Data Management System for Energy Analytics and its Application to Forecasting

Francesco Fusco
IBM Research Ireland
francfus@ie.ibm.com

Ulrike Fischer
IBM Research Ireland
ufischer@ie.ibm.com

Vincent Lonij
IBM Research Ireland
vincentl@ie.ibm.com

Pascal Pompey
IBM Research Ireland
papompey@ie.ibm.com

Jean-Baptiste Fiot
IBM Research Ireland
jean-baptiste.fiot@ie.ibm.com

Bei Chen
IBM Research Ireland
beichen2@ie.ibm.com

Yiannis Gkoufas
IBM Research Ireland
yiannisg@ie.ibm.com

Mathieu Sinn
IBM Research Ireland
mathsinn@ie.ibm.com

ABSTRACT

The effective management of a power grid with an increasing share of (distributed) renewables and more and more available data, e.g., coming from smart meters, heavily relies on advanced data analytics such as demand and supply forecasting. In this context, data management is one major challenge in electric grids. Large amount of data from multiple heterogeneous sources require transformations, e.g., spatio-temporal alignment or anomaly detection, to serve data analytics tasks and are often applied on different views of the data, e.g., on state, substation or feeder level.

In this paper, the progress on the development of an energy data management systems for the electricity grid is presented. The design of the system was inspired by the real-world use case of forecasting short-term energy demand in Vermont, using data from a combination of SCADA, smart meters and weather forecasting services. A general data model addressing the aforementioned challenges and aimed at supporting advanced data analytics is introduced. The proposed data model views a time series as an abstract concept that might represent raw measurements or arbitrary operations. The benefits of the system is demonstrated for the design and live update energy demand forecasts.

1. INTRODUCTION

The smart grid is the next-generation power system characterized by the inclusion of highly-distributed intelligent devices and communication technologies that manage electricity demand in a sustainable, reliable and economic manner. Advanced data analytics, such as load classification or forecasting, are essential operations for the optimization of the power flow in a smart grid. For the successful application of such operations the reliable processing and management of the data collected by a smart grid is a key factor. Due to the increasing number of diverse devices considerable amount of data is produced by a smart grid, leading to a trend of emerging big data architectures and the discussion of technical challenges as well as potential use cases [1, 2].

Data management is one major challenge in electric grids as data is incomplete in nature, heterogeneous, difficult to merge and arrives at different rates [3]. Energy data arrives from various distributed sources, e.g., supervisory control and data acquisition (SCADA) systems, smart meters, renewable generation systems, and differs significantly in terms of format, resolution and quality. Moreover, data might represent different views of a power system, e.g., load at feeder level or over a whole substation. Analytics tasks, such as demand forecasting, also require contextualisation with additional data sets, such as calendar information and weather forecasts. The latter might come from different weather services, again arriving at different rates and in different time and location resolutions. The system needs to be able to consolidate all these diverse data sets and provide a common view. Analytics are then rarely performed on raw data, but require transformations of the data such as the computation of weighted averages or anomaly detection. The system has to support, store and continuously re-compute such transformation so that analytics can be continuously applied.

In this paper, we explicitly address the challenge of data management in a smart grid, proposing a data management architecture for the smart grid that is, on the one side, able to manage such diverse data sets and, on the other side, supports operations, such as forecasting, that perform various transformation on the available data. To achieve this, we introduce a generic data model for the energy domain and show how this data model can be applied for the use case of short term energy demand forecasting.

There have been other efforts in the design of smart grid architectures. For example, Yang et. al. [4] give a high-level overview for a smart grid big data management system, introducing a simple data model, but mainly discussing data

EnDM '16 Bordeaux, France

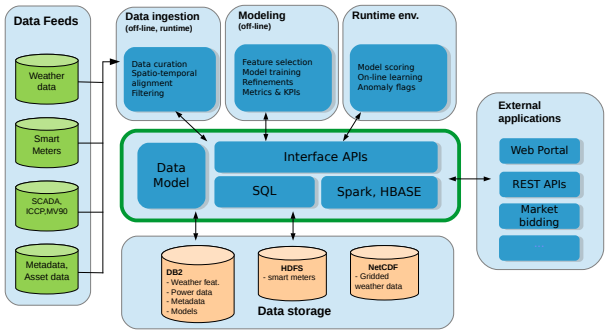


Figure 1: Architecture diagram.

