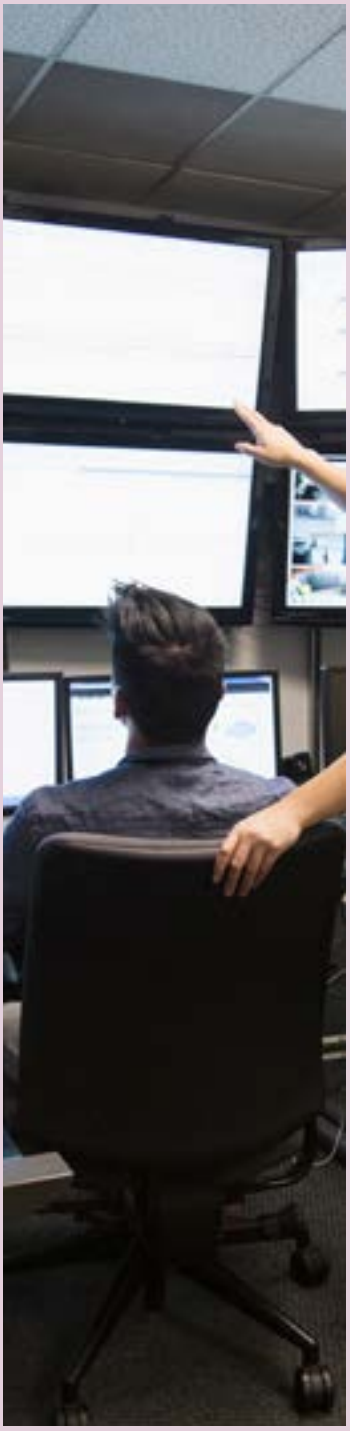
PERSPECTIVE



END-TO-END TEST AUTOMATION – A BEHAVIOR-DRIVEN AND TOOL-AGNOSTIC APPROACH

Anand Avinash Tambey
Product Technical Architect, Infosys

Abstract

In today's fast changing world, IT is under constant pressure to deliver new applications faster and cheaper. The expectation from the quality assurance (QA) organization is to make sure that all the applications are tuned to deliver to every rising user expectation across devices, locations, and typically, at no additional cost. And with an exponential growth in the diversity and number of end users in almost all sectors, requirements are fairly fluctuating and demanding too.

Infosys®
Navigate your next

This fast-paced software engineering advancement is also posing challenges to software engineers to build an ecosystem that enables rapid prototyping and design, agile development and testing, and fully automated deployment. For the QA community, this translates to a need to maximize automation across all stages of the software engineering and development life cycle and more importantly, do it in an integrated fashion. Consequently, `extreme automation' is the new mantra for success. And there is more.

According to the 2014 State of DevOps report, high-performing organizations are still deploying code 30 times more frequently, with 50 percent fewer failures than their lower-performing counterparts. High IT performance leads to strong business performance, helping boost productivity, profitability, and market share. The report counts automated testing as one of the top practices correlated with reducing the lead time for changes. However, automated testing puts together another set of challenges. The latest technologies stack advocates multiple choices of test automation tools, platform-specific add-ins, and scripting languages. There is no inherent support available for generic, tool-agnostic, and scriptless approach with easy migration from one tool to another. Therefore, a significant investment in training, building expertise, and script development is required to utilize these tools effectively. The cost and associated challenges inadvertently affect the time-to-market, profitability, and productivity although it also creates an opportunity to resolve the issues using a combination of an innovative tool-agnostic approach and latest industry practices such as behavior-driven development (BDD) and behavioral-driven test (BDT).

Business challenges

A persistent need of businesses is to reduce the time between development and deployment. QA needs to evolve

and transform to facilitate this. And this transformation requires a paradigm shift from conventional QA in terms of automation achieved in each life cycle stage and across multiple layers of architecture.

Technical complexity

The technology and platform stack is not limited to traditional desktop and the web for current application portfolios. It extends to multiple OS (platforms), mobile devices, and the newest responsive web applications.

Infrastructure, licensing, and training costs

To test diverse applications, multiple test automation tools need to be procured (license cost), testing environment needs to be set up (infrastructure), and the technical skills of the team need to be brought to speed with training and self-learning / experimentation (efforts).

Late involvement of users

The end user is not involved in the development process until acceptance testing and is totally unaware of whether the implemented system meets her requirements. There is no direct traceability between the requirements and implemented system features.

