Research on Multi-source Data Integration in Data Platform of Power Distribution and Utilization

Shanyi Xie¹, Hui Xu¹, Hui Cai¹, Yilong Yu²

¹ Information Center of Guangdong Power Grid Co., Ltd, Guangzhou 510080, China
² Weihai CIMSTech Co., Ltd, Weihai 264209, China

Abstract. This paper presents a design for integrating the data of multiple systems of power distribution and utilization into a platform system. Two ways of using source data to build power grid model, based on object tree or object classification, are described in detail. With the help of path name, association, and normative name, the data of the source system and the existing data of the platform are matched and mapped, the data from multiple source systems are integrated together, and the code mapping between the systems is constructed. Cleaning and standardizing the data in the process of data collection, data problem will be submitted to the maintenance staff to repair, so the quality of the data into the platform is strictly controlled. Record key data in the process of data collection makes the source and changes traceable. The data collection system implemented in this design can improve data quality, facilitate manual correction and make the data collection process controllable.

1. Introduction

The power company's multi-year power distribution, power automation and information system construction has increased the coverage and monitoring level of distribution automation, and the data in the distribution grid has seen an explosive growth. Intelligent analysis and control of distribution and utilization power grid requires the comprehensive utilization of data from multiple systems [1]. However, the information models, object coding, and data publish methods adopted by each system development company are different. Data splicing and integration are time-consuming and labour-intensive, which restricts the development of advanced applications for distribution and utilization grid.

By constructing a standardized, integrated distribution and utilization grid data management and control platform that integrates grid model, asset, environment, fault, defect, and test data, it can effectively support comprehensive analysis applications [2]. The integration and normalization of data from multiple source systems is the largest part of platform construction workload, and it is also the key to whether the platform data quality can effectively support the application.

This paper studies the integration of multi-source data of distribution and utilization grid data management and control platform. Data integration is based on the unified information model of distribution and utilization grid, which is constructed from IEC TC57 Common Information Model (CIM) [3]. The integration is carried out in two major contexts: data classification and data tree level hierarchy. The task scheduling framework module coordinates multiple data splicing integration tasks. Process data of source data conversion and splicing are recorded in detail to support data source tracing. The problem of data integration process and the quality problem of source end data are quickly discovered by tracing the source, so that the data integration process of the data platform is controllable, and at the same time it can help the source system to improve the data quality.
2. Data analysis of distribution and utilization grid

Distribution and utilization power grid data management platform is used to integrate data from different systems, including: (1) distribution and utilization power grid model (often called Function Location part, mainly refers to equipment container, equipment, measurement, topology, etc.); (2) asset information and technical parameters; (3) customer data and metering value; (4) weather forecast and weather report; (5) real-time and historical data; (6) monitoring data; (7) object location data, etc. These data are modeled by CIM's existing schema elements and necessary extensions, and are published through standardized services, which allows an application system to acquire all dimensions of power grid data from one stop.

All kinds of data use the functional location model as the core related target, and they form a whole by coupling the relationship between the objects and the dependencies between the data [4]. Data objects establish connections through associations, and data access can navigate through these associations.

In the power distribution and utilization model, most objects have one or more paths that are directly or indirectly associated with a geographical region. If the "geographical region" object is considered to be the top level, and the other objects are stratified according to the minimum length of their arrival through the associated path to the geographical region, the tree (vertical) hierarchy of the grid model is formed.

The tree type structure is mainly composed of geographical region, substation, line (or feeder), equipment container, equipment and other functional location objects, and related measurement, equipment terminal and connectivity node in the equipment container. The geographical region is the root of the tree (upper-right of figure 1).

![Figure 1. Data objects and relationship.](image)

Assets, monitoring devices, and other objects, through N to 1 or N to N associations directly or indirectly linked to node (object) in tree hierarchy, formed a network hierarchy model of object association (lower right of figure 1).

The analysis of distribution and utilization power grid requires that the data has a complete and accurate tree hierarchy and mesh correlation and incorporates a variety of dynamic data (operation, monitoring, test, weather, etc.). The purpose of splicing and integrating multi-source data is to construct this complete set of standardized data.
3. Data normalization and collection
The data extraction, processing, reorganizing, standardization and splicing of different source systems are called data normalization and collection (integration). The data collection is not only to copy, but also to cleanse and standardize the source data to ensure the data which entered the platform conforming information model and correct. Thus data error accumulating and amplifying along data flow chain growing can be avoided.

The data collection standardization is mainly based on the CIM schema of the distribution and utilization power grid to organize model data, adopting a consistent expression format for the values of the same type of attributes, limiting the enumerable attributes to enumeration types, using standard, reasonable and consistent unit to describe data, and to constrain numerical precision.

One data collection task can be decomposed into three phases (figure 2).

![Diagram of data collection process](image)

Figure 2. Process of data collection.

In stage 1, read data of the source system, and then organize and standardize the data of the source system according to the schema of the platform.