distribution and load balancing. Using cloud computing as a platform for smart grid data management [5] and providing time series analytics as a service [6] poses another trend in this area. A cloud-based demand response platform is introduced in [7], which provides a workflow for data ingestion and scalable demand forecasting on Hadoop. To handle large amount of data and enable real-time reactions data streaming techniques have also been applied for the smart grid context [8]. A real-time data management system was developed within the MIRABEL project with focus on storing and processing special energy planning objects for demand-response [9]. In the specific context of modeling energy data, the representation of time series and flex-offers within the context of distributed energy markets has been studied in [10], while an ontology for big data in the smart grids has been considered in [11], with particular focus on offshore wind farms. The proposed data model specifically addresses the management of dynamic time series data, as well as the operations and analytics applied to them. It can therefore be seen as complementary to the reviewed research efforts and might serve as a basis for other existing designs of smart grid architectures.

In the remainder of the paper a general system architecture supporting data management and analytics tasks for energy utilities is introduced in section 2. The paper then focuses on the data model and query operations at the core of the proposed system in section 3. To demonstrate the usability of our system, in section 4, the use case of short-term energy demand forecasting is discussed and concrete examples on the stored data and query operations are given. Final conclusions and future possibilities are discussed in section 5.

2. SYSTEM ARCHITECTURE

A data management system was developed, conceptually shown in Fig. 1, in order to support general data analytics and services for energy utilities by overcoming the challenges involved in dealing with the ingestion of data from multiple, heterogeneous sources.

The system was developed to support the specific use case of short-term energy forecasting for a number of distribution utilities in Vermont, United States. In particular, the systems provides predictions of hourly energy consumption and distributed solar photovoltaic (PV) generation up to 2 days ahead at various aggregation levels. Time series of energy demand are derived from a combination of thousands of active power measurement points available from SCADA and interval energy data from MV-90. Where required, energy demand or distributed generation is obtained by aggregating data from AMI up to the desired level of the grid (feeder, substation). IBM Deep Thunder [12] was used to obtain weather predictions up to 72 hours ahead. Deep Thunder produces weather forecasts on a grid with 1 km resolution and 10 minute time steps. Forecasts are updated twice per day and each run produces about 300 gigabytes of data. Internally, the data are stored in a combination of relational database management systems (RDBMS) for electrical asset data and SCADA/MV-90, Hadoop distributed file system (HDFS) for the AMI data, network common data format (NetCDF) for the weather gridded data.

As shown in Fig. 1, the data management system supports the tasks of: data ingestion, curation and spatio-temporal alignment of energy and weather data; training forecasting models, which requires retrieving raw data and designing input features (covariates); retrieving data of covariates and scoring forecasting models at runtime; interfaces to client applications (e.g. web portal, market bidding services).

3. THE DATA MODEL

In order to manage highly heterogeneous sets of data and support a variety of analytical operations typical of energy and utilities, as the ones detailed in section 2, a data model was developed. The main objective of the data model was providing a transparent, high-level interface to the users and client applications, as well maintaining consistency, integrity and traceability between the various data sources.

Figure 2 shows a conceptual structure of the proposed data model. All dynamic data in an electricity grid can be represented as a time series. Analogue, operations on the data applied by analytics also produce time series, such as lags or rolling-window forecasts. Consequently, at the core of the proposed data model is the representation of time series, including timestamp/value pairs and abstract operations, as described in section 3.1. The data model also contextualises the time series with respect to a physical quantity (e.g. energy demand, power, temperature) and entity (e.g. substation, service territory), through the concept of signals, as discussed in section 3.2. Finally, the representation of analytical models, which links the output time series of complex operations, such as energy forecasts, to one or more several input time series, is then detailed in section 3.3.

3.1 Time series

At the core of the data model is the abstract concept of a `TimeSeries`, which in general can represent materialized data or abstract operations.

