Intelligent Information Management, 2020, 12, 43-62
https://www.scirp.org/journal/iim
ISSN Online: 2160-5920
ISSN Print: 2160-5912

Secure and Performant Handling CIM-Based Data Streams in Control Room Software Interaction for Multi Vendor Solutions

Benjamin Requardt, Sebastian Wende-von Berg, Martin Braun

Department of Grid Planning and Operation, Fraunhofer Institute for Energy Economics and Energy System Technology, Kassel, Germany
Email: Benjamin.Requardt@iee.fraunhofer.de

How to cite this paper: Requardt, B., Berg, S.W. and Braun, M. (2020) Secure and Performant Handling CIM-Based Data Streams in Control Room Software Interaction for Multi Vendor Solutions. *Intelligent Information Management*, **12**, 43-62.
https://doi.org/10.4236/iim.2020.121004

Received: December 20, 2019
Accepted: January 14, 2020
Published: January 17, 2020

Abstract

Due to the increasing share of renewable energy, new requirements are placed on control room software. Such software is often exclusive to the supplier, but other suppliers could offer new and better methods. For security reasons, external applications often have no direct data access to control room software. Such software can provide information about the power grid via a periodic file transfer in CIM (Common Information Model) format. These files are often very large, containing complete records, delivering information not always relevant to the external applications. Extracting the relevant information required by external applications can be time-consuming, thus presenting a problem for time-critical applications. This paper presents a method allowing different applications to efficiently access the relevant data from the massive data stream contained in the CIM files. This method has been tested with a distribution system operator and clearly increases performance, allowing different applications to access the relevant data.

Keywords

Common Information Model (CIM), Grid Control Room Software, Electrical Grid

1. Introduction

Exchanging and processing data for electrical grid information is one of the challenges faced by a transforming energy system. The complexity of operating electrical grids is constantly increasing. The growing amount of variable distributed energy resources needs coordination with schedules and forecasts. This

requires new approaches (software, architecture and communication protocols) [1] [2] [3] [4] [5]. Software (such as a control system) for smart grids allows the coordination of renewable generators in general to be improved and as a more specific example, it enables a more effective coordination of ancillary services such as reactive power provision [6] or the smart integration of electric vehicles into the electricity grid [7]. The monitoring and operation of the power grid is carried out in the control room. Control rooms are operated with software systems that are often provided by a single vendor to the grid operator. It might be helpful for other providers to extend software systems to include new functionalities, since this increases the diversity of the solutions. These new functionalities would represent extensions to the control room software of the grid operator. However, adding extension modules to existing control room software for such innovative control approaches is difficult and costly, and in most cases only possible for the original manufacturer. Control room extension requires continuous information about the power grid (e.g. topology or measurement data). Control room software can have the option of periodically providing information about the power grid as a CIM (Common Information Model) file to external applications. This functionality is often installed in the control room software of those grid operators who are obliged to forward information about their power grid to third parties (e.g., a distribution system operator needs to forward information over the grid to a transmission system operator) [8]. External modules can use this functionality to offer new innovations to the grid operator. Some modules require continuous state information about the power grid (such as measurements and switching states) and this at a high-performance speed. Collecting the CIM files periodically and extracting the information they need can be a time-consuming process for a single application, and many applications require only part of the information about the power grid. The question arises as to how the relevant data from the constantly changing CIM files can be made available to external modules at high speed. In this paper, an approach is presented for distributing the information from the CIM files to many external applications with the assistance of modern technologies.

1.1. CIM: Standard for Data Exchange in the Energy Industry

The Common Information Model (CIM) is a UML (Unified Modeling Language) based data model for describing the electrical grid [9]. The goal of CIM is to generate a data model that describes electrical grids as well as related data (e.g. load schedules and generator models). The CIM uses classes with attributes to describe the different object type (e.g., "analogvalues" are used for measured values). The CIM also allows the user to define profiles, which are a subset of the semantic model and designed for the user's application. The Common Grid Model Exchange Standard (CGMES) is such a CIM profile and is published by ENTSO-E (European Network of Transmission System Operators for Electricity). CGMES divides the CIM into different profile classes: equipment profile

(EQ) for device information, topology profile (TP) for grid topology information, steady state hypothesis profile (SSH) for load flow simulation and state variables profile (SV) for measurement information. The CIM uses RDF (resource description framework) files for data exchange. The RDF model uses statements about resources in the form of subject-predicate-object expressions (triples), e.g. the wind turbine produces two MW. Each object in RDF is tagged with a Universally Unique Identifier (UUID) and can be referenced (RDF ID). RDF/CIM files are in XML (extensible markup language) format by default. CIM offers many advantages, among others it can speed up complex processes and reduce errors, as its goal is to increase automation in the energy sector [10]. The CIM exchange format is clearly defined and the definition of profiles increases the consistency between producer and consumer during data transfer [11]. Data integration is a big challenge in the smart grid, which can be solved with the help of CIM [12] [13].

1.2. Extensions for Control Room Software

Control room software is used to monitor and control the power grid and its devices. New tools and applications for control room software, e.g. optimization modules can improve grid operation. Connecting control room software to other systems or applications is often very difficult, especially for security reasons [14] [15] [16]. Adding extension modules for control room software or updating this with new functionalities for innovative control approaches requires much effort [17] and is in most cases only possible for the original manufacturer [18]. The CIM interface, if the control room software for electrical grid can offer it, is suitable for third-party provider's access and expansion, so that the existing system does not need to be replaced or modified beyond standard user customizations. This interface provides CIM files at periodic intervals (e.g. every 5 minutes).

