TECHNIQUES FOR AUTOMATING HARDWARE-IN-LOOP TESTING OF PROTECTIVE RELAYS AND PROTECTION SCHEMES

B S Rigby

etalumiSe (Pty) Ltd

INTRODUCTION

Real-time simulator based factory acceptance tests (FAT) are by now a necessary and well-established part of the process of developing and implementing new protection schemes by electric power utilities, in part because of the sheer complexity of modern numerical protection relays and the rapid rate at which the functionality of such relays continues to evolve and be improved by their manufacturers. Such FAT testing commonly takes the form of mandatory testing of a particular manufacturer's protection relay by a utility to allow the utility to satisfy itself that the particular device being considered meets the requirements for future usage within its protection schemes based on its general settings design philosophies and particular operating requirements.

However, an emerging approach for some utilities is to require that, before each new protection scheme can be approved for commissioning on their networks, real-time simulator based FAT testing must be carried out on the protection scheme as a whole (i.e. on all the relay hardware making up the protection scheme, together with its final settings), for a range of realistic network operating conditions. In such instances the FAT tests would typically involve a complete simultaneous performance analysis of the Main 1 and Main 2 protection relays, as well as of any reclosing relays, for both ends of the protected line, including communications between the relays at each end of the protected line. Although real-time simulators are purpose-designed specifically to be used to carry out the kind of closed-loop testing of protective relaying hardware required by such FAT tests, nevertheless, the data recording desired by utilities during such FAT tests, in particular, can present considerable logistical challenges.

This paper will start by providing an overview of the approaches typically used to carry out real-time simulator based FAT testing (and associated data gathering) of protection equipment in general, and will describe the tools and techniques available to allow automated application of the large numbers of test-shot conditions typically required during FAT tests, as well as automated recording of the test results to properly-formatted reports in a coordinated manner by the simulator itself during such tests. The paper will then also describe one particular approach that can be used to carry out automated gathering of trip logs and fault records from multiple different protection relays during FAT testing of full protection schemes in a manner that is centrally coordinated by the simulator being used to conduct the fault tests after each test-shot has been completed.

REAL-TIME SIMULATOR TESTING OF HARDWARE PROTECTION RELAYS

A real-time digital simulator allows actual power system equipment, such as protection relays, to be connected in closed loop with a real-time simulation model; this hardware in the loop (HIL), closed-loop arrangement enables detailed testing and investigations to be carried out into the performance of protection schemes under highly-realistic conditions [1].

Fig. 1 is a schematic diagram showing how protection relay hardware is connected in closed-loop with a real-time simulation model of the protected plant for the particular case in which the performance of a full line protection scheme is to be evaluated as part of factory acceptance tests (FAT tests) of the scheme using a real-time simulator. In this particular case the transmission line whose protection scheme is being tested is one of two parallel lines, both of which are suspended from a common set of double-circuit line towers. In such cases of parallel lines suspended from common transmission towers, the conductors of the two lines are located sufficiently close to one another that the mathematical representation of the parallel lines in the real-time simulation model needs to include the effects of mutual magnetic coupling between them, since this mutual coupling can strongly influence the performance of the impedance (distance) functions in the line protection relays. Consequently, in the case of such lines, particular categories of tests are included within the FAT test schedule in order to ensure that the performance of the line protection is evaluated thoroughly for all practical and worst-case operating scenarios in which mutual coupling is known to influence protection performance.

The generic diagram in Fig. 1 depicts the sources feeding each end of the lines under study using dashed lines to highlight the fact that the actual real-time simulation model will typically include appropriate representations of additional surrounding loads, lines and generators so as to properly represent the infeeds at each end of the protected line under steady-state and faulted conditions. As a natural consequence of this modelling approach, the real-time simulation model will properly represent the relative strengths of the sources feeding each end of the transmission line whose protection is being tested. The relative strengths of the