Some examples of specific time series which can be used to represent raw observation data are `TimeSeriesUnstructured`, with values of type `double`, and `TimeSeriesCategoricalIndexed`, with values coming from a set of labels represented as a `CategoricalIndex`. The values of a `TimeSeries` are represented through the concept of `TimeSeriesMaterialization` entity, which points to a `TimeSeriesStore` where the unique pairs timestamp and values, `TimeSeriesMaterializedValues`, are stored. Note that the `TimeSeriesStore` could be one or more tables in the database itself but could also be a different system, for example an Hadoop Distributed File System (HDFS) in the case of very large data

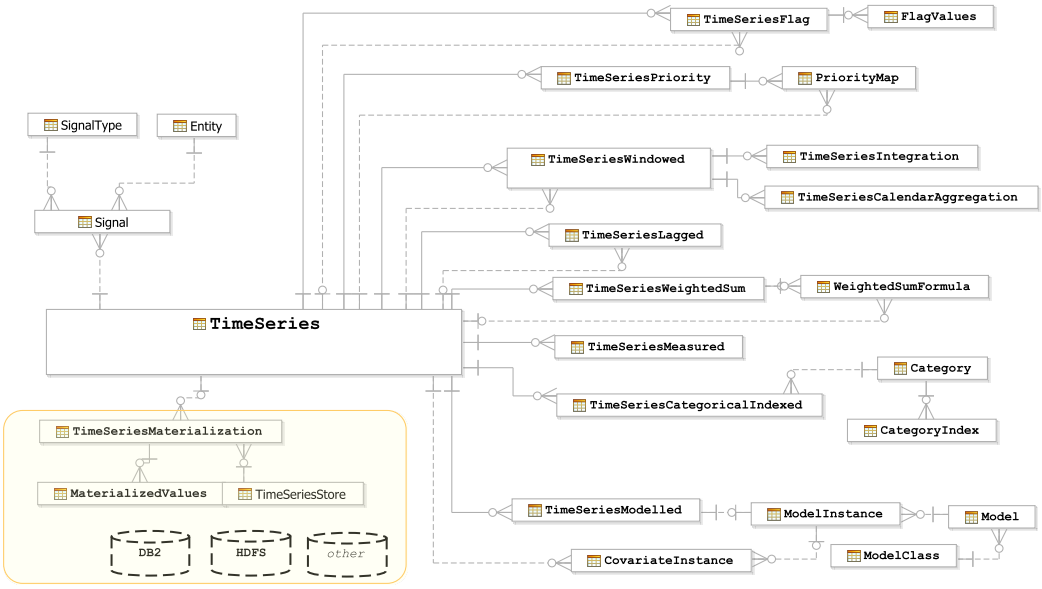


Figure 2: Conceptual diagram of the proposed data model.

sets such as the smart meter data.

Typically the raw time series data are not immediately applicable for analytics purposes and several operations are required for cleaning and spatio-temporal alignment. In the proposed data model, the concept of `TimeSeries` is also used to support and trace such operations, which can be calculated on the fly where possible or in batch processes using the materialization concept. Some examples are:

- `TimeSeriesWeightedSum`: Time series whose value x_t, at time t, is the weighted sum of the value of n input time series y_t^j, $j = 1, \ldots n$, at the same time, namely $x_t = \Sigma_{j=1}^{n} w_j y_t^j$. Typical use cases of weighted sums are spatial aggregations (interpolation of high-resolution weather data, extraction of PV generation in a spatial region, etc.) or applications of power flow equations to derive the electrical load at a substation.

- `TimeSeriesLagged`: Time series whose value x_t, at time t is given as the value of a reference time series y_t at time $t-h$, namely $x_t = y_{t-h}$. Lagged time series are critical for representing delayed dynamical effects in statistical models. Note that the combination of lags and weighted sums can be used to represent quite complex time series models.

- `TimeSeriesWindowed`: Represents a time series with values x_t, at time t, resulting from an operation applied to the values of an input time series y_τ at times within a window $\tau \in [t-\Delta, t]$. Use cases for such an operator are time interpolation and integration, when raw data come at irregular sampling intervals or moving averaging is required (e.g. SCADA gives instantaneous power but we are interested in modelling energy). An equally important use case is the computation of summary statistics of high-resolution data, such as daily minimum/maximum of temperature or energy consumption, which can be quite useful in build forecasting models, as demonstrated in section 4.

