[1] Zhihao Shen
[2,*] Yuanbin Xu
[3] Yanjie Wang
[4] Yifan Hu
[5] Gang Chen
[6] Ying Gao
[7] Pang Zhang
[8] Yongchao Pan

Research on the Application of Intelligent Multi-branch Version Management Strategy of Electricity Marketing Service System Software

Abstract: - In recent years, the Market Marketing Department of State Grid Corporation of China, in collaboration with the Internet Department, has initiated the construction of an intelligent energy internet marketing service system. This paper aims to improve the development efficiency and quality of the power marketing service system. To achieve this goal, an intelligent multi-branch version management strategy has been introduced. Firstly, the necessity of multi-branch version management is analyzed in relation to the characteristics of the power marketing service system software and the potential issues that may arise in its development and version management process. Then, based on the concept of productizing the general functions of the entire network and projecting the functions of each provincial network, a multi-branch version management strategy is designed, which includes the core components of intelligent multi-branch version management and intelligent multi-branch version protection. Finally, an application case of power marketing service system development is presented. This solution not only standardizes the version management process, improves software development efficiency, ensures the order and control of versions, but also demonstrates the enormous potential of intelligent technology in enhancing system development and management efficiency.

Keywords: Service System; Intelligent Multi-Branching; Version Management; Version Protection; Software Quality Enhancement

I. Introduction

The rapid development of intelligent information technology globalization has raised higher demands for software development, such as enhancing productivity and meeting the uncertainty and variability of demand. In 2019, the State Grid Marketing Department, in collaboration with the Internet Department, initiated the construction of the Smart Energy Internet Marketing Service System and completed the full-service online test run of the Jiangsu pilot in August 2022, achieving the province-side dual centralized platform architecture of the marketing service "One System". The intelligent power marketing service system is an application system covering the full range of intelligent marketing services and intelligent digital management. It includes three intelligent front-ends for customer service, employee operations, and management decision-making, as well as a large intelligent middle platform (comprising dual middle platforms for business and data, and five intelligent support class platforms). This system has become the main intelligent platform for the company's business revenue, serving as the total intelligent window for external services and comprehensively supporting the company's digitization of marketing services, business operations, and management processes. In this paper, we study the multi-branch version management strategy to improve the collaborative software development efficiency[1] and software quality[2] of each project team in the electric power marketing service system.

II. Necessity

With the promotion of the power marketing service system and the emergence of new demands, marketing personalized applications must be continuously released to meet new demands and optimize system services. However, there are many problems in the development process of the unified software of the power marketing service system, which leads to difficulties in the synchronisation and management of the main branch of the unified software and the feature branches of multiple provinces, thus affecting the promotion of the system iteration work[3]. At present, the unified software version management may have problems:

[1] Information Service Center, Info and Comm Branch, State Grid Zhejiang Electric Power Co., Ltd, Hangzhou, Zhejiang, China
[2] State Grid Info & Telecom Group Co., Ltd, Beijing, China
[3] Information Industry Research Institute, State Grid Info & Telecom Group Co., Ltd, Beijing, China
[4] Information Service Center, Info and Comm Branch, State Grid Zhejiang Electric Power Co., Ltd, Hangzhou, Zhejiang, China
[5] Information Industry Research Institute, State Grid Info & Telecom Group Co., Ltd, Beijing, China
[6] Information Industry Research Institute, State Grid Info & Telecom Group Co., Ltd, Beijing, China
[7] Information Industry Research Institute, State Grid Info & Telecom Group Co., Ltd, Beijing, China
[8] Marketing Department, State Grid Liaoning Electric Power Co., Ltd, Shenyang, China.
* Corresponding author: Yuanbin Xu

- Multi-version management problems of baseline version and province-specific personalized version. More and more province-side branches and function branches will generate a large number of technical problems and management problems, and there is a risk that the baseline master version will be bypassed and split.
- Integration of the personalized marketing services of the province-specific into the standardization of the main version. At present, the system lacks the specification of the whole link through the "front, middle and back office", which makes it impossible to realize rapid and accurate positioning and whole-link investigation and analysis when the system functions are abnormal.
- The problem of publishing marketing personalized applications under uninterrupted service conditions. The system, as an information system undertaking marketing services, must provide uninterrupted services, and there is a lack of standardized management of personalized demands in the release scenario of the unified software of the power marketing service system.