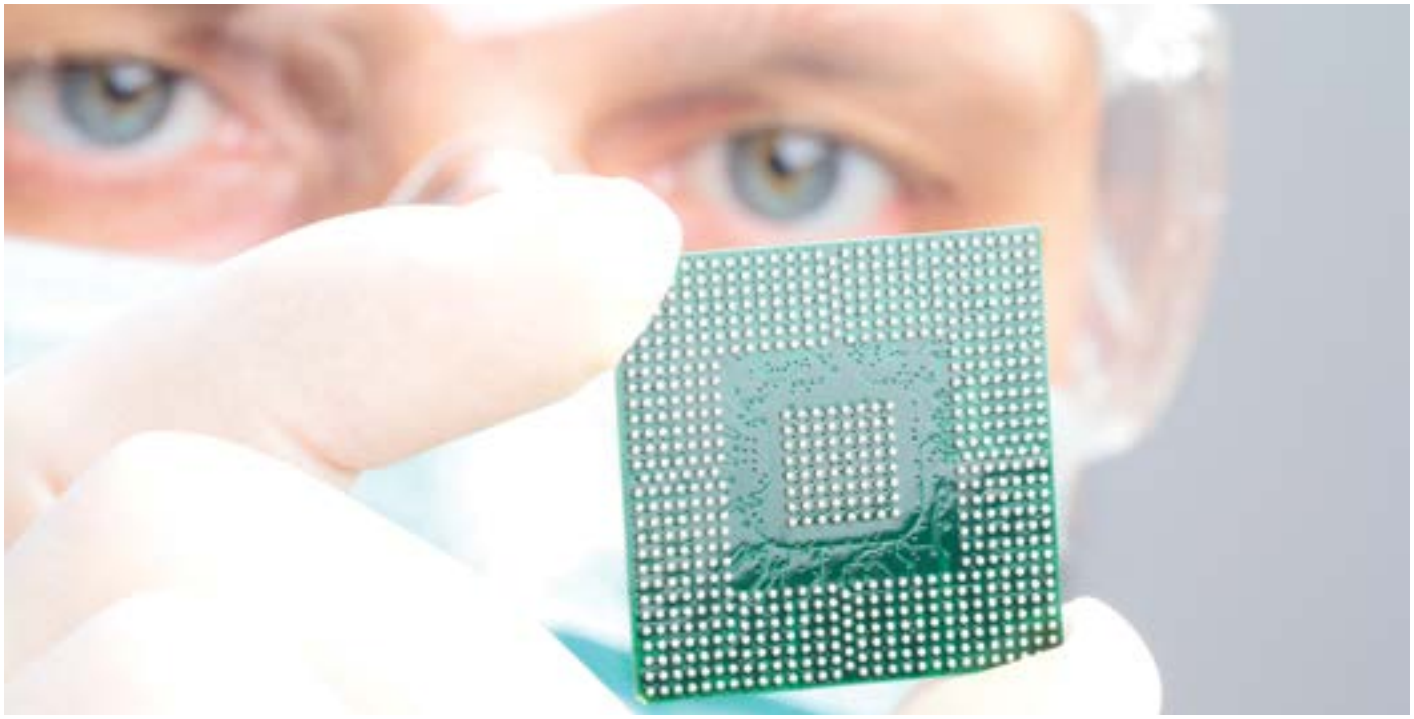
Late involvement of testers

Testing and automation also need to start much earlier in the life cycle (Shift-Left) with agility achieved through the amalgamation of technical and domain skills of the team, as well as the end user.

How to face these challenges

Challenges	Approaches
• Technical complexity	• De-skilling
• Infrastructure, licensing, and training costs	• Using open source stack
• Late involvement of users	• Using behavior-driven techniques
• Late involvement of testers	• Utilizing the testers and users effectively

Let us discuss these approaches in detail to face the above challenges.



Using open-source stack

To reduce the cost of commercial tool license and infrastructure, utilize open-source tools and platforms.

De-skilling

Using easy modeling of requirements and system behaviors, accelerated framework, and automated script generators, reduces the learning curve and the dependency on expert technical skills.

Using behavior-driven techniques

Behavioral-driven development and testing (BDD and BDT) is a way to reduce the gap between the end user and the actual software being built. It is also called `specification by example.' It uses natural language to describe the `desired behavior' of the system in a common notation that can be understood by domain experts, developers, testers, and the client alike, improving communication. It is a refinement of practices such as test-driven development (TDD) and acceptance test-driven development (ATDD).