Figure 1 describes a possible way for control room software to provide CIM files for other applications. The control room software is usually located in a so-called high security area that is separated from the other applications (referred to here as office world) by one or more firewalls. The control room

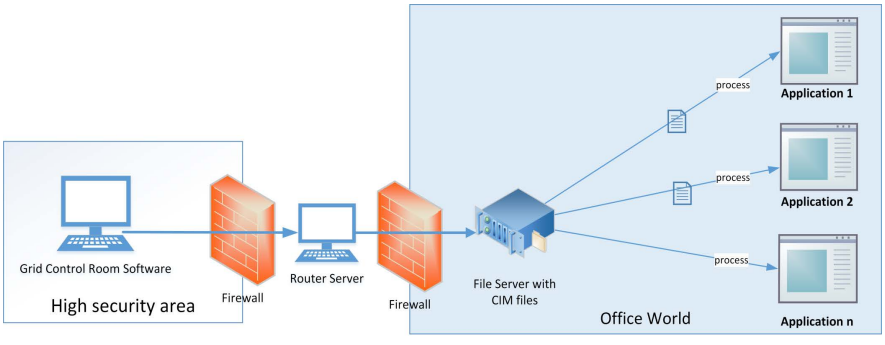

Figure 1. Providing CIM files from the control room software for external applications or processes.

software can consist of several servers (represented by one symbol in the figure). Within the high security zone, multiple servers may also be available. After the control room software has created the CIM file, it is often first copied to other computers within the high security zone (usually via secure copy protocol). Then the files are put on a so-called router server. This is also protected by firewalls and serves as a link between the high security zone and the office world. Subsequently, the files are stored on a file server. Due to the many transport routes, there may be delays until the files have arrived in the final system (e.g. file servers). Therefore, lower cycles are usually selected for the file transport (e.g. every 5 - 15 minutes), to avoid a time overlap of the delivery process. The applications have access to the CIM files through the file server. However, this means that each process must have implemented its own capture and storage of the CIM file. Furthermore, an application does not necessarily require the complete data set from the control room software, but only certain data records. The delivery of the CIM files for the control room can be used to request information about the power grid to be passed on to third parties (e.g. once a day or manually by a user). For the control room software, it may be irrelevant whether the third party already has information about the power grid. Thus, the complete data set is provided by the control room software, regardless of whether information (e.g. measured values) has changed from the last file sent to third parties. This can lead to a massive increase in CIM data. The CIM standard (IEC 61968-100) defines how existing technologies for distributing and relaying information (e.g. web service or Java Message Service) can be used to distribute or forward CIM data. The standard defines the interface for the web service and concrete message envelopes (such as header or request information) for the transmission the CIM data. The aim is to increase interoperability for the exchange of CIM data [19]. However, the interfaces defined in the standard could pose a security problem for the control room software. For example, a web service would have to be made accessible from outside through the firewall; this would mean a permanently open interface to the control room software, which could increase the security risk. A file transfer from the control room software requires no open access from outside. Furthermore, transferring the file takes place via several servers and only for a certain period of time. These mechanisms make unauthorized access or targeted overloading (a so-called denial of service attack) of the control room software much more difficult. This increases the security of the control room software, but makes access to required information more difficult for external applications. Capturing the complete CIM record and extracting the required information can increase the complexity of the application and lead to more computation time. This raises the question of how the massive data flow of information about the grid in the CIM file format can be distributed to different applications or services using modern technologies. At the same time, security must be ensured and existing functionalities of the control room software are to be accessed.

1.3. Related Work

An important aspect for the distribution of information concerning the power grid is the connection to a control room or a SCADA (Supervisory Control and Data Acquisition) system. For instance, a SCADA system was integrated into the IT landscape with the help of an ESB (Enterprise Service Bus) to increase its interoperability with other applications [20]. However, the extraction of data from the SCADA system can be difficult [21]. In order to communicate between different applications, a so-called middleware is often used. This allows independent software components to exchange data [22]. This was, for example, used by an optimal power flow application (OPF) to receive and process data from a SCADA system. The middleware transforms data from the low-level devices to CIM and a central database is used for the storage [23]. Another important issue for the distribution of information concerning the power grid is the interoperability of software systems in the energy sector. CIM is often used to improve communication between these systems. For example, networking different software systems (e.g. geographic information system) via an ESB with CIM as a unified data model can lead to new features [24]. Another solution provides data from various software systems installed at a distribution system operator via adapters. These adapters provide the data as a CIM data model through REST (Representational State Transfer) [25]. An alternative approach to increasing the interoperability of a grid operator is to create a reference architecture. This is a monolithic approach based on the CIM data model and uses an ESB and a central database [26]. The use of an open, standards-based platform can also lead to higher interoperability for applications that require data from different software systems [27]. Until now, data has been obtained from SCADA systems or control room software. These data, or data from other software systems, were converted to CIM to increase interoperability. So far, no consideration has been given to how massive CIM data (e.g. as files) from a control room software can be effectively made available to other applications without endangering the security of the system.

1.4. Scope of This Paper

Control room software has the option of providing information about the grid as CIM files for other applications. However, many applications do not only require unique information about the power grid (e.g. which devices are installed), but also regular information such as switching states or measured values. Collecting CIM files periodically can be computationally expensive and complex for a single application. This paper describes a new aspect: how information from regularly transmitted large CIM files can be distributed to many applications using modern methods. The innovation here is that applications can search for relevant CIM data and be informed when new CIM records are available and which records have changed. This new aspect has been examined with regard to performance using the control room software of a German grid operator. First,

the methods are presented (2), followed by the selected software architecture (3.1) and the existing technologies which are used. The following is an example implementation of the methods (3.2) and an example of an application (3.3) using the new CIM data method. To evaluate the new method, the implementation was tested on the premises of a German grid operator regarding data volume and performance (4).

2. Method for Processing Large CIM Files and Distributing Their Content

The method describes a modern but in practice thoroughly applied approach to a new problem. For a better and easier processing of CIM files in object-oriented programming languages, the RDF model is transformed into an object-oriented data model. For this purpose, classes are generated from the CIM data model for the desired object-orientated programming language and the CIM profile. To store the CIM objects, which are generated from the CIM classes, in a database, a corresponding database model is set up. The first step is to establish that the control room software has sent a new CIM file and then CIM objects are created from this information and synchronized with the database. After that, the CIM files are validated. The syntax is checked to ensure the CIM file corresponds to the desired profile (attributes and association are set correctly). It is also determined whether CIM objects have references to non-existing CIM objects, so it can be validated how far a CIM file is self-contained. For the use of historical CIM data, a version object (creation time, file path, profile, version ID, etc.) is created for each acquisition process and the CIM objects are associated with this object. To enable applications to recognize which CIM objects have changed from the previous version, a hash value is created across each CIM object and mapped to its RDF ID. Furthermore, this allows a faster capture, because only CIM objects are recorded that have changed or are new. Thus, it can be detected in the next acquisition process whether a CIM object has been deleted, attributes have changed or a new object has been added. This information, along with the result of the validation is transmitted to the applications via a message broker. Applications gain access via a web interface to the CIM objects. The use of a database language e.g. structured query language (SQL) and the resulting concatenation of strings can be time-consuming and complex for querying CIM data (the CGMES profile has over 300 classes). Appropriate APIs (application programming interface) provide a simpler solution for filtering data. Furthermore, these enable a consistent concept for SQL and SQL-proximate query languages. For better use of the API, so-called query classes were generated from the CIM data model. The possibility of filtering the data is offered via the web interface.

