

View Point



Realizing the Enterprise Test Automation vision

Manoj Narayan, Harish Krishnankutty, Kaushik Ramakrishnan

Abstract

Several organizations have tried to replace manual testing with test automation tools. While they did reap its benefits, most also encountered a few roadblocks. Typical drawbacks being that the tools are UI driven, are based on ad-hoc definitions, and very often are person-dependent. These roadblocks have deterred some organizations from investing in test automation, while some others have been able to leverage it only for part of their business processes. This paper provides the vision of an end-to-end test automation framework that helps overcome key challenges in currently available test automation tools.

Traditional approach to test automation

Test automation delivers tremendous benefits to organizations. It helps optimization of the resources necessary to perform testing, viz, hardware, people and time. Automated functional tools are used to build and execute automated test scripts of these tools.

Most test automation tools test applications through the front-end and provide for UI-based automation. The front-end approach is similar to manual testing, for which a wide range of test tools have long been in the market. Although the UI-based automated testing approach of the currently available test tools is fairly mature, we feel that they fail in three areas:

- **Test automation is not end-to-end:** The UI-based testing features provided by most of the available tools are limited in their ability to test back-end systems, systems that do not have an UI, database level operations, and several other key components of a typical enterprise-level system. **Many** versions of test automation have tried to get around this limitation by building the automation framework through a pool of different tools – each used to develop one part of the framework. But this approach also has some shortcomings. The use of multiple tools necessitates dealing with multiple vendors and increases management complexity. It is not cost effective, and poses several challenges in integration of the different tools.
- **Clear requirements are a pre-cursor:** The lack of clear requirements to build a test automation framework has stalled many test automation efforts even before they got off the starting line. Even in cases where requirements are defined, they are often based on the 'gut-feel' of a team, rather than through a systematic functional decomposition of the system.
- **Testing is Subject Matter Expert (SME) dependent:** Standard testing software like Retek, MTS, etc, lack a clear statement of requirements, requiring organizations to engage a product SME to conduct the testing. SMEs typically are highly experienced and skilled resources. Engaging them for prolonged periods can be cost intensive, besides making the process highly person dependent.

The vision of an enterprise test automation strategy

Our vision is of an enterprise automation strategy that addresses all the key challenges posed by currently available test automation tools. It consists of a framework that helps define the business process, validation rules, etc and has components to test each aspect of the process and rules.

End-to-end test automation framework

Our test automation framework is capable of tracing a business process from the front-end input, all the way through several interfacing or back-end systems, to its final end point(s).

Consider an application used by a bank to transfer money from one account to another. The operation goes through several systems and several levels of business logic. To begin with, the user input is provided on the UI and client-side validations take place. The input is then passed on to the application business logic for processing. This may require database entries to be made in different tables – for example, an audit log table. Further checks and validation are done, and the transaction may then be sent to a legacy system where the account balances are stored and managed. At this point, the transfer amount gets debited from the source account and credited to the destination account. The scenario is depicted in Figure 1 below.

