Research on Technologies and Application of Automated Testing Oriented OS2 Pas Function

Bochuan Gu[1,a], Yi You[2,b], Jinhua Huang[3,c], Fei Liu[4,d]

[1,2,3,4]Electric Power Research Institute of Guangdong Power Grid Corporation, China

[b]Youyi1983@163.com

[c]seafoodhao@126.com

Key words: OS2; Pas function; Automation testing framework; Test-case generation

Abstract An automation testing framework oriented OS2 PAS application is constructed in order to solve the deficiencies that there is no effective methods for the PAS application of OS2 master station system. An Automated Test-case Generation method (P-ATGM) is proposed. ROI evaluation of automation testing framework is carried out. The analysis shows that the method can improve testing efficiency and degree of automation of OS2 Pas application test.

1. Introduction

China Southern Power Grid Operation Smart System (OS2), according to principle SOA, is an integrated, modular, intelligent next-generation power grid operation support system. It is built to solve the operation information complication and fragmentation of secondary system which will improve operation integration level [1, 2]. As OS2 system is currently in the stage of experimental construction, it lacks adequate operation experience, test criteria and systematic evaluation methods.

Since OS2 system is evolved from the traditional dispatching automation system EMS, the testing technology did not consider the inspection in code development stage. It mainly focuses on the evaluation of power system service layer. For the function inspection relied on artificial means in OS2 platforms, assisted with recording / playback or structured script automated test technology, it can realize the functional verification of power data acquisition, monitoring and control and other aspects. While for the analytical and computing business closely related to the operational data, it still needs to build a series of pointed and widely covered test cases. The key issues restricting OS2 running capacity is the scarcity of calculation index analysis, which leads to difficulties to mine in-depth weak points of business analytical applications.

There are two purposes to establish the automated testing for software systems. One is to execute a large number of ordinary and repetitive testing tasks, such as OS2 data processing capabilities avalanche test and user-oriented concurrent access test; The second purpose owing to that a very complicated test case is beyond the capacity of large-scale testing by hand. Both of the above are the best candidate conditions for test automation [3]. For the first condition, standardized interface test can be applied by means of mature automated testing tools. While the non-standardized interface test needs testers to write the structured script. As important application of OS2, Power grid analysis belongs to the second condition. In previous tests, attentions mainly have been paid on the general function test. Testers care more about whether the application function meets the business requirements or whether the HMI and other surface features is perfect. There's no dedicated cases and performance analytical means to test the functional models.

For the lack of effective means of testing to OS2 analysis applications, an automated testing framework models was built on the basis of data-driven automated testing technology. And this paper proposed a model-based test case derived technology, to solve the complex problem caused by design and huge workload of artificial preparation of cases. Finally, the above automated testing framework was verified and assessed by ROI. The results show that the proposed automated testing technology has effectively enhanced the level and efficiency of automated testing in OS2 analysis applications.

2. Overview of Automated Testing

Automated testing is the process that transforms the manual testing into machine testing. By using pre-programmed test strategies and tools to reduce manual intervention, automated testing has a high degree of repeatability and complexity characteristics [4, 5]. With the continuous scale expanding of software systems, the requirements of function and performance was also increasing. Automated testing is becoming pretty important. To some extent, automated testing can significantly reduce resource overhead and increase test coverage in a limited time. It brings a very significant effect on ensuring the quality of software systems, reducing testing costs and testing cycles [6].

Compare to manual testing, automated testing has three significant advantages. First, to test the new version of the program for regression, most functions and interfaces are similar or identical to the previous version. The verification of the update procedures did not introduce new bugs, automated testing is particularly suitable for this situation. The second is suitable for testing tasks which are difficult or impossible to complete by manual testing, such as stress test, avalanche test, high complexity data testing etc. Third, Automated testing is consistent and repeatable, which ensures that each test run the same case in the same environment. It easily reproduce the software defect.

