Vertical software controllable system design for grid service operation process

ZHANG Jinbo¹, SHEN Wuqiang ², SHEN Guiquan ³

¹ Information Center Guangdong Power Grid Company Limited
Guangzhou, China
² Information Center Guangdong Power Grid Company Limited
Guangzhou, China
³ Information Center Guangdong Power Grid Company Limited
Guangzhou, China

*Corresponding author’s shenwuqiang@gdxx.csg.cn

Abstract: The traditional software control system with partition management is applied to the grid service operation process with more complex data structure, which will lead to slow system response due to high concurrency pressure. In view of the above problems, a vertical software controllable system is designed for the grid service operation process. After designing the hardware part of the two systems, grid server management board and communication module, the system software is designed. In the system software part, the grid service operation functional components are designed to realize the modular management of the service operation. The real-time and relational database of the system is designed, and the keyword query algorithm is used to realize the business item query to realize the system functions. The simulation experiment results show that the response time of the controllable system is all below 2000ms, which meets the demand of grid service operation, and the system has good concurrency resistance.

1. Introduction
With the increase of users in different industries and the difference in requirements for various services, users are flooded with a huge amount of various information in the process of using software, which provides ample space for the development of specialized and segmented network platforms and network information services. Under the power market, the basic function of power grid enterprises is to provide reliable and high-quality transmission and distribution services for power generation and users without discrimination as an independent third party. Power grid enterprises have always attached great importance to the promotion of management information technology for enterprise development, and actively explore the road of information technology in line with enterprise strategy [1]. A large amount of data will be generated in the process of grid service operation, and because these data contain a large amount of professional information, professional data management software is required, while comprehensive service operation software has been unable to meet the actual demand of grid service operation. Vertical software, which specializes in one or several specialized fields, has gradually replaced the traditional integrated software and won the time for development [2]. In the process of grid service operation, vertical software aids in data, project management, and information recall with good results.
In practice, the network environment of power grid enterprises is complicated, the most prominent is the information system management problems, regular inspection type inspection and maintenance has been unable to meet the needs of the entire network information system good operation and maintenance. Therefore, when the vertical software used for data management in the process of power grid operation is managed in a controlled manner, it is one of the ways to ensure the normal operation of power enterprises and provide services. Literature [3] traditional software controllable system uses distributed monitoring approach to realize the control of operation service data management software according to the data source area. This method is better for the management of management software with small data volume and poor data heterogeneity, but for the data integration and heterogeneity of data management software caused by the operation of modern power grid services, the system cannot guarantee real-time control is difficult to meet the needs of power enterprises to provide operational services. The software controllable system mentioned in the literature [4] is based on spring's front-end framework to control the grid service operation management software, which runs on a container server that is easy to deploy and provides rich functionality, but the development process is tedious and repetitive tasks, and a large number of source code files have to be created and maintained, which is less effective for vertical software control and has limitations in actual use. Based on the above analysis content, in order to improve the control efficiency of the data management vertical software in the process of grid service operation, this paper will study and design the vertical software controllable system for the process of grid service operation.

2. Hardware design of vertical software controllable system for grid service operation process

2.1. Power grid server management motherboard design

A large amount of data is required to support the operation of the grid service, all data is stored in the database, and for effective management of the grid server, the server management motherboard is designed. Figure 1 below shows a basic motherboard management controller architecture [5].

![Server management motherboard architecture](image)

Figure 1 Server management motherboard architecture

The motherboard management controller is a server management processor with an internal integrated RAM processor that uses internal physical variables from physical sensors to monitor the status of the grid server or other external devices, supporting both local in-band access and remote out-of-band access. Temperature and voltage sensors and power monitoring are connected via the I2C bus, and the motherboard management controller monitors system status by measuring internal physical variables. Server administrators can manage the server locally and remotely using the motherboard management controller [6].
The motherboard management controller provides an efficient and convenient server management interface and has a very mature ecosystem, plus its management capabilities are also recognized by a wide range of data center customers, and its general main components are shown in table 1 [7].

| Serial number | Hardware equipment and composition | Function Description |
|---------------|-----------------------------------|---------------------|
| 1             | Integrated 400MHz 32-bit ARM9 processor with memory processing unit and 16K cache support | Master management control program operation |
| 2             | Frequency 200MHz 32-bit co-processor | Control system data processing, storage |
| 3             | DDR2/3 16bits 800 MHz Memory Interface | Memory Access |
| 4             | 8 independent pulse width modulators, 16 voltage monitoring | Control signal modulation and system voltage monitoring |
| 5             | Two independent 10/100/1000 Mbps network processor interfaces | Supports Internet access at different network speeds |
| 6             | 16 registers for system and motherboard management controller communication | Temporary storage of instructions, data and addresses |
| 7             | 2 independent watchdog timers, 6 general-purpose timers | In the event of a software failure, the microcontroller is reset by resetting the device (if the software does not zero out the device) to protect system operation in a timely manner |

The main board management control module is connected to the communication module via I2C bus, and the communication module is connected to the grid service operation terminal to realize the communication between the grid service operation server and the vertical software control system.