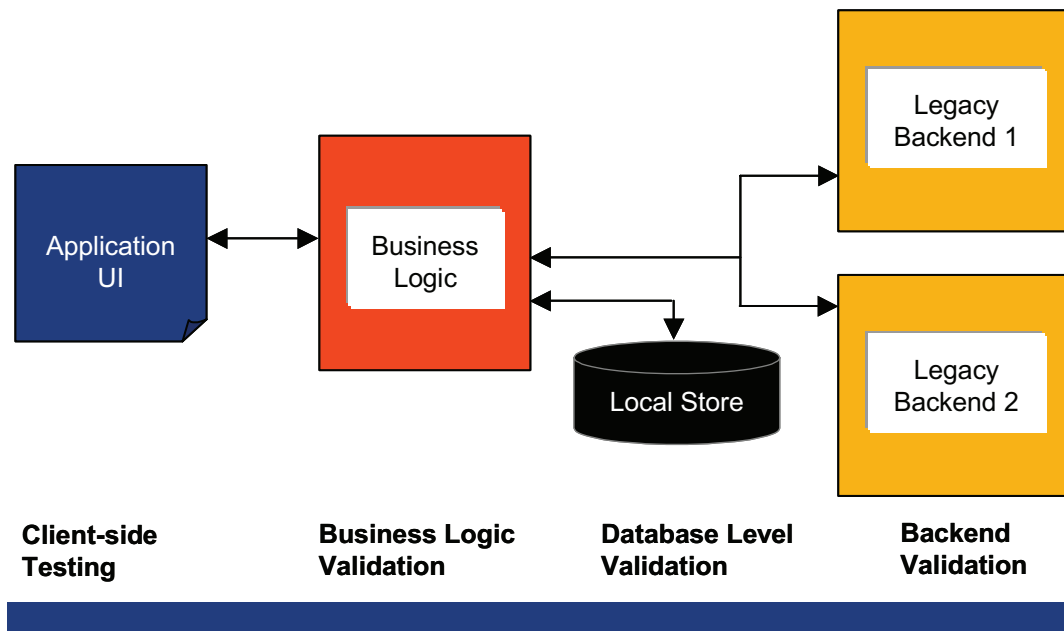


Figure 1- Fund transfer

All along the path of the transaction, there are several opportunities for branching off. The application layer business logic may determine that the user is not entitled to make money transfers, the legacy system may find that the source account does not have sufficient funds, the destination account may belong to a different bank and the transaction may need to be passed on to an external system, and so on.

Reduced dependence on SMEs

The framework has a set of libraries that contain domain-specific knowledge and pre-defined test cases for common business processes. Having domain-specific knowledge built into the framework will significantly reduce dependency on SMEs. In this case, banking related knowledge built into the test automation framework will be employed.

Eliminating the need to pre-define requirements

The process of defining specific requirements for implementation of an enterprise test automation framework is simplified by abstracting the pre-defined knowledge base and extending only the gaps. This approach is much easier than defining the complete requirements from scratch.

The table below summarizes the key drawbacks in traditional test automation and how the enterprise test automation strategy can help address them.

Drawbacks of the traditional approach	Enterprise test automation solution
Test automation is not end-to-end	The enterprise test automation solution recognizes that there are several areas of an enterprise system, beyond just the UI layer, that need to be tested. The strategy envisions setting up of a framework to initiate a transaction through the UI and follow it all the way through to the back-end systems.
Clear requirements are a pre-cursor	The enterprise test automation framework consists of a component that will aid the generation of test cases and scripts. This eliminates the need for concrete requirements defined outside the framework.
Testing is SME dependent	The envisioned enterprise test automation framework has in-built domain-specific knowledge in the form of libraries. This knowledge can be leveraged in defining test cases that reduces dependence on SMEs.

The enterprise test automation framework

The enterprise test automation framework enables implementation of the strategy discussed thus far. The framework has the capability to:

- Model business processes and validation rules
- Leverage a pre-defined knowledge base and generate a set of domain-specific test cases (banking – money transfer applications, in this case)
- Initiate a transaction, such as the money transfer transaction, and validate it all along its path

A framework with these capabilities will, therefore, consist of four key parts (Figure 2):

- Modelers
- Engines for test case / script generation
- Libraries
- Adapters for specific technology platforms