In stage 2, divide the data into three parts: (1) Model data that contains a complete hierarchical information and can be integrated into the platform; (2) Data which accuracy is doubtful or association is incomplete and need to be corrected manually; (3) Isolated data object group which is unable to attach by hierarchy because of the data in part (2) need to be solved firstly. The first part data can be imported into the platform directly. The second and third part data need to be stored separately for further processing.

In stage 3, the maintenance personnel manually maintain second part data, these data can be amended or deleted. Combine the corrected data in part (2) with data in part (3) and move to stage 2.

Repeat the above steps until all the data is imported into the platform or the whole remaining data discarded by user.

The three stages can be decomposed into two data collection tasks. The first task includes stage 1 and stage 2. The second task includes stage 2 and stage 3. These two tasks and the data correction tool work together to complete data import.

4. Software modules
Software modules and functions to perform data collection is shown in figure 3.
The entire data collection system includes one or more task scheduling services, one task monitoring support service, and the client API for task monitoring support service. Notification of module registration and changes is provided by the underlying network services of the data platform system.

A task scheduling service manages one or more data collection tasks. Because there are many kinds of data access tasks, placing multiple tasks with similar functions and scheduling tasks that are staggered with each other not only facilitates management, but also makes the memory requirements relatively stable.

There is only one task monitoring support service. It is the management centre of all tasks, through its client API, the monitoring client can query task status, execution plan, execution progress, and log information, and can start and stop tasks.

Task monitoring support service is used to integrate, centrally record, and store valuable data (such as the evolution records of objects, ID mappings of source system objects to platform objects, and tracking information indicating which system is the source for a certain type, object or attribute) generated during the data collection process, and provide query service to response client operation.

5. Model building

The source system that provides the grid model, can provide data through mirror database, data file, Web Services or the CIS interface, etc. After acquiring the data of the source system, the platform data collection software module adopts a type-construction method and a tree-style construction method to perform data normalization processing.

A model object is not directly built on a single source data. First, a “draft” (temporary) object is generated according to the object of a data source. Then, combined with other sources, the high-quality property values are deduced based on object path, voltage level attributes, etc., and redundant and illegal data are removed, and the error data are corrected. When the multi-source data processing is complete, the temporary object set is ready. Then, temporary objects are compared with existing data objects in the platform to generate incremental update data. And incremental data is submitted to the model server to complete the model update.

5.1. Construction according to type

According to the target CIM type, the object data is filtered from the source data, and is imported into the platform after standardization. Data objects need to be imported in sequence according to the hierarchical structure shown in figure 1, "from top to bottom", and the relationship construction follows the principle that "lower objects is linked to the upper object, 'multi end' is associated to 'less end'".
Table 1 Methods of association update.

| Objects and relationship example | Update Method | Execution order |
|----------------------------------|---------------|-----------------|
|                                  | Method-1      | Update_1{…,1-2,1-3} |
|                                  |               | Update_2{…}     |
|                                  |               | Update_3{…}     |
|                                  | Method-2      | Update_1{…}     |
|                                  |               | Update_2{…,2-1} |
|                                  |               | Update_3{…,3-1} |

Table 1 gives a brief description of the update order in different updating methods. The "..." in the sequence of parentheses representing object properties, "1-2", "1-3" and so on, representing object associations. Uses Method-2 to establish or modify "one-to-many", "many-to-many" associations. In table 1, there is a one-to-many association between class A and B. Updates that establish 1-2 and 1-3 associations are placed in object updates 2 and 3 to facilitate model comparison, generate effective updates, and record object evolution information more efficiently.

The process of import data of one class is mainly composed of three parts: querying source data, constructing objects one by one, and submitting all temporary objects. There are three extension points in the processing program, through which additional processing program can be hooked: (1) before querying source data. At this point, we can cache data and modify dynamic filtering conditions; (2) after an object is read. Object path name can be modified and cache related object; (3) after temporary objects of one class are built up. We can analyse the objects as a whole, delete redundant and illegal objects, and release the cache.

5.2. Construction according to object tree

Use following process to construct standardized object according to object tree: (1) The source model data is organized into an efficient traversal tree according to its original level; (2) Source object tree is mapped to platform object hierarchy; (3) The depth first algorithm is used to traverse the data tree of the source model and extract the data needed by the platform.

Using object tree, when a source object is processed, not only the information of its type ID, but also the hierarchical information, can be accurately mapped to the type of platform object. And the information of the ancestor or descendant nodes of the source object can be easily obtained as the attribute of the target object. The path name of the object that the object match mainly relies on can be naturally formed based on the object tree.

Because the branches of the object tree are relatively independent, the depth-first algorithm can also reduce memory resource consumption. When a node and all its descendants are processed, these nodes are released. With the progress of the processing, the source model tree is shrinking and the memory footprint is decreasing.