2.2. Communication module design

Due to the need to achieve real-time control of the vertical software when operating the grid service, the advantage of higher I2C bus communication speed can satisfy the communication connection between the control system and the vertical software. The I2C bus connection between the main board management control module and the communication module is made using a shielded twisted pair cable with RJ45 interface [8].

The communication module uses Socket communication protocol for data transfer. Socket is a communication protocol and a socket consists of five main pieces of information: protocol, local address, local port number, remote address, and remote port number. It is a protocol to communicate with any computer that has a socket interface and all messages are sent and received through this socket interface [9-10]. The operation of a socket is simple and in application development it is like using a file handle, you can read and write to the socket handle. A client uses a Socket object to send a connection request to a specified port of a server on the network, and if the connection is successful, the session is opened. The client does not need to specify an open port, the operating system usually assigns a temporary, dynamic port of 1024 or more. The server uses the Server Socket object to listen to any specified port and wait for a client connection request. When the client connects successfully, a session is generated; when the client completes the session, the connection is closed.

Based on the above design of the hardware part of the vertical software controllable system, the software part of the vertical software controllable system is designed to meet the requirements of the grid service operation process.

3. Software design of vertical software controllable system for grid service operation process

3.1. Design of functional components for grid service operation

During the system design process, in order to realize the vertical management of each service
operation project of the power grid, it is necessary to divide each specific application function of the control system into interface control components, business logic components, etc., and all business application functions are decomposed into basic units that conform to the change business needs through the combined component technology [11].

The project management component of the vertical software controllable system is mainly for the management of power grid service operation projects. Since power companies often provide multiple operation services at the same time, it is important to manage the service operation project information through the functional component. The project manager is able to manage the information of each service project through the functional component and enter the latest information of the project into the system through the project data collection. The project data analysis function can display the project data in the form of graphs and charts, which makes it easier to visualize the current status of the project. The project information query function is a channel for the user to query information about a given project [12].

The timing diagram of the functional components of the grid service operation module is shown in figure 2 [13].

Grid service operation staff enter the main page of the system through their account and password, click on the project management module to enter the project management function interface, click on the grid service project information management function to obtain detailed information of the selected project, and can add, modify and delete the project information and other operations. Once the information is entered, it can be analyzed and processed by the project information analysis function, which can be in the form of tables or graphs. This makes it easier for grid service operations staff to visualize the status of the project [14]. In addition, the power grid service operation staff can query the information and progress of the service project through the project information query function of the functional component module.

The project manager analyzes the results of the comparison and controls the project progress through the project progress control function. The project manager analyzes the comparison results and controls the project schedule through the project schedule control function, which enables the project to proceed smoothly and improves the efficiency of the grid service operation [15]. After designing the grid service operation functional components of the vertical software controllable system, the data storage database of the system is designed.
3.2. Database Design
The database of the vertical software controllable system uses a combination of real-time database and relational database, using a real-time library to meet the demand for providing fast real-time data access, while introducing a relational database, which organically combines the real-time database with the relational database. The high reliability, versatility and scalability of the relational database are fully utilized, while the high speed of the real-time database is fully utilized [16].

3.2.1. Real-time database
The real-time database is stored in the physical hard disk in the form of files, including three levels: state, application, and table. The whole real-time database corresponds to a directory on the hard disk, i.e. the real-time library root directory. The real-time database includes several vertical software control states, each corresponding to a subdirectory under the root directory of the real-time database, i.e., the state directory. Each state contains several applications, which correspond to a subdirectory under the vertical software control state directory, i.e., the application directory, and each application corresponds to a binary file under the application directory for storing control data in real time.

The real-time database is specifically designed to provide efficient access to real-time data to monitor, control and analyze the development of each service operation in the course of grid service operation. In the grid service vertical software controllable system, all applications with high requirements for real-time need to be constructed on top of the real-time library, which is also the basis for data interaction between applications and platforms and between applications and applications [17]. Real-time databases support multiple applications and polymorphism and provide various access interfaces, including network interfaces and local interfaces. The main functions of the real-time database include: data download, data storage, data maintenance, data validation, data browsing, data access, data synchronization, data replication, etc.

