Functional Testing: A SOA & BPM Approach

William Gontier[1], Franck Hennequin[2], Fabien Lloansi[2]

[1] Cygnus Systems S.A.R.L., Meudon sur Seine, France
[2] SoftAtHome S.A., Nanterre, France

E-mails: william.gontier@cygnus-systems.eu, franck.hennequin@softathome.com, fabien.lloansi@softathome.com

Abstract— **In this paper, a new approach to automate software reliability verification and validation activities is described. The project has been developed focusing on functional testing of digital television and network home appliances. Nevertheless, at this stage of the project, the developed approach has proven to be sufficiently generic to support a wide range of domains and so it can be of interest for people dealing with functional test automation in a wide industrial range. Most commonly used approaches followed by the industry to automate functional testing require important efforts and skilled human resources in software development to build large sets of specific scripts. Consequently, these approaches cannot be conducted by professionals without programming skills since they are not sufficiently involved in the design, development and maintenance of the tests scenarios. We present in this paper an innovative strategy to overcome the main difficulties. We also show how this new strategy based on a zero-code approach can offer new exciting roles to test team members and deliver an optimized cost structure of test automation activities.**

Keywords-Functional Testing; Test Automation; Zero Code; Cost Optimization; Service Oriented Architecture; Business Process Modeling.

I. INTRODUCTION

The software development activities are each day more "test driven". The main reasons for such a trend mainly lie in the increasing complexity of the developed systems, the integration of third-party software modules (commercial and open source), the interactions with external systems partially mastered, the pressure put on the R&D teams with respect to constraints of "time to market", the difficulties in getting clear, complete and detailed specifications before the project starts, the widespread and success of agile methods which significantly helped software organizations detect the benefits of test driven development methodologies.

As an immediate consequence, functional testing is each day a more strategic step in the product development cycle but it is also a more costly activity for software organizations due to the increasing number of tests required to deliver adequate test coverage of products.

Thus, poor testing coverage and/or inadequate test automation strategies and/or inadequate testing tools can prove to be detrimental to the competitiveness in terms of product quality and cost.

In this context, automating functional tests becomes each day a more challenging topic for software organizations. This paper reviews the commonly followed approach in the industry to carry out test execution and automation, for which the creation of scripts is based on coding, possibly simplified with an interface to abstract this layer. Identifying the key weaknesses of these approaches, we present an innovative strategy to overcome the main difficulties encountered in automating functional test activities. We show how this new strategy improves the quality of execution during functional test campaigns while providing an optimized cost structure.

Beyond this introduction, Section 3 presents the most frequently encountered test execution and automation strategies in the industry to compare them with the approach described in this paper. We also introduce the Saturn software framework implementing this new strategy in the following section. Saturn is currently used for functional testing of products developed by the company, SoftAtHome, including its digital television receivers/decoders and xDSL/fiber Internet gateways. Section 4 deals with the main results delivered by Saturn. Finally, before concluding, Section 5 introduces the evolutions of Saturn that are currently under development.

II. State-of-the-art

An efficient and fully automated system must combine an easy interface to generate automation tests (no coding skill required) and the possibility for users to add their own modules (new functionalities or devices).

Main actors in automated testing proposed solutions based on scripting that requires lengthy and expensive coding phases (often in python language). In order to bring more flexibility for the user, these solutions can include a package of additional libraries (for example, *RT-RK* company anticipated from future users include list of libraries [7]). These solutions require coding skills that often differ from tester to tester, according to their profiles.

To abstract the coding layer, they subsequently developed additional interface (drag and drop). User can access to the toolbox that simplifies scripts developments. However these glue layers are fully linked with the automation tool and the user of this tool is dependent on these solution concept companies to supply him with future evolutions of their product (sometimes the solution needs a proprietary language, as *StormTest* [8], the solution developed by *S3 Group*). Consequently, there is no more versatility for the user to generate evolutions.

Our new approach is to directly start with a modeling system (BPM) that can interface with a toolbox suitable. In others words, to use a system adapted for sequencing and add application layers used through independent connectors.