5.3. Model splicing

If different source systems adopt a unified coding specification and have the same code for the same object described, then model splicing can be directly based on the object coding. However, the more common situation is that different source systems have their own coding specification, which causes data objects cannot be matched by object code. In this case, the relationship between objects and the normalized name must be used to deduce the unknown object matching relationship. Using path names to match is the most effective way.

The path name is a system unique name composed from all node names in a naming (tree) hierarchy path from the object to the root. The path name uses the normalized name of each node. The normalized name is processed according to the source data using the established naming convention.
The model data of different source systems may have some deviations for the name of the same object. For example, for the same substation, some may take the highest voltage level of this station in front of the substation, and some do not take it, such as "south suburb substation", "220kV south suburb station", "220kV south suburb substation" and so on. Only the path name constructed from normalization names can be used between homologous systems as a basis for matching. The name normalization is based on naming rules defined by power grid corporations [5].

The model data that cannot be matched after normalization may be problematic data. Maintenance personnel should analyse and correct the data or improve the mapping rules.

In summary, the model splicing is based on object path to realize object matching, and then completes attributes merging according to the configured source-platform model mapping table, and then complements the object associations.

6. Data source tracing
The distribution and utilization power grid model data is spliced and integrated from multiple source systems, including the following cases: (1) objects of different CIM types are derived from different source systems; (2) objects of the same CIM type are created from different source systems; (3) some attributes of certain CIM type comes from specific system; (4) attributes of the CIM type are selected from multiple source systems according to data quality scoring algorithm.

Data traceability information is created to track which system the object and attributes come from and what transformations the current values have taken. Users and developers can use data tracing to grasp the source and evolution of data, analyse suspect data, and help repairing data.

The information contained in an object's evolution record is shown in table 2.

Table 2. Evolution record of one object.

| Creation time of the object | Name of the source system on which the object was created | Name of tool or data collection task which created this object | Object status (Active or Deleted) | Name of tool or data collection task which deleted this object (null if object status is ‘Active’) |
|-----------------------------|----------------------------------------------------------|-------------------------------------------------------------|---------------------------------|-------------------------------------------------------------------------------------|
|                             |                                                          |                                                              |                                 |                                                                                      |
|                             |                                                          |                                                              |                                 |                                                                                      |
|                             |                                                          |                                                              |                                 |                                                                                      |
|                             |                                                          |                                                              |                                 |                                                                                      |
|                             |                                                          |                                                              |                                 |                                                                                      |
|                             |                                                          |                                                              |                                 |                                                                                      |

Editing and operating history of the object (use following stack to store)

| Operation type | Name of tool or data collection task | Time of operation |
|----------------|--------------------------------------|-------------------|
|                |                                      |                   |
|                |                                      |                   |
|                |                                      |                   |
|                |                                      |                   |
|                |                                      |                   |
|                |                                      |                   |
|                |                                      |                   |
|                |                                      |                   |
|                |                                      |                   |

(There are three types of operation: create, edit, delete)

Name of attribute-1
[association-1]

Name of source system
Name of tool or data collection task
Snapshot of source data
The original value [original associated object code] in platform
Value (written in platform) [newly added associated object code ]
[Deleted associated object code ]
Operation type
Timestamp

… (other attributes and associations)

a Association specified content is marked in square brackets.

The object's evolution record data is stored in HBase [6]. Use platform object code to query evolution record when needed.

7. Conclusion
According to the characteristics of the data objects used in power grid of distribution and utilization, this research adopts the combination of classification and object tree construction, and uses the
collaboration of multiple data tasks to organically splice and integrate multi source data together to form standardized platform data. Store process data of model object transformation and splicing, and support source analysis of platform data.

The research results have been applied to the construction of standardized data platform based on data from existing business systems, and the correctness and effectiveness of data collection, model splicing and data tracing design has been validated.

References
[1] Teng Zhao, Yan Zhang, Dongxia Zhang 2014 Application technology of big data in smart distribution grid and its prospect analysis. *Power System Technology* 38(12) 3305-12.
[2] Ran Hao, Qian Ai, Fei Xiao 2017 Architecture based on multivariate big data platform for analysing electricity consumption behaviour. *Electric Power Automation Equipment* 37(8) 20-27.
[3] IEC 2009 IEC 61970 energy management system application program interface (EMS-API) - part 301: common information model (CIM) base 2nd Edition.
[4] Zonghua Li, Xiaofeng Zhou, Aihua Gu, Qinfeng Li 2015 A complete approach for CIM modelling and model formalising. *Information and Software Technology*, 65 39-55.
[5] Wenchao Li 2014 Implementation and application of coding standard for dispatching data objects. *Guangdong Electric Power* 27(3) 80-83.
[6] Fuzheng Zhang, Wenqi Huang, Huajun Chen, Xiaobin Guo, Chengzhi Chen 2017 Research on HBase-based massive time series data storage for distribution grid. *Modern Electronics Technique* 40(13) 159-163.