



Agile Testing: Your Key to Better Software

What's in the Way of Testing at the Speed of Agile?

Testing at the speed of agile drives quality at speed. But the question is, "When do you think the testing process should begin?" If you answered something like "after the code is written", or "once developers hand it off to QA", you are not thinking Continuous Testing. To truly achieve continuous testing, you need shift left—all the way left.

The challenge is to enable requirements design, test automation and development that fits into the same sprint, while allowing stakeholders—from business analysts to testers to developers—to stay in alignment and remain flexible. This is a tall order that requires replacing the typically slow, manual and error-prone testing process with a powerful, model-based approach to agile testing.

An Agile Testing approach addresses the key challenges that business analysts, testers and developers face when they attempt to create better software, faster. These pain points span across the testing lifecycle and include:

- Ambiguous requirements
- Poor test case design and limited coverage
- Waiting for test data
- Unavailability of system components
- No automation

Let's take a look at each of these problem areas and see how CA Agile Requirements Designer, a solution from CA Technologies for agile testing using a model-based approach, can help.



Of practitioners say testing is the biggest bottleneck¹



Of testing is still manual



Of critical dependencies unavailable



Of time spent looking for test data

Ambiguous Requirements

More often than not, software challenges are introduced at the very beginning of a project, during the requirements phase. Requirements are often ambiguous and incomplete and usually stored in disparate, static formats. Consequently, test cases are manually derived from incomplete requirements and the problem continues to compound as the development process proceeds. The result is that software often fails to deliver a good customer experience, or even worse, defects are detected later in the development lifecycle where they require far more time and resources to resolve. Because most testing teams operate manually, there is no way to automatically or easily update a test when requirements change. And there is a high probability that your requirements *will* change.

There is a better way— automatically model requirements as an active flowchart.

Here's how:

- Build a formal model automatically using CA Agile Requirements Designer that is accessible and understandable to the business analysts that already use VISIO, BPM or other tools.
- Eliminate ambiguity and incompleteness by creating a mathematically precise representation of a system.
- Bring users, business analysts and IT into close alignment with a model that everyone can share, review and use.