3.2.2. Relational Database
The relational database is mainly used to store all data that needs to be stored permanently, such as grid devices, parameters, alarm and event records, system configuration, historical statistics, etc. The relational database data management function includes two parts: model data management and historical data management. The relational database data management function provides a set of database access interfaces, i.e., database middleware, which makes it easy for various applications and public services, etc., to access the data in the relational database [18].

Cluster and parallel processing technologies are used to ensure secure, reliable, and fast database access with openness, real-time, and security. The data storage of the historical database is equipped with a caching mechanism to guarantee that the temporarily lost historical data can be automatically recovered from the SCADA server when the historical database server is out of operation for a short period of time.

After establishing the vertical software controllable system database according to the above, keyword query algorithm is introduced in the system for the convenience of front-end grid service operation staff operation.

3.3. Service business data keyword query
An \( n \times n \) grid is used to partition the data space in the database, and for a given data set \( R \), assuming its dimension to be \( d \), base to be \( c \), and ideally the number of individual interval points to be \( l \), the specific steps to solve for \( n \) are given below. The data space is partitioned in the order of the values of \( n \) from 2 iterations to \( n' = \sqrt[3]{c} \), and for each particular value of \( n \), a local bit string is generated for that scratch task input data slice. After dividing the data by global operations, the specific \( n \) value is found. The individual instances in the dataset are distributed in grid intervals, then the keywords contained in each instance also have their subordinate grid intervals. For a given
keyword, this paper uses the grid inverted index to quickly locate the grid interval to which it belongs [19].

For a given dataset $R$, a grid of $n \times n$ is used to partition the data space, where the value of $n$ is found by the above procedure. First, the data set $R$ is divided into mutually disjoint data slices $R_1, R_2, \ldots, R_m$, which are used as inputs for keyword queries. The local inverted index of each data slice is generated according to the data partitioning mechanism and Equation (1).

$$iibs[i] = \begin{cases} 1, & \exists t \in p_i \land w \in t.W \\ 0, & \text{else} \end{cases}$$

(1)

In the above equation, $t$ is the data space division node. $p_i$ is the $i$-th interval into which the data are divided. $w$ is the keyword; $W$ is the set of keywords. In this paper, the approximate string matching algorithm is used to calculate the degree of matching between the keywords and the elements in the query items. The approximation degree is calculated as follows [20].

$$sim(x, y) = \sqrt{\sum_{k=1}^{n} \left(\frac{x - y}{s_{xy}}\right)^2}$$

(2)

In the above equation, $x$ is the keyword submitted by the system user; $y$ is the query item summarized in the system database. $s_{xy}$ is the standard deviation of the two data elements. A match is given for each keyword based on the approximate string matching algorithm, and if the element and keyword match above a given threshold, it can be identified as a possible match. After the matching algorithm, a series of possible matches are generated for each keyword, forming a match table. By forking each keyword match table, a series of query terms are generated. The work required before submitting the final query results to the user is to rank the relevance of the query templates obtained from the keyword aggregation query.

The score of a query template is determined by a combination of node weights and edge weights. The scoring function assigns a score to each query template that measures the relevance of the query results to the query keywords, with the most relevant results being returned first. The sorting function is based on the scoring function. The sorting function ranks the query templates in a certain order based on this score. It is assumed that the nodes in the query template are all corresponding to the query keywords. The number of keywords contained in the query results varies, and the higher the keyword match, the higher the score should be according to the query keyword match. The improved scoring formula is shown in the following equation.

$$S(QT, Q) = \frac{1}{sif(QT)} \left(\frac{m}{m'}\right)^a \sum_{i=1}^{sif(QT)} S(T_i, k_i)$$

(3)

In equation (3), $QT$ is a query template that constitutes a query. $sif$ is the number of nodes contained in the query template. $T_i$ is the node in the query template. $Q$ is a keyword query and $m$ is the number of keywords in $Q$. $m'$ is the number of nodes contained in the query template. $a$ is a constant used to ensure that queries containing more keywords have a higher score than those containing fewer keywords. $S(T_i, k_i)$ is the match between a given query keyword and the corresponding node keyword. In this paper, the value of $S(T_i, k_i)$ is normalized in the range of $[0,1]$, where 0 means no match and 1 means perfect match. The keyword query result items are displayed and output to the front-end of the operation according to the descending order of the scoring results.
Through the above design process of system hardware and software, the design of a vertically software controllable system for grid service operation process is completed.