Another important type of `TimeSeries` is the

- `TimeSeriesPriority`, internally composed of a sorted map of `TimeSeries` indexed according to a priority order. For a given timestamp, by default, the `TimeSeriesPriority` returns the value from the first `TimeSeries` in the map where a value is available. Alternative behaviors can be implemented, where the value from the `TimeSeries` at a specific index is returned, or from the first `TimeSeries` starting at the index above a certain threshold.
  The main use case of the `TimeSeriesPriority` is dealing with rolling-horizon forecasts, which was the primary application behind the development of the proposed data model, as mentioned in section 2. Rolling-window forecasts, e.g. from weather forecasts produced by numerical models, are multi-step ahead forecasts that are refreshed at regular intervals. Such quantities cannot be represented as a one-dimensional time series because there is a one-to-many relation between timestamps and values. An option could be to overwrite the values with the latest available forecasts, at the loss of potentially valuable information (most recent forecasts are not necessarily more accurate) and traceability of operations between live and batch calculations. By using the `TimeSeriesPriority`, rolling-window forecasts are represented as multiple `TimeSeries` indexed by a quantity proportional to the forecasting horizon h, specifically where each `TimeSeries` is $\hat{x}(t+h|t)$, with h fixed. By default, the most recent forecast is returned, but one could select the value from forecasts at least 24-hour ahead, or many other alternative behaviors could be implemented.
  An alternative use case of the `TimeSeriesPriority` is the fallback mechanism between multiple forecasting

models applied to the same energy signal. If the models are prioritised based on some accuracy measure, the `TimeSeriesPriority` allows to deal with temporary issues in one model (e.g. data anomalies or missing inputs) and to transparently fall back to the next available model output. Such mechanism will be demonstrated in section 4.

Finally, the `TimeSeriesFlag` is also defined, as a mechanism to associate flags to particular data points of a time series, in order to deal with anomalies or data quality issues. Flagging time series points prevents them from being used, for example, as input to analytical models and increase the robustness of the live system, for example in the case of faulty autoregressive model features.

The proposed data model is quite general and can be easily extended with other fundamental types of `TimeSeries`. The listed examples already form the basis for quite a rich grammar able to drive powerful features from the raw data.

3.2 Signals and Entities

One of the main objective of the proposed data model is to provide context to the existing data with respect to high-level physical entities and types of quantities of interest in the specific domain of application. As a result,as also shown in Fig. 2, two main dimensions are utilised in the system for identifying one or more `TimeSeries`:

- `Entity`: represents a physical entity of interest. In the context of energy utilities, for example, we have administrative or geographical entities such as `State`, `DistributionUtility`, `County`, `Town`, and grid assets such as `DistributionSubstation`, `DistributionFeeder`, `NetworkBus`, `NetworkBranch`, `ServicePoint`.

- `SignalType`: represents the type of a physical quantity for which it can be expected to have observation data or for which estimated time series data are expected to be required. Some examples are `TEMPERATURE`, `ENERGY_DEMAND`, `ENERGY_RESIDUAL_DEMAND`, `ENERGY_GENERATION_PV`, `ACTIVE_POWER`. Signal types are the mechanism for cataloguing time series according to some high-level human-understandable meaning, and to maintain consistency between data of the same type, for example with respect to `UnitMeasure`.

The two dimensions of `Entity` and `SignalType` are gathered within the concept of `Signal`, which is a required property of a `TimeSeries`. The proposed constructs allow the data and more abstract time series available within the system to be navigated with very high-level queries of the type:

```
SELECT * FROM TIME_SERIES TS
INNER JOIN SIGNALS S ON S.ID=TS.SIGNAL
INNER JOIN SIGNAL_TYPES ST ON ST.ID=S.STYPE
INNER JOIN ENTITIES E ON E.ID=S.ENTITY
WHERE E.NAME='Substation_name'
      AND ST.NAME='ENERGY_RESIDUAL_DEMAND'.
```

3.3 Models

Another important component of the proposed data model was designed in order to represent the output of statistical models, which relies on yet another type of time series, the `TimeSeriesModelled`. The details of the model are represented through the following entities:

- `ModelClass`: Specifies the structure of the model, in terms of requires set of inputs, as pairs of `SignalType` and variable name, and the `SignalType` of the output.

- `Model`: A realization of a `ModelClass`, with specific values for the parameters and trained to model a specific `Signal` (pair `SignalType`/`Entity`). The parameters are stored using an XML file following the principles of the Predictive Model Markup Language[1] (PMML).

- `ModelInstance`: An instantiation of a `Model`, where each required input, a `CovariateInstance`, is linked to a specific `TimeSeries`.

Such a representation of the analytical models is powerful in supporting the offline tasks of model design and training of the data scientist: the relevant data can be transparently extracted and aligned by relying on queries of the type given in section 3.2; derived features can be designed by using the abstract fundamental operation described in section 3.1. Moreover, when dealing with thousands of statistical models, the system makes it quite easy to navigate between models to verify performance and retrain them where needed.