To solve the above problems, it is necessary to adopt a multi-branch version management strategy in the development process of the power marketing service system. On the one hand, the multi-branch version management strategy can meet the demand for stable iteration of the main branch of the unified software to avoid bypassing and splitting the main baseline version of the unified software; on the other hand, it can flexibly support the personalized iteration in multiple provinces-specific and at multiple levels. Based on ensuring the steady iteration of the baseline main version of the unified software for the power marketing service system, it meets the needs of different provinces-specific for the iteration of marketing front-end services, marketing business centre services, marketing domain models[4], etc.

III. Multi-Branch Version Management Strategy

A. The Connotation of Multi-branch Version Management

The electric power marketing service system is developed based on the integrated support platform for public shared service research and operation and is oriented to the construction of personalized demands of multiple provinces-specific. Based on this, the version control adopts single trunk and multi-branch management, combining the features of network-wide common functions, multi-network and provincial personalized functions, multi-level version iteration and maintenance needs, etc. The multi-branch version management divides the unified software into a trunk branch, a development branch, a pre-release branch, a fix branch, and province-specific feature branches[5].

- The backbone branch is the baseline branch of the unified software of the power marketing service system promoted for the whole network, which is used to unify the release of major versions and promote the stable evolution and development of the unified software of the power marketing service system.
- The development branch is the research and development branch of the power marketing service system unified software general marketing business model, middle-end service, front-end, etc., which is promoted for the whole network.
- The pre-release branch is a test and validation release branch before the official launch of the unified software of the power marketing service system for network-wide promotion.
- A fix branch is a branch used to fix defective issues against a baseline version of the Unified Software for Electricity Marketing Services System after the official release of a major baseline version.
- The feature branch of network and province is to provide the feature branch of network and provincial companies to deal with personalized needs under the premise of maintaining the stability of the main branch of the unified software of the power marketing service system.

The trunk is the core code, without any development activities; the branch is the customized code, whenever there is a special customization demand access to pull a new custom development branch. The daily development mainly focuses on the branch development, and changes involving the core code are merged into the trunk. Branches often need to go to the trunk to pull the core code and synchronization to avoid duplication of development work. The collaborative management of the five branch versions, not only ensures the steady iteration of the main version of the unified software baseline of the power marketing service system, but also meets the needs of different network provinces for the iteration of the version of the marketing front-end service, the marketing business middle-end service, and the marketing domain model at multiple levels[6].