III. Strategies for Test Automation

A. *Test Execution & Automation* ***methods***

Manual testing generally delivers imprecise test results and fails to reproduce tests, mainly because it is largely subject to the interpretation of a human operator whose judgment is more likely to evolve over time.

On the other hand, test campaigns using robots can significantly improve the stability of test procedure results and reproduction over time (which is essential, for example, in the case of tests performed within the context of certification processes).

Nevertheless, it is quite frequent to see test automation reduced to its simplest form, and thus, limited to the development of scripts specifically developed for the product to be tested. Such scripts are sometimes even developed by the teams who have contributed themselves to the development of the tested product.

Many experiences show that this approach generally gives poor results. Indeed, if a test case can be decomposed as a succession of steps to execute, this structure can't be kept with a script in a coding language. Consequently, the understanding of an automation script will required a code review.

Furthermore, this bias, associated with the implementation of complex tests for which developers in charge may lack time and/or skills to implement the complex algorithms required to replace human skills (e.g., computer vision), leads the tests implementation to be based, most of the time, on optimistic use cases.

In addition, some defects resulting from omissions committed during the design and / or development will typically be perpetuated when the product and the test scripts are done or described by the same developers. Thus, using the common approach, the automation process leads both to replace a human operator by an automatic operator, but also to adapt the tests to be implemented within a reasonable time frame and since its use requires programming skills, thus needing a engineering profile different from that of an expert in validation.

Consequently, with the common approach to test automation, significant bias can be introduced, typically leading to degraded functional tests coverage.

Just the opposite, the approach followed by the Saturn project (modeling system BPM) ensures that the test team - whose role is to systematically search for defects of any kind in the products they have in charge - retains its prerogatives during the design, the development, the maintenance and the execution of the test campaigns.

To achieve this goal, Saturn delivers to test team members a suitable toolbox to deal with all the activities related to test scenario management but without requiring expertise in the field of computer programming.

Furthermore, the Saturn toolkit provides the tools necessary to replace human capacities in the field of testing activities. This includes, for example, computer vision algorithms, image quality assessment (taking into account the human visual system), identification of soundtracks, and more others features.

In addition, Saturn tools can provide valuable information about the diagnosis of the system due to their inherent ability to quickly process large amounts of data (e.g., protocol analysis).

B. *Costs of Test Execution & Automation*

While manual testing triggers operating expenses (OPEX) without offering any possibility of cost sharing on the number of test campaigns to be achieved on time, the development of tests automation tools constitutes an investment opportunity (CAPEX).

It is important to note that compared to the traditional automation approach (i.e., specific scripts development), the approach proposed by Saturn, which consists in offering a generic and reusable toolbox, can generate some revenue. Indeed, generic tools can be marketed and therefore likely, to attract customers and business partners in a given industry. These business opportunities can help finance investments in tests automation through income generation. Furthermore, the development of generic tools is susceptible to help in extending the amortization period of the developments due to a longer depreciation period thanks to a better sustainability over time of the developed tools. All these aspects contribute to minimize overall labor costs since it spares using engineers for the development of automatic tests. In the Saturn approach, only widely reused generic tools require contribution from specialized software developers while as automated tests are implemented by technicians.

These considerations, summarized by in Figure 1, make the strategy deployed through Saturn quite an optimized strategy in terms of TCO (Total Cost of Ownership) of an automated test infrastructure.

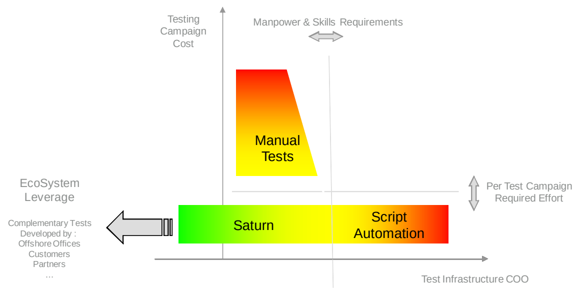

Testing Campaign Cost
Manpower & Skills Requirements
Manual Tests
EcoSystem Leverage
Per Test Campaign Required Effort
Complementary Tests Developed by : Offshore Offices Customers Partners …
Saturn
Script Automation
Test Infrastructure COO

Figure 1. Cost structures of manual testing vs. script developments vs. Saturn approaches.

