information in CIM is implemented through RDF (Resource Description Framework) models [48] stored in a standard format, for example, RDF/XML (extensible Markup Language).
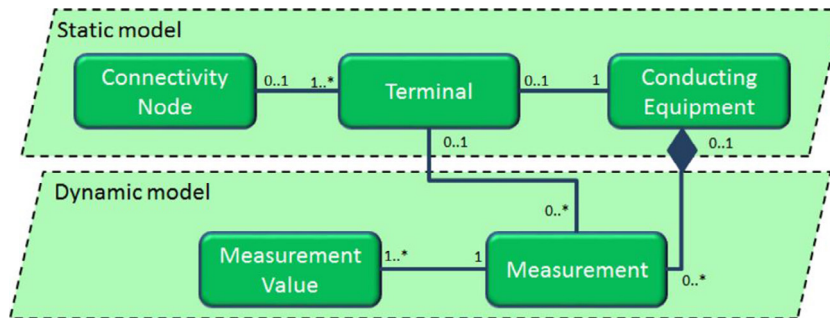
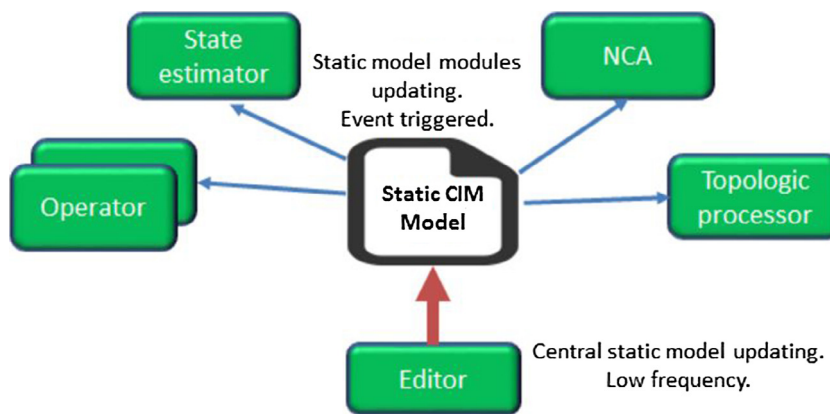**Fig. 4.** Components defined in the static and dynamic models.

Fig. 5. Updating process of the static model.

The type of the module determines the model that need to be updated. The static models are updated by the network editors with a low frequency. This must be reflected in all the modules as a trigger engine (Fig. 5).

The dynamic model is updated by the broker and the DAS publishes measurement values with a high frequency. The modules subscribed to the corresponding topic updates the dynamic model automatically with the arrival of a measure (Fig. 6).

At present, CIM cannot model all the information that needs to be used and exchanged by the electrical sector. Namely, in the proposed modular division, the network is formed by the interconnection of physical elements, which can be stored in RDF format files. However, there are cases where these format files do not characterize the physical elements. For such cases, an extension of CIM is necessary in order to support some parameters, for example, resistance and reactance matrix of lines, reactance percent between windings in transformers.

4. Architecture

An ADD methodology (Attribute Driven Design) is employed to build the DMS architecture 7. The main quality attributes required to build the architecture are usability, extensibility, flexibility, interoperability and security. To meet these attributes, an Event Driven Architecture (EDA) which is based on a paradigm publish-subscribe was defined. EDA allows monitoring, detect and control different events in real time through several components. These components are uncoupled and can be modified isolately. This property improves the flexibility and modifiability of the system [26].

EDA is complemented with the Service Oriented Architecture (SOA), which allows every application to be freely connected to the platforms. In the same way, these features improve the modularity, flexibility and independence between the supplier and the platform.

The DMS architecture is divided into four main modules: DAS, applications, middleware services and persistence. DAS is the first communication point with several information sources such as measuring equipment and SCADA systems. All the received information is published in the middleware service ActiveMQ, which sends the captured information to the other parts of the system. This historical information is stored in a database to be employed by several algorithms, as state estimation and demand forecasting, among others. The process of storing the information is executed by the persistence module, and the DAO pattern (Data Access Object) is used to implement it [49]. Thus, the design of the database can be altered without modifying the code implementation. The applications module contains independent applications that provide the functionality of the DMS: Schematic and Geographic representation, State estimator, NCA and Alarm Processor.

The values on the screen are updated using a change notification mechanism based on the MVC [40] pattern, which enables the view and the controller components to be notified by the model. These components are registered in the model, providing an immediate update. This feature allows an updated process that do not violate the independence of the model.