Although automated testing generates an increasing role in software testing, it is still restricted by the inherent shortcomings, such as the technical reserve, organization management, and ongoing maintenance of test scripts [7]. Only relying on automated testing tool does not guarantee that the test will achieve the desired effect, it is necessary to design, customize and maintain the entire testing framework. Automated testing tool is not substitute for professional design and consideration of the entire test [8].

3. OS2 application automated testing framework design

Selecting an appropriate automated testing framework is the prerequisite to achieve an effective test. A good automated testing framework can effectively reduce the cost of implementation and maintenance. Especially for highly professional business systems (OS2), testers should shift the emphasis to the design of testing case.

3.1 Framework Design Strategy

OS2 is a technical support system for power grid operation, which is responsible for the monitoring and control within the scope of its jurisdiction. It also considers the System importance and risk tolerance. At the same time, OS2 belongs to non-invasive system, therefore, the automated testing framework should be independent from the SUT (system under test). In order to avoid the automated test introducing any application deployment on SUT, it requires the system itself has a good interoperability on the basis of the appropriate testing framework. Secondly, test case design should be independent with the test framework, to ensure that the implementation of the existing cases and the new test cases will not be affected by the maintenance of automation testing framework.

3.2 Construction of automated testing framework

To achieve the aforementioned frame design strategy, data interaction layer between test frameworks and SUT will be constructed in a loosely coupled way. The framework is mainly composed of five parts in modular form, such as automatic test engine (test management, test execution), data base (database, case derivative instruments), test drive layer (data interface, data mapping and type conversion), test evaluation layer (test evaluation tools, test report generator), test interface (HMI, log management).

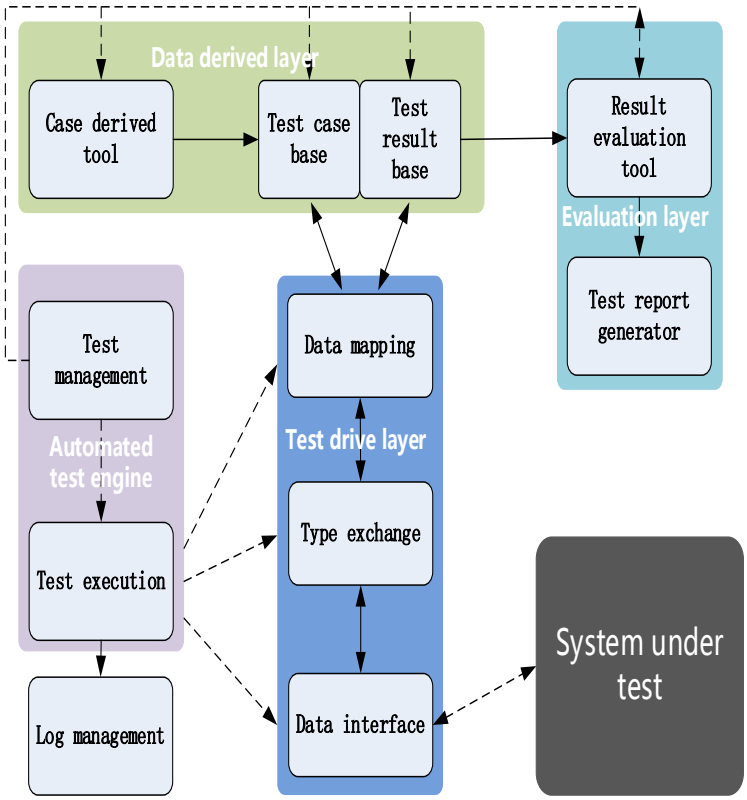

Fig.1 OS2 analytical application automated testing framework overview

(1) Taking the automated test engine as main control, in OS2 application automated testing, the default strategy calls the data derived layer according to the test management. For different targets, test data is automatically generated. The test case is submitted to the test drive layer. After data mapping and type conversion, test case was sent to the SUT by OS2 standard service interface, and the test drive layer would sends the test results to the test evaluation layer. The automatic test engine will call the log management module during the execution of the test procedure, and writes the result to the log. Then returns to test management to check the strategy execution progress, finally plans to start a new mission.