The idea behind this approach is to describe the behaviors of the system being built and tested. The main advantage is that the tests verifying the behaviors reflect the actual business requirements / user stories and generate the live documentation of the requirements, that is, successful stories and features as test results. Therefore, the test results generated can be read and understood by a non-technical person, such as a project sponsor, a domain expert, or a business analyst, and the tests can be validated.

Utilizing testers and users effectively

Our accelerated automation approach provides a simple modeling interface for a scriptless experience and thereby utilizes non-technical staff effectively. It introduces the `outside-in' software development methodology along with BDT, which has changed the tester's role dramatically in recent years, and bridges the communication gap between business and technology. It focuses on implementing and verifying only those behaviors that contribute most directly to the business outcomes.

Solution approach

Our solution approach is threefold, to resolve challenges in a holistic way. It applies a behavior-driven testing approach with a tool-agnostic automation framework, while following an integrated test life cycle vision. It ensures that business users and analysts are involved. It provides the flexibility of using any tool chosen and helps save cost and effort. It also provides a simple migration path to switch between tools and platforms, if such a case arises. With an integrated test life cycle approach, it ensures seamless communication between multiple stakeholders and leverages industry-standard tools.

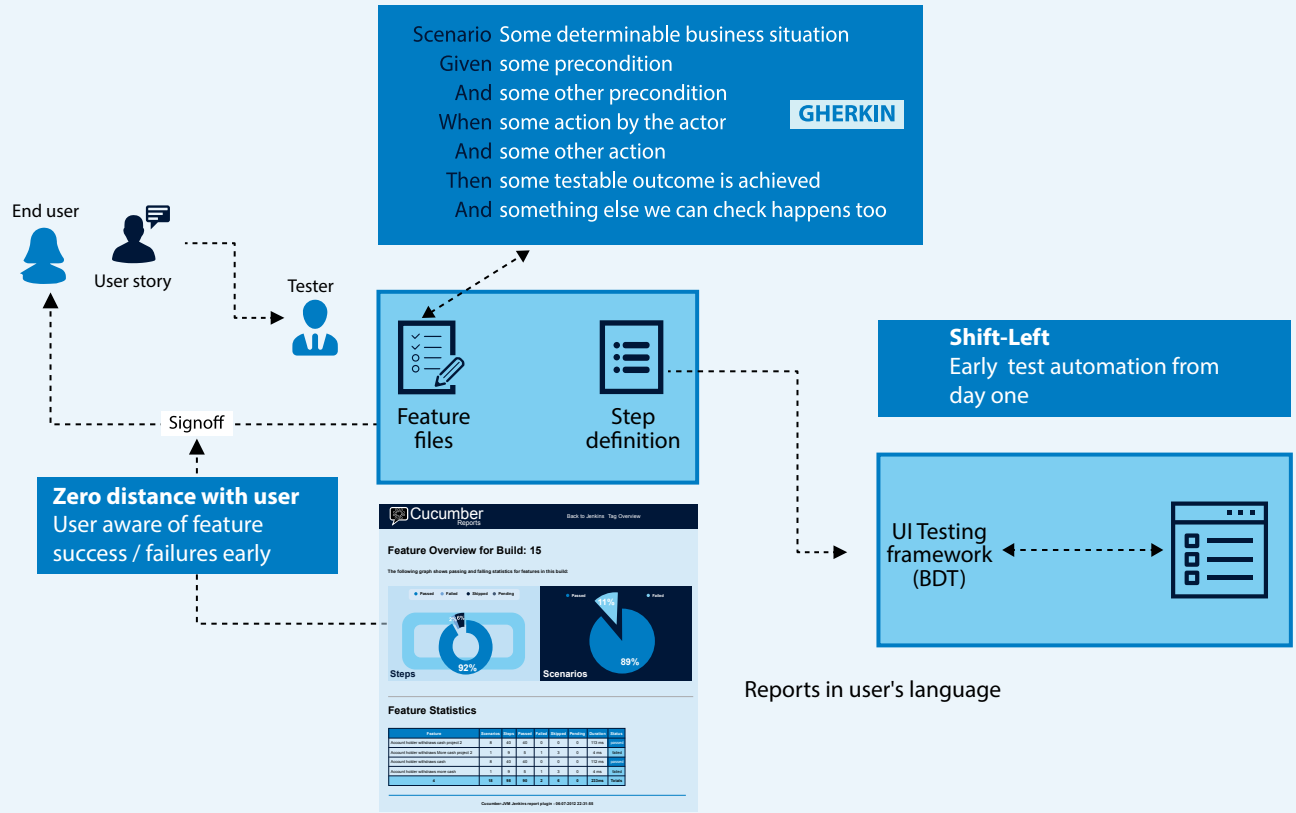
Behavior-driven test (BDT)

To facilitate real-time traceability of user stories / requirements and to aid live documentation, we have implemented a BDT approach across multiple Infosys projects as described below.