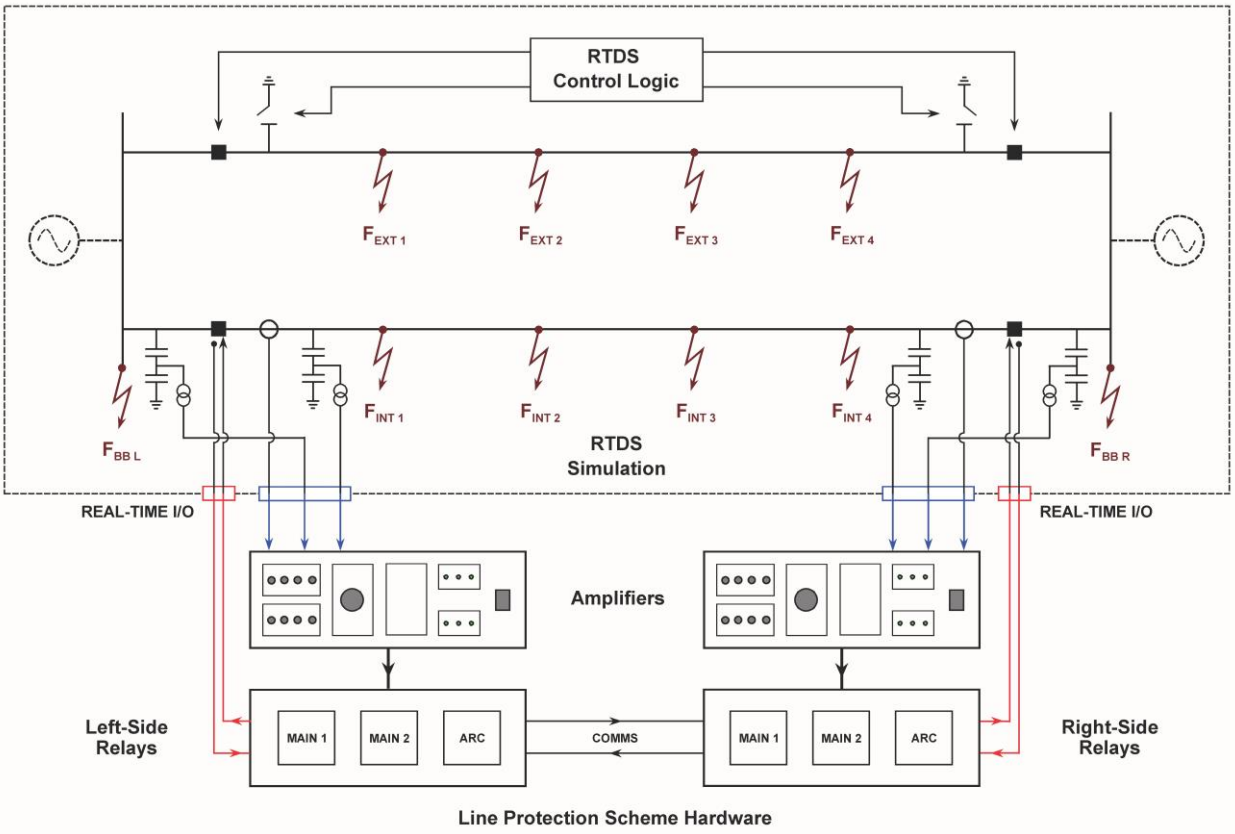

Fig. 1: Schematic diagram showing HIL connection of a full line protection scheme to a real-time simulation model of the protected plant for FAT testing of the complete protection scheme and its final settings.

sources feeding each end of the line can have important influences on the performance of the line protection, and should therefore be represented properly during the FAT tests of the scheme and its settings.