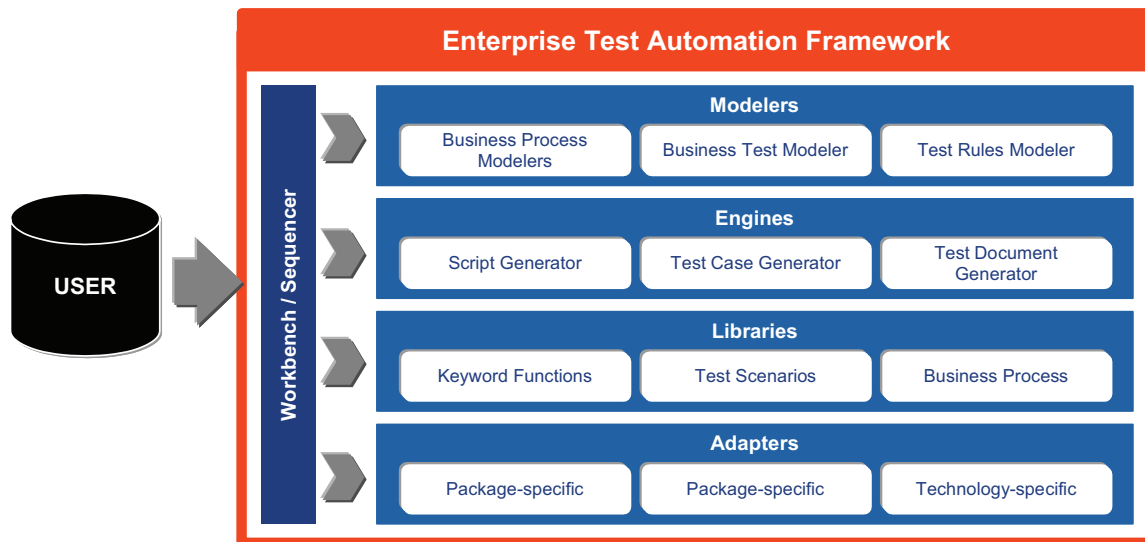


Figure 2 – Components of an enterprise test automation framework

Modelers

Modelers enable definition of business processes, validation rules, etc. They abstract the test scenario definitions from the scripting.

Engines

Test case generation engines generate scripts for multiple platforms. They will have the capability to generate not only UI-level test cases, but also test cases to test back-end interfaces and components. The engine will take in the list of parameters for a particular interface and the input values for each of these parameters. The intelligent generation engine then uses this information to generate a set of test cases that are mutually exclusively, but collectively exhaustive. The test case generation engine will also have the capability to take in the properties of a rules engine and generate test cases that can traverse all the logical paths within the rules engine.

Libraries

The enterprise test automation framework will normally come with pre-built knowledge or libraries of critical operations in various domains. This knowledge may be in the form of high-level test cases. The team implementing the automation framework will need to select the set of appropriate test cases in the given context. Some amount of customization may be needed for implementation of the selected test cases.

Adapters

The final component of the framework is the set of adapters. Adapters allow the test automation framework to work with multiple tools, packages and technology platforms. They form a level of abstraction between the test scripts, test cases that may be business process dependent, and the actual execution of the test scripts that are technology or platform dependent.

The adapters or components that drive test execution can be classified into three broad areas:

UI testing component

The UI testing component of this framework initiates transactions through the front-end. It will also initiate the transaction with various combinations of input data simulating positive and negative test conditions.

Server-side testing component

The server-side testing component of the framework initiates the transaction directly on the server by making calls to the appropriate interfaces or APIs and passing the required input parameters. The server-side test component can provide various combinations of parameters and hence, tests out all possible ways of calling a particular API. The parameter combinations can be designed to test every possible input sequence that a rules engine can accept and test. The server-side test can also capture the response from a call and use that response in the next call. This allows a flow or sequence of calls to be set up, thereby simulating a business process on the back-end. The sequence, however, need not be restricted to calls to a particular system. It can span multiple systems to simulate the effect of an end-to-end business process.

Data validation component

The UI and server-side testing components work in a request-response mode. Usually, responses to a request are validated against an ideal or expected response. Between the request and response, there may be many background operations. The third component of the enterprise test automation framework data validation, tests these operations. Going back to our example of banking application, the data validation component can come in to play at many stages. For example, when the transaction is initiated from the UI, an entry of the transaction details is made into an audit log table as the transaction is processed. A data validation component of the test automation framework can be invoked to verify if this operation is completed successfully.