3. Architecture and Implementation

The distribution of CIM data from the control room software to various applications and services is performed by a microservice that communicates with other

applications via REST and a message broker (in this case ActiveMQ). A microservice in general can be understood as a stand-alone (runs as one operating system process) and a small service. Several microservices exchange data and work together [28]. The communication takes place over the computer network and the microservice can offer an API that can be used by others.

3.1. Architecture

Distributing CIM data from the control room software to various applications and services is an important part of a modular platform (called beeDIP) that offers extension modules (e.g. reactive power optimization for PV and wind generators) for control room software.

Figure 2 shows the developed system architecture of the system platform. Under "control room software extensions" the extension modules (e.g. state estimation, optimal power flow, CIM converter to other formats etc.) are displayed. Each block represents a microservice which then may use various technologies for its task. The control room extensions are a loose component of the system architecture and can be adapted, redeveloped or discarded depending on project needs. A uniform interface (proxy) for the programming languages Java, Python and C-sharp is available for connecting to the message broker and retrieving the data via REST. The proxy allows the application or service to register with other applications or services: as soon as new data is available (e.g. new CIM files from the control room software), it is communicated to the registered applications via the message broker and the data is retrieved via REST. For the corresponding programming language, the proxy provides callback methods, which are invoked if new data from the desired service is available. This allows

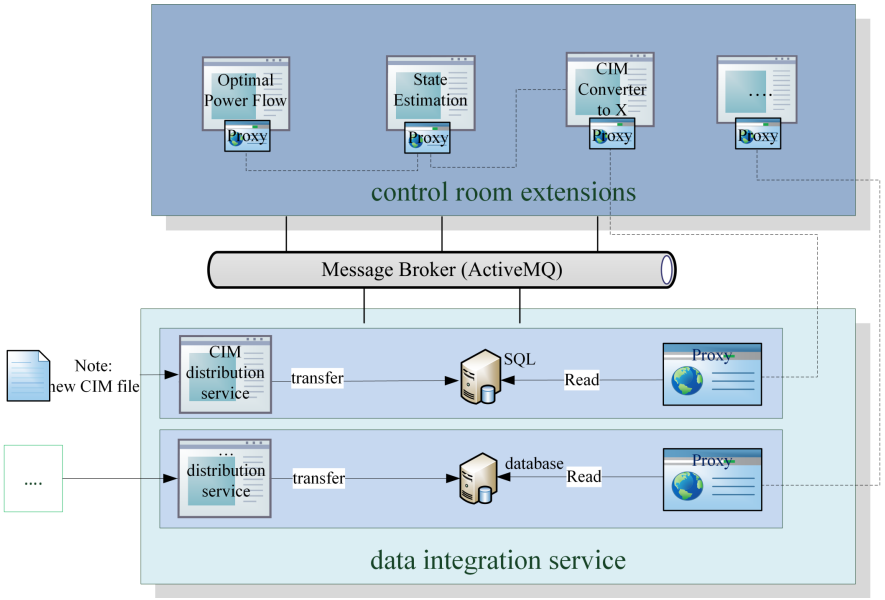

Figure 2. Architecture of the system platform. The CIM distribution service is an important part of it. The dashed lines are example connections.

the extension to focus on its tasks and does not have to deal with the handling of REST and the message broker. Extensions for control room software can also exchange data with each other via the proxy. The service for the distribution of CIM data from the control room software is part of the data integration services. In this category, further distribution services are grouped for other communication standards e.g. TASE.2 (Telecontrol Application Service Element 2). Extensions or other applications can also use the CIM distribution service via REST and the message broker without the use of the proxy and integration into the system platform.

3.2. Example Implementation for the CIM Distribution Service

Data (topology, device information, measurement) can be regularly provided from the grid control room software as CIM/CGMES (RDF) files. Many power flow tools and applications can import and export CIM data [29]. This implementation focuses on capturing CIM/CGMES data with high performance and extracting the desired information, which is necessary for many control room software extensions. The CIM file integration service attempts to capture the regular CIM/CGMES data as efficiently as possible. A previous version of the service is described in [30]. To recognize new files in the control room software, this service (named CIM distribution service) monitors a folder in which the CGMES files are copied from the control room software. As with other tools, information concerning the CIM objects is extracted using Apache Jena [31], [32]. In detail, this service produces Java objects from this information before the objects are stored in a database. These objects are created using Java classes generated from the CGMES data model. This uses Java reflection to generate CIM objects and set attributes. As a result, the classes and associated attributes are determined at runtime instead of being permanently stored in the code. This significantly reduces the number of lines of code and the implementation can be used for several CIM profiles. The hash value via the CIM attributes is generated with the hash code function of Java. This determines whether a Java CIM object has changed in comparison to the previous CGMES file provided by the control room software. A CIM Java object can have the following status: new, deleted, changed, error and equal. Thus, CIM Java objects are only established if they are also new in the CGMES file. This increases the performance, because creating new objects in Java could be a time-consuming operation [33], especially when the object is created using Java reflection [34]. Objects that have just changed will only match the attributes and objects that have been deleted are no longer recorded. Via the proxy, this status information is communicated to the other services. The individual profiles (CGMES files) can be captured in parallel, which also leads to a higher performance. Database types for control room software [35] but also for CIM [36] have been analyzed in the past. In this service, the object-relational database PostGreSQL is used to take advantage of object-oriented design (e.g. speed) but still have access to the data through SQL

[37]. The database structure is set up with Flyway and the individual CIM Java objects are transferred to the database via EclipseLink. Even if a CIM object has not changed from the previous version, it will still be saved in the database. Thus, a complete image of the power grid is stored in the database every time. This service was implemented with the help of Spring Boot and is integrated into the platform via the proxy. For the documentation of the REST interfaces, the framework Swagger is used. To increase security, the REST service can only be used over HTTPS (Hypertext Transfer Protocol Secure). An API key is used to determine which services make use of the offering service and can access the provided data [38]. The REST service allows the extensions to search for specific CIM objects (e.g. CIM objects for wind generators) and, in addition, filters them according to certain criteria (e.g. wind generators with high power). This allows each extension to receive only the required CIM objects. QueryDSL enables filtering for certain CIM objects. For this purpose, a corresponding QueryDSL class was generated for each CIM class and initialized also via Java reflection according to the request, e.g. only wind turbines with a nominal value (CIM attribute pNominal) above 2 MW and of type onshore (CIM attribute type) should be returned. The String for the filter function is as follows: filter = (nominal P > 2) & filter = (type = onshore). Two QueryDSL expressions are formed from this dynamically (pNominal.gt (2) & type.eq ("onshore")), after this, it is transformed by QueryDSL into an SQL query. To use the REST interfaces, the proxy will automatically construct a URL (Uniform Resource Locator), based on the address of the REST interface of the service, the desired CIM type (e.g. Windgenerating Unit), profile (e.g. CGMES EQ) and the desired filter. With the help of these search and filter methods, applications have the possibility of extracting data for a certain grid area e.g. to perform grid calculations.