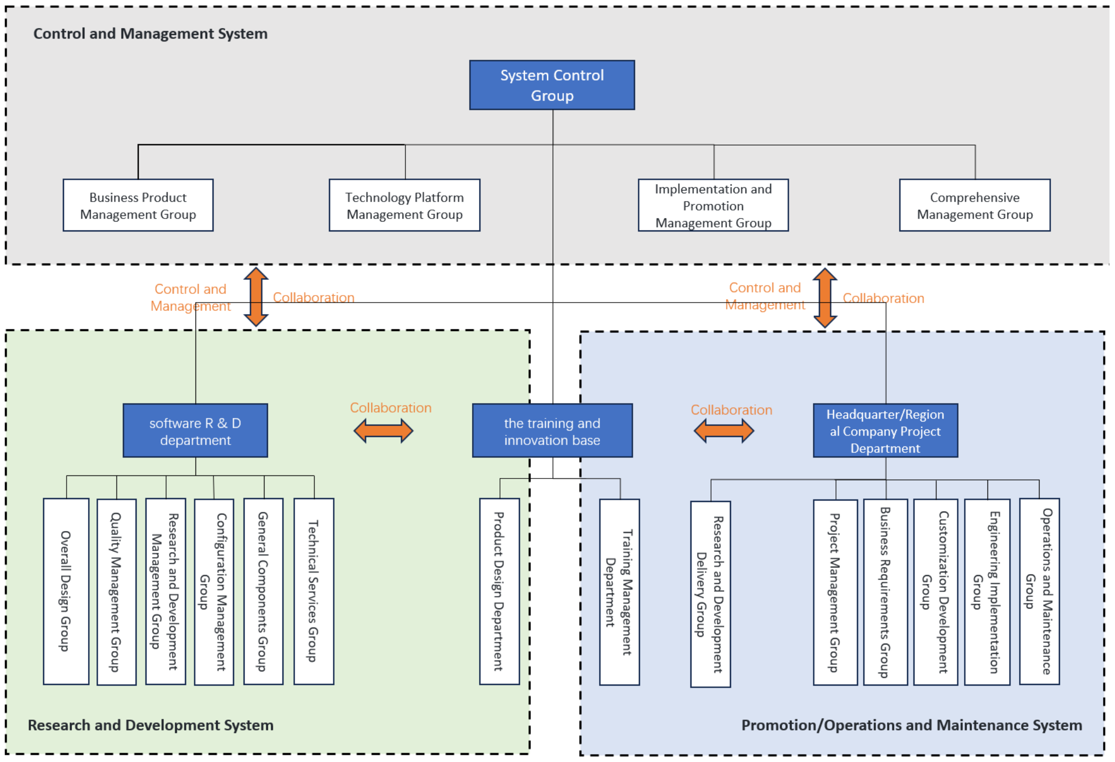
B. Multi-branch version management Uses

The single trunk development model has only one version of forward iterative development, and does not support parallel development; the modifications of the software are concentrated in the latest version, can not cope with the frequent changes in the needs of multi-user scenarios; can not distinguish between general-purpose and customized functions, is not conducive to the realization of the software productization.Multi-branch version control can make up for the above shortcomings, mainly including the following[7-8]:

- There are several different branches in the system, such as development, pre-release development and fix branches, etc., and each branch is relatively independent and does not affect each other.
- Each branch is independently maintained, and if necessary, merging between branches or branches can be merged into the main trunk.
- Avoiding that the newly developed functions are wrongly entered into the production environment along with the on-line of the other functions.
- Precisely distinguish between the common functions of the whole network and the individual functional requirements of each network province, which is conducive to the realization of software productization.

C. Multi-Branch Version Management

1) Organizational Structure and Responsibilities: The construction and promotion of the electric power marketing service system adhere to the overall idea of unified control, centralized R&D and integrated implementation[9]. "Unified control" means being centrally controlled by the headquarters control group; "centralized R&D" indicates that the training and innovation base and the software R&D department jointly take responsibility for business requirements, including product design, research and development, verification, and delivery; "integrated implementation" refers to the promotion and implementation of the provincial companies by the network's "one game" of the idea of coordinated promotion, and by the training and innovation base docking network-wide business product training, software R&D department docking provincial side to provide standardized deployment training and technical support, its structure is shown in Figure 1.



Control and Management System
System Control Group
Business Product Management Group
Technology Platform Management Group
Implementation and Promotion Management Group
Comprehensive Management Group
Control and Management
Collaboration
Control and Management
Collaboration
software R & D department
Collaboration
the training and innovation base
Collaboration
Headquarter/Regional Company Project Department
Overall Design Group
Quality Management Group
Research and Development Management Group
Configuration Management Group
General Components Group
Technical Services Group
Product Design Department
Training Management Department
Research and Development Delivery Group
Project Management Group
Business Requirements Group
Customization Development Group
Engineering Implementation Group
Operations and Maintenance Group
Research and Development System
Promotion/Operations and Maintenance System