The user applications could run on dedicated machines or external servers. This facilitates modifiability and the use of components that can be developed and updated independently. For example, to add the OMS (Outage Management System), only communication with the middleware service has to be defined. This communication is performed by the topics and the OMS will be subscribed to an existing topic and will send the new information. Thus, these changes do not have an impact on the other applications and system extensibility is maintained.
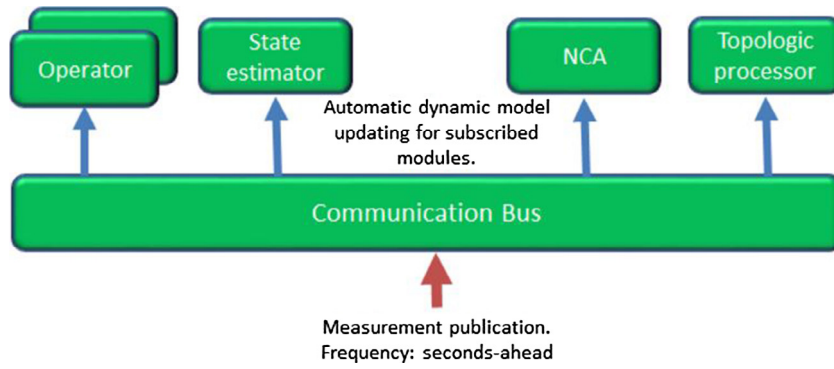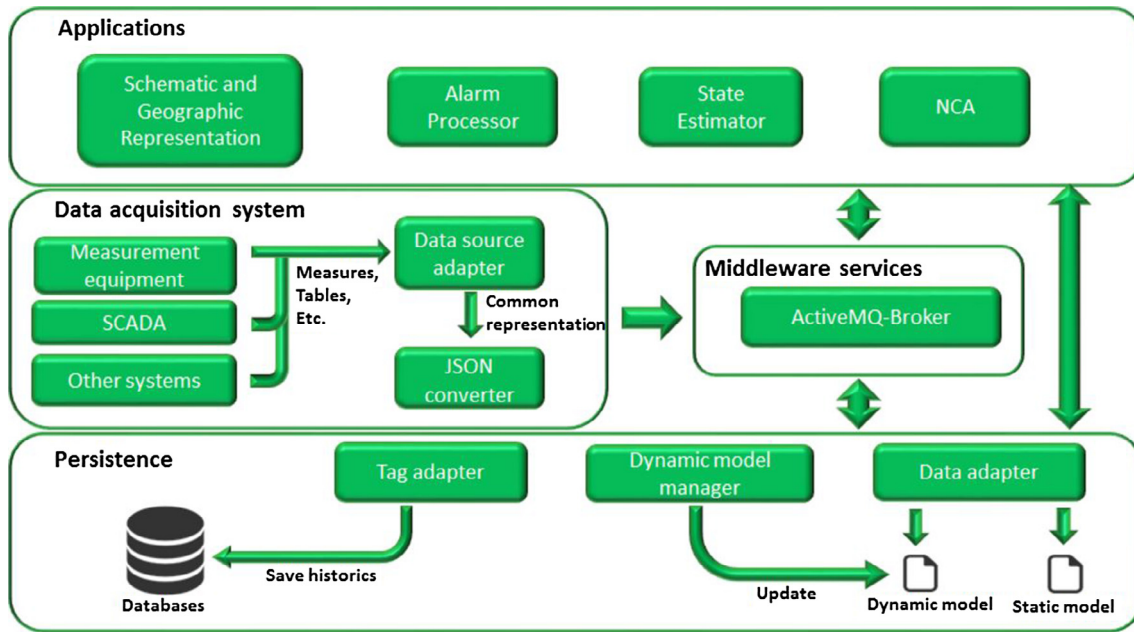
Fig. 6. Updating process of the dynamic model.
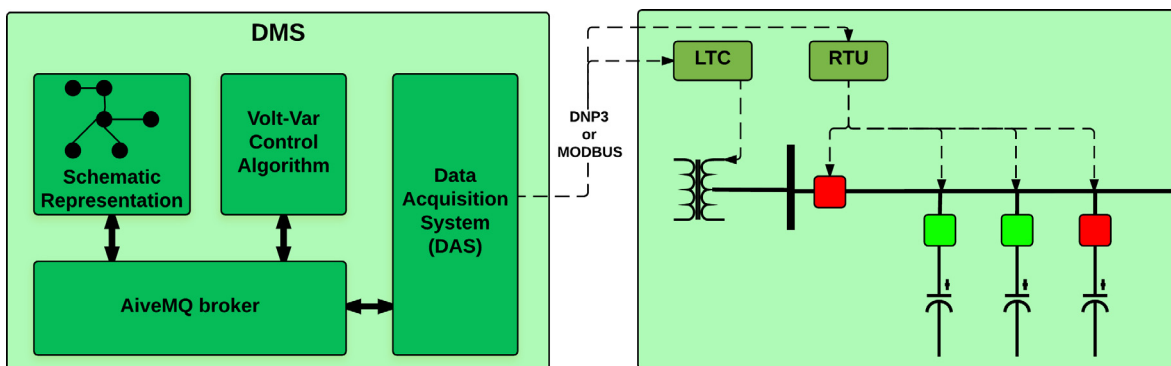