4. System Experiment

The purpose of software testing is to check whether the system meets the standards, assess the software quality, and test whether the system is functionally complete, available, and reliable, and whether the interfaces are smooth. This section focuses on functional testing, stress testing, and rectification of test results/problems after the completion of the vertical software controllable system for grid service operation process.

4.1. Experimental Content

The purpose of performance testing is to check whether the capacity and concurrency of the system meet the performance requirements. The data stress test and concurrency test are used to determine whether the system meets the actual grid service operation requirements and whether the response time of the system meets the requirements under large concurrency conditions. The vertical software controllable system designed in this paper is used as the experimental group, and the control systems mentioned in literature [3] and literature [4] are used as the reference group. In the computer simulation platform, the grid service operation state is simulated, and the three groups of control systems are applied to control respectively, and the experimental conclusions are drawn by processing and analyzing the data in the comparison experiments.

4.2. Experimental results and analysis

The experimental data of the control system of the experimental group and the comparison group in the simulation condition environment for the control process of the vertical software are shown in Table 2 below.

| Serial number | System in this paper | System in literature [3] | System in literature [4] |
|---------------|----------------------|--------------------------|--------------------------|
|               | System instantaneous capacity / GB | Maximum system concurrency / TPS | Response time / ms | System instantaneous capacity / GB | Maximum system concurrency / TPS | Response time / ms | System instantaneous capacity / GB | Maximum system concurrency / TPS | Response time / ms |
| 1             | 50.1                 | 12751.8                  | 1689.9                  | 24.1 | 8604.9                  | 2724.1                  | 17.2 | 6851.1                  | 4923.6                  |
| 2             | 48.9                 | 12522.6                  | 1744.4                  | 24.2 | 9695.6                  | 2586.8                  | 17.1 | 6943.6                  | 4866.5                  |
| 3             | 49.5                 | 12618.7                  | 1797.1                  | 23.8 | 9650.4                  | 2728.2                  | 17.7 | 6044.3                  | 5099.2                  |
| 4             | 48.2                 | 12750.5                  | 1765.3                  | 24.3 | 9591.2                  | 2643.3                  | 17.6 | 6573.7                  | 4968.9                  |
| 5             | 48.4                 | 12977.5                  | 1676.2                  | 24.2 | 9562.3                  | 2632.4                  | 17.9 | 5723.4                  | 4801.7                  |
| 6             | 48.8                 | 12637.3                  | 1726.4                  | 24.5 | 8954.7                  | 2530.5                  | 17.3 | 5885.8                  | 5274.8                  |
| 7             | 47.3                 | 12541.5                  | 1821.7                  | 24.2 | 8884.5                  | 2493.6                  | 17.4 | 6882.9                  | 4914.1                  |
| 8             | 48.6                 | 12530.6                  | 1872.5                  | 24.0 | 9524.0                  | 2734.7                  | 17.8 | 6632.3                  | 4810.4                  |

Analysis of the data in Table 2 above shows that the instantaneous capacity and maximum concurrency of the system in this paper are the largest in several experiments, and the response time of the system is much less than 2000 ms. The response times of the software control systems in both papers are greater than 2000 ms, and their system resistance performance test index data are smaller than that of the system in this paper. From the specific numerical point of view, the average response time of the system in this paper is 1761.89ms, the average response time of the system in literature [3] is 2634.2ms, the average response time of the system in literature [4] is 4957.4ms, and the response time of the system in this paper is shortened by at least about 33.3%. In summary, the vertically software controllable system designed in this paper for grid service operation process has high concurrent resilience and effectively reduces the response time by about 33.3% on average, which is superior.

5. Conclusion

Vertical software does not pursue big and comprehensive, but only focuses on familiar fields and
becomes an authority in the corresponding industry field. In the process of grid service operation, applying vertical software helps to manage data. To improve the efficiency of controlling software operations, a vertical software controllable system for grid service operation process is designed. Comparative experiments conducted on the simulation platform proved the reliable performance of the system. In future research, further in-depth studies on enhancing the scope of application of the system are needed.