Figure 1: Schematic diagram of the organizational structure

After the product design requirements of the power marketing service system, the technical architecture optimisation requirements, and the operational elimination requirements of each province are validated by the management and control group and the Jiangsu Training and Innovation Base, the software R&D Department completes the revision of the unified software system design, data model, standard coding, algorithm model, etc., and completes the product research and development, iteration, and functional factory testing according to the plan.The Software R&D Department, led by the Information Industry Group, is composed of the Power Marketing Service System Software R&D Center under the Group and the R&D strength of Jiangsu Company, Zhejiang Company and other industrial units. R&D centre for the State Grid Corporation headquarters and training and innovation base to carry out unified software requirements for research and development work, including the headquarters side of the main support headquarters departments, offices, State Grid customer service centres, big data centres, responsible for the headquarters side of the business centre, data centre, online State Grid R&D work, the provincial (municipal) side of the demand for the completion of the product design in the Jiangsu training and innovation base, to support the unified software on the ground after the development of the power marketing service system. The unified software draws on the Scrum (iterative incremental software development process) management model and carries out construction and operation work according to the idea of overall iteration. The R&D Center is responsible for the operation and maintenance of the system's unified software R&D simulation environment; it is responsible for providing unified software R&D delivery to 27 provincial companies, including the deployment, adaptation, and validation of the unified software on the provincial side of the tenant; it is responsible for cooperating with the provincial side engineering team in the release of the business validation environment, and it is responsible for docking of problems on the provincial side, as well as providing technical training and technical support to support the smooth implementation of the provincial side of the project.

2) Parallel Development Process: Power marketing service system software is developed in parallel on multi-branch versions, which can ensure that the branch versions are independent of each other[10] and reduce the probability of code conflicts when code is merged in the final release version[11], and the multi-branch version control process is shown in Figure 2.

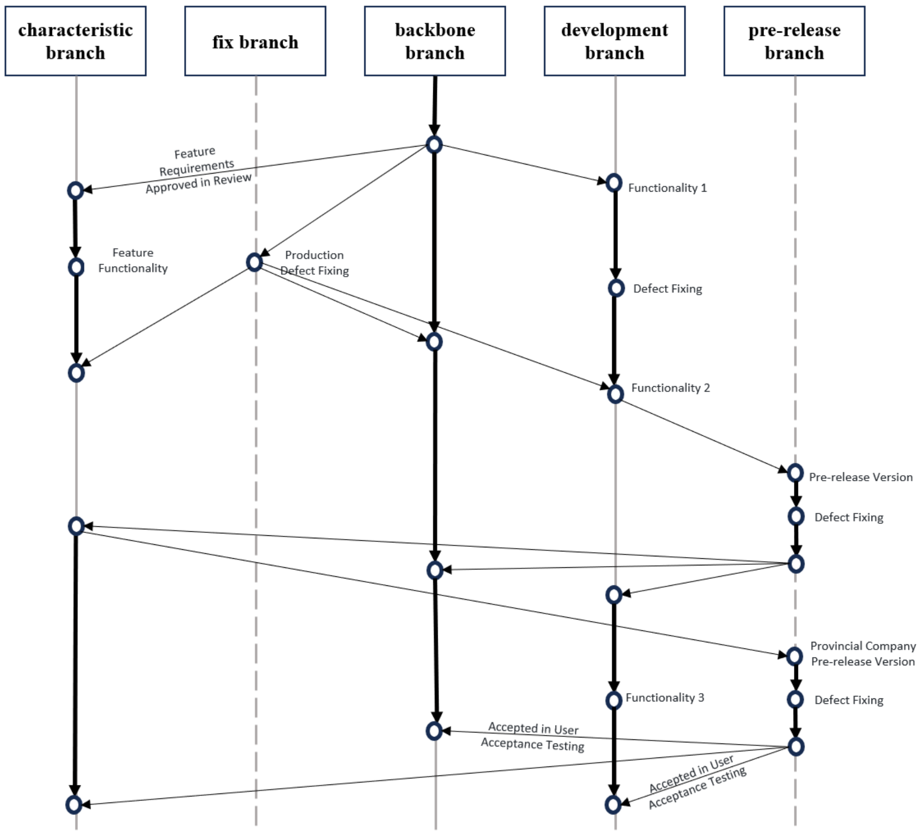


Figure 2: Development Flowchart

- In the beginning stage, developers are carrying out software development activities on the trunk, and form the initial version after completing the development of the basic functions of the system, that is, the trunk branch.
- The development branch belongs to the R&D branch and is responsible for developing new features and completing defect fixes for new features. When the feature meets the requirements, the code is pulled to the pre-release branch. After acceptance in the pre-release branch, the code is merged into the development branch.
- The pre-release branch is responsible for pulling code from the network province feature branch and the development branch, and merging it into the trunk branch, the development branch and the network province feature branch respectively after testing and development.
- After the review and approval of the feature requirements of the network province, the feature branch of the network province pulls the code from the trunk branch for the development and testing of the feature functions. When the feature meets the generalization of the whole network, it will be merged into the pre-release branch.
- The fix branch is responsible for the repair of production defects in the trunk branch, and after the repair is completed, it is merged into the trunk branch, the network province branch, and the development branch.