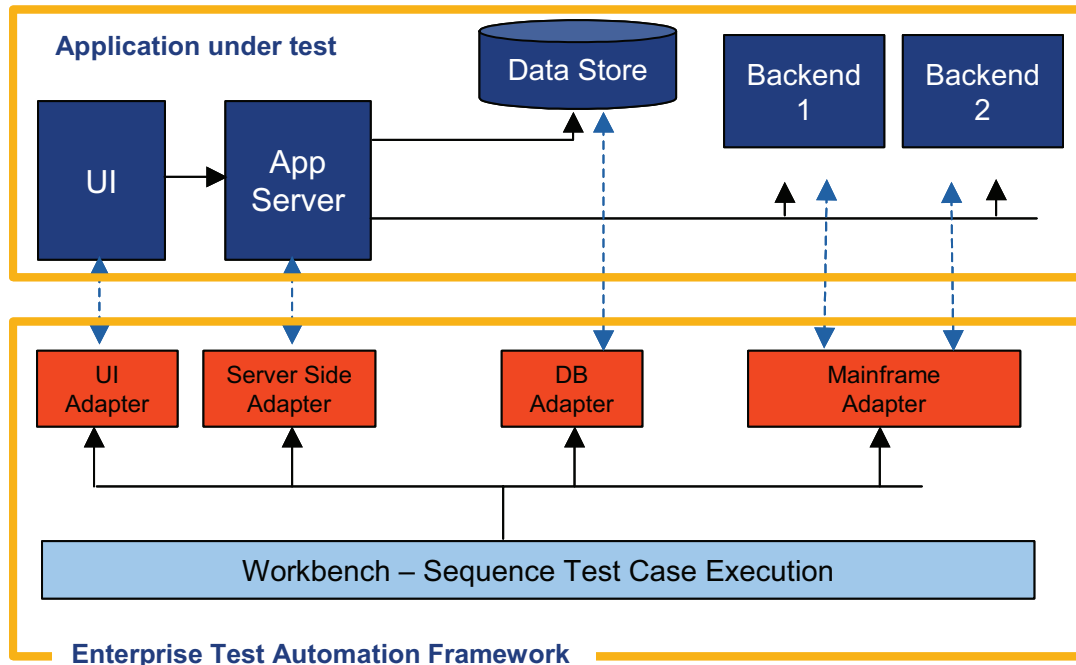


Figure 3 – Enterprise test automation framework at work

The final component of the execution engine is the sequencer or workbench that coordinates test script and documentation generation, and controls the sequence of test execution. The test workbench is a core component that drives the execution of tests. This component sequences the various operations to simulate business flows. For example, there is no point in running the data validation component to check the audit log table before the UI script to initiate a transaction is run. A consolidated workbench that coordinates test script generation and drives the entire test execution process is an essential part of the enterprise test automation framework (Figure 3).

Case in point

A Fortune 500 telecommunications service provider scheduled six releases of their CRM application, every year. Forty days of manual testing was required for each release, impacting time-to-market. Infosys worked with the client to implement an automation framework comprising:

- A keyword-driven framework to automate transaction entry and front-end check-point verification
- A custom-built back-end verification tool to automate legacy system based back-end checkpoint verification
- A custom-built result reporting engine to combine the results of front-end testing and back-end verification of each test case, and to present it in the form of a user-friendly dashboard that permitted causal analysis

Infosys' Test Automation framework helped the client to:

- Reduce testing effort by 70%, enabling faster time-to-market
- Achieve payback on the investment on automation in ~10 releases
- Increase application service availability to 99.5% by enabling more comprehensive testing

Conclusion

The envisioned enterprise test automation framework addresses the major drawbacks of traditional test automation techniques and can test a system end-to-end. It reduces manual effort significantly, besides reducing the number of tools and frameworks required to test each component of an application individually.

The benefits of test automation - savings in time, cost savings, higher efficiency, and better quality software are well recognized. With an enterprise test automation framework these benefits can be leveraged in all areas of an enterprise system.

About the Authors

[Manoj Narayan](#) manages the Solutions and Consulting group that focuses on developing Validation and QA solutions. He has over 11 years of IT experience spanning complex solution consulting, program management, and delivery focused on functional and automated testing solutions. He has managed large Testing Center of Excellence programs and automation competency centers for major financial institutions in North America.

[Harish Krishnankutty](#) is a Senior Test Manager with the Solutions and Consulting group. He has over nine years of IT experience in application development and maintenance for multiple clients in the Banking and Capital Markets domain. He has extensive experience in business process management and business rules management systems. He has built multiple solutions on these technology platforms for exception management, data quality management etc. Presently he leads the development of ready-to-market test solution offerings, including the Enterprise Test Automation solution, at Independent Validation Solutions.

[Kaushik Ramakrishnan](#) is a Test Analyst with the Solutions and consulting group and is a champion for Enterprise Test Automation solution. He has four years of IT experience in consulting and QA solutions delivery. He has been in large testing engagements with major financial institutions in North America



For more information, contact askus@infosys.com

About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.