3.3. Example Extension for Using the CIM Distribution Service: CIM to Pandapower Converter

Services and applications for grid calculation require constant topology and device information about the grid that can be provided periodically via CIM from the control room software. CIM is often too complex and detailed for grid calculations, other models are more appropriate. Furthermore, CIM can be in node-breaker model and system simulations for example are performed in the bus-branch model [39]. There are already tools for converting CIM into physical models for calculations [40]. An open source library for grid calculation and analysis is available within the pandapower framework [41]. It provides functionalities such as state estimation, optimization, short-circuit calculation or contingency analysis. Since this framework is based on the Python programming language, grid calculation services using this framework can be flexibly supplemented and further developed. In order to use these already available functions, a converter has been developed from CIM to pandapower data model (load flow grid). This service uses the CIM distribution service and applies the status in-

formation about each CIM object. When the data is retrieved, it filters the CIM objects that are new or have changed. As a result, the data model of the power grid (in pandapower format) is selectively updated and not rebuilt from each CIM file. Thus, new CIM data or information can be converted with high performance into a load flow grid model and online calculations are possible.

4. Evaluation of the CIM Distribution Service

The following was analyzed for the CIM distribution service evaluation: What amount of data do the extensions for control room software have to expect, how can the CIM data be retrieved via the interfaces and at which speed can the information about the grid be distributed? The investigation was carried out on the premises of a German distribution system operator. For this purpose, the CIM distribution service, pandapower converter and its required software systems (such as ActiveMQ v. 5.15.5 and Postgre SQL v.10.5) were installed on a local Windows server (8 GB of memory; Intel Core I7 with 2 cores and 2.8 GHZ). Grid control room software from PSI (http://www.psienergy.de/) provided CIM files every 15 minutes.

4.1. Analysis of the Data Volume of a Distribution Grid

This authentic grid of a distribution grid operator consists of 270 devices (producers and consumers) and their electrical equipment (over 6000 cables, switches etc.). For the description of the devices in the grid, CIM needs numerous (about 18.000) additional virtual CIM objects (such as base voltage, geographical region etc.). In total, there were over 24.000 CIM objects in the equipment profile (EQ file). The Topology profile (TP file) comprised of 3444 CIM objects, the steady state hypothesis profile (SSH file) 15.800 and the state variables profile (SV file) 22.000 objects.

Figure 3 shows the required data volume of different data formats for storing the CIM profiles. As expected, we can see a significant difference in data volume (in megabytes) between RDF-file, JSON-file and database (db). This is due to the following reasons: CIM in RDF format (XML syntax) require more space than JSON format, because XML syntax has a larger syntax overhead than JSON

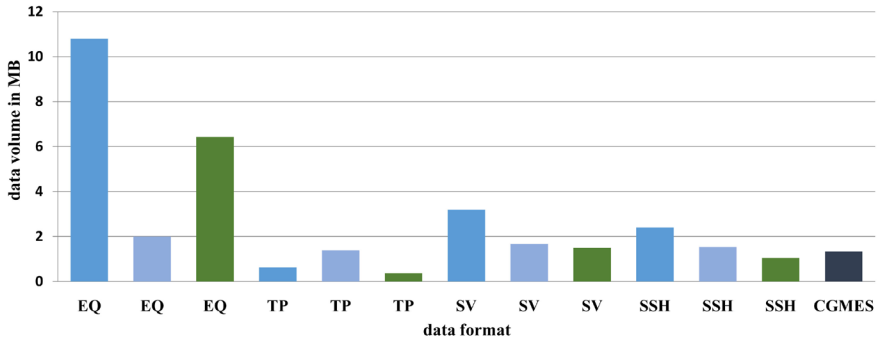

Figure 3. Required data volume of CIM in different formats.

format. For databases, the Meta information (tables, columns, etc.) is the same for all profiles, since they all use the same database schema and there is no individual database per profile. This means that even if information for one of the profiles is not contained in the database, memory storage is required for storing the metadata (see "db schema only" in Figure 3). In general, there is less information stored in SV, SSH and TP than in the EQ profile and the Meta information exceeds the actual information. It must be mentioned that Figure 3 shows only one record (RDF file). If several data records (RDF files) are transformed into the different formats (JSON or database), the actual information requires more data volume than the Meta information in the database. The data storage evaluation has shown that storing topological and device information (which does not change frequently) in object-relational databases is sufficient for a single grid operator (for the EQ profile, approximately 22 GB is required for a year with a 15 minute resolution). In view of the volume of data, it is recommended that CIM data is not stored as RDF files for a longer period, but stored in a database. Considering the result regarding the data volume of the database, solutions for processing large amounts of data e.g. NoSQL (Not only SQL) are not necessary for the time being. Of course, this may change if several grid areas or longer periods (several years) are considered. Nevertheless, not every application should store the complete CIM data itself, depending on the format (RDF, database etc.), as this can lead to unnecessary memory usage and not every application requires all the CIM data. An application should, if needed, store only the relevant CIM data.

4.2. Analysis of the Interfaces for Retrieving the CIM Data

After the CIM files have been recorded in the database, the data can be retrieved via the proxy or via REST. Before that, status information concerning the CIM file can be retrieved from the message broker.