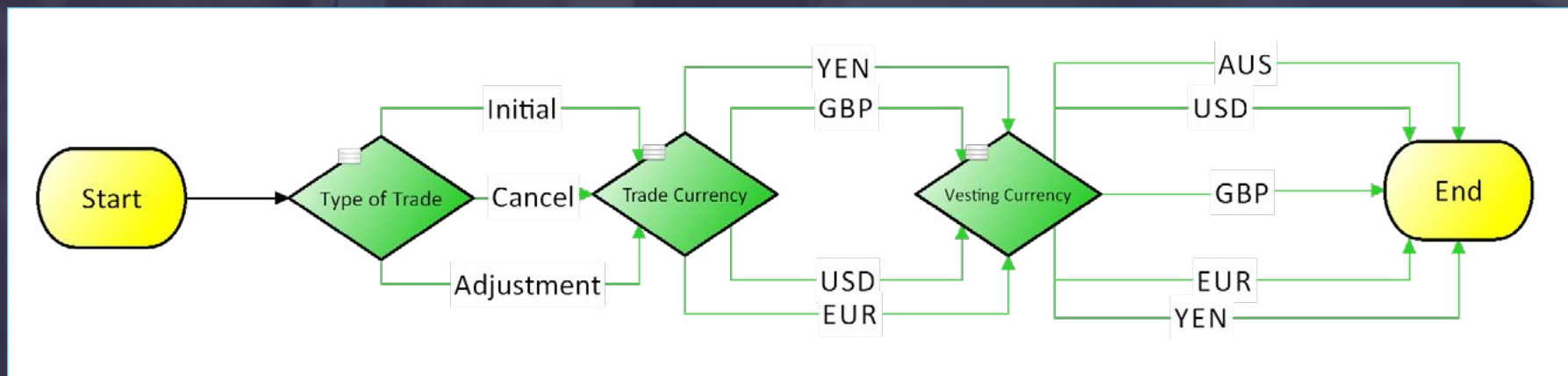


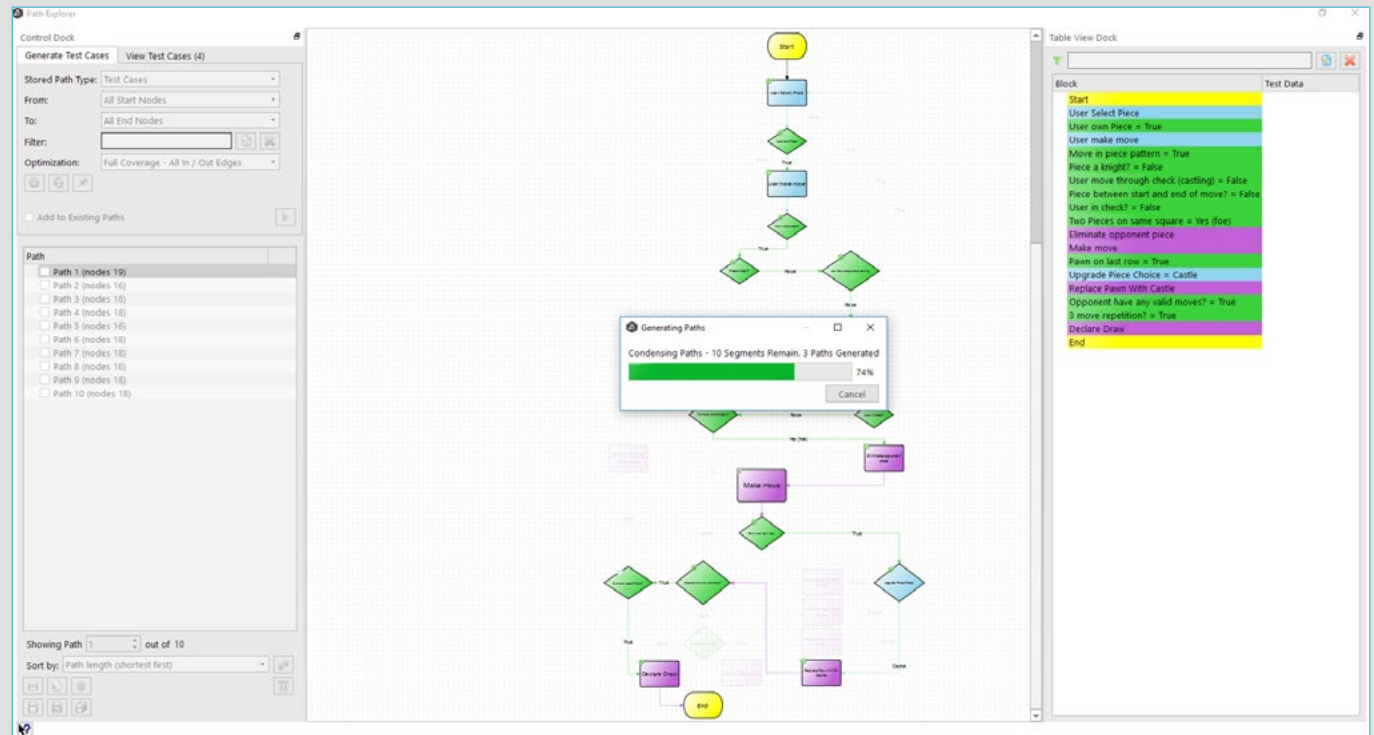
Figure 1. A basic flowchart depicting data flowing through a trading system application.

Poor Test Case Design and Limited Coverage

Manual test case design is a time consuming, error-prone process. Poor requirements get translated into poor test cases that lead to design flaws and faulty code.

There is no real notion of coverage because testing is usually conducted in an unsystematic and ad hoc fashion, leading to just 10 to 20 percent functional test coverage. Poor test case design also leads to significant over-testing of the same features of the application.

Figure 2.
An example of generating test cases using CA Agile Requirements Designer.