Fig. 1 also illustrates the complete hardware-in-loop connection of the relays in the protection scheme under test (Main 1, Main 2 and Auto-Reclose as applicable). Detailed models of the current transformers and capacitive voltage transformers at each end of the protected line in the real-time simulation model feed the external protection relays with live, real-time injections of the secondary measurement quantities. These live secondary-measurement quantities can be fed from the real-time simulation to the relay hardware under test from analogue output cards on the simulator, via high-bandwidth amplifiers (as shown in blue in Fig. 1) for tests on a protection scheme comprising conventional analogue measurement inputs, or directly in the form of IEC 61850-9-2 compliant data streams via Ethernet for tests on a protection scheme comprising sampled value measurement inputs. The live status of each pole of the circuit breakers at the two ends of the protected line within the real-time simulation model is likewise fed to the relays, and the trip and close signals from the external relays are fed back into the real-time simulation model to operate these circuit breakers in response to any relaying decisions taken by the protection scheme during each fault scenario considered in the FAT tests. These live binary quantities passed between the real-time simulation and the relay hardware under test can be in the form of conventional binary I/O or in the form of IEC 61850 GOOSE messages as required by the particular scheme under test.

The real-time simulation model is configured to allow a range of faults to be applied both internally to the protected line, or externally to the protected line (either on the parallel line, or on the busbars at the left and right ends of the protected line). Depending on the specifications called for in the customer's FAT test schedule the faults applied during a particular test may comprise only internal faults, only external faults, or combinations of internal and external (cross-country) faults. In addition, in cases where the real-time simulation model includes representations of additional surrounding loads, lines and generators, it is possible to conduct external fault tests at other locations to those shown in Fig. 1 in order to verify, for example, the performance of over-reaching back up impedance zones of the line protection.

In the example in Fig. 1, the real-time simulation model also includes the circuit breakers and earth switches of the mutually-coupled parallel line. These circuit breakers and earth switches are controlled by specially-designed control logic within the real-time simulation model itself according to the requirements of the FAT

test schedule. This control logic may be used to either keep the parallel line open and earthed during the tests on the protected line, to ensure that the parallel line remains in service during the tests on the protected line, or to replicate the response of the protection relays on the parallel line during external fault tests, as specified by the particular requirements in each section of the customer's FAT test schedule.

The diagram of Fig. 1 thus shows how a real-time simulation environment allows detailed, hardware-in-loop testing of a full protection scheme, operating in closed loop with the protected plant (including all inter-relay communications) in response to a range of detailed, practical fault scenarios. The real-time simulation environment provides tools not only to apply the test fault scenarios (test shots) specified within the customer's FAT test schedule, but also, for each test shot, to record time plots of the pre- and post-fault responses of both the simulated plant and the externally-connected protection equipment (in the form of its trip and close signals fed back into the real-time model). It is also possible to use the real-time simulator to record additional relaying variables from the protection scheme under test by marshalling these variables to any spare binary output contacts available on the relays, and then connecting these outputs to spare digital input channels on the simulator. However, the number of variables that can be recorded in this manner from any one protective relay may be limited by the number of spare binary outputs available on that device, and in the case of FAT tests on entire protection schemes, a further limitation may be the number of spare digital input channels on the simulator itself that are available to record binary variables from several different devices in the scheme under test.

Nevertheless, during FAT tests it is common practice to use the real-time simulator to record as many additional relaying variables as possible (over and above the trip and close outputs) from each hardware device in the protection scheme to aid in the analysis and interpretation of the scheme's response to each test shot. For example, in the case of impedance protection functions within a scheme under test, variables such as forward and reverse zone pick up, earth fault blocking, permissive carrier send and receive and switch on to fault signals are typically recorded; in the case of differential protection functions within a scheme under test, variables such as differential earth fault forward and reverse pick up, differential current threshold pick ups ($I_{DIFF}>$ and $I_{DIFF}>>$) and differential earth fault carrier send and receive signals are typically recorded.

ADDITIONAL CONSIDERATIONS FOR HIL TESTING OF FULL PROTECTION SCHEMES

Once a real-time simulation model and hardware-connected protection scheme such as that shown in Fig. 1 has been developed on a real-time simulator, each test shot called for in the FAT test schedule can be applied manually, and the results saved manually, by a user from a Windows PC based graphical user interface programme that is used to execute the real-time simulation model. However, the number of test shots that have to be carried out for a FAT test of a scheme such as that shown in Fig. 1 can be considerable (typically hundreds of unique test conditions per scheme) and the configuration of the control logic inputs required to correctly set up each one of these unique test shot conditions can be complex. Hence, even when conducting the FAT tests one test shot at a time, in order to minimise human operator error it is common to make use of automation tools provided within the real-time simulation environment [2] to develop pre-programmed, and carefully pre-checked, scripts that correctly set up the test conditions and apply the faults prescribed for each test shot within the FAT test schedule.