(2) Data derived layer is the core part of the testing framework for generating test cases automatically. Based on the ground state test case, test derivative tools are able to generate large-scale case sets in accordance with the specified target. During the test process, data derived layer receives feedback data from test evaluation layer and adjust the strategy accordingly which aimed to make several times large-scale similar testing to analyze the suspicious point of the application. So that with the adequate number of samples obtained from the statistical results, the accuracy of the evaluation is enhanced. The features of OS2 test analytic case are reflected in that the calculation need to satisfy a strict mathematical model. The details of the test cases derived technique will be discussed in the third chapter.

(3) Test drive layer design has considered SOA architecture system. Based on unified standards of power grid model and service interfaces, the drive layer realizes the information exchange and business integration with external systems via the service bus OSB, which is shown in Fig.2. Testing system receives and analyses the data from SUT by Web Service or in FTP mode. The data contains CIM model of power grid, static basic data of incremental CIM model and the dynamic operation data of power section. Data exchange interface design follows the IEC61970 specifications. Related properties are expanded on the basis of above interface, which can be used for automated testing framework, automatic inspection or preparation for future access to the secondary monitoring system.

(4) Testing evaluation layer is responsible for the online or offline analysis of the results returned from the SUT. The automated test engine simultaneously sends test cases to the test drive layer.

Also the ground state use case of the test case derivation tool is transmitted to the test result evaluation tool. Providing a data section which meets the power system data model as a benchmark, online assessment will find the suspicious points from the results of SUT. Then modify test case design strategy to generate large-scale sets of test cases. Test report generator will produce comprehensive evaluation offline report after the completion of the automated testing specified task.

(5) Testing interaction layer provides man-machine interface, log management and other functions to achieve the following three main tasks. One is used for controlling the entire automation testing process; The second is used for some special needs such as unconventional single test or other validation operations. Man-machine interface provides the tools to modify the related grid computing models and data sections; The third is used to view the test analysis report and test framework running log.

4. Automated test cases derived technology

For dispatching automation system in the past, the test case is generated mainly by artificial means. Due to particularity of grid computing business analysis, manual generating a test case is extremely complex. CIM model can be imported directly into the SUT, but the real-time data in a data section generally is stored accordingly by the private ID of the system's internal measuring point. It needs to build up a data mapping between private ID and GID, which makes the system to identify the data section. On this basis, the data also need to check again to filter out the unreasonable part. The test case is generated by manual modification in many times according to different test purposes. For the new OS2 system, there is no real running data. It's difficult to create a convergent data section through artificial means, no mention the required test coverage rate. Therefore, this paper proposed an automated test case derivation techniques for OS2 grid analytic calculation function (P-ATGM, PAS Automated Test-case Generation Method).

The core of P-ATDG is the generation of the ground state of test case. Based on the existing network equipment parameters and topology model, set the operation mode and key measurement. The none-critical measurement could be selected as the rated value or a random one around the rated value. As there are some bad data among the measurement value, so it is necessary to use system iterative state estimation to get the final data section according to a mathematical model. The state ground power flow calculation is realized by the state estimation function of BPA. After one state estimation calculation, chose portion of the reasonable estimated value to correct the ground state data section and increased the credibility weight of the measurement value. Taking the out-of-limit estimated results as suspicious data and reduce the weight of its credibility, then run the next iteration. By iteration, a satisfied ground state section can be obtained in limited steps. If adding the real time operation data, the ground state solutions will be obtained in fewer iterations.