This approach acted as a single point of continuous interaction between the tester and the business users.

The user story is divided into various scenarios by the testers into a feature file.

- These scenarios are written using Gherkin language. These scenarios include the business situation, the pre-ions, the data to be used, and the acceptance criteria.
- The end user signs off the features / scenarios. This provides the user control to execute and validate the scenarios based on the data as per the user need and bring up feature reports / dashboard.
- The underlying technical implementation is abstracted from the business user.
- Tester creates the underlying test scripts for the scenarios which could be a business layer test script, service test script, or a UI automated test script.
- The tool then converts the scenarios to `step definitions,' which act as the binder between the test scripts and the scenarios. This ensures that a single point / interface is used to execute any type of tests.



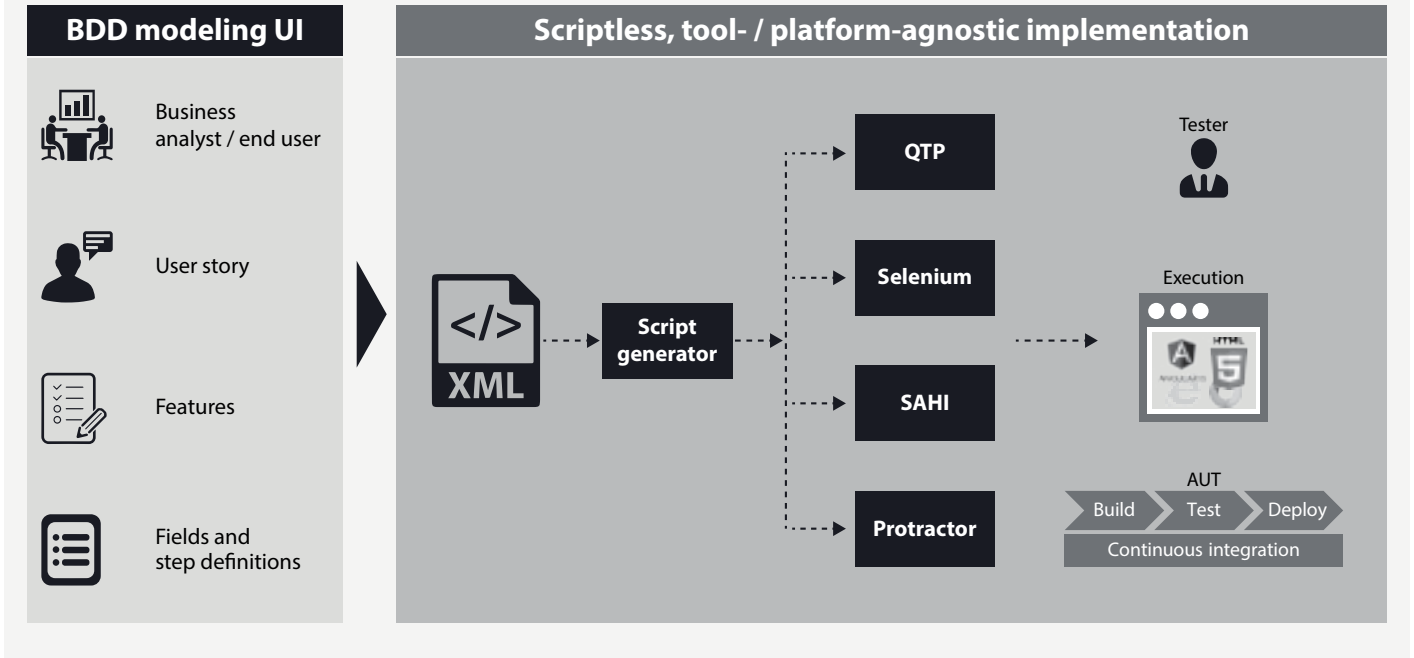
Finally, it introduces automation in early stages to realize the benefit of Shift-Left.