These automation and scripting tools provided within the real-time simulation environment can also be used to automatically output the results recorded by the simulator from each test shot (in the form of both analogue and binary time plots and text-based summaries) to a properly-formatted test report in the form of a Microsoft Word document, and then to save this test report using a filename indexed to the particular test shot number. An overview of how this automated test and reporting procedure can be set up using the scripting tools single-shot operation of the tests in a FAT test schedule (i.e. manually conducting one test shot at a time) is illustrated in Fig. 2.

EXTENDED AUTOMATION OF FAT TESTS

Batch-Mode Application of FAT Tests

The approach described in the previous section can be considered as a partial automation of the process required to conduct FAT tests on a real-time simulator. However, there are two further ways in which the process of automating the execution of FAT tests on protection schemes can be extended. Firstly, once an automation script such as that shown in Fig. 2 has been developed and tested it is possible to add conditional branches and flow control loops to the automation script in order to allow a specified range of test shots from the FAT test schedule to be carried out as a batch-mode task as illustrated in Fig. 3.

Fig. 3 shows that, if so chosen by the user, this automation script can now be operated in multi-shot mode: in this mode the script will then repeatedly configure the required test conditions and apply the required faults for each test shot within the specified range, and generate a formatted test report for each such test shot. Once such an automation script has been thoroughly tested, it can be used to run multi-shot tests on the full protection scheme in batch mode, unattended, over night or over a weekend, and once all of the tests in the batch have been completed the formatted test report files for each test can then be

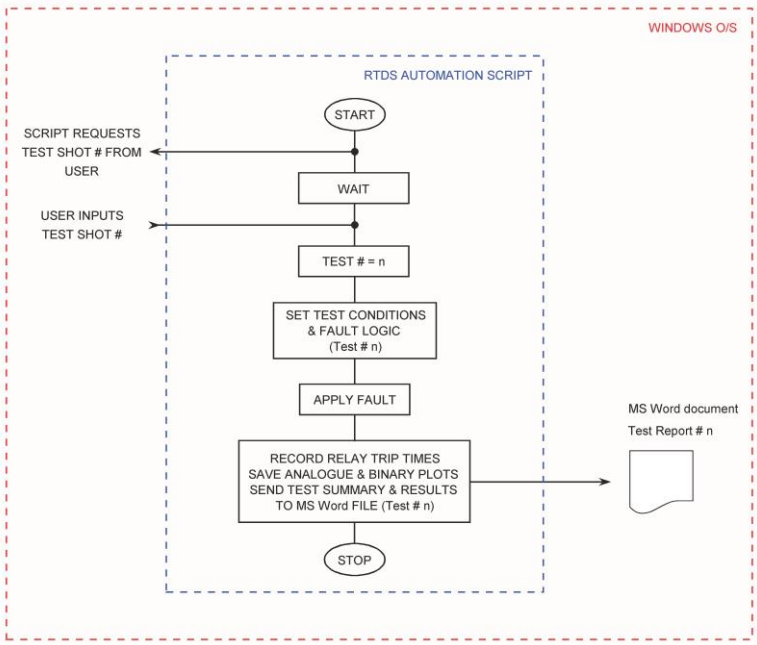

Fig. 2: Flow chart of a real-time simulator script file used to automate the configuration of test conditions, apply fault control logic, and generate a test report when conducting FAT tests manually, in one test shot at a time mode.