Fig. 7. DMS architecture.

Fig. 8. Monitoring and controlling system components.

This architecture is instantiated and tested for many re-closers provided by a local distributor. In this case, DNP3 protocol and MODBUS are used to operate the equipments. In the first, different libraries to implement this protocol are analyzed. The selected library is OpenDNP3[50] because it presents more benefits than the others. This implements all the DNP3 layers. It allows the system to retrieve and decode packages from re-closers, and also to write registers into them. Moreover, it is necessary to adapt the information provided from the devices to another format understandable by the DMS. Therefore, extra functionality is added to the application layer to convert the values and commands to a JSON format as intermediary between the
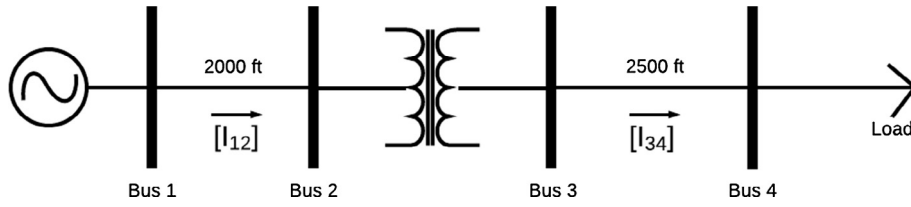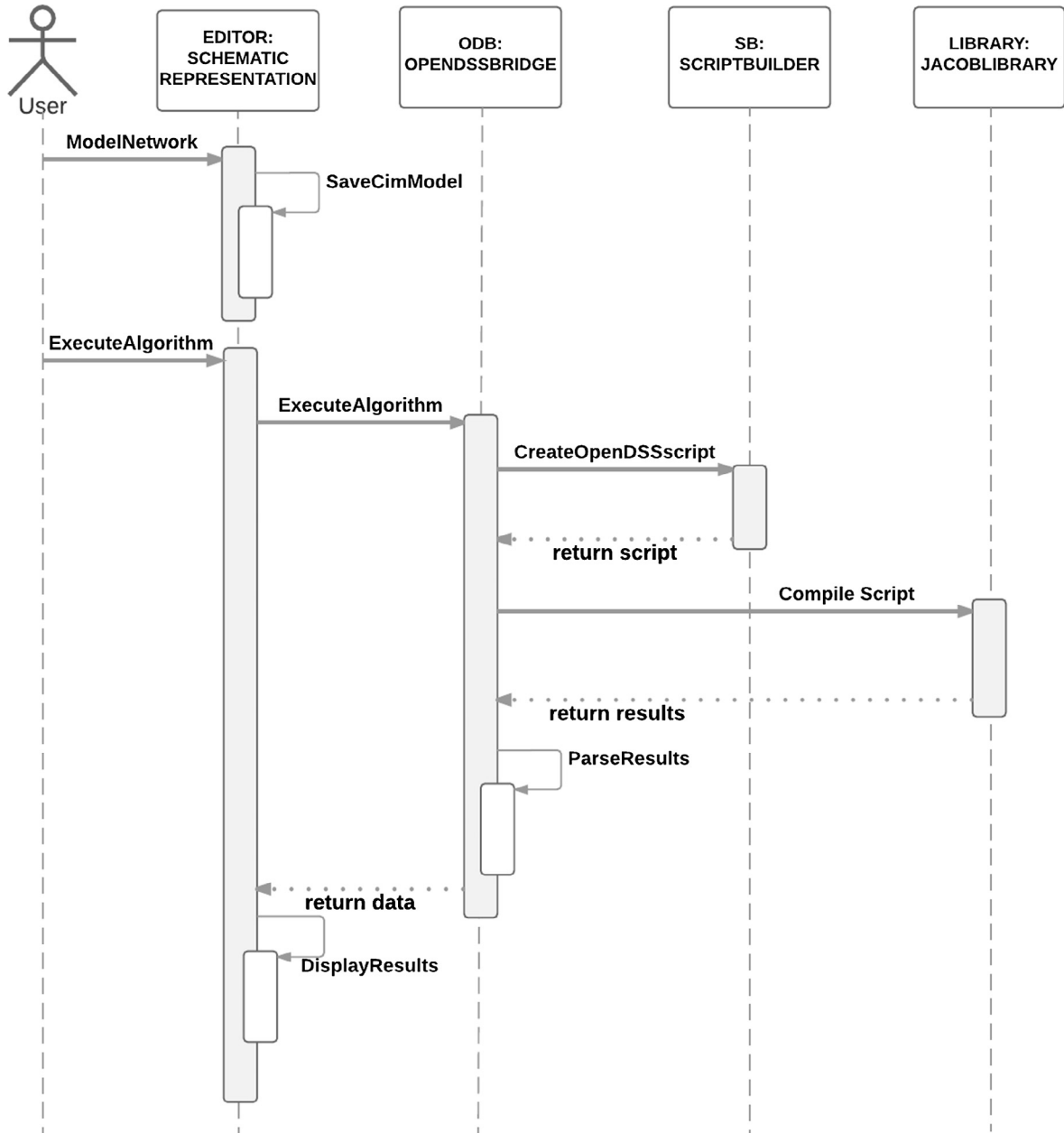
Fig. 9. IEEE 4 nodes test feeder.