Tool-agnostic approach

To remove dependencies on technical skills, tools, and platforms, our solution proposes modeling of system behaviors using generic English-like language via an intuitive user interface. This model would be agnostic to any specific tool or UI platform and reusable.

This model will be translated into a widely acknowledged format, XML, and will act as an input to generate automation scripts for specific tools and platforms. Finally, it would integrate with a continuous integration platform to achieve an end-to-end automation of build-test-deploy cycle.

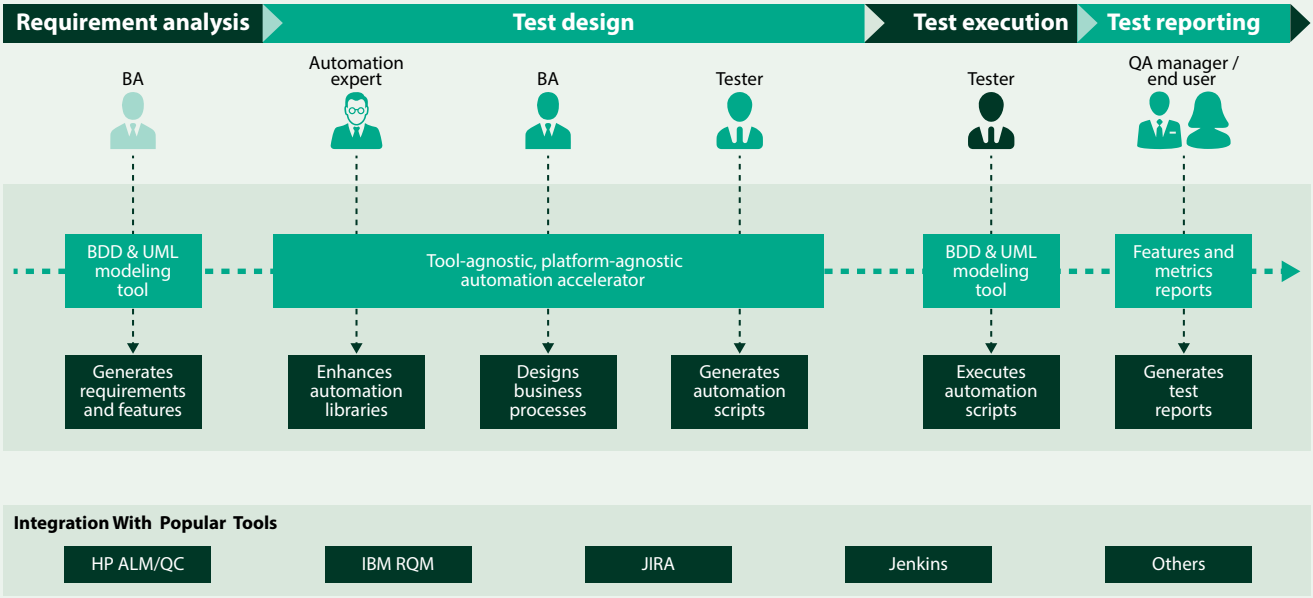
Tool- / platform-agnostic solution (BDD release)



Integrated test life cycle

The role of the traditional tester and end users is changing in the era of DevOps and Shift-Left. The integrated-solution approach enables a larger stakeholder base to contribute towards the quality of the system under development. It also ensures the satisfaction of stakeholders via early validation and early feedback of system progress while using the industry-standard toolsets seamlessly.