3) Province-Specific Feature Branch Management: The province-specific feature branch pulled from the backbone branch serves as the main branch for maintaining the province-specific branch, based on which the province-specific feature development branch, province-specific pre-release branch, province-specific fix branch, etc. are set up as needed. Additionally, the R&D branches within the province-specific feature branch are maintained according to different levels of demand flow: Front-end service layer demand flow: it involves updating the front-end services; 2) Middle-end service layer demand flow: it involves updating the middle-end shared services and front-end services; and 3) Domain model layer demand flow: it involves updating the models created by the domain driver, the middle-end shared services, and the front-end services.

The integration management of network and provincial feature branches with the main version. After the realization of the network and provincial feature functions, the contents of models, middle-office services, front-end services, etc. that have network-wide promotion value and are suitable for the expansion of the baseline version of the Unified Software for Electricity Marketing Service System can be merged into the main branch of the Unified Software for Electricity Marketing Service System after examination by the Training and Innovation Base and the Software Research and Development Department.

Net-provincial feature branch release management, for personalized requirements that do not have the value of network-wide promotion, after the function is realized, it will not be merged with the backbone branch, and the subsequent release will create a net-provincial pre-release branch to carry out pre-release testing, and the pre-release branch will be merged with the net-provincial feature branch and marked with a version number before preparing for the release. The net-provincial feature branch regularly pulls code from the backbone branch as needed, resolves conflict, merges, and maintains code synchronization with the backbone branch to reduce the occurrence of code conflicts and overrides in the subsequent development of net-provincial personalized demand functions[12-14].

4) Naming Management: The naming conventions for source code multi-branch versioning include the naming of the baseline master version and the network province feature branch, as well as the naming rules for the release tag. In the baseline master version, the main branch is named master, the development branch is dev, the pre-release branch is released, and the fix branch is hot fix, while the naming of the network province feature branches are based on the English names of the provinces, such as feature_Anhui, feature_Jiangsu, feature_Zhejiang, etc. The Tag of the release version follows the specification of the application name - branch type - version number, e.g. xxxx-release-v1.0.2. The version number consists of the main version, the feature version and the fixed version. When there is an architectural modification or a modification that is not forward-compatible, the first digit of the master version number is added; the second digit of the version number is added when adding a new feature that is forward-compatible; and the third digit of the version number is added when fixing only a system problem. These specifications ensure that version naming is uniform and semanticized identification is achieved[15].

D. Multi-Branch Version Protection

1) Main Version Baseline Branch Protection: Trunk branch protection strategy. To protect the stability of the backbone branch, all development branches, province-specific feature branches and other branches are pulled from the power marketing service system backbone branch specification. All codes submitted to the backbone branch are subject to review and endorsement. The project can only have one trunk branch, and the code in the trunk branch is only updated when new deployable code is released[16]. Each time a trunk branch is updated, a tag in the specified format is added for release or rollback purposes. The master branch is protected and cannot be pushed directly to the master branch of a remote repository. The code on the trunk branch can only be updated by merging it with a pre-release branch or a fix branch. The trunk branch protection policy helps to ensure the management and control of the trunk branch to ensure the stability of the trunk branch.

Development branch protection strategy. The development branch for the unified software baseline version of the electric power marketing service system involves the development of the marketing business model, middle-office services, front-end, etc. The code submitted to the development branch should be clearly annotated with the content of the submitted updates, and the submitted code triggers the automated continuous integration, completes the process of code scanning, software construction, unit testing, etc., and then submits the code to the pre-release branch after the code audit is passed. A project is allowed to have only one development branch. In the early stages of development, the branch manager signs out the development branch from the master branch, which is used to integrate new development results and resolve defects in the development process. However, in subsequent phases of development, the development branch will no longer interact directly with the master branch. This protection strategy is designed to ensure reliable code management and control during the development process while providing an independent and stable development environment in which team members can efficiently carry out their development work.