Fig. 10. Sequence diagram: power flow calculation.

devices and the DMS. Finish, libmodbus [51] is adopted to implement Modbus protocol. This library is a free software tool to send/receive data according to the Modbus protocol which is written in C, and supports RTU (serial) and TCP (Ethernet) communication.

Under the proposed schema, every physical register of the equipment is identified by a unique code called Tag. The operation selected by the user is translated to a set of tags which is sent through an ActiveMQ topic. Then, the DAS module subscripted to the same topic, receives that set of tags and converts it into a valid frame (depending on the protocol), resulting in a physical operation into the device.

Nowadays, the volt-var control algorithm is being developed. This algorithm is executed from Python with the OpenDSS library,

Table 1
Refresh rate results.

| | Visual interface | | |
# Measurements	Max	Min	Average
10	0.003	0.001	0.002
100	0.009	0.002	0.006
1000	0.0018	0.015	0.0016
10000	0.0057	0.037	0.042

and returns the optimal device configuration, which can be sent by an ActiveMQ topic to change the turn ratio for LTC transformers, control the reactive of the photo-voltaic system or change the state of the capacitor bank 8.

5. Test cases

The new tools have to be robust and integrate seamlessly with other software systems. In this sense, several Test Feeders [41], which are central to verify new distribution system software, have been published. Test feeders will also be advancement drivers in distribution system analysis technology, being designed to stretch incrementally the capabilities of the tools.

Considering that this paper mostly refers to a new DMS, the user's interface will have to support new smart grid components and behaviors. Middleware services and databases usually contain software for transaction and persistence management, remote communication primitives and threading policies, which have a great impact on the aspects mentioned above.

The following section shows three test cases linked to OpenDSS interface, ActiveMQ broker, database operations and visualization rate.

5.1. OpenDSS interface

To test the OpenDSS interface, it is proposed to execute a power flow computation following the scheme explained in 3.2. For this purpose, the IEEE 4 nodes test feeder is selected (Fig. 9). The sequence diagram for this case can be visualized in Fig. 10. This diagram shows the operation flow, starting from the creation of the network on the schematic representation (network modeling

with the electrical parameter of each component). Next, the Script-Builder module is responsible for building the OpenDSS script and making communication through the COM interface. Finally, the power flow is executed and the results are shown on the schematic and geographic representation of the proposed DMS. As a result, the values obtained for voltages, currents, and power flows are equal to the ones reported by OpenDSS. In addition, the performed tests demonstrate that the times of script generation are insignificant compared to the communication times through the COM interface.

### 5.2. Schematic representation update

Firstly, the refresh rate is analyzed on the schematic network representation module. This is essential for choosing the JGraph library. In this test, different scenarios with 10, 100, 1000 and 10000 tags are modeled. It must be mentioned that each scenario is updated every 1 s (Table 1). After analyzing each case, it can be concluded that the refresh rate increases linearly with the number of tags on the screen. That is to say, if the number of tags increased by 10, the total time grows 2.6 ms (Fig. 11). This is the case of large scale distributors like Edenor [52] and Edesur [53].