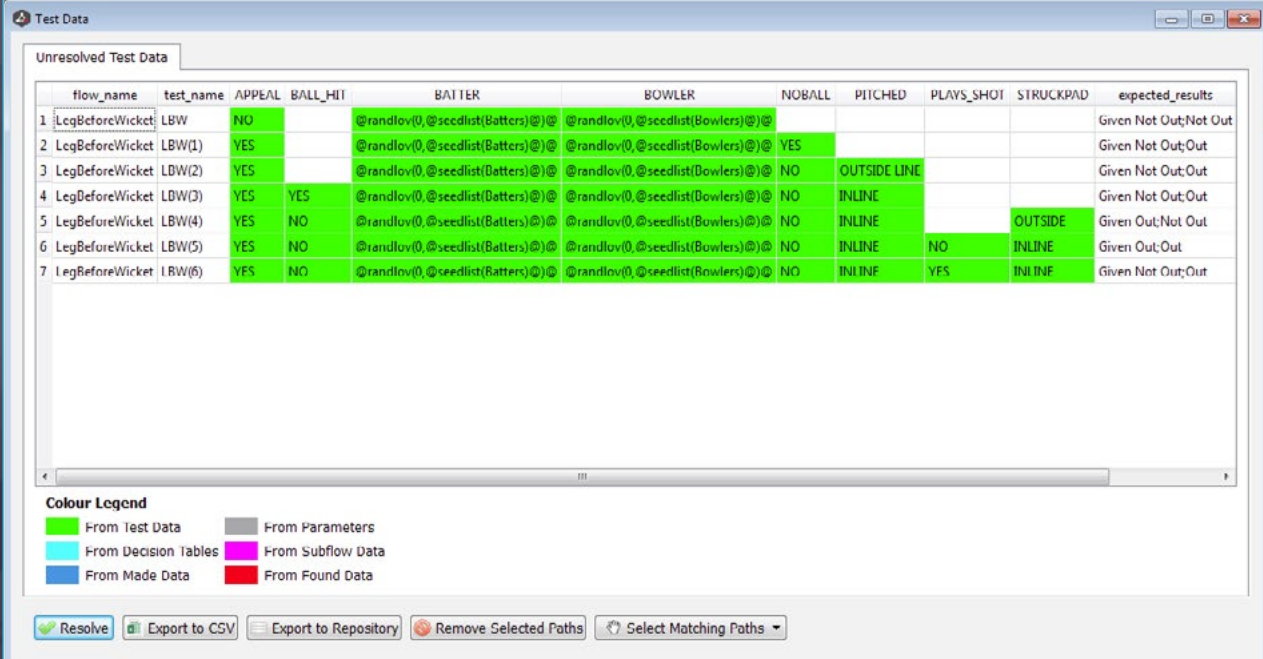
There is an easier way—automatically generate optimal sets of test cases.

Here's how:

- Generate test cases automatically from the solution's flowchart when the user stories are created.
- Test up to 100 percent of the specified functionality.
- Link the right data and expected results to the relevant user stories.
- Detect defects earlier and shorten test cycles so that testers can deliver software sooner and with a better customer experience.

Waiting for Test Data

The right data is never available when testers need it. Data is not linked to tests and testers have to sift through high-volume, low-variety production data sets, which don't provide adequate coverage. CA estimates that 20 percent of a software development lifecycle is spent waiting for data. These data constraints force testers to wait for data to become available upstream, which means data is not available in parallel, across teams, projects or releases. Data refreshes can take days or weeks to complete, causing significant delays in testing.