4. SYSTEM DEMONSTRATION

In this section, the application of our system to short-term forecasting of electricity demand is demonstrated. Due to confidentiality reasons, electricity demand data of Vermont from January 1st, 2012, till August 31st, 2015, was obtained from the website of ISO New England[2]. Those data came in hourly format, with the measurements describing energy usage (in MWh) over the previous hour. When ingesting those data into our database, the time stamps were converted to Eastern Standard Time (EST). Three classes of covariates were used in the forecasting model: weather data, calendar variables, and lagged demand values. Next, the workflow that was used for designing and training forecasting models is described, and the configuration of the system to apply forecasting models in a "live" environment is demonstrated.

4.1 Modelling

For the modeling and forecasting of electricity demand, a popular class of non-linear regression models, which represent the effect of covariates in an additive fashion was used: Generalized Additive Models (GAMs). For more background on GAMs and their application to electricity demand data, we refer to [13]. Note that, in principal, the proposed data model would support any other class of regression or classification methods, as it only describes the inputs and outputs of the models, but not the exact form of the functional relation. The `mgcv` package in `R` was used for training GAMs (see [14]), as part of the following general work-flow:

1. A training data frame is compiled by querying historical electricity demand data and the covariates aligned with it. For the Vermont data, the choice of the covariates had already been defined and implemented through abstract time series in the data model. Since the relation between the model and covariates is also represented in our data model, the compilation of the training data frame could be done fully automatically.

[1] `http://dmg.org/`
[2] `http://iso-ne.com`

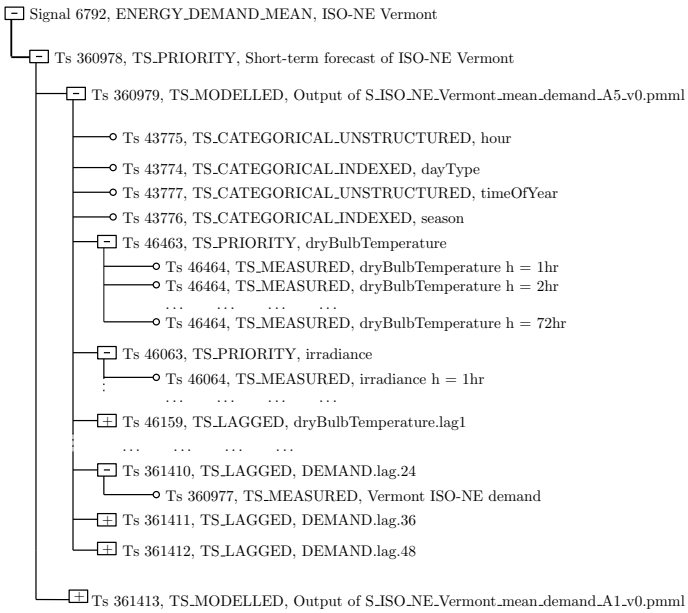


Figure 3: Conceptual relations between the output short term energy forecast and the input time series.

2. Two models were produced: a model using autoregressive features (`S_ISO_NE_Vermont_mean_demand_A5_v0` in Fig. 3), specifically the demand at various lags; a more robust model without autoregressive features (`S_ISO_NE_Vermont_mean_demand_A1_v0` in Fig. 3), which could be used in case of data anomalies. The two models are represented through a `TimeSeriesPriority`, as described in section 3.1, which allows the implementation of a fallback mechanism where the preferred model (the one with autoregressive features) fails to produce a value because of data anomalies.

3. The models were trained on a specified period of time, in-sample and out-of-sample statistics were calculated to assess its accuracy, and finally exported into PMML format, required for model scoring in the live system described in the following section.

4. Besides the GAM models for the conditional mean, which served as forecast of electricity demand, a model for the conditional variance was also trained and exported, which served as forecast of the associated uncertainty. Using the methodology in [15], this model was obtained by fitting a GAM to the squared model residuals in the training data.

Figure 3 shows a conceptual structure of the forecasting models for the mean of energy residual demand, and their links to the input covariates. The one short-term energy forecast time series that a user would see as an output is internally represented by two statistical models applied to around 500 time series each. Note how each input coming from a Deep Thunder forecast is internally represented as a `TimeSeriesPriority` composed of 72 time series.