Figure 4 shows an example of status information concerning the CIM file. It presented: The file path, the CGMES profile, a hash value for the RDF file, the time it was captured, its length of validity, status about the capture (successful, warnings or errors) and status information concerning individual CIM objects, with the following categories: is equal, has been deleted, has changed, has errors and is new. In the respective category, the RDF ID and the type of the CIM object is displayed. After viewing the status information, the data can be retrieved via REST. Table 1 shows examples of retrieving CIM objects through the REST

"CimParseInformation": {"file": "...\\toParse\\20180116T0230ZZ1_EQ.rdf","profile": "http://entsoe.eu/CIM/EquipmentCore/3/1","hashRDF": -6738970429236,"validFrom": "2018-09-03T14:04:38.000+02:00","validTo": "2018-09-03T14:19:38.000+02:00","warnings": { },"status": "SUCCESS","statusCIMObj": {"isEqual": {"a24b640a-...": ["VoltageLevel"],"f6bb9d8c-...

Figure 4. Example of status information of the captured CIM file.

Table 1. Examples of retrieving CIM data through the REST interface. The performance could be increased by filtering.

Profile	CIM Type	Filter	Size	Duration
EQ	Wind Generating Type	None	35	0.171 s
EQ	Wind Generating Type	Nominal P > 30	15	**0.087 s**
EQ	Sv Power Flow	None	68,000	2 s
EQ	Sv Power Flow	p > 0 & q > 0	175	**0.04 s**

interface or proxy. The first column (header Profile) shows the CGMES profile (here EQ or SV), the second column (header CIM Type) shows the CIM type being retrieved (here Wind Generating Unit or Sv Power Flow). The third column (header Filter) shows the respective filters for each CIM type. Either no filter is set or corresponding attributes must meet certain criteria, e.g.: The rated output (attribute: nominal P) of the wind farm must be above a certain value (here 30), or the current measured values (CIM type Sv Power Flow) must be above zero. Here it is also possible to filter CIM objects based on the RDF ID. By setting the version ID to zero (version Id = 0), the REST interface returns the last received CIM data. A separate REST interface is provided for retrieving the version information. Here the version number, the time of capture and the CGMES profile are displayed. The fourth column (header Size) shows how many JSON elements have been returned for the grid specified above via the REST interface. In the last column (header Duration) the time (in seconds) is listed, until the receiver (here for test purposes Chrome browser) has received the CIM data.

Figure 5 shows a section for receiving CIM data formatted as JSON. Here the CIM type Wind Generating Unit with its attributes is represented. For references to other CIM objects (here Equipment Container) their RDF ID is used, instead of nested JSONS. This is similar to the RDF syntax and reduces the volume of data for each retrieval. Since CIM objects are strongly networked with each other, the JSON documents could become very large with nested representation. Furthermore, many CIM objects would be listed multiple times because many CIM objects point to the same object. The presented interfaces (status information and REST interface) can enable applications to deal with large CIM files better and easier. The status information can help applications to analyze CIM files and their objects better. It is now possible to view individual CIM objects e.g. that have changed or that have been newly added. If a file has errors (e.g. syntax errors), warnings (e.g. empty references to other CIM objects) or has not changed compared to the previous file (detectable by the hash value), this captured file can be ignored by the application. These mechanisms can lead to higher performance in the application, since the entire content no longer needs to be considered. Table 1 illustrates that filtering (e.g. only displaying measurements with a positive value or only certain changed CIM objects based on the RDF ID) can also significantly reduce response time. Different applications can

[{"windGenUnitType": "onshore","nominalP": 32,"equipmentContainer": "_e5085c12-9787-4dc8-a1a1-7611aeaf4333","name": "Test Wind Generating Unit","rdfId": "_85ac63da-a124-483d-9819…"

Figure 5. Extract of receiving CIM data formatted as JSON.

thus filter unnecessary data and increase performance. A unified interface (e.g. the same REST URL schema) can simplify the development of applications that require CIM data because existing mechanisms (e.g. REST data retrieval) can be transferred to new applications.

4.3. Analysis of the Performance for the Capture and Conversion of CIM Files

Figure 6 illustrates the time taken (in milliseconds) to parse and save to the database of the incoming CGMES profile files (EQ, TP, SSH, SV) every 15 minutes. Furthermore, the conversion time from CIM to pandapower (includes loading the data via REST) is shown. **Figure 7** shows the total time for the process of capturing the CIM files into the database, loading the data via REST and then converting the CIM data into the pandapower data model (every 15 minutes). In addition, it is displayed that the CIM files in each iteration completely re-capture the data, load via REST and convert to pandapower (ignoring the differences to the previous data). The pandapower converter needs about 80 percent of the different CIM types to convert the grid to a pandapower model. Since the profiles are acquired in parallel, the total time is: max (time for storage of [EQ, SSH, SV, TP] into database) + (time for conversion of the CIM data into the pandapower data model; includes loading the data via REST). **Figure 7** shows that the first acquisition and conversion of the CIM files takes the most time (about 29 seconds). After that, the capture and conversion time is about 13 seconds. This is due to the creation of status information concerning the individual CIM objects, since only new objects or objects with changed parameters must be captured and converted. Since there are fewer activities in the grid (such as switching events) during the night (02:00 - 06:00) and thus the CIM files stay largely constant, the capture and conversion time is also constant at approx. 13 seconds. In the morning hours and during the daytime, there are significantly more fluctuations in capture and conversion times due to ever-changing CIM files. These changes in CIM data are related to, for instance, grid operations such as switching events or tap changer positions.

Figure 8 shows an overview of the average time saved in percent by parallel acquisition and observation of changes in the CIM files in contrast to methods without parallel acquisition and observation of the changes. The capture and conversion time is on average 16 seconds for the entire period. When the previously received data is not taken into consideration, an average acquisition time of 23 seconds is the result. This means that the capture and conversion of the CIM files, taking into account the previously received data, leads to a time saving

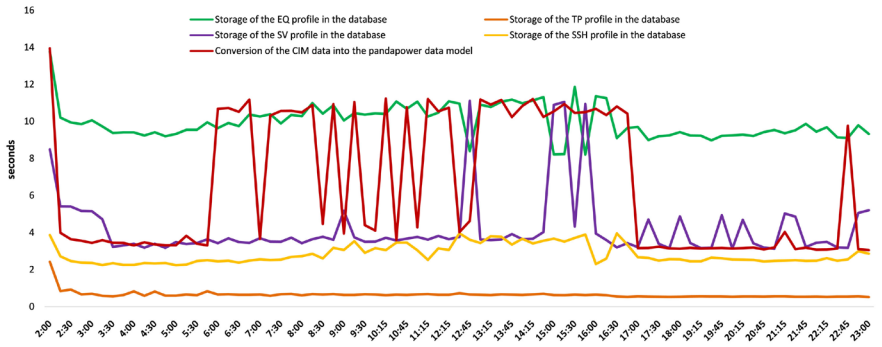

Figure 6. Storage time for different CIM profile (EQ, SV, TP, SSH) files and converting CIM to pandapower data model.