IV. Implementation

To enable test team members to develop their test scenario without requiring computer programming skills, the Saturn system had to provide a way to describe tests procedures in a graphical form.

A. *The Business Process Modeling Approach*

The approach consisted in selecting the BPMN (Business Process Modeling Notation) language in its 2.0 version [6]. This language offered all the key characteristics for the development and the maintenance of test scenarios, as well as both easy to learn and intuitive to use.

With our approach, a test scenario is developed as a BPMN process made of connected activities (cf. Figure 2). Each connection can activate - at the next execution step - an activity if it is connected to the currently active activity and the condition associated to the connection is evaluated to TRUE at execution time. An activity can execute a sub-process, perform some local actions such as updating local or global variable values, or call connectors. Connectors are typically predefined routines performing some frequently required tasks. In the case of Saturn, the connectors are employed to access the toolbox API (Application Programming Interface). Each Saturn connector implements a web service call to a wrapper delivering a specific service (e.g., checking the presence of a given pattern on the television screen thanks to the computer vision wrapper managing video acquisition hardware). Connectors are used by the scenario developer as a way to access services delivered by the wrappers of the Saturn framework and so, interacting with the external devices to be tested. The state of the tested device is known from the test scenario thanks to the results returned by the wrapper calls.

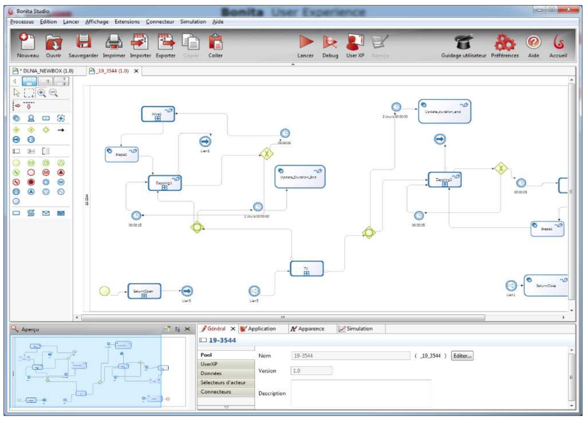


Figure 2. Test scenario example developed in BPMN 2.0 with the Bonita Studio editor. Thanks to the BonitaSoft solution the BPMN 2.0 files are then migrated to the Java framework and executed on a JBOSS server.

B. *The Wrappers*

The wrappers are server applications hosted by the tests robots whose role is to provide the services required to test scenarios for what concerns specialized functions typically interacting with the physical world (e.g., pattern detection on a television screen).

Wrappers typically incorporate SDK (Software Development Kit) to manage the robots' hardware components (e.g., video capture card) and / or specific algorithms (e.g., audio identification algorithms). A description of the main algorithms developed for the project can be found in [1][2][3][4].

With our approach, the BPMS (Business Process Modeling System) acts as a sequencer calling - through Java connectors - the services rendered by the wrappers (Figure 3). Communication between the BPMS and the wrappers is done via a session oriented API whose structure is common to all the wrappers. At this date, the main Saturn wrappers are:

- Vision (shown in Figure 4) deals with the computer Vision algorithms toolbox such as pattern matching, video detection, screenshots, optical character recognition, etc.
- Audio dealing with audio processing services such as audio watermarking, audio detection, audio track identification, etc.
  Audio contents are tagged using different amplitude modulation (cf. Figure 5)
- Studio (shown at Figure 6) dealing with audio/video content management services such as stream generation, video frames identification, video quality assessment (PSNR, SRSIM, etc.), "lip synching" computation.
- Web UI dealing with Internet browser control used for web user interface and web services testing.
- Power dealing with external devices power supply management services.
- RCU dealing with remote control unit services, such as infrared and radio frequency based remote control simulators.
- Traces dealing with equipments traces and logs management services.