Pre-release branch protection strategy. A pre-release branch is created before a new version of the baseline version of the unified software for the power marketing service system is about to be released, and the testing and verification of this pre-release version are all carried out on this branch after the code is reviewed and approved, the code is submitted to the main branch, and the version number of the main branch is updated according to the naming convention. If you need to prepare a new version of the product for release, a pre-release branch is derived from the development branch. This pre-release branch, which can be derived from a specified commit on the development branch, is used to test the new version of the code and must be merged back into the master branch after it passes the test and is tagged with a specific version number. Once a pre-release branch is created, no more code can be pulled from other branches. In addition to bug fixes, the code in the pre-release branch should include all the information needed to prepare for the release, such as the version number, release date, and compile time. At the same time, all business requirements outside the current upcoming release should be ensured not to be mixed into the pre-release branch to avoid introducing some uncontrollable system defects. Finally, the pre-release branch should be deleted after the pre-release branch has been tested and merged back into the development branch and the master branch.

Fix branch usage strategy. After a major baseline release of the Unified Software for Electricity Marketing Services System is officially released, it is derived from the trunk branch to specify the tag version for quick fixes of defective issues or fine-tuning of features. The fix branch must be merged back into the trunk branch and the development branch once the defect fix is complete and the fix branch is deleted. When the trunk branch is merged, it should be tagged with a new version number to ensure versioning accuracy. Once a fix branch is created, it can no longer pull code from other branches, ensuring the independence and stability of the fix branch.

2) Province-Specific Feature Branch Protection: The province-specific individualized requirements are classified into three different levels of requirement flows based on their impact on the overall architecture of the unified software.1) The front-end service layer of the baseline version does not satisfy the requirement, and the requirement flow hierarchy type is the Front-end Service Layer Requirement Flow.2) The Front-end Service Layer and the Middle-Office Shared Service Layer that already existed in the baseline version do not satisfy the requirement, and the requirement flow hierarchy type is the Middle-Office Service Layer Requirement Flow.3) The front-end service layer, the middle-office shared service layer, and the domain model layer cannot satisfy the demand, and the demand flow hierarchy type is the domain model layer demand flow[17].

Demand review and protection strategy. When the existing front-end services and middle-stage shared services of the baseline version of the unified software of the electric power marketing service system cannot meet the personalised needs of the network provinces, the network provinces will submit the personalised demand function description and demand flow hierarchy type, and the training and innovation base will audit the personalised demands of the network provinces.

Review strategy for network province personalized demand design plan. After the approval of the individualized requirements of the network provinces, the network provinces will submit software outline design and detailed design documents describing the functional development of the individualized requirements of the network provinces involving the expansion of the model layer, the middle platform layer and the front-end of the baseline version of the unified software of the power marketing service system, and the Training and Innovation Base and the Software Research and Development Department will evaluate and review the impact of the baseline version of the unified software of the power marketing service system and whether it has value for network-wide promotion. The Training and Innovation Base and the Software R&D Department will evaluate and review whether the baseline version of the unified software for the power marketing service system has network-wide promotion value.

Network province personalized branch creation protection strategy. Net provincial personalized demand to achieve the design plan review and approval, the network provincial companies are permitted to pull code from the unified software version of the power marketing service system backbone branch baseline version, to create a personalized branch of the network province to create[18].

IV. Application

The following is an example of the power marketing service system project shown in Fig 3, which describes the multi-branch version management implementation process[19].