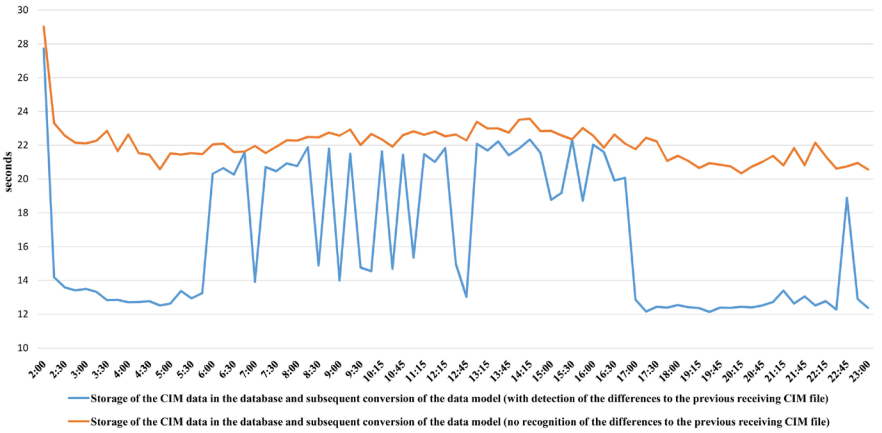

Figure 7. Total time of acquisition, storage and subsequent conversion according to the pandapower data model. The time is illustrated both with recording of the differences to the previous receiving file and without it.

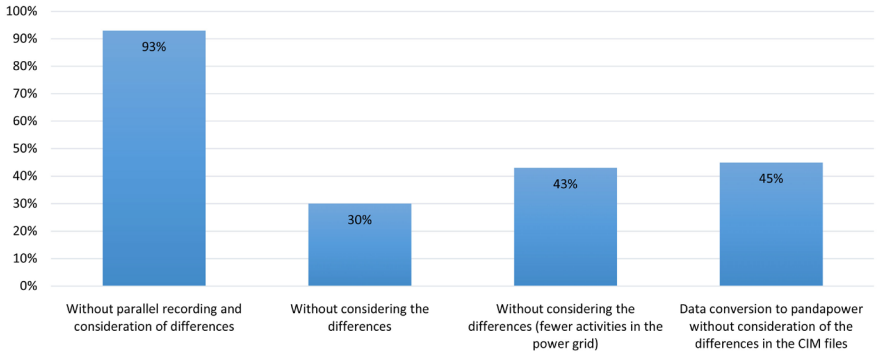

Figure 8. Average time saving by parallel recording and consideration of the differences compared to other methods.

of 30% on average. With fewer activities in the electricity grid (02:00 - 06:00) the time saved is 43%. The largest savings are for the pandapower data model converter (45% on average) when considering the differences to the previously received data. The outcome of this is that the performance could be increased by keeping the changes applied by CIM data to a minimum. In an earlier version of

the implementation without parallel capture and viewing the differences to the previous receiving files, the total time was about 4 minutes. Here, a timesaving of approx. 93% results from parallel recording and consideration of the CIM file changes. This version is more likely to be used for applications that are not time-critical.

4.4. Conclusion of the Results

Because a control room software is often a heavily sealed system and usually has no more modern interfaces (e.g. web services) for security reasons, simple file transfer (usually over many server addresses) of grid data formatted in CIM is often the only way for external applications to access such complex data as a power grid. However, this can lead to a delay in data flow that could pose a problem for many external applications (e.g. module for renewable energy reactive power management). Nonetheless, the evaluation shows that the smart method can minimize this delay, so that even time-critical applications can process using the CIM data from the control room software. With this method, it would be possible for the control room software to make the CIM files available to external applications at a significantly higher rate (e.g. every minute instead of every 15 minutes). This assumes that no large delays occur in the data transport of the CIM files to the file servers. The programming languages` independent technologies (REST and message broker) can access a variety of applications (here Python application) to the relevant data of the power grid. In addition, it has become clear that for a single grid operator, data output in the CIM format does not yet have to be managed by solutions for large amounts of data (e.g. NoSQL databases). Nevertheless, if necessary, an application should only store the relevant CIM data, which is simplified by this approach. Some applications require only part of the data from the control room software. Through defining the CIM type, setting the filters and data of the period (e.g. by specifying the version ID), the counting of data for the individual application can be reduced to a minimum and thus increase access speed. The administration and storage of grid data is reduced as both current and historical grid data can be retrieved via the interface with high performance. The fact that the developed software has its own server in an office world and no direct access to the control room software does not increase the security risk of the control room software.

5. Summary and Outlook

The increasing share of distributed generation and the volatile behavior of renewable generation from photovoltaic plants and wind turbines require advanced approaches for the integration into the electricity grid. Adding extension modules to control room software for innovative control approaches is difficult and in most cases only possible for the original manufacturer. CIM offers a standardized data model for power grids, which allows many external applications to interpret the receiving data unambiguously. For security reasons, most

control room software does not have modern interfaces making the complex CIM data available to external applications. However, control room software may be able to offer CIM data via periodic file transfer (e.g. every 15 minutes), as this does not endanger the access protection of the control room software. Capturing the CIM files and extracting the necessary information might cause a significant delay for external applications. This paper presents a method that uses modern technologies to deliver data about power grids, modulated as CIM files periodically provided by the control room software, to external and different applications. Through filtering and using status information from CIM data (changes in the CIM data compared to the previously received file), external applications can retrieve and process data relevant to them. The methods presented here make it possible to transfer CIM files at a higher rate (e.g. every minute) from the control room software, so also time-critical applications are enabled. Furthermore, the data volume required for a single application can be reduced, since historical and current grid data can be retrieved via the interface and the application no longer needs to store and manage this data itself. This method ensures the security and access control of the control room software while still allowing external applications access to the required data within a reasonable amount of time. This is also evident in the evaluation where the implemented method is tested using real control room software on the premises of a distribution system operator. The sample application (a data model converter) was able to process the CIM data much more efficiently with the help of the filter and status information. In the future, further approaches for processing data streams will be investigated, e.g. with the help of Apache Hive Hadoop [42]. Another option for accessing data from the control room software is provided by the TASE.2 interface. This raises the question of the advantages that arise for external applications through this interface and whether these can complement the CIM interface or whether both interfaces can be combined with each other. A further important issue for external applications for control room software is how data (e.g. calculated set points for reactive power management for distributed generators) can be transmitted with high performance without compromising the security and access protection of the control room software.