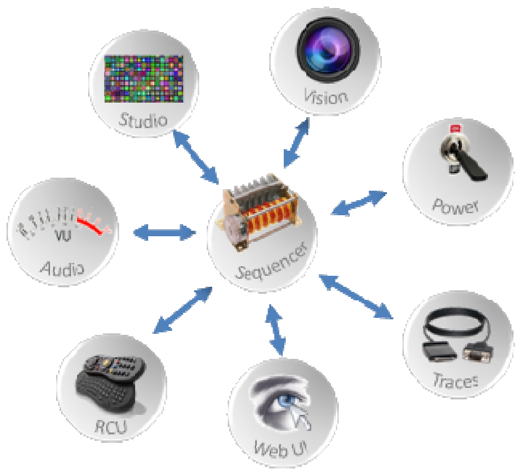
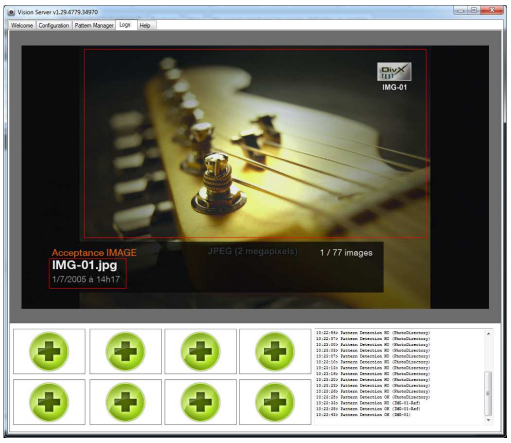


Figure 3. SOA (Service Oriented Architecture) illustrated: test scenarios executed by the BPMS (Business Process Modeling System) server interact through a standardized web services API with the wrappers applications hosted by the robots.



```
10:22:54> Pattern Detection KO (PhotoDirectory)
10:22:57> Pattern Detection KO (PhotoDirectory)
10:23:00> Pattern Detection KO (PhotoDirectory)
10:23:03> Pattern Detection KO (PhotoDirectory)
10:23:07> Pattern Detection KO (PhotoDirectory)
10:23:10> Pattern Detection KO (PhotoDirectory)
10:23:13> Pattern Detection KO (PhotoDirectory)
10:23:16> Pattern Detection KO (PhotoDirectory)
10:23:20> Pattern Detection KO (PhotoDirectory)
10:23:23> Pattern Detection KO (PhotoDirectory)
10:23:26> Pattern Detection KO (PhotoDirectory)
10:23:28> Pattern Detection OK (PhotoDirectory)
10:23:33> Pattern Detection KO (IMG-01-Ref)
10:23:35> Pattern Detection OK (IMG-01-Ref)
10:23:43> Pattern Detection OK (IMG-01)
```

Figure 4. Saturn powerful computer vision system. Example of patterns recognition and localization on a set top box video output.

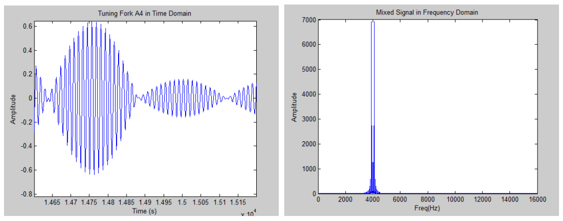


Figure 5. Audio watermarking by amplitude modulation (resp. time and frequency domains).

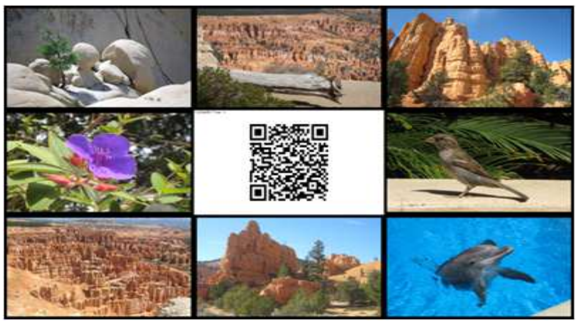


Figure 6. Video frame identification and clock synchronization by QR Code insertion/decoding.

C. *The Catcher Application*

As shown in Figure 7, the catcher application, allowing the test execution infrastructure to communicate with the information system in charge of the test plans and test results management, plays a central role in Saturn.