flow_name	test_name	APPEAL	BALL_HIT	BATTER	BOWLER	NOBALL	PITCHED	PLAYS_SHOT	STRUCKPAD	expected_results
1 LcgBeforeWicket	LBW	NO		@randlov(0,@seedlist(Batters))@	@randlov(0,@seedlist(Bowlers))@					Given Not Out;Not Out
2 LcgBeforeWicket	LBW(1)	YES		@randlov(0,@seedlist(Batters))@	@randlov(0,@seedlist(Bowlers))@	YES				Given Not Out;Out
3 LegBeforeWicket	LDW(2)	YES		@randlov(0,@seedlist(Batters))@	@randlov(0,@seedlist(Bowlers))@	NO	OUTSIDE LINE			Given Not Out;Out
4 LegBeforeWicket	LDW(3)	YES	YES	@randlov(0,@seedlist(Batters))@	@randlov(0,@seedlist(Bowlers))@	NO	INLINE			Given Not Out;Out
5 LegBeforeWicket	LBW(4)	YES	NO	@randlov(0,@seedlist(Batters))@	@randlov(0,@seedlist(Bowlers))@	NO	INLINE		OUTSIDE	Given Out;Not Out
6 LegBeforeWicket	LBW(5)	YES	NO	@randlov(0,@seedlist(Batters))@	@randlov(0,@seedlist(Bowlers))@	NO	INLINE	NO	INLINE	Given Out;Out
7 LegBeforeWicket	LRW(6)	YES	NO	@randlov(0,@seedlist(Batters))@	@randlov(0,@seedlist(Bowlers))@	NO	INLINF	YES	INLINF	Given Not Out;Out

Colour Legend

- From Test Data
- From Decision Tables
- From Made Data
- From Parameters
- From Subflow Data
- From Found Data

Buttons: Resolve, Export to CSV, Export to Repository, Remove Selected Paths, Select Matching Paths

Figure 3. Unresolved test data linked to stored paths in CA Agile Requirements Designer.

There is another way—test data on demand. Here's how:

- Automate data mining using CA Agile Requirements Designer to find existing data from multiple back-end systems.
- Use a comprehensive set of combinable data generation functions, system variables and seed lists to create any missing data needed for maximum coverage.
- Include future scenarios, outliers and unexpected results, which are not often found in production data.
- Link data to the test cases that feed directly into the test automation engine and avoid the delays created when the wrong data is delivered.

Unavailability of System Components



Environmental and data constraints create further delays as distributed teams sit idle waiting for unavailable system components (such as a production database) or data to become available. There are also delays when testing teams wait for test data to be created and provisioned by a central team, and data is usually not available in parallel. What's more, many organizations still copy and mask production data to their test environments. These copies are costly to maintain, carry compliance risks and cover only a fraction of the tests that need to be run.

There must be a better way—use self-service data.

Here's how:

- Use CA Agile Requirements Designer to define within the requirements and test the components that need to be virtualized.
- Use the self-service, on-demand portal to define virtual data needed and receive it automatically.
- Generate realistic virtual data when the optimized test cases are needed.
- Quickly and accurately define request/response pairs directly from a message definition or sampled traffic.