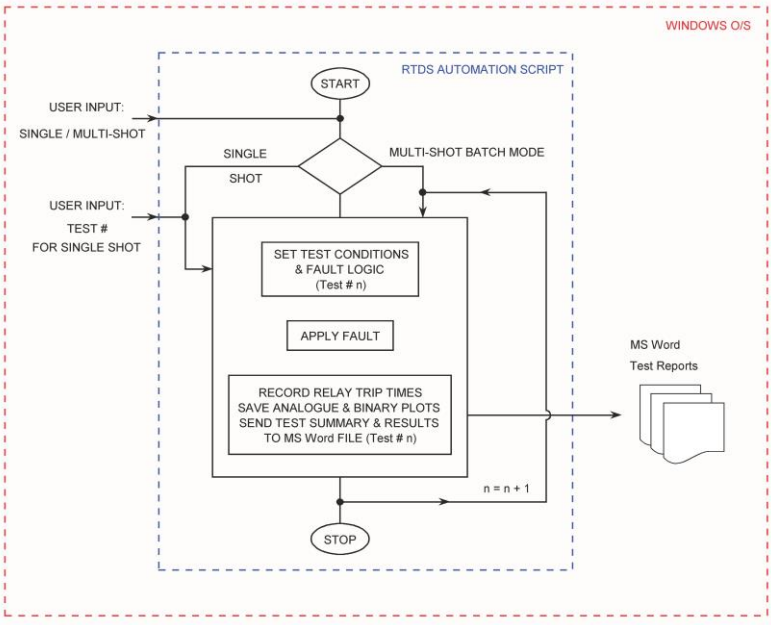

Fig. 3: Flow chart of a real-time simulator script file that allows automated application of a range of test shots from a FAT test schedule, in batch mode.

inspected by the user. Note, however, that in the example shown in Fig. 3 the structure of the extended script file has been designed still to allow, if chosen by the user, operation of the automated application of test shots from the FAT test schedule in single-shot mode. This facility is important, since, after a range of tests have been conducted in batch mode, and the test reports inspected, it may be necessary to repeat selected test cases of interest for the utility customer witnessing the tests, or in order to aid in iterative revision of the scheme's settings if any shortcomings are identified during the batch-mode test runs.

It should also be noted that, in the extended automation approach shown in Fig. 3, the test reports generated for each test shot within the FAT test schedule contain only those variables that are available within the real-time simulation environment. Even when the real-time simulator is used to gather an extended set of diagnostic binary variables from the external relays in the

protection scheme under test, and these additional variables are included in the graphical results saved in the test reports for each test shot, such test reports are saved in the format of a Microsoft Word document and cannot therefore be used later for the kind of diagnostic analyses commonly carried out using software tools that allow post-processing of fault records.

One simple way to allow for later, post-test diagnostic analysis of each captured test shot is to include within the automation script a command to output each set of captured test results as a COMTRADE file (in addition to generating the required test report in Word document format). This is a straight forward feature to add to the automation script, but such COMTRADE files will still be limited by the number of variables that can be retrieved from each of the many relays under test. Furthermore, some utilities stipulate, as part of the documentation and reporting requirements for their FAT tests, that full trip logs and fault recordings must be downloaded from each relay in the scheme under test, and saved along with the results recorded by the real-time simulator itself, for each individual fault test-shot carried out as part of the FAT testing schedule. This latter requirement by some utility customers can make extended automation of FAT tests more challenging to achieve, particularly if – as is not uncommon – the schemes being tested comprise four to six different relays, and not all of the relays are from the same manufacturer.

Automated Retrieval of Information from External Relays

In cases where the end customer requires that full trip logs and fault recordings be retrieved from each relay in the scheme during FAT tests, one option is simply to retrieve and save this information manually from each relay after completion of each test shot, using the relay manufacturers' proprietary software tools. However, this approach can take a considerable amount of time for FAT test schedules involving hundreds of unique test shots on protection schemes comprising several relays. Alternatively, if one wants to conduct such FAT tests in automated batch mode, a mechanism is required whereby the process of retrieval of information from the relays in the scheme following each test shot can itself be automated, and in a way in which this automated information retrieval can be coordinated and controlled from within the scripting environment used to control the operation of the real-time simulation. Although the real-time simulator's scripting tools do provide a mechanism to communicate directly to some manufacturers' relays (using MODBUS commands), it may not be possible to use this approach to communicate to all of the relays being tested in any particular scheme. Furthermore, even for those relays that do support MODBUS communication, such an approach might not be suited to the specific task of retrieving fault records and trip logs from the relay in the particular format needed by a utility customer.