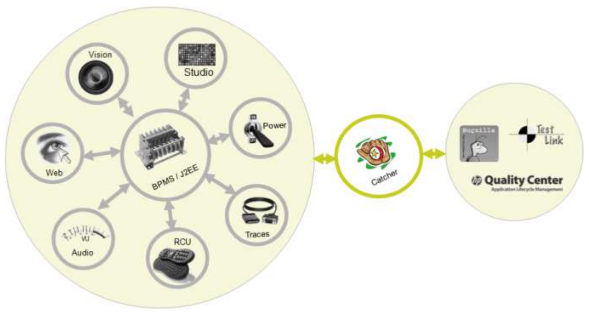
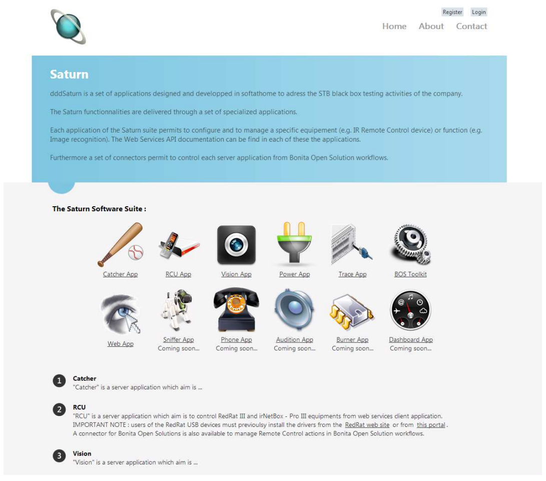


Figure 7. The "catcher" application in the system's architecture.

The Catcher application main roles consist in extracting the test campaigns descriptions stored in a third party application of a test management system (for example, TestLink, HP Quality Center), in presenting tests scenarios to the operator in different levels of aggregation such as unit testing, test sequence or functional modules, in deploying the Java code corresponding to the tests to be executed by the JBOSS server, in controlling the execution of the test scenarii (e.g., abort a test sequence in case of critical error), in collecting traces obtained during tests execution, in attaching these traces and various additional information (e.g., TV screen captures) to the test reports, in posting the test results to Testlink, HP Quality Center or any test management third party application, in keeping track of files versions, in managing the files versioning system (as *Git* [11]) and in centralizing the Saturn's configuration parameters.

D. *The Saturn Portal*

The Saturn web portal shown in Figure 8 has been developed to provide a single point of access to system's users. It mainly hosts: the wrappers applications providing versioned automatic updates of the applications installed on the robots, the files repositories for multi-sites deployments (tests scenarios written in BPMN 2.0, Java classes, logs, screenshots, test traces, A/V streams, patterns, etc.), the wrappers' databases, the Saturn portal and the user documentation.



Saturn

dddSaturn is a set of applications designed and developped in softathome to adress the STB black box testing activities of the company.

The Saturn functionnalities are delivered through a set of specialized applications.

Each application of the Saturn suite permits to configure and to manage a specific equipement (e.g. IR Remote Control device) or function (e.g. Image recognition). The Web Services API documentation can be find in each of these the applications.

Furthermore a set of connectors permit to control each server application from Bonita Open Solution workflows.

The Saturn Software Suite :

Coming soon... Coming soon... Coming soon... Coming soon... Coming soon...

1 **Catcher**
"Catcher" is a server application which aim is ...

2 **RCU**
"RCU" is a server application which aim is to control RedRat III and irNetBox - Pro III equipments from web services client application.
IMPORTANT NOTE : users of the RedRat USB devices must previoulsy install the drivers from the RedRat web site or from this portal.
A connector for Bonita Open Solutions is also available to manage Remote Control actions in Bonita Open Solution workflows.

3 **Vision**
"Vision" is a server application which aim is ...

Figure 8. Screenshot of the Saturn web portal.