### 5.3. The performance of middleware services

This section describes a performance test executed on ActiveMQ broker. Several data sources are simulated while they are concurrently publishing information of 10 tags. In all the cases 500,000 tags are sent and the throughput is analyzed in each case. Fig. 12 shows that the throughput increases with the amount of data sources able to be processed, and an average of 30,000 tags per second are obtained. However, Table 2 shows that the percent of errors grows with a higher amount of data sources. This happens due to the overload of ActiveMQ buffers and the overhead of communication between consumers and producers [54].

5.4. The performance of databases

Finally, a load test on the persistence module, which allows to evaluate the reading and writing latency, is performed. To measure reading latency, the average time for executing a query is calculated. That query selects a total of 100, 10,000 and 40,000 measurement values. The results can be seen in Table 3, where the response
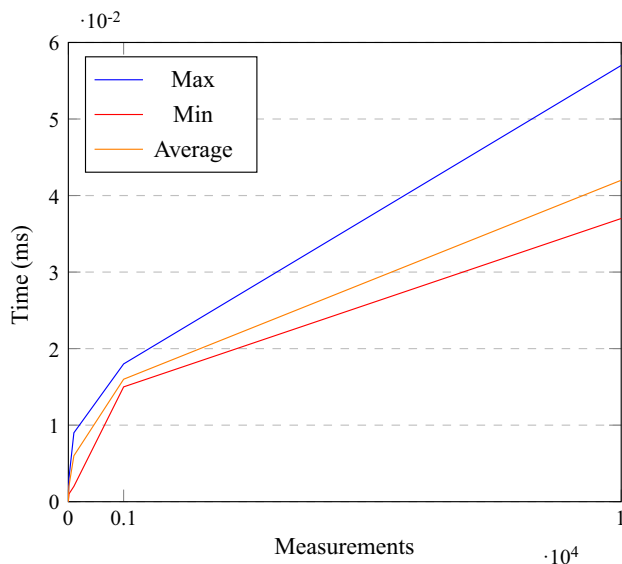
Fig. 11. Comparison of the refresh rate results.
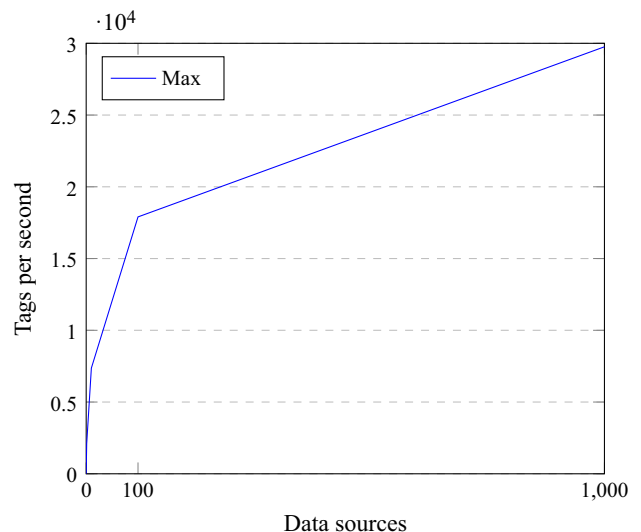

Fig. 12. ActiveMQ throughput.

Table 2
Results of the throughput test.

	Middleware service (ActiveMQ)					
# Data sources	# Samples	Average (ms)	Min (ms)	Max (ms)	Error (%)	Throughput (per sec)
1	500,000	2	1	245	0	2105
10	500,000	5	1	279	0	7367
100	500,000	20	1	470	0	17902
1000	500,000	50	1	1017	1.14	29757

Table 3
Results of the read latency test.

|	Database (Cassandra)			
# Operations	Average (ms)	Min (ms)	Max (ms)	Error (%)
100	60.32	27.00	1196.00	0.00
1000	51.75	34.00	1929.00	0.00
40000	528.98	593.00	4715.00	0.00
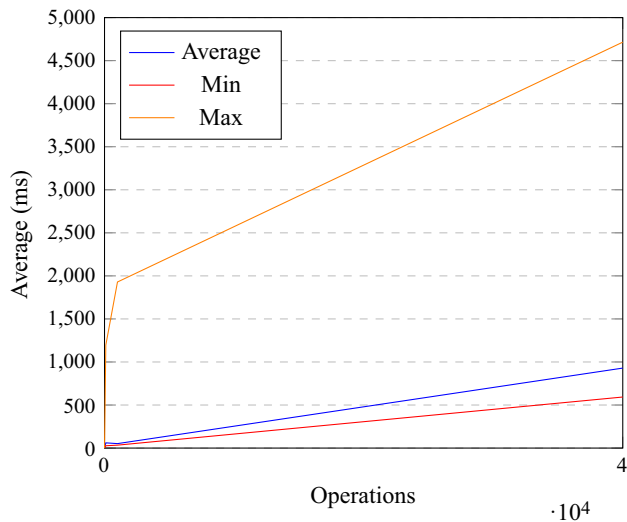
Fig. 13. Comparison of the read latency results.