References
[1] Chae Sang Heon, Kim Gi Hoon, Choi YeongJun, et al. Design of Isolated Microgrid System Considering Controllable EV Charging Demand[J]. Sustainability, 2020, 12(22) : 9746-9746.
[2] Brooks Justin, Minnick Grayson, Mukherjee Prithvijit, et al. High Throughput and Highly Controllable Methods for In Vitro Intracellular Delivery.[J]. Small (Weinheim an der Bergstrasse, Germany),2020,16(51): e2004917.
[3] Abdullah Kamadan, Gullu Kizilta¸s, Volkan Patoglu. A Systematic Design Selection Methodology for System-Optimal Compliant Actuation[J]. Robotica, 2019, 37(4) : 656-674.
[4] Hae-Yong Park, Sang-Hoon Oh. Design range of the damper of a T-stub damage-controlled system[J]. Journal of Constructional Steel Research, 2019, 162(2):105719.
[5] Michael Felderer, Andrea Herrmann. Comprehensibility of system models during test design: a controlled experiment comparing UML activity diagrams and state machines[J]. 2019, 27(1) : 125-147.
[6] Khudier Khalid Hussien, Mohammed Khalid G., Ibrahim Mayyadah Sahib. Design and Implementation of Constant Speed control System for the Induction motors Using Programmable logic Controller (PLC) and Variable Frequency Derive (VFD)[J]. IOP Conference Series: Materials Science and Engineering, 2021, 1076(1) : 012007.
[7] Huynh Van Van, Tran Phong Thanh, Minh Bui Le Ngoc, et al. New Second-Order Sliding Mode Control Design for Load Frequency Control of a Power System[J]. Energies, 2020, 13(24) : 6509-6509.
[8] Rasmus Ros, Mikael Hammar. Data-driven software design with Constraint Oriented Multi-variate Bandit Optimization (COMBO)[J]. Empirical Software Engineering: An International Journal, 2020, 25(5) : 3841-3872.
[9] Järvenpää Eeva, Siitala Niko, Hylli Otto, et al. Capability matchmaking software for rapid production system design and reconfiguration planning[J]. Procedia CIRP, 2021, 97 : 435-440.
[10] Baouya A., Mohamed O A., Bennouar D., et al. Safety analysis of train control system based on model-driven design methodology[J]. Computers in Industry, 2019, 105:1-16.
[11] Polo Óscar R., Sánchez Jonatan, da Silva Antoniom et al. Reliability-oriented design of on-board satellite boot software against single event effects[J]. Journal of Systems Architecture, 2020, 114(1): 101920.
[12] Alper Bayrak. Design of an experimental wind tunnel system for control applications[J]. International Journal of Electrical Engineering & Education, 2019, 56(3) : 265-281.
[13] Pueo Basilio , Penichet Tomas Alfonso , Jimenez Olmedo Jose Manuel. Validity, reliability and usefulness of smartphone and Kinovea motion analysis software for direct measurement of vertical jump height[J]. Physiology & Behavior, 2020, 227(1) : 113144.
[14] Suryaningsih F., Dewi D C. Correction of vertical point of projection images using the correct axis tilt parameter in the Octopus software package[J]. Journal of Physics: Conference Series, 2020, 1436(1):012087.
[15] Grzegorz Bocewicz , Zbigniew Banaszak ,Izabela Nielsen. Multimodal processes prototyping subject to grid-like network and fuzzy operation time constraints[J]. Annals of Operations Research, 2019, 273(1-2) : 561-585.
[16] Zhanfeng Song, Yun Yu, Yaqi Wang, et al. Transient-Performance-Oriented Discrete-Time Design of Resonant Controller for Three-Phase Grid-Connected Converters[J]. JOURNAL OF POWER ELECTRONICS, 2019, 19(4) : 1000-1010.

[17] Larbi Djilali, Edgar N. Sanchez, Mohammed Belkheiri. Real-time neural sliding mode field oriented control for a DFIG-based wind turbine under balanced and unbalanced grid conditions[J]. IET Renewable Power Generation, 2019, 13(4) : 618-632.

[18] Bojan Banković, Filip Filipović, Nebojša Mitrović, et al. A Building Block Method for Modeling and Small-Signal Stability Analysis of the Autonomous Microgrid Operation[J]. Energies, 2020, 13(6):1-28.

[19] Federico Gaetani, Patrizio Primiceri, Giovanni Antonio Zappatore, et al. Hardware design and software development of a motion control and driving system for transradial prosthesis based on a wireless myoelectric armband[J]. IET Science, Measurement & Technology, 2019, 13(3) : 354-362.

[20] V.N. Burkov, A.K. Enaleev, V.I. Strogonov, et al. Models and Management Structure for the Development and Implementation of Innovative Technologies in Railway Transportation. I. Mechanisms of Priority Projects Selection and Resource Allocation[J]. Automation and Remote Control, 2020, 81(7) : 1316-1329.