Test Life Cycle Management View



Benefits

Reduced time-to-market

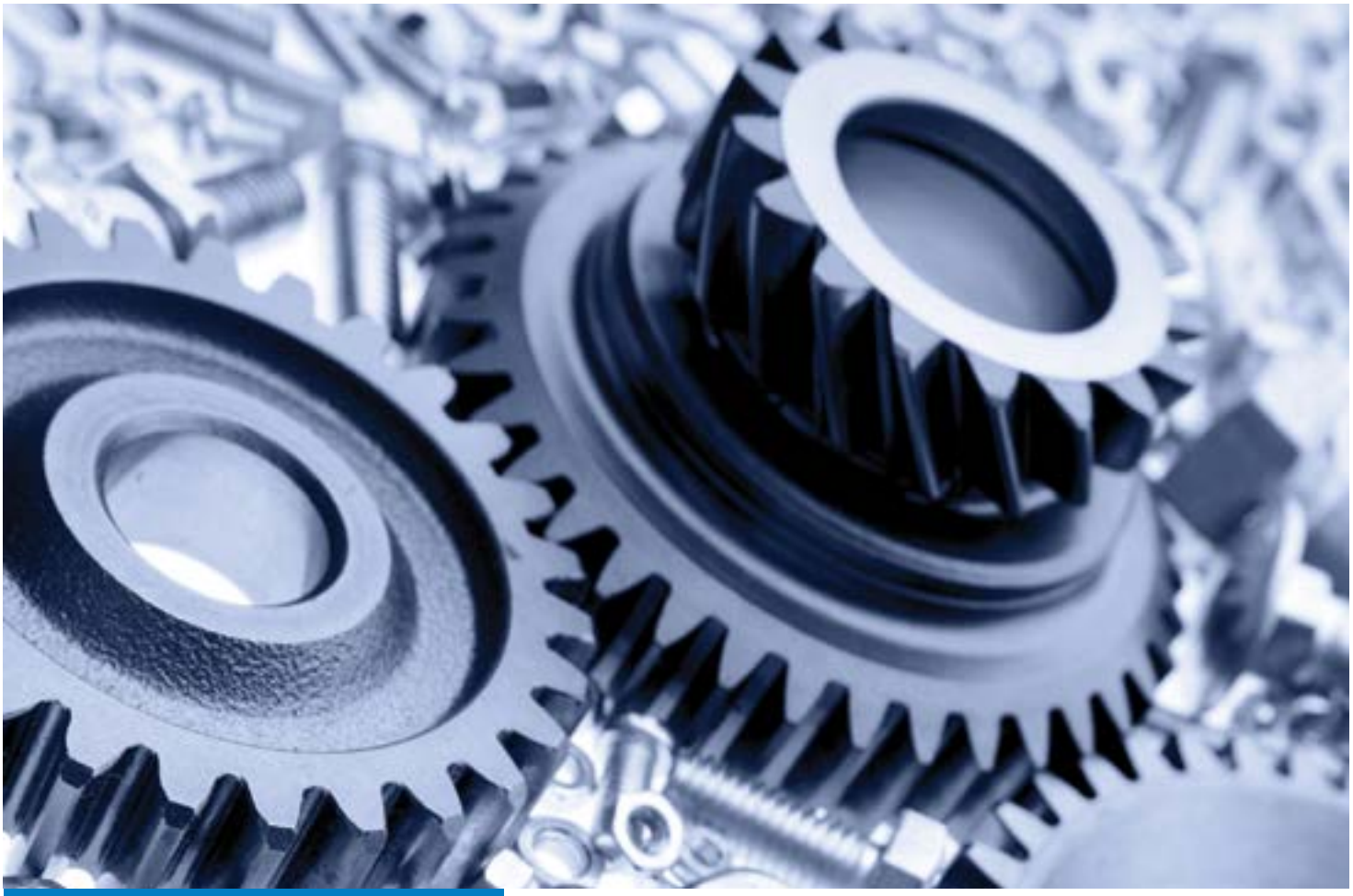
- Shift-Left, early automation, and early life cycle validation
- Single-click generation and execution of automated scripts

Reduced cost

- 40–60 percent reduction in effort for automated test case generation over manual testing
- Detailed error reporting reduces defect reporting effort considerably
- Easy maintenance of requirements, stories, features, and the automated test suite
- No additional cost involved in building integration components for test management tools (HP-ALM)
- The agnostic approach works for a broad range of applications irrespective of tools, technology, and platform

Improved quality

- Enhanced business user participation and satisfaction due to the live documentation of features and user stories, available at fingertips
- Developer, tester, and client collaboration possible due to a common language
- High defect detection rates (95–99 percent) due to high test coverage



Conclusion

Our solution approach is the first step towards reducing the complexity of test automation and making it more useful for the end user by providing early and continuous feedback to the incremental system development. Moreover, it advances automation at every level to achieve rapid development and faster time-to-market objectives.

With the advent of multiple technologies and high-end devices knocking at the door, using a tool- and platform-agnostic approach will increase overall productivity while reducing the cost of ownership.

References

https://puppetlabs.com/sites/default/files/2014-state-of-devops-report.pdf

http://www.ibm.com/developerworks/library/a-automating-ria/

http://guide.agilealliance.org/guide/bdd.html

http://www.infosysblogs.com/testing-services/2015/08/extreme_automation_the_need_fo.html

For more information, contact askus@infosys.com

Infosys®
Navigate your next