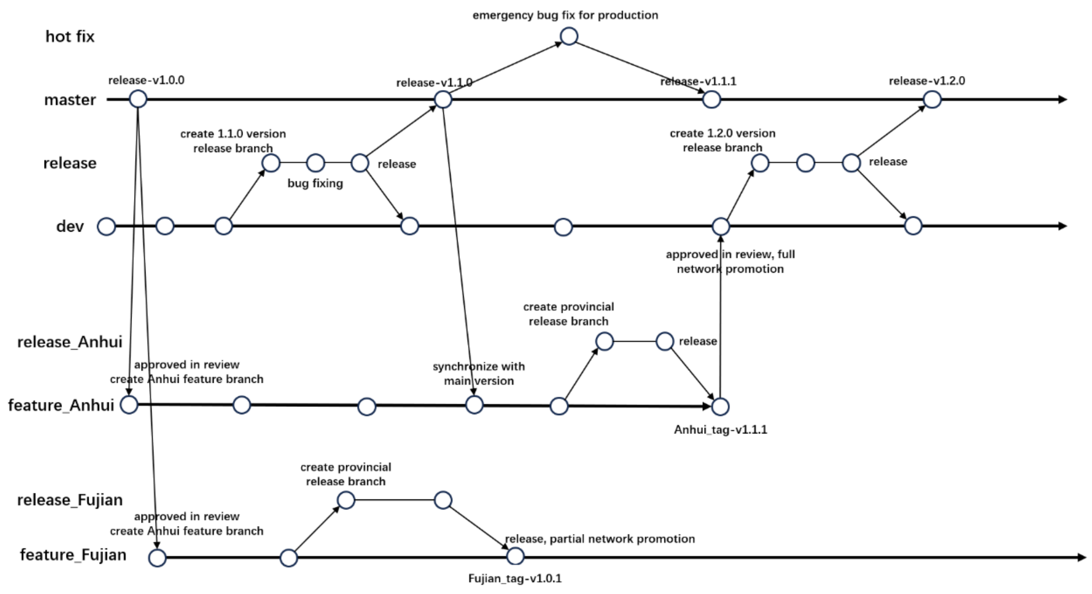

hot fix
emergency bug fix for production
master
release-v1.0.0
release-v1.1.0
release-v1.1.1
release-v1.2.0
release
create 1.1.0 version release branch
bug fixing
release
create 1.2.0 version release branch
release
dev
approved in review, full network promotion
release_Anhui
create provincial release branch
release
approved in review create Anhui feature branch
synchronize with main version
feature_Anhui
Anhui_tag-v1.1.1
release_Fujian
create provincial release branch
approved in review create Anhui feature branch
release, partial network promotion
feature_Fujian
Fujian_tag-v1.0.1

Figure 3: Example Engineering Diagram

The implementation process is described below:

- When the audit of the network province is passed, you can pull the code from the master of the trunk version and create the network province feature branches, such as Anhui feature branch feature_Anhui and Jiangsu feature branch feature_Fujian.
- After creating the branch of Jiangsu features, according to the Jiangsu features of the function create a network province release branch, after the successful release, not to the whole network promotion.
- After the creation of the Anhui feature branch, it will be updated and iterated according to the Anhui feature. When the version of the main branch is updated, the feature branch needs to synchronize with the version of the main branch. After the iteration, you can create a branch for publishing. At the same time, you can submit the audit to the development branch dev, after the audit, the development branch creates a new version of the release branch for release[20].
- After the development branch is updated and iterated, a new version of the release branch can be created and released after updating and fixing bugs, and the released version is synchronized to the development branch and the master branch.
- When the trunk branch update iteration encounters an emergency bug, the emergency bug fix will be produced through the fix branch[21].

V. CONCLUSION

The multi-branch parallel development and version control model proposed in this paper emphasizes productization and network province personalization. The model appears to be particularly outstanding in terms of precise and differentiated version management, parallel development, efficiency improvement, and productization operation. Through the implementation of the multi-branch version management control process, traditional project-oriented thinking can be changed, emphasizing the concept of productization of general functions and the projectization of personalized functions in each network province. Furthermore, the intelligent characteristics of this model, through the incorporation of a smart decision support system, further enhance the precision and efficiency of version management and parallel development, making the productization operations and the projectization of personalized functions more intelligent and efficient. Practice shows that each branch has a clear purpose and is independent of each other, which can effectively solve the problem of chaotic version management, ensure an orderly and controllable version, standardize the software development process, and improve the efficiency of collaborative development. This intelligent approach will bring a higher level of intelligence and benefits to the development of the power marketing service system.

