THE ACT FRAMEWORK

Continuous Testing for Enterprises

In association with:

accenture epam TRICENTIS

CONTENTS

introduction	03
introducing the ACT framework	04
the foundation of the ACT framework	05
implementing at the team level	06
implementing at the program level	07
implementing at the enterprise level	08
shared teams	09
value tools	10
conclusion	10

INTRODUCTION

Enterprises across the world are exploring ways to create more innovative business models. They are driven by a new digital economy that affects their ways of doing business and can no longer solve problems the same way they have over the past several decades.

The new age of technology allows even the smallest startups to disrupt the market by providing greater value, with a better experience than their slow-moving, less innovative counterparts can offer. There are many examples of startups adopting new technology to become market leaders in industry after industry.

To stay ahead of the game, enterprises must constantly think, rethink, adapt, and replace their approaches to doing business. Speed of change and speed of learning have become the latest sources of strategic advantage. For this reason, today's enterprises have digital transformation on their agenda. Agile and DevOps are among the key ingredients of faster innovation.

Speed, however, cannot come at the expense of accuracy or quality. And while development and delivery of software have sped up, software testing practices have remained untouched due to their complex nature. The increased speed of software delivery requires continuous testing, but today's practices don't support that.

In the past, minor wins in efficiency have been gained through automated test execution. This is no longer enough. Continuous testing is needed, and in addition to automation, it demands:

- **An effective test strategy** in which tests are designed based on potential business risk.
- **Seamless integration** to create a continuous feedback loop in the delivery lifecycle and DevOps toolchain.
- **Safety nets** that help protect the user experience.
- **Predictable environments** for fast, stable test execution and reliable results.
- **End-to-end visibility** to increase agility through quick and thoughtful decision-making.

These demands require a new design for today's testing practices—a design built on a solid foundation that defines the vision, mindset, and core values. This design will require a transformation at all levels of the enterprise.

There is a need for standardization and a framework that acts as the blueprint to scale continuous testing for the enterprise. This is the purpose of the ACT Framework (ACT). It describes the roles, processes, capabilities, and tools you will need to transform your testing practices.

The new age of technology allows even the smallest startups to disrupt the market by providing greater value, with a better experience...