It is necessary to further process the ground state test case for generating the actual test cases. According to the test target, set the modification program of the measurement data, such as modification for data in a particular voltage level or a certain type of variable. It can also set the topology distance to modify the measurement data in a nearby area of a node. Then automatically generate batch test cases by a program script. Generally, the test cases are generated off-line. The test framework engine will call the test cases from the case base under the on-line automated testing. However, when the results returned from OS2 appear abnormal regularly such as the out-of-limit of the voltage level, the testing system will feedback the relevant information to the automated test engine. The test engine will trigger the on-line case generation and send the case to test drive layer. Through the on-line modification strategy, large number of the statistical results can be obtained which is conducive to mine the defects of SUT. The specific implementation steps are shown in Figure 2.

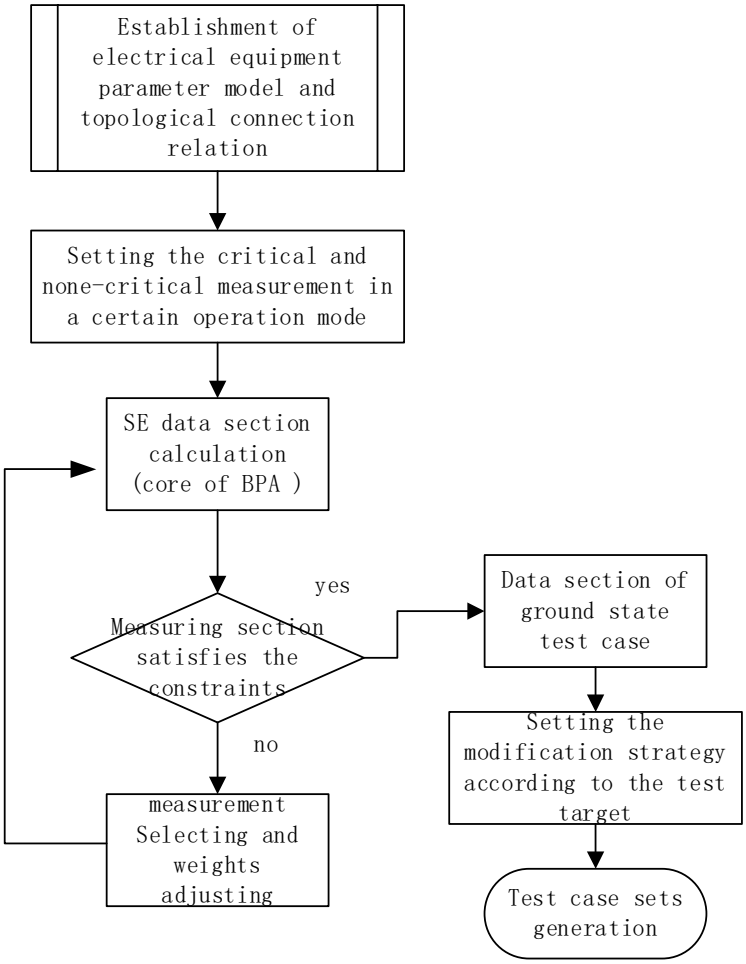

Fig.2 Automate test cases derived flowchart of OS2 analysis and calculation test

5. Automated Testing ROI assessment

The purpose of using automated testing tasks typically to analyze the same mission and compare resources spent on manual and automated test. The benefits and costs of automated testing are demonstrated with a real test task in ROI assessment.

Table 1 Test Task Overview

Test mission	State estimation on-line verification of OS2
Test location	Key Automation Laboratory of Southern China Power Grid
Test model	IEEE 10 Generator 39 Bus System
Test case	Thirty (26 cases for state estimation +4 cases for parameters estimation)
Test tool	Independent research and development of automated testing frameworks and tools
Test scale	4 sets of different manufacturers mainstream OS2 online estimation module

Before the execution of OS2-line state estimation test, IEEE 39 node CIM model file was previously sent to manufacturers for importing the file to SUT. Which unites the data exchange interfaces, including grid data section exchange format, network parameters adjustment incremental model interchange format, calculation results exchange format. And system uses WS/FTP mode or swap file to exchange files. After obtaining test cases, it will startup state estimation calculation and returns the result. Or the OS2 open state estimation service will be called by the computing engine to implement task automatically. The test framework is shown in Figure 3.