Note that, in cases where the user wants to design new forecasting model from scratch, the work-flow is more involved. Typically, the user would start by querying historical energy data and raw inputs from which - often in an iterative fashion - new model covariates are derived. Some of the key operations supported by our data model are, e.g., the registration of new predictive models for a given entity, and the linking of new time series to the model covariates.

4.2 Live system operations

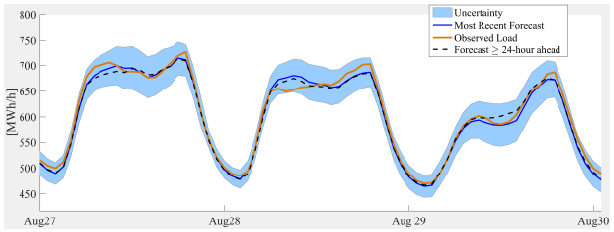
In the live system context, the following work-flow was used for applying the forecasting models to live data:

1. Adapters for automatically ingesting weather forecasts from IBM Deep Thunder, extracting spatio-temporal weather features as described in Section 2, and inserting them into the database were developed.

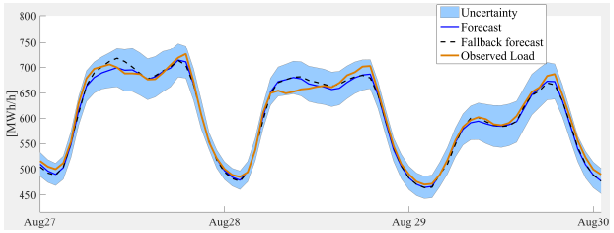
2. Adapters for automatically ingesting live data feeds from SCADA, MV90 and AMI were also developed. Besides being able to display to the user the latest actual measurements, this also helps improving forecasting accuracy, e.g., by using the current electricity demand for predicting the demand in 24 hours from now. Data from SCADA systems is available typically with very low latency (seconds or few minutes); in the case of Vermont, the data from ISO New England becomes available only after a couple of days. However, the scenario where data would be available in real-time was emulated and a 12-,24-,36-hours lagged demand variable were included in the forecasting models. In order to avoid that anomalous values distorts the output of forecasting models, a filter was implemented for flagging such values, such that the system would avoid scoring the corresponding models. In this case, through the fallback mechanism based on the `TimeSeriesPriority`, the system would fall back to the model without lagged variables.

3. Upon the availability of new weather forecasts, a list of timestamps over the 72-hour forecasting horizon was given as input to the model scoring engine of the system. For the implementation of the engine, IBM InfoSphere Streams was used. Basically - for all models registered in the database - the required covariates for the given timestamps are retrieved, the GAM model specified in PMML format is applied, and the forecasts are written back to the database. If covariates are missing for a particular model and timestamp, then a log message is generated and no forecast is produced.

Figure 4(a) shows the forecasts for August 27th-29th, 2015, based on models that were trained with data from January 1st, 2012 till July 31st, 2015. The graph also displays uncertainty bands obtained from the conditional variance forecasts. Note that the forecasts of the conditional mean are based on a model which uses 12-,24-,36-hours lagged demand values. The graph also shows, in a dotted line the less recent forecasts, $>$ 24-hours ahead, which are also produced by the system and can easily be retrieved using the estimation horizon and the concept of `TimeSeriesPriority`. Figure 4(b) illustrates the fall-back mechanism in the case of missing inputs: the solid line shows the forecasts from a model with 24-hours lagged demand values; the dashed line corresponds to the forecasts from a fall-back model without lagged values. In the case where real-time demand information is missing or anomalous (and hence flagged at data ingestion data), our system would automatically return the output of the fall-back model, rather than not providing any



(a)



(b)

Figure 4: Illustration of forecasts.

forecasts for those instances at all. If real-time information is available, it will return the forecasts from the model with lagged demand values, which are more accurate in general.

5. CONCLUSION AND FUTURE WORK

A data and analytics management system for energy utilities was described. In particular, focus was put on the design of the core data model required for providing a transparent, high-level interface to the users of the data and the client applications. The data model was also designed for maintaining consistency, integrity and traceability between the various complex data sources relevant to energy utilities, particularly energy and weather data.