However, the scripting automation language in the real-time simulation environment also provides a *system* command (similar to the system call in the C programming language) that will allow any external Windows program or executable file to be invoked from within the automation script and will then wait until the invoked program finishes execution. This system command can be used in different ways to achieve automated retrieval of information from external relays under test, two of which are discussed below.

Information Retrieval Using Commercial Third-Party Automation Software. Since relay manufacturers generally all have proprietary Microsoft Windows based software that can be used to retrieve trip logs and fault records from their own relays, in principle each hardware relay in a protection scheme undergoing FAT tests can have its trip logs and fault records retrieved using that relay manufacturer's own Windows-based proprietary software tool, but under the master coordination of the real-time simulator being used to conduct the FAT test: what would be required would simply be some form of intermediate software program capable of automating the tasks that would otherwise have to be carried out by hand when using the each relay's proprietary software tool to retrieve information from that relay.

There are a number of commercially-available software programmes that can be used for automating tasks in the Windows operating system, each providing varying degrees of flexibility and robustness. Of these available programmes, the ones most suited to this application are those that allow automation tasks to be programmed using a structured macro programming language (supporting conditional statements and control loops, and with proper debugging tools provided) and which allow compilation of a developed automation procedure into a stand-alone executable programme that can be invoked from within the real-time simulator's own automation environment.

Fig. 4 shows a flow chart of a script file to extend the automation of FAT tests so as to include retrieval of fault records and trip logs from each protection relay in a scheme under test using this latter approach. The structure of the automation script's main flow control loops is unchanged from that shown in Fig. 3, i.e. the automation of the FAT tests within the real-time simulation environment is still designed to be able to operate in either single-shot mode or multi-shot batch mode when carrying out fault scenarios specified within a utility's FAT test schedule. However, when operating in either of these modes, the automated script within the real-time simulation environment uses a system call after each completed test shot to invoke a separate executable programme in the Windows operating system environment (developed using a third-party Windows automation programming tool), which in turn schedules the tasks and actions needed to retrieve fault