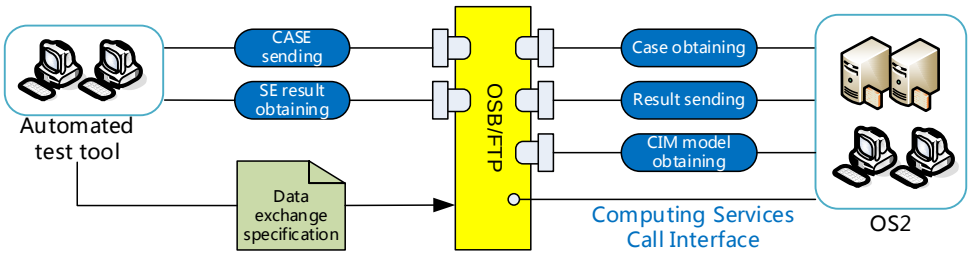

Fig. 3 Automated testing framework of OS2 online state estimate

Table 2 OS2 online state estimate test ROI calculations

Operation	Time required for manual testing	Time required for automated testing
Test creation	Test case design: 2hours*30=60hours	Create 2 ground state tests: 2hour* 2=4hours Test case generation: 1min* 30=0.5hour
Structure preparation	N/A	Adaptability Improvement and debugging of test tool 90hours
Environmental construction	N/A	Building FTP server: 2hours OS2 interface development: 8hours
Workload	60 hours	104.5hours
Test execution	Case Manual Induction: 10min*30=5 hours Manually start computing： 2min* 30=1 hour Artificial export: 10min*30=5 hours Manually adjustment for Parameter estimation 1hour*4 =4 hours	Test automatically import / export: Time can be ignored Automated calling for computing services: 1min* 30=0.5hour
Testing evaluation	Data reconciliation between manual analytical results and standard ground state data 20min*30 =12 hours	Test results automated assessment: 0.5 min*30=0.25 hour
Workload	27hour*4 test =108 hours	0.75hour*4 test =3 hours
Total workload	168 hours	107.5 hours

As the ROI calculations results shows, the main consuming of automated testing framework mainly concentrates on the adaptability and commissioning work which belongs to a fixed time-consuming. With the expansion of the scale of the test, the return on investing automated testing will far surpass manual testing.

6. Conclusions

With respect to manual testing, automated testing has obvious advantages in that automated testing framework can be gradually extended by developers to improve test coverage. With the gradual improvement of OS2, its application modules number increased and the system become more complex. Only relying on the traditional manual testing methods have been unable to fully verify the system. Automated testing is a necessary technology for promoting the construction of OS2 in China Southern Power Grid.

References

[1] WANG Jifeng: Southern Power System Technology Vol. 6 (2012), no. 2. P. 1-5
[2] WANG Jifeng: Automation of Electric Power Systems Vol. 35(2011), no. 34. p. 1-6
[3] Daniel J.Mosley, Bruce A.Posey: Software testing automation (Machinery Industry Press, Beijing 2002)
[4] Dorothy Graham, Mark Fewster: Experiences of Test Automation: Case Studies of Software Test Automation (Pearson Education Press, 2012.)
[5] LI Feng, SHENG Zhuang in: Action-driven automation test framework for Graphical User Interface (GUI) software testing, Volume 17-20 of IEEE Autotestcon (2007), no. 9. P. 22-27
[6] DONG Nana, ZHAN Huiq in: Electronic Testing, Vol. 11(2010), P. 47-50
[7] MO Xi, ZHAO Fang: Computer engineering, Vol. 35 (2009), no. 21. P. 78-81
[8] G. Whyte, D.L. Mulder, in: The Impact of Software Test Constraints on Software Test Effectiveness, Proceedings of the 2nd International Conference on Information Management and Evaluation (ICIME11 Toronto), Academic Publishing International Limited (2011), P. 450-510