Table 4
Results of write latency test.

	Database (Cassandra)			
# Operations	Average (ms)	Min (ms)	Max (ms)	Error (%)
100	44.30	25.00	843.00	0.00
1000	28.74	20.00	775.00	0.00
10000	24.30	17.00	2194.00	0.00

times oscillate between 60.32 and 528.98 ms. Fig. 13 shows that the response time increases when the number of consulted data grows.

The average insertion time of a measurement value is analyzed to evaluate the writing latency of the database. For this reason, several numbers of writing operations (100, 1000, 10,000) are evaluated. Table 4 shows the results of each test. Fig. 14 depicts that the writing time decreases with the increase of writing operations. This happens because Cassandra maintains in memory the result of the last operations. Therefore, it is not necessary to bring the information from the hard disk every time a query is executed.
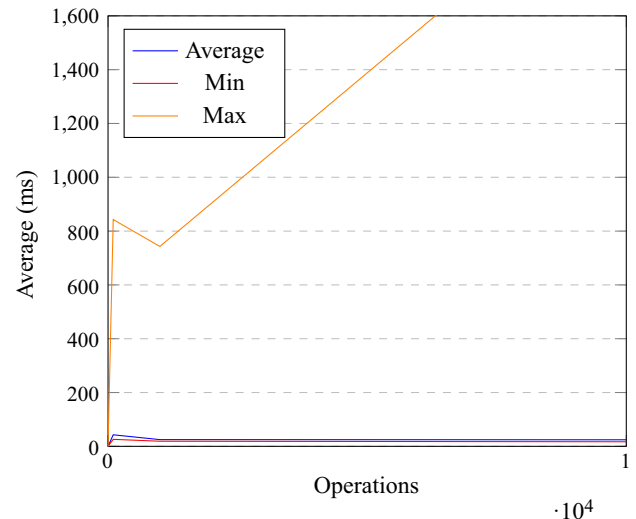
Fig. 14. Comparison of the write latency results.

6. Discussion

The proposed DMS can edit and operate an electrical network and perform several data analyses on a dedicated database. Moreover, the adopted architecture allows the incorporation of additional modules to bring solution to other required functionalities. Hence, this implementation is more suitable to be adapted to new requirements, in contrast with existing monolithic structures. Also, it can be integrated with other on-line systems to help decision makers during the system operation. In fact, the authors are currently working on adding other features to the system such as state estimation, volt/var control and fault management, all of them integrated in the implemented DMS. Furthermore, a new tool for visualizing results in GIS environment is being developed.

It worth noting that security challenges were properly addressed to avoid risk situations that could endanger the integrity of the system's data. As seen in Section 3, large amounts of information are transmitted through the modules in order to be processed for each application. As a consequence, it was necessary to implement tactics to avoid any external agent to publish information into a broker and send it to the modules. We followed a security approach based on session and encryption, which only allows to authorized modules and external systems to publish information.

Finally, it is worth mentioning that this tool will be used by three distribution companies in Argentina in the short term.

7. Conclusion

Nowadays, there are several challenges related to distribution networks. The reliably operation of an electrical network is an important concern for the power system operators. The applica-

tions with on-line monitoring must be able to recognize the dangerous situations that can be leading to network collapse. Thus, it is necessary to have a system that supports decision-making which allows the analysis of the state of the power network.

Moreover, the integration of renewable energy is taking an important role in industry. Thus, it is necessary to have a system that supports decision-making which allows the analysis the state of the power network. This paper presents a DMS development based on open source tools. Under this scheme, several subsystems can be easily added. In this way, it is possible to adapt the system to the new challenges that involve renewable energies. This new paradigm allows the incorporation of tools that help operation.

It worth mentioning that important characteristics such as interoperability, event orientation and flexibility were taken into account during the development, and were implemented making use of state of the art architectural patterns such as SOA, EDA, publish/subscribe and broker. In addition, the tool was integrated with NoSql storage in order to obtain data to analyze several aspects of the power network. The design combines: OpenDSS, which provides several electrical algorithms; Jgraph, as a graphical library; ActiveMQ, as a broker; Cassandra, as a storage mechanism; and CIM as an intermediate model. Using this existing open source software, it was possible to build a platform which provides functions that allow enhanced real-time operation and simulation of a changing electric distribution system.