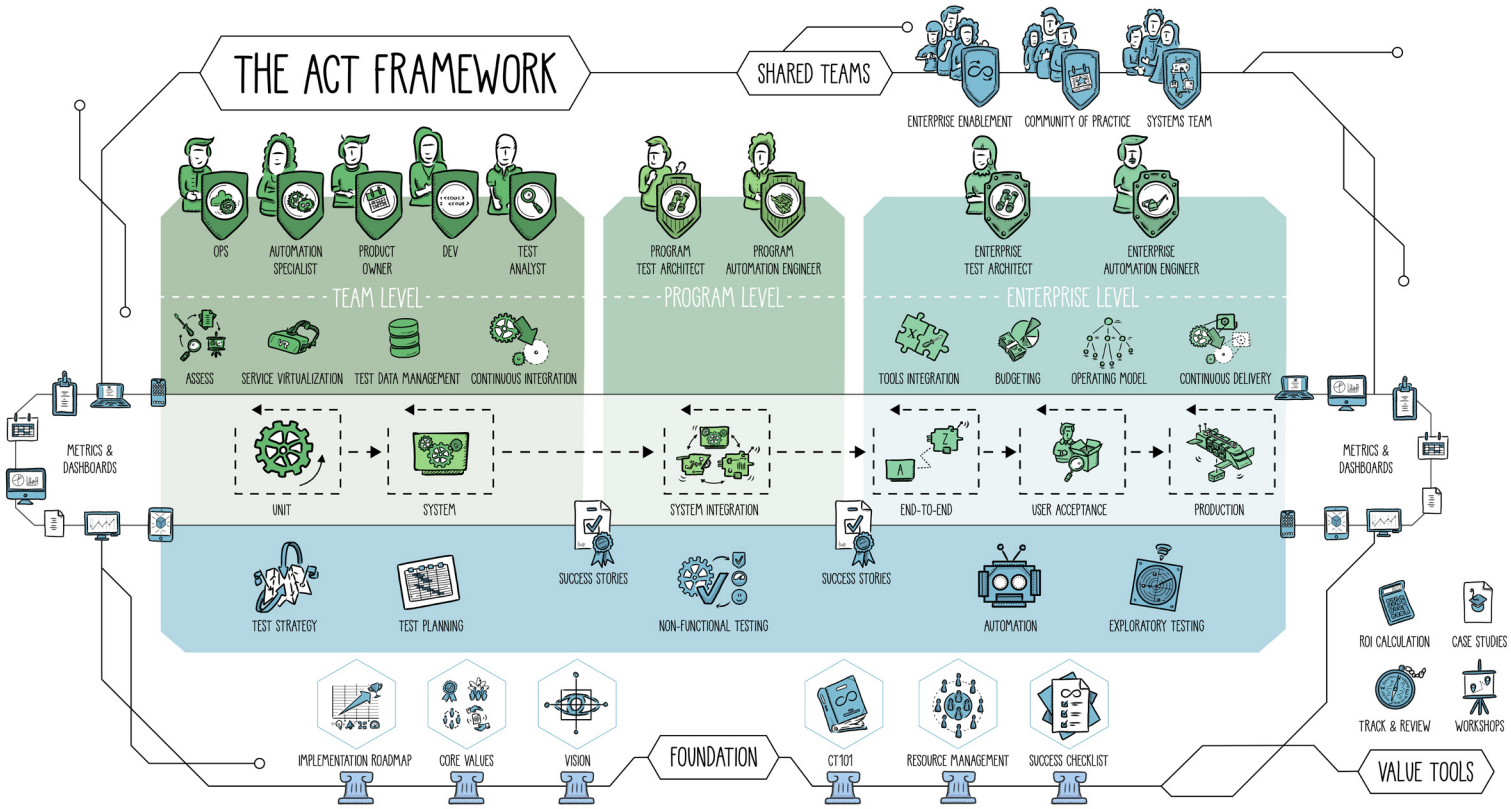
INTRODUCING

THE ACT FRAMEWORK

The ACT Framework defines the coordination of people, activities, processes, and technology implementation at three different levels: Team, Program and Enterprise.

ACT defines the testing stages at the different levels (e.g. Unit testing) as well as the common practices (e.g. Automation) that span across the levels. Depending on the size of the organization, the activities may shift left or right between the levels. Regardless of where the activity is performed, these three levels will ensure that a feedback loop on the quality of the product is present and transparent to all the stakeholders throughout the entire development life cycle.

In addition to the three levels, ACT also defines **Shared Teams** (who operate across all three levels) and **Value Tools** (that can help with the implementation of the Framework). Finally, the ACT Framework is built on top of a **Foundation** that is key to the successful implementation at any organization.



THE FOUNDATION OF THE ACT FRAMEWORK

The true foundation of ACT is the mindset you adopt towards continuous testing. There are six main pillars that make up the foundation:

CORE VALUES

The four core values below represent the fundamental beliefs that are key to successfully implementing ACT:

- "Testing is everyone's responsibility." Software quality is no longer the responsibility of the QA team—it belongs to everyone.
- "Quality First." Before you even begin development, anticipate the ways something could go wrong. Then think of ways to embed quality into every development process throughout your organization.
- "Collaborate." Although quality is now everyone's responsibility, that doesn't mean teams should work in silos. Make sure teams are sharing assets and knowledge.
- "Embrace failure." When developers and testers fear failure, they blame each other for the problems that arise. When they embrace failure, they're more eager to learn from it and work together to prevent it from happening again.

VISION

Before you adopt ACT, ask yourself why your organization is even beginning this journey. If you're doing it with specific business outcomes in mind, you won't be disappointed.

CONTINUOUS TESTING 101

Be sure you're beginning with a solid understanding of what continuous testing really is. Continuous testing is the process of executing automated tests as part of the software delivery pipeline so that you can obtain rapid feedback on business risks.

RESOURCE MANAGEMENT

As you set up ACT in your organization, you'll likely work not only with your own teams, but also with third-party vendors. You'll need to plan how you'll find the right people for the right positions, and then make a plan for engaging them—from the interview process on to the day to day implementation activities.

SUCCESS CHECKLIST

According to an old adage, "what gets measured gets done." A continuous testing checklist is an essential tool for ensuring that you've implemented all the elements we discuss in this paper.

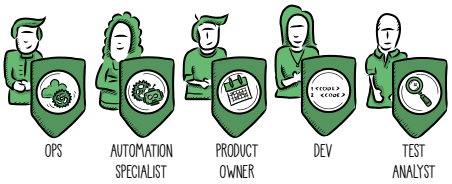
IMPLEMENTATION ROADMAP

Although there is no single step-by-step implementation process that would work in every situation, the Implementation Roadmap provides a pattern that can be adapted to each unique environment.

the Implementation Roadmap provides a pattern that can be adapted to each unique environment

IMPLEMENTING AT THE TEAM LEVEL

Your testers who work at the Team Level will focus on performing unit testing and system testing on new and existing functionality in a sandboxed environment to ensure that it is functioning properly. Through a fully automated continuous integration pipeline, the team will get fast, reliable feedback for optimal performance.