V. Main results

A. *Saturn Key Benefits*

Among the main benefits experienced with the Saturn solution deployments, can be mentioned: a clear separation of the R&D and test team roles, the increased motivation of the test team members in dealing with the whole test strategy and not only with repetitive task execution, an increased autonomy of test teams in the test automation process, fully automated process supported by an easy integration of test management tools (TestLink [9], HP Quality Center [10]) dealing with IT and reporting tasks automation, integrated test scenarios versioning and reviewing process (using the version control system GIT), easy reuse of already developed test scenarios via BPMN 2.0 sub-processes mechanism, version tracking of the totality of the elements involved in test results, quality of test results related to the reproduction of the test runs, multi-site robots deployment with file versioning and repositories synchronization, limited training required for newcomers thanks to easy use of intuitive tools, scalable and easy to maintain architecture with new testing requirements managed by adding separate wrappers and connectors in an incremental approach without any impact on the existing system.

B. *Quantitative Results*

The Saturn test automation framework implements this strategy and is used by the company SoftAtHome, in the implementation of its automated testing infrastructure for "Set Top Boxes" and "Home Gateway". Saturn is currently deployed in 4 countries: France, Belgium, UAE and Tunisia with about 20 test robots connected through the internet to a shared infrastructure. One third of the manual validation can be executed with automation system Saturn. Each month, more than 6 middleware releases are tested (robustness and no regression campaign).

VI. CONCLUSION

We presented a novel approach to deal with functional tests automation. This approach - built around a Business Process Modeling System and according to a Service Oriented Architecture - instead of targeting specific scripts development for tests automation - focuses on the delivery of a generic toolkit which aims to deliver a set of human replacement tools that can be used by testers without programming skills.

The versatility to add new wrappers brings many perspectives for Saturn tool in particular to interface with additional devices useful for Set Top Boxes validations: EDID Extended Display Identifier Data generator (as *Quantum* [12], a EDID generator that can simulate a connection with all kind of TV sets), stream player (as *DekTek* modulator [13] able to broadcast a specific stream content mandatory for the automated test), etc.

Moreover, a new Saturn wrapper offering innovative IP Network datagram analysis services [5] is currently under development. The main goal is to offer a toolbox to develop automation test cases for Home Gateway (basic network) and to check the interoperability with Set Top Boxes, by the way of common scripts.

ACKNOWLEDGMENTS

The Saturn project is financially supported by the company SoftAtHome. SoftAtHome leverages the best of connectivity, Pay TV and Digital Home Services into SOP, its unified software platform for Set Top Boxes and Home Gateways.

The main contributors to the Saturn project development are in alphabetical order: Mr. Christian Couder, Mr. Vincent Courtot, Mr. Olivier Delfosse, Mr. William Gontier, Mr. Franck Hennequin, Ms. Alison Jorre, Mr. Saddek Kadji, Mr. Fabien Lloansi, and Mr. Marc Toffanin.

REFERENCES

[1] William Gontier, Franck Hennequin and Marc Toffanin, "Automated Detection of Video Artefacts", SoftAtHome S.A., September 2013 (see condition 1).

[2] William Gontier, Franck Hennequin and Marc Toffanin, "Audio watermarking and Identification Algorithms", SoftAtHome S.A., September 2013 (see condition 1).

[3] William. Gontier, Franck Hennequin and Marc Toffanin, "Lip Sinching computation in Saturn Test Automation Framework", SoftAtHome S.A., November 2013. (see condition 1)

[4] William Gontier, " RCU Simulator SDK Specifications - Saturn Framework", SoftAtHome S.A., March 2014 (see condition 1).

[5] William Gontier, "IP Network Test Automation - A big Data Approach", SoftAtHome S.A., May 2014 (see condition 1).

[6] Robert Shapiro and Stephen A. White, "BPMN 2.0 Handbook Second Edition: Methods, Concepts, Case Studies and Standards in Business Process Modeling Notation (BPMN)", Layna Fischer, 2012.

[7] http://bbt.rt-rk.com/en/test-suites

[8] http://www.s3group.com/tv-technology/services/stormtestr-test-services/

[9] TestLink Open Source Test Management https://wiki.openoffice.org/w/images/1/1b/Testlink_user_manual.pdf

[10] Quality Center Enterprise http://www8.hp.com/us/en/software-solutions/quality-center-quality-management/

[11] Git http://git-scm.com/

[12] http://quantumdata.com

[13] http://www.dektec.com/ /

(1) Delivered subject to non disclosure agreement.

Contact SoftAtHome : http://www.softathome.com/pages/contact