Acknowledgements

This work is based on results from the EU-SysFlex project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 773505. The authors are solely responsible for this publication.

This work presents selected results from the project "SysDL 2.0", which is funded by the Federal Ministry for Economic Affairs and Energy under grant no. 0325744E on the basis of a decision by the German Bundestag. The authors take full responsibility for the content of this paper.

This work presents selected results from the project "RPC2", which is funded by the Federal Ministry for Economic Affairs and Energy under grant no. FKZ 0350003A on the basis of a decision by the German Bundestag. The authors take full responsibility for the content of this paper.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Ramchurn, S.D., Vytelingum, P., Rogers, A. and Jennings, N.R. (2012) Putting the "Smarts" into the Smart Grid: A Grand Challenge for Artificial Intelligence. *Communications of the ACM*, **55**, 86-97. https://doi.org/10.1145/2133806.2133825

[2] Güngör, V.C., Sahin, D., Kocak, T., Ergüt, S., Buccella, C., Cecati, C. and Hancke, G.P. (2011) Smart Grid Technologies: Communication Technologies and Standards. *IEEE Transactions on Industrial Informatics*, **7**, 529-539. https://doi.org/10.1109/TII.2011.2166794

[3] Albano, M., Ferreira, L.L. and Miguel Pinho, L. (2015) Convergence of Smart Grid ICT Architectures for the Last Mile. *IEEE Transactions on Industrial Informatics*, **11**, 187-197. https://doi.org/10.1109/TII.2014.2379436

[4] Wu, F.F., Moslehi, K. and Bose, A. (2005) Power System Control Centers: Past, Present, and Future. *Proceedings of the IEEE*, **93**, 1890-1908. https://doi.org/10.1109/JPROC.2005.857499

[5] Saad, A., Youssef, T., Elsayed, A.T., Amin, A., Abdalla, O.H. and Mohammed, O. (2019) Data-Centric Hierarchical Distributed Model Predictive Control for Smart Grid Energy Management. *IEEE Transactions on Industrial Informatics*, **15**, 4086-4098. https://doi.org/10.1109/TII.2018.2883911

[6] Wende von Berg, S., Bornhorst, N., Gehler, S., Schneider, E. and Hänchen, H. (2016) SysDL 2.0 Systemdienstleistungen aus Flächenverteilnetzen: Methoden und Anwendungen. https://www.tugraz.at/fileadmin/user_upload/Events/Eninnov2016/files/pr/Stream_C/Session_C4/PR_Wende_von_Berg.pdf

[7] Palensky, P. and Dietrich, D. (2011) Demand Side Management: Demand Response, Intelligent Energy Systems, and Smart Loads. *IEEE Transactions on Industrial Informatics*, **7**, 381-388. https://doi.org/10.1109/TII.2011.2158841

[8] ENTSO-E AISBL (2018) GLDPM-v2 Approved. Brussels. https://www.entsoe.eu/news/2018/03/16/gldpm-v2-approved

[9] Uslar, M., Specht, M., Dänekas, C., Trefke, J., Rohjans, S. and González, J.M. (2013) ICT and Energy Supply: IEC 61970/61968 Common Information Model. In: *Standardization in Smart Grids*, Springer-Verlag, Berlin, 99-114. https://doi.org/10.1007/978-3-642-34916-4_6

[10] Britton, J., Brown, P., Moseley, J. and Bunda, M. (2016) Optimizing Operations with CIM: Today's Grid Relies on Network Analysis (and a Lot of Data). *IEEE Power and Energy Magazine*, **14**, 48-57. https://doi.org/10.1109/MPE.2015.2481783

[11] Österlund, L.-O., Hunter, K., Demaree, K., Goodrich, M., McMorran, A., Iverson,

B. and Kostic, A. (2016) Under the Hood: An Overview of the Common Information Model Data Exchanges. *IEEE Power and Energy Magazine*, **14**, 68-82. https://doi.org/10.1109/MPE.2015.2485859

[12] Hargreaves, N., Taylor, G. and Carter, A. (2012) Information Standards to Support Application and Enterprise Interoperability for the Smart Grid. *IEEE PES General Meeting*, San Diego, 1-6. https://doi.org/10.1109/PESGM.2012.6345134

[13] Marten, F., Hammermeister, I., Ringelstein, J., Requardt, B. and Martin, B. (2019) CIM CGMES-Extensions for the TSO-DSO Data Exchange in the EU-Project "TDX-ASSIST". *ETG*, Esslingen am Neckar, 69-74.

[14] Dán, G., Sandberg, H. and Ekstedt, M. (2012) Challenges in Power System Information Security. *IEEE Security Privacy*, **10**, 62-70. https://doi.org/10.1109/MSP.2011.151

[15] Tan, S., De, D., Song, W.-Z., Yang, J. and Das, S.K. (2017) Survey of Security Advances in Smart Grid: A Data Driven Approach. *IEEE Communications Surveys & Tutorials*, **19**, 397-422. https://doi.org/10.1109/COMST.2016.2616442

[16] Kostic, T. and Cukalevski, N. (2005) Data Exchange Issues within the Power System Operation and Control Environment. *CIGRE/IEEE PES*, New Orleans, 5-7 October 2005, 290-298.

[17] Stojkovic, B. and Vukasovic, M. (2006) A New SCADA System Design in the Power System of Montenegro—ICCP/TASE.2 and Web-Based Real-Time Electricity Demand Metering Extensions. *IEEE PES*, Atlanta, 2194-2199. https://doi.org/10.1109/PSCE.2006.296282

[18] Mercurio, A., Di Giorgio, A. and Cioci, P. (2009) Open-Source Implementation of Monitoring and Controlling Services for EMS/SCADA Systems by Means of Web Services—IEC 61850 and IEC 61970 Standards. *IEEE Transactions on Power Delivery*, **24**, 1148-1153. https://doi.org/10.1109/TPWRD.2008.2008461

[19] Johnson, M. (2011) IEC 61968-100. Austin, TX, USA. http://www.ucaiug.org/Meetings/Austin2011/Shared%20Documents/CIMug/CIM%20University/IEC%2061968-100%20Implementation%20Profile%20Overview.pdf

[20] Sánchez-López, A., Islas-Pérez, E., Espinosa-Reza, A. and Quintero-Reyes, A. (2015) Deploying SCADA Data to Web Services for Interoperability Purposes. *GIIS*, Guadalajara, 28-30 October 2015, 1-8. https://doi.org/10.1109/GIIS.2015.7347172

[21] Gray, A.D.L., Pisica, I., Taylor, G.A. and Whitehurst, L. (2017) A Standardised Modular Approach for Site SCADA Applications within a Water Utility. *IEEE Access*, **5**, 17177-17187. https://doi.org/10.1109/ACCESS.2017.2654685

[22] Ruh, W.A., Maginnis, F.X. and Brown, W.J. (2001) Basic Building Blocks. In: *Enterprise Application Integration: A Wiley Tech Brief*, Wiley & Sons, New York, 52-57.