Finally, more Smart Grid/DMS applications can be implemented over the framework to facilitate development tasks.

Acknowledgments

References

[1] Taylor T, Ohrn M. Network management for smart grids: innovative operations centers to manage future distribution networks. Revista ABB 2009;3:45–9.
[2] Roytelman I, Landenberger V. Real-time distribution system analysis-integral part of DMS. In: Power Systems Conference and Exposition, 2009. PSCE'09. IEEE/PES. IEEE; 2009. p. 1–6.
[3] Rezkalla M, Heussen K, Marinelli M, Hu J, Bindner HW. Identification of requirements for distribution management systems in the smart grid context. In: Power Engineering Conference (UPEC), 2015 50th International Universities. IEEE; 2015. p. 1–6.
[4] Bakken D. Smart grids: clouds, communications, open source, and automation. CRC Press; 2014.
[5] Meliopoulos AS, Polymeneas E, Tan Z, Huang R, Zhao D. Advanced distribution management system. IEEE Trans Smart Grid 2013;4(4):2109–17.
[6] Vaccaro A, Loia V, Formato G, Wall P, Terzija V. A self-organizing architecture for decentralized smart microgrids synchronization, control, and monitoring. IEEE Trans Ind Inform 2015;11(1):289–98.
[7] Terzija V, Valverde G, Cai D, Regulski P, Madani V, Fitch J, Skok S, Begovic MM, Phadke A. Wide-area monitoring, protection, and control of future electric power networks. Proc IEEE 2011;99(1):80–93.
[8] Gungor VC, Sahin D, Kocak T, Ergut S, Buccella C, Cecati C, et al. A survey on smart grid potential applications and communication requirements. IEEE Trans Ind Inform 2013;9(1):28–42.
[9] D. PowerFactory, Version 14.0. DIgSILENT International, Germany. URL http://www.digsilent.de/.
[10] E.P.S.A.. O. Software, ETAP; 2017. URL https://etap.com/.
[11] S.E.D. NS, Schneider; 2014. URL http://www.schneider-electric-dms.com/.
[12] Tanskanen A, Raussi T, Partanen J, Lohjala J. Cost and benefit analysis for a distribution management system in electricity distribution networks. Int J Energy Sect Manage 2010;4(2):256–72.
[13] Katic N. Benefits of smart grid solutions in open electricity market. Acta Polytechnica Hungarica 2013;10(2):49–68.
[14] Katic N, Marijanovic V, Stefani I. Smart grid solutions in distribution networks cost/benefit analysis. In: CICED 2010 Proc. IEEE; 2010. p. 1–6.
[15] Simmins JJ. The impact of PAP 8 on the common information model (CIM). In: Power Systems Conference and Exposition (PSCE), 2011 IEEE/PES. IEEE; 2011. p. 1–2.
[16] Pradeep Y, Seshuraju P, Khaparde SA, Joshi RK. CIM-based connectivity model for bus-branch topology extraction and exchange. IEEE Trans Smart Grid 2011;2(2):244–53.
[17] Moffet M-A, Sirois F, Beauvais D. Review of open-source code power grid simulation tools for long-term parametic simulation. CanmetENERGY, Tech. Rep. 137.
[18] E.P.R. Institute, EPRI, 2017. URL http://www.epri.com/.
[19] Tsampasis E, Sarakis L, Leligou HC, Zahariadis T, Garofalakis J. Novel simulation approaches for smart grids. J Sens Actuat Netw 2016;5(3):11.
[20] Kumar Y, Ringenberg J, Depuru SS, Devabhaktuni VK, Lee JW, Nikolaidis E, Andersen B, Afjeh A. Wind energy: trends and enabling technologies. Renew Sustain Energy Rev 2016;53:209–24.
[21] Kouzapas D, Yoshida N, Hu R, Honda K. On asynchronous eventful session semantics. Math Struct Comput Sci 2016;26(02):303–64.
[22] Ionescu VM. The analysis of the performance of RabbitMQ and ActiveMQ. In: 2015 14th RoEduNet International Conference-Networking in Education and Research (RoEduNet NER). IEEE; 2015. p. 132–7.
[23] Rabl T, Gómez-Villamor S, Sadoghi M, Muntés-Mulero V, Jacobsen H-A, Mankovskii S. Solving big data challenges for enterprise application performance management. Proc VLDB Endowment 2012;5(12):1724–35.
[24] Moniruzzaman A, Hossain SA. Nosql database: new era of databases for big data analytics-classification, characteristics and comparison, arXiv preprint arXiv:1307.0191.