An implementation of the proposed data model was demonstrated in the context of a short-term energy forecasting system, particularly in support of the model training/deployment tasks and in the system live operations. Further applications and extensions can be considered along the direction of analytical model management, automation of model (re)-training and support for model feature design. Further research will also go in the direction of a more formal study of the time series grammar and its potential in support many other use cases. The Big Data aspect of the data was not discussed, but the definition of a hybrid architecture where data are stored in a mix between traditional RDBMS and HDFS is also scope for further study.

6. REFERENCES

[1] M. Aiello and G. A. Pagani, "The Smart Grid's Data Generating Potentials," *Proceedings of the Federated Conference on Computer Science and Information Systems*, vol. 2, pp. 9–16, 2014.

[2] P. D. Diamantoulakis, V. M. Kapinas, and G. K. Karagiannidis, "Big Data Analytics for Dynamic Energy Management in Smart Grids," *Big Data Research*, vol. 2, no. 3, pp. 94 – 101, 2015.

[3] N. Yu, S. Shah, R. Johnson, R. Sherick, M. Hong, M. Ieee, and K. Loparo, "Big Data Analytics in Power Distribution Systems," in *Proceedings of the Innovative Smart Grid Technologies Conference*, Washington, DC, USA, 2015.

[4] Z. Yang, Q. Zhou, A. G. Ma, P. X. Cheng, and Y. Gao, "The Design and Implementation of Smart Grid High Volume Data Management Platform Architecture," in *Proc. of the Innovative Smart Grid Technologies Conf.*, Washington, DC, USA, 2014.

[5] S. Rusitschka, K. Eger, and C. Gerdes, "Smart Grid Data Cloud: A Model for Utilizing Cloud Computing in the Smart Grid Domain," *First IEEE Int. Conf. on Smart Grid Communications*, pp. 483–488, 2010.

[6] Y. C. Xiaomin Xu, Sheng Huang and K. Browny, "TSAaaS: Time Series Analytics as a Service on IoT," in *Proc. of the IEEE Int. Conf. on Web Services (ICWS)*, Alaska, USA, 2014, pp. 249–256.

[7] Y. Simmhan, S. Aman, A. G. Kumbhare, R. Liu, S. Stevens, Q. Zhou, and V. K. Prasanna, "Cloud-Based Software Platform for Big Data Analytics in Smart Grids," *Computing in Science and Engineering*, vol. 15, no. 4, pp. 38–47, 2013.

[8] M. Couceiro, R. Ferrando, D. Manzano, and L. Lafuente, "Stream analytics for utilities. Predicting power supply and demand in a smart grid," *Proceedings of the 3rd International Workshop on Cognitive Information Processing (CIP)*, 2012.

[9] U. Fischer, D. Kaulakiene, M. E. Khalefa, W. Lehner, T. B. Pedersen, L. Šikšnys, and C. Thomsen, "Real-Time Business Intelligence in the MIRABEL Smart Grid System," in *Enabling Real-Time Business Intelligence*, ser. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, 2013, vol. 154, pp. 1–22.

[10] L. Šikšnys, C. Thomsen, and T. B. Pedersen, "MIRABEL DW: Managing Complex Energy Data in a Smart Grid," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7448, pp. 443–457.

[11] T. Nguyen, V. Nunavath, and A. Prinz, "Big Data Metadata Management in Smart Grids," in *Big Data and Internet of Things: A Roadmap for Smart Environments*, ser. Studies in Computational Intelligence. Springer International Publishing, 2014, vol. 546, pp. 189–214.

[12] L. A. Treinish, A. Praino, H. Li, E. Novakovskaia, J. Drexel, R. Derech, and B. Hertell, "Operational Evaluation of a Meso-Scale Weather and Outage Prediction Service for Electric Utility Operations," in *First Conference on Weather, Climate, and the New Energy Economy*, 2010.

[13] A. Ba, M. Sinn, Y. Goude, and P. Pompey, "Adaptive Learning of Smoothing Functions : Application to Electricity Load Forecasting," in *Proceedings of the Neural Information Processing Systems (NIPS) Conference*, Lake Tahoe, Nevada, USA, 2012.

[14] S. Wood, *Generalized Additive Models: An Introduction with R*. CRC Press, 2006.

[15] T. K. Wijaya, M. Sinn, and B. Chen, "Forecasting Uncertainty in Electricity Demand," *AAAI-15 Workshop on Computational Sustainability*, 2015.