ACKNOWLEDGMENT

This work is supported by the Science and Technology Project of State Grid Corporation of China (Research and Application of Dynamic Release and Rapid Data Processing Technology of Electricity Marketing Service System Application for the Whole Grid Users, No. 5108-202218280A-2-400-XG).

REFERENCES

[1] Rahul Premraj,Antony Tang,Nico Linssen, et al. To branch or not to branch? Proceedings of the 2011 International Conference on Software and Systems Process, 2011: 81-90.

[2] Emad Shihab,Christian Bird,Thomas Zimmermann. The effect of branching strategies on software quality. Proceedings of the Acm-ieee International Symposium on Empirical Software Engineering and Measurement, 2012: 301-310.

[3] Nazatul Nurlisa Zolkifli,Amir Ngah,Aziz Deraman. Version control system: A review. Procedia Computer Science, 2018, 135: 408-415.

[4] Shaun Phillips,Jonathan Sillito,Rob Walker. Branching and merging: an investigation into current version control practices. Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering, 2011: 9-15.

[5] Brad Appleton,Steve Berczuk,Ralph Cabrera, et al. Streamed lines: Branching patterns for parallel software development. Proceedings of Plop, 1998: 14.

[6] Mojtaba Shahin,Muhammad Ali Babar,Liming Zhu. Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. Ieee Access, 2017, 5: 3909-3943.

[7] John Plaice,William W Wadge. A new approach to version control. Ieee Transactions on Software Engineering, 1993, 19(3): 268-276.

[8] Diomidis Spinellis. Version control systems. Ieee Software, 2005, 22(5): 108-109.

[9] Dustin Heaton,Jeffrey C Carver. Claims about the use of software engineering practices in science: A systematic literature review. Information and Software Technology, 2015, 67: 207-219.

[10] Hoai Le Nguyen,Claudia-Lavinia Ignat. An analysis of merge conflicts and resolutions in git-based open source projects. Computer Supported Cooperative Work (cscw), 2018, 27: 741-765.

[11] H Christian Estler,Martin Nordio,Carlo A Furia, et al. Unifying configuration management with merge conflict detection and awareness systems. 2013 22nd Australian Software Engineering Conference: Ieee, 2013: 201-210.

[12] Rajesh Vasa,Markus Lumpe,Philip Branch, et al. Comparative analysis of evolving software systems using the Gini coefficient. 2009 Ieee International Conference on Software Maintenance: Ieee, 2009: 179-188.

[13] Jules White,José A Galindo,Tripti Saxena, et al. Evolving feature model configurations in software product lines. Journal of Systems and Software, 2014, 87: 119-136.

[14] Audris Mockus,Roy T Fielding,James D Herbsleb. Two case studies of open source software development: Apache and Mozilla. Acm Transactions on Software Engineering and Methodology (tosem), 2002, 11(3): 309-346.

[15] Daniel M German,Bram Adams,Kate Stewart. cregit: Token-level blame information in git version control repositories. Empirical Software Engineering, 2019, 24: 2725-2763.

[16] Stephan Krusche,Mjellma Berisha,Bernd Bruegge. Teaching code review management using branch based workflows. Proceedings of the 38th International Conference on Software Engineering Companion, 2016: 384-393.

[17] Hong Mei,Lu Zhang,Fuqing Yang. A component-based software configuration management model and its supporting system. Journal of Computer Science and Technology, 2002, 17(4): 432-441.

[18] Abdel Salam Sayyad,Tim Menzies,Hany Ammar. On the value of user preferences in search-based software engineering: A case study in software product lines. 2013 35th International Conference on Software Engineering (icse): Ieee, 2013: 492-501.

[19] Md Tajmilur Rahman,Louis-Philippe Querel,Peter C Rigby, et al. Feature toggles: practitioner practices and a case study. Proceedings of the 13th International Conference on Mining Software Repositories, 2016: 201-211.

[20] Samera Obaid Barraood,Haslina Mohd,Fauziah Baharom. A comparison study of software testing activities in Agile methods. Knowledge Management International Conference (kmice), 2021.

[21] Xuan Bach D Le,David Lo,Claire Le Goues. History driven program repair. 2016 Ieee 23rd International Conference on Software Analysis, Evolution, and Reengineering (saner): Ieee, 2016: 213-224.