ASSEMBLING YOUR TEAM

The **product owner**, **developers**, **operations**, **test analysts**, and **automation specialists** are the roles that will make your team successful. The developers and operations staff manage overall software delivery, while the test analysts and automation specialists are responsible for exploratory testing and test automation. The product owner defines and prioritizes the items the team works on.

IMPLEMENTING CONTINUOUS TESTING

Once your team is in place, you can start implementing the automation processes and technologies that will increase your efficiency and reduce your business risk.

As you work, keep these principles in mind:

- **Follow automation best practices.** Avoid a "bottom-line" mentality in which the only priority is to go live with automation as soon as possible.
- **Put the right environment in place.** If your environment isn't well built and consistently managed, tests will fail, and the credibility of your results will be lost.
- **Consider how virtualization can help.** By virtualizing your interfaces to the system under test, you can run your tests without their involvement and maintain complete control over your testing environment.
- **Have your test data ready.** Define and implement a test data management strategy to avoid unnecessary delays during test execution.
- **Automate as many stages as possible.** The ideal scenario is to automate everything from code check-in to the distribution of test results.
- **Work to eliminate false positives.** You can begin to reduce them through proper test data management, service virtualization, and resilient test case design.
- **Add Exploratory Testing into the testing process.** This will help you capture bugs more quickly, especially during progressive testing.

MEASURING YOUR PROGRESS

Gauge the progress and success of your testing transformation initiative by tracking relevant metrics. **Defect Detection and Leakage**, **Test Cycle Time**, and **Effort Savings** will help you measure cost, quality, and speed for your ROI. **Test Coverage**, **Automation Degree**, and **False Positive Rates** will help you optimize your team's productivity.

Through a fully automated continuous integration pipeline, the team will get fast, reliable feedback for optimal performance.

IMPLEMENTING AT THE PROGRAM LEVEL

At the Program level you will focus on testing the integrations between systems. The roles at this level facilitate integration testing while working closely with the individual teams to create and manage the right processes for collaboration across teams. They also work to help teams get up to speed with continuous testing.

ASSEMBLING YOUR TEAM

The Program Test Architects and Program Automation Engineers orchestrate testing at the Program level. Test Architects orchestrate the integration tests, foster collaboration across teams, and work with individual teams to maximize productivity. They are overall responsible for continuous testing in the program. The Automation Engineers take away any technical hurdle teams may encounter and develop clever solutions that can be reused across teams to increase quality and performance.

Automation Engineers take away any technical hurdle teams may encounter and develop clever solutions that can be reused across teams to increase quality and performance.

IMPLEMENTING CONTINUOUS TESTING

At the Program level, a Systems Team is responsible for executing test scenarios that run across multiple teams. Integration tests are identified and designed based on potential business risk. Ideally, automated test cases will be created by reusing assets that were already created by the individual teams. This will reduce maintenance efforts over time. You should also implement exploratory testing to fill in any gaps missed by your automated checks.

To succeed in test automation at the Program level, you'll need a coherent strategy on how to store and reuse test data, what kind of test environment to use across teams, and where and how often to test.

MEASURING YOUR PROGRESS

You may discover that certain defects could have been detected earlier. If so, you'll count these bugs as defect leakage. By providing this feedback to the individual teams, you can help them improve their processes and optimize the overall delivery lifecycle.

At the Team level, you tracked metrics such as automation degree, risk coverage, effort savings and test cycle time. Combine these metrics to create **program level dashboards** for maximum visibility across the teams on overall speed and quality increases as well as time and labor savings through automation.

Document your testing transformation successes and create **success stories** to share with the rest of your enterprise. You'll encourage other programs to optimize their efficiency, while also helping to justify to your executives the investment they have already made or will need to make.

IMPLEMENTING AT THE ENTERPRISE LEVEL

At the Enterprise level, your focus should be on your company's overall testing strategy. This should encompass your tools landscape, testing budget, operating models, and shifting your company culture around quality.

The testing focus should cover the most-used and highest-risk end-to-end business processes, simulating end user behavior. This is also the level where you define production tests.