[25] Cassandra A. The apache software foundation. The Apache Cassandra project.
[26] Pradeep Y, Seshuraju P, Khaparde S, Joshi RK. Flexible open architecture design for power system control centers. Int J Electr Power Energy Syst 2011;33(4):976–82.
[27] Dugan RC. Reference guide: the open distribution system simulator (OpenDSS). Electric Power Research Institute, Inc.
[28] Alder G. Design and implementation of the JGraph swing component. Techical Report 1 (6).
[29] Falstad P. Circuit simulator v1.2, 2013. URL http://www.falstad.com/circuit/.
[30] Garzón MA, Aljamaan H, Lethbridge TC. Umple: a framework for model driven development of object-oriented systems. In: Software Analysis, Evolution and Reengineering (SANER), 2015 IEEE 22nd International Conference on. IEEE; 2015. p. 494–8.
[31] MindFusion, JDiagram (2016). URL http://www.mindfusion.eu/onlinehelp/jdiagram/.
[32] Zhou M, Zhou S. Internet, open-source and power system simulation. In: Power Engineering Society General Meeting, 2007. IEEE. IEEE; 2007. p. 1–5.
[33] Steiniger S, Hunter AJ. Free and open source GIS software for building a spatial data infrastructure. Geospatial free and open source software in the 21st century 2012:247–61.
[34] p. OSGeo, The open source java gis toolkit (2016). URL http://www.geotools.org/.
[35] Boundless. OpenGeoSuite; 2016. URL http://suite.opengeo.org/.
[36] Barron C, Neis P, Zipf A. A comprehensive framework for intrinsic OpenStreetMap quality analysis. Trans GIS 2014;18(6):877–95.
[37] Dugan R, Arrit R, Henry R, McDermott T, Sunderm W. Opendss distribution system simulator-harmonic load modeling documentation. retrieved december, 2014; 2014.
[38] Dugan RC, McDermott TE. An open source platform for collaborating on smart grid research. In: 2011 IEEE Power Energy Soc General Meet. IEEE; 2011. p. 1–7.
[39] Adler D. The jacob project: A java-com bridge; 2005.
[40] Gamma E. Design patterns: elements of reusable object-oriented software. Pearson Education India; 1995.
[41] Feeders DT. IEEE power and energy society (2014). URL https://ewh.ieee.org/soc/pes/dsacom/testfeeders/.
[42] Snyder B, Bosanac D, Davies R. Introduction to apache ActiveMQ. Active MQ in Action 6–16.
[43] A.S. Foundation, Apache ActiveMQ availability (2011). URL http://activemq.apache.org/failover-transport-reference.html.
[44] A.S. Foundation, Apache ActiveMQ security (2011). URL http://activemq.apache.org/security.html.
[45] Rademakers T. Activiti in action: executable business processes in BPMN 2.0. Manning Publications Co.; 2012.
[46] A.S. Foundation, Apache camel crypto; 2011. URL http://camel.apache.org/crypto.html.
[47] The Common Information Model CIM: IEC 61968/61970 and 62325-A practical introduction to the CIM, author=Uslar, Mathias and Specht, Michael and Rohjans, Sebastian and Trefke, Jörn and González, José M, year=2012, publisher=Springer Science & Business Media.
[48] Cyganiak R, Wood D, Lanthaler M. Rdf 1.1 concepts and abstract syntax. W3C Recommendation 2014;25:1–8.
[49] Song J, Sierra SC, Rodríguez JC, Perandones JM, Jiménez GDC, Buján JO, García RM, Galdón AS, et al. Data consistency management in an open smart home management platform. In: Modelling Symposium (EMS), 2014 European. IEEE; 2014. p. 366–71.
[50] Curtis K. A DNP3 protocol primer. DNP User Group; 2005.
[51] Raimbault S. A modbus library for linux, mac os x, FreeBSD, QNX and win32; 2016. URL http://libmodbus.org/.
[52] E.D. y Comercializadora Norte Sociedad Annima, EDENOR; 2008. URL http://www.edenor.com.ar/cms/SP/EMP/ACE/COM.html.
[53] E.D.S.S. Annima, EDESUR; 2016. URL http://www.edesur.com.ar/masedesur/conoceedesur.aspx.
[54] A.S. Foundation, Apache ActiveMQ; 2011. URL http://activemq.apache.org/what-is-the-prefetch-limit-for.html.