[23] Maffei, A., Srinivasan, S., Meola, D., Palmieri, G., Iannelli, L., Holhjem, Ø.H., Marafioti, G., Mathisen, G. and Glielmo, L. (2018) A Cyber-Physical Systems Approach for Implementing the Receding Horizon Optimal Power Flow in Smart Grids. *IEEE Transactions on Sustainable Computing*, **3**, 98-111. https://doi.org/10.1109/TSUSC.2017.2737144

[24] Rozic, B., Mlakar, D., Gruden, M. and Petrovic, N. (2017) Elektro Gorenjska CIM Project. *CIRED*, Glasgow, 12-15 June 2017, 2263-2264. https://doi.org/10.1049/oap-cired.2017.0222

[25] Ascher, D. and Kondzialka, C. (2018) Towards Model-Driven CIM-Based Data Exchange for DSOs. *Energy Informatics*, **1**, 213-224.

https://doi.org/10.1186/s42162-018-0032-4

[26] Goering, A., Meister, J., Lehnhoff, S., Herdt, P., Jung, M. and Rohr, M. (2017) Reference Architecture for Open, Maintainable and Secure Software for the Operation of Energy Networks. 24*th International Conference & Exhibition on Electricity Distribution*, 12-15 June 2017, 1410-1413. https://doi.org/10.1049/oap-cired.2017.0965

[27] Melton, R.B., Schneider, K.P., Lightner, E., Mcdermott, T.E., Sharma, P., Zhang, Y., Ding, F., Vadari, S., Podmore, R., Dubey, A., Wies, R.W. and Stephan, E.G. (2018) Leveraging Standards to Create an Open Platform for the Development of Advanced Distribution Applications. *IEEE Access*, **6**, 37361-37370. https://doi.org/10.1109/ACCESS.2018.2851186

[28] Newman, S. (2015) Microservices. In: *Building Microservices*, O'Reilly Media, Sebastopol, 2-8.

[29] McMorran, A.W., Ault, G.W., Morgan, C., Elders, I.M. and McDonald, J.R. (2006) A Common Information Model (CIM) Toolkit Framework Implemented in Java. *IEEE Transactions on Power Systems*, **21**, 194-201. https://doi.org/10.1109/TPWRS.2005.857846

[30] Requardt, B., Wende von Berg, S., Wagner, T., Toebermann, C. and Braun, M. (2017) Modular System Architecture for Processing of CIM. *ETG*, Bonn, 28-29 November 2017, 1-6.

[31] Schulte, S., Berbner, R., Steinmetz, R. and Uslar, R. (2007) Implementing and Evaluating the Common Information Model in a Relational and RDF-Based Database. *ICSC*, Irvine, 17-19 September 2007, 109-118. https://doi.org/10.1007/978-3-540-71335-7_13

[32] Gómez, F.J., Aguilera Chaves, M., Vanfretti, L. and Olsen, S.H. (2018) Multi-Domain Semantic Information and Physical Behavior Modeling of Power Systems and Gas Turbines Expanding the Common Information Model. *IEEE Access*, **6**, 72663-72674. https://doi.org/10.1109/ACCESS.2018.2882311

[33] Simmons Jr., R. (2004) Data Modeling. In: *Hardcore Java*: *Secrets of the Java Masters*, O'Reilly Media, Sebastopol, 198.

[34] Oracle (2017) The Java™ Tutorials. https://docs.oracle.com/javase/tutorial/reflect/index.html

[35] Wu, J., Cheng, Y. and Schulz, N.N. (2006) Overview of Real-Time Database Management System Design for Power System SCADA System. *IEEE SoutheastCon*, Memphis, 31 March-2 April 2006, 62-66.

[36] Ravikumar, G. and Khaparde, S.A. (2017) A Common Information Model Oriented Graph Database Framework for Power Systems. *IEEE Transactions on Power Systems*, **32**, 2560-2569. https://doi.org/10.1109/TPWRS.2016.2631242

[37] Tomlinson, R.F. (2007) Choose a Logical Data Model. In: *Thinking about GIS*: *Geographic Information System Planning for Managers*, 3rd Edition, ESRI PR, New York, 97.

[38] De, B. (2017) API Authentication and Authorization. In: *API Management*, Apress, Bangalore, 113-114.

[39] Pradeep, Y., Seshuraju, P., Khaparde, S.A. and Joshi, R.K. (2011) CIM-Based Connectivity Model for Bus-Branch Topology Extraction and Exchange. *IEEE Transactions on Smart Grid*, **2**, 244-253. https://doi.org/10.1109/TSG.2011.2109016

[40] Gomez, F.J., Vanfretti, L. and Olsen, S.H. (2018) CIM-Compliant Power System Dynamic Model-to-Model Transformation and Modelica Simulation. *IEEE Transactions on Industrial Informatics*, **14**, 3989-3996.

https://doi.org/10.1109/TII.2017.2785439

[41] Thurner, L., Scheidler, A., Schäfer, F., Menke, J.-H., Dollichon, J., Meier, F., Meinecke, S. and Braun, M. (2018) Pandapower—An Open Source Python Tool for Convenient Modeling, Analysis and Optimization of Electric Power Systems. *IEEE Transactions on Power Systems*, **33**, 6510-6521. https://doi.org/10.1109/TPWRS.2018.2829021

[42] Rodger, J.A. (2015) Discovery of Medical Big Data Analytics: Improving the Prediction of Traumatic Brain Injury Survival Rates by Data Mining Patient Informatics Processing Software Hybrid Hadoop Hive. *Informatics in Medicine Unlocked*, **1**, 17-26. https://doi.org/10.1016/j.imu.2016.01.002