ENTERPRISE TEST ARCHITECT

ENTERPRISE AUTOMATION ENGINEER

ASSEMBLING YOUR TEAM

Just like at the Program level, **Enterprise Test Architects** and **Enterprise Automation Engineers** will orchestrate testing at the Enterprise level across all the programs. The Test Architects focus on overall strategy, collaboration, enablement, coaching, internal marketing, and orchestration of end-to-end tests. They are responsible for continuous testing within the enterprise. The Automation Engineers work with the community to integrate tools and implement enterprise-wide solutions to increase overall productivity.

IMPLEMENTING CONTINUOUS TESTING

As you did at the Team and Program levels, you'll need to develop a strategy at the Enterprise level for writing test cases, building test environments, determining how often each round of testing will be triggered, and so on. Exploratory Testing plays a vital role in uncovering any unforeseen defects in the software before it goes to production.

Depending on the product of your company, Continuous Delivery and/or Continuous Deployment are goals that are delivered at the Enterprise Level. You can achieve this by integrating tools and systems and using automated tests as gateways to the next testing stage, delivering feedback at every step of the journey.

MEASURING YOUR PROGRESS

At the Program level, you established a dashboard to report on testing metrics. You should also create a dashboard for Enterprise-level testing, but with a different goal. Rather than displaying metrics on defect leakage, cycle times, and effort savings, you should aim to report on business outcomes such as **enhanced quality**, **increased speed**, and **reduced costs**.

Exploratory Testing plays a vital role in uncovering any unforeseen defects in the software before it goes to production

SHARED TEAMS

Although ACT describes the distinct processes and personnel roles at the Team, Program, and Enterprise levels, there are other activities that span across these levels. These activities are owned and managed by the Shared Teams.

Shared Teams are teams that operate across different levels of testing within your enterprise. Some members of a shared team may also hold roles within specific testing teams at the Team, Program, or Enterprise level. Others may not be involved in testing beyond their participation on a Shared Team.

Three common types of shared teams are the Community of Practice, the Enterprise Enablement Team, and the Systems Team.

COMMUNITY OF PRACTICE

A Community of Practice is a virtual group of people that consists of anyone in your enterprise who has anything to do with testing. This group represents the backbone of your organization's continuous testing efforts. They share strategies, lessons learned and best practices as well as collaborate to collectively decide on new tools and processes. Your community will bring together people who share the same goals around software quality.

ENTERPRISE ENABLEMENT TEAM

An Enterprise Enablement Team functions as a slim version of a test center of excellence. This team includes architects and engineers from the Enterprise level who are responsible for setting teams' testing strategy across the organization. They gather and propagate best practices across teams, define and manage operating models, enable new teams, orchestrate decision-making on testing tools and infrastructure through the Community of Practice, and represent QA in strategic discussions.

SYSTEMS TEAM

Systems Teams are teams within the Program and Enterprise levels who specialize in orchestrating integration testing and end-to-end testing. These teams could be made up of testers who will work with various testing teams to write integration tests and will then execute these tests.

Three common types of shared teams are the Community of Practice, the Enterprise Enablement Team, and the Systems Team.

VALUE TOOLS

Throughout your implementation of ACT, there are several Value Tools you can use to help you through your journey and accelerate adoption. These tools include:

ROI CALCULATION

Every major investment needs to show payback. It's essential to calculate ROI at each stage of testing transformation, so that you can not only motivate your teams, but also win additional funding for the project.

ASSESSMENTS AND REVIEWS

Equally as important as showing ROI is the need to assess your progress. By creating a feedback loop among all stakeholders, you ensure alignment and can implement changes that keep you moving towards your goals.

CASE STUDIES

As you progress through your ACT implementation, it's a good idea to research how others have implemented the framework to learn from their successes—and their struggles.

WORKSHOPS

Make it a priority to organize periodical tactical and strategic workshops across different levels of your organization to identify current strengths and future opportunities around quality. You'll learn how to optimize the processes and tools you're already using.

There are several Value Tools you can use to help you through your journey and accelerate adoption.

CONCLUSION

Implementing continuous testing across your organization may be a complex, multilayered process, but it doesn't have to be a daunting one. The ACT Framework is created to provide guidance based on lessons learned by many large organizations that have been successful on their own journeys. With a clear view of the business outcomes you hope to achieve, you can implement a continuous testing program that increases collaboration, shortens your development cycles, and enhances your overall product quality.

This paper only touches the surface of ACT. We have much more to share about it and how it can benefit your organization.

To make sure you don't miss out on any of these updates, subscribe now at **theactframework.com**