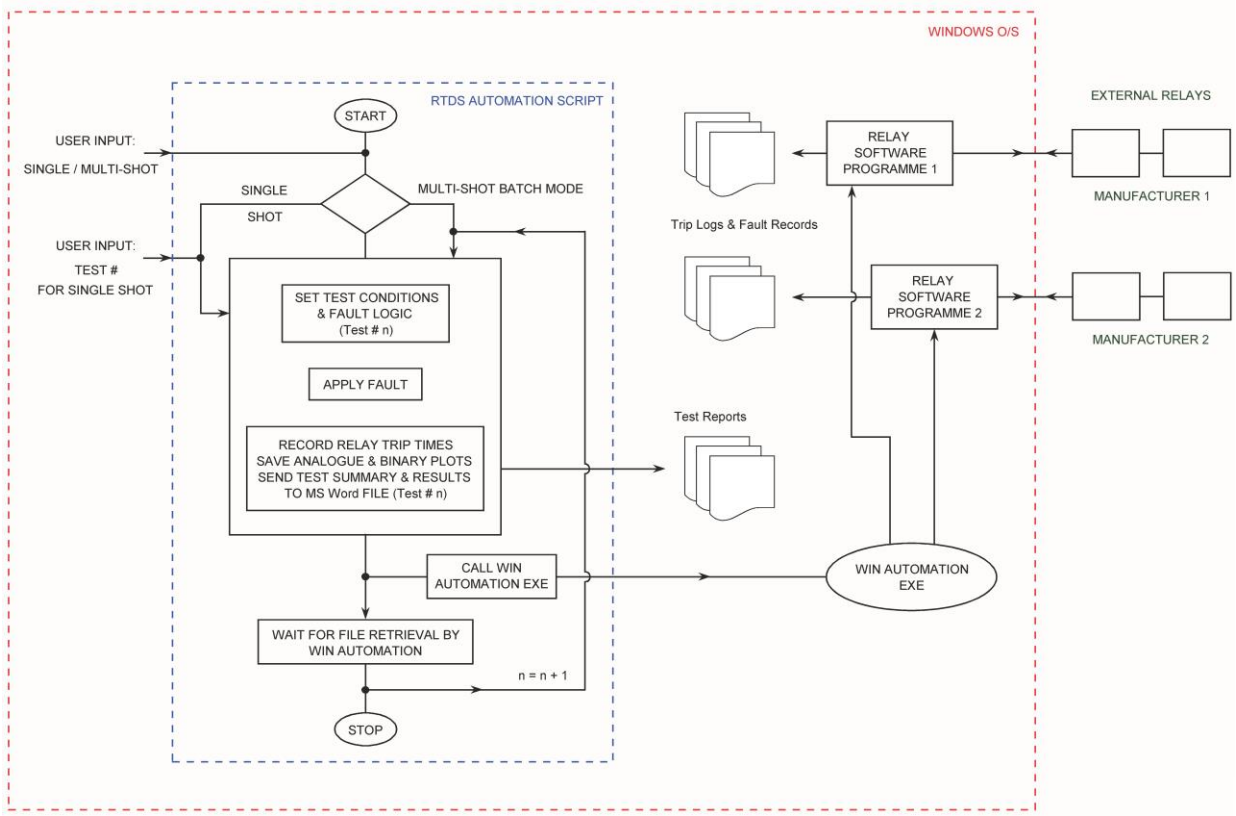

Fig. 4: Flow chart of a real-time simulator script file that allows automated application of a range of test shots from a FAT test schedule, in batch mode, using a system call to a separate, Windows-based software tool used to automate retrieval of records from multiple relays following each test shot.

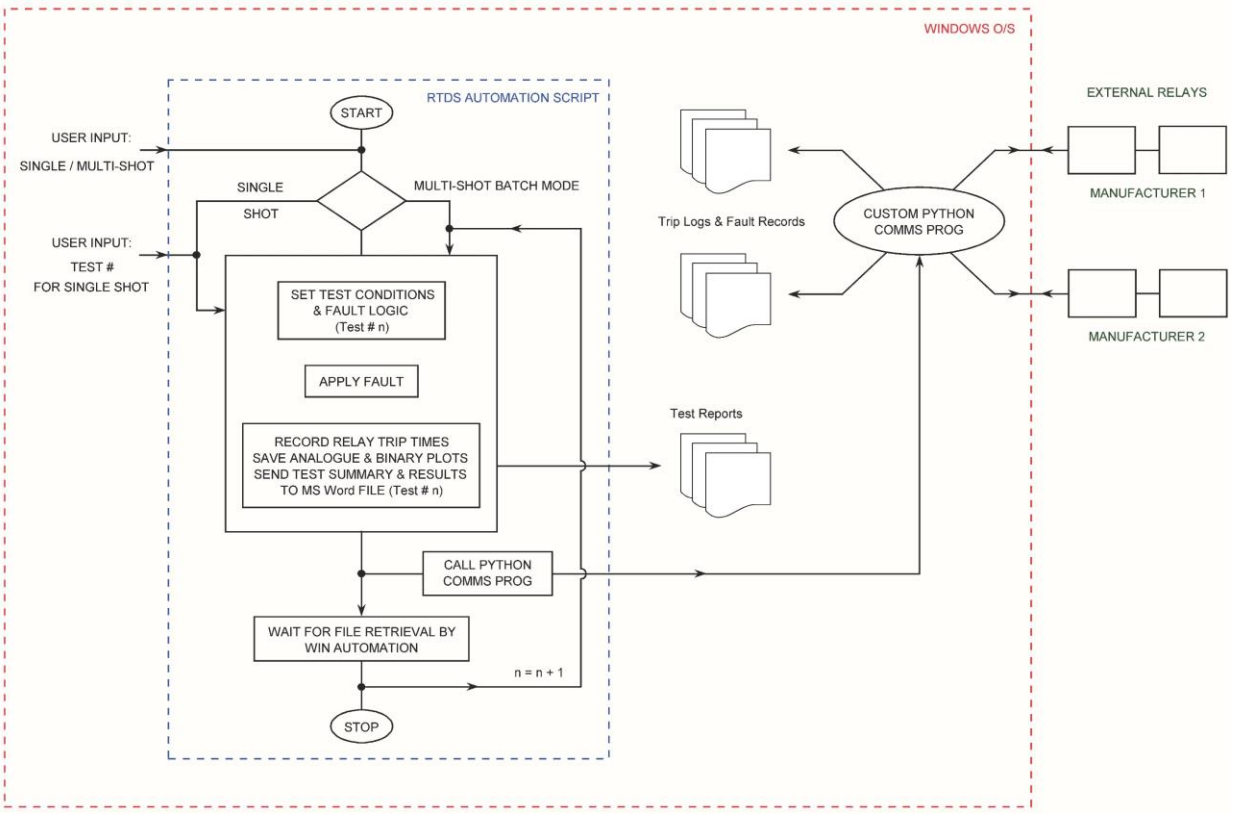

Fig. 5: Flow chart of a real-time simulator script file that allows automated application of a range of test shots from a FAT test schedule, in batch mode, using a system call to a separate, custom-written Python program used to retrieve records from multiple relays following each test shot.

records and trip logs from each protection relay under test, using that relay's own proprietary Windows-based software tool in each case.

The approach shown in Fig. 4 can be implemented practically provided that each relay vendor's proprietary software tools have been written so as to allow all of the necessary user-operated menu and sub-menu commands, pop-up windows and dialogue boxes used when retrieving event records from a relay to be openly accessible and controllable by means of the macro-language programming commands of the third-party automation software being used. In practice, however, it has been found that not all relay vendors' software tools are amenable to this approach.

Information Retrieval Using Custom-Written Communication Code. A more robust and reliable method to retrieve information from external relays under test is to write custom executable code to communicate directly with, and fetch data from each relay, via one of its communications ports, making use of low-level communications protocol commands provided by the relay manufacturer for this purpose.

Fig. 5 shows a flow chart for extended automation of FAT tests in which a custom-written communications program, written in the Python language, is used to respond to the system call after each test shot carried out by the real-time simulator's scripting language. When called by the real-time simulator's scripting language, this custom-written Python communications program communicates with each of the relays under test over Telnet, using the commands of that relay's proprietary low-level communications protocol, in order to retrieve the event record for the test shot in question from each relay and store it in a format that can later be opened and analysed using that relay vendor's specific post-processing software tool.

CONCLUSION

This paper has described techniques that have been used to successfully carry out real-time simulation based FAT testing of protection relay hardware. Whilst real-time simulators provide all the tools necessary to automate the large numbers of test shots typically required for FAT tests, carrying out such tests on full protection schemes with their final settings can nevertheless be a cumbersome and time-consuming process if the utility customer requires retrieval of trip logs and fault records from every relay in the scheme following every test shot, particularly when the scheme comprises relays from different manufacturers. The paper has described two approaches that can be used to speed up FAT testing under such conditions by using either a third-party software programme or custom-written communications code to automate the tasks involved in retrieving records from different manufacturers' relays, coordinated from within the real-time simulator environment.

REFERENCES

[1] McLaren P G, Kuffel R, Wierckx R, Giesbrecht J, Arendt L, "A Real Time Digital Simulator For Testing Relays", *IEEE Transactions on Power Delivery*, Vol. 7, No. 1, Jan. 1992, pp. 207 – 213.

[2] Rigby B S: "Automated Real-Time Simulator Testing of Protection Relays", Proceedings of the IEEE PowerAfrica 2007 Conference, Johannesburg, July 16 – 20, 2007, ISBN 1-4244-1478-4.

[3] Kuffel R, Forsyth P, Meiklejohn H, Holmes J, "Batch Mode Operating Software for Relay Test Applications of the RTDS™ Simulator", Proc. EMPD '98, Singapore, March 1998, pp. 356 – 361.