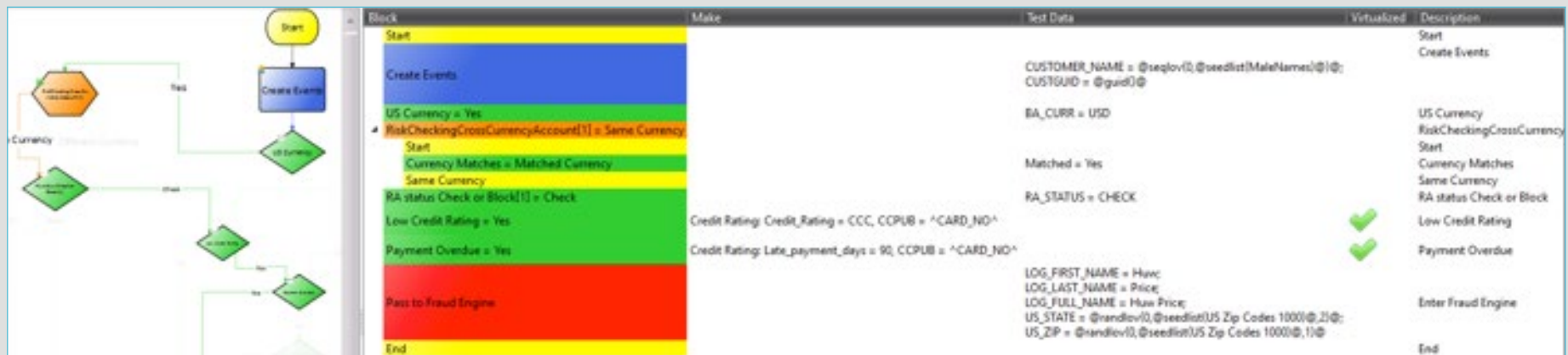
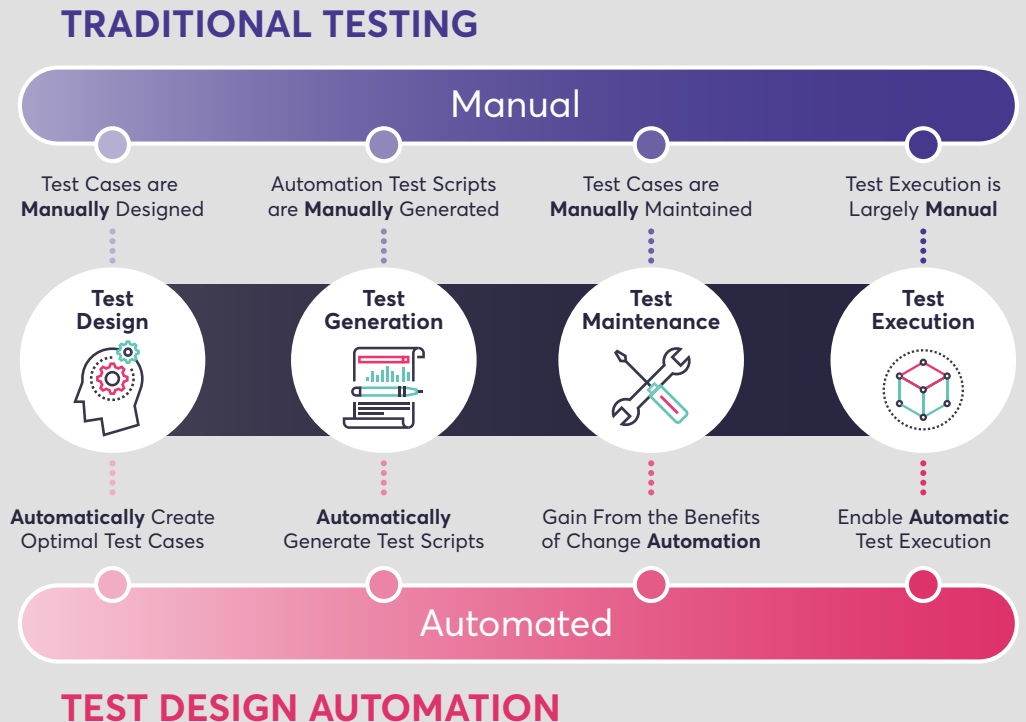


Figure 4. Link virtual end points to test cases in CA Agile Requirements Designer.

No Automation

Managing change is possibly the biggest issue for automation. In an ideal world, you want to automatically generate and maintain optimized, automated tests, that link to the data and the expected results needed to execute them. And when changes are made, you want to avoid going back to existing scripts and editing them.

For most organizations, the challenge is that automated testing frameworks are heavily scripted and the script generation is done manually, as is script maintenance. Some organizations use alternate solutions, such as record/playback or script-less automation frameworks (keywords). However, these approaches still bring you back to manual test case design. Ideally, you want to avoid going back around existing scripts and editing them again, which is simply impossible without an automation generator.



There is a faster way—use automated testing. Here's how:

- Use CA Agile Requirements Designer to automate test case design and test case creation to make exhaustive testing possible.
- Eliminate the time wasted on manual test design and maintenance.
- Keep up with changing user needs easily using an automation generator.
- Maximize the value of existing testing frameworks.
- Build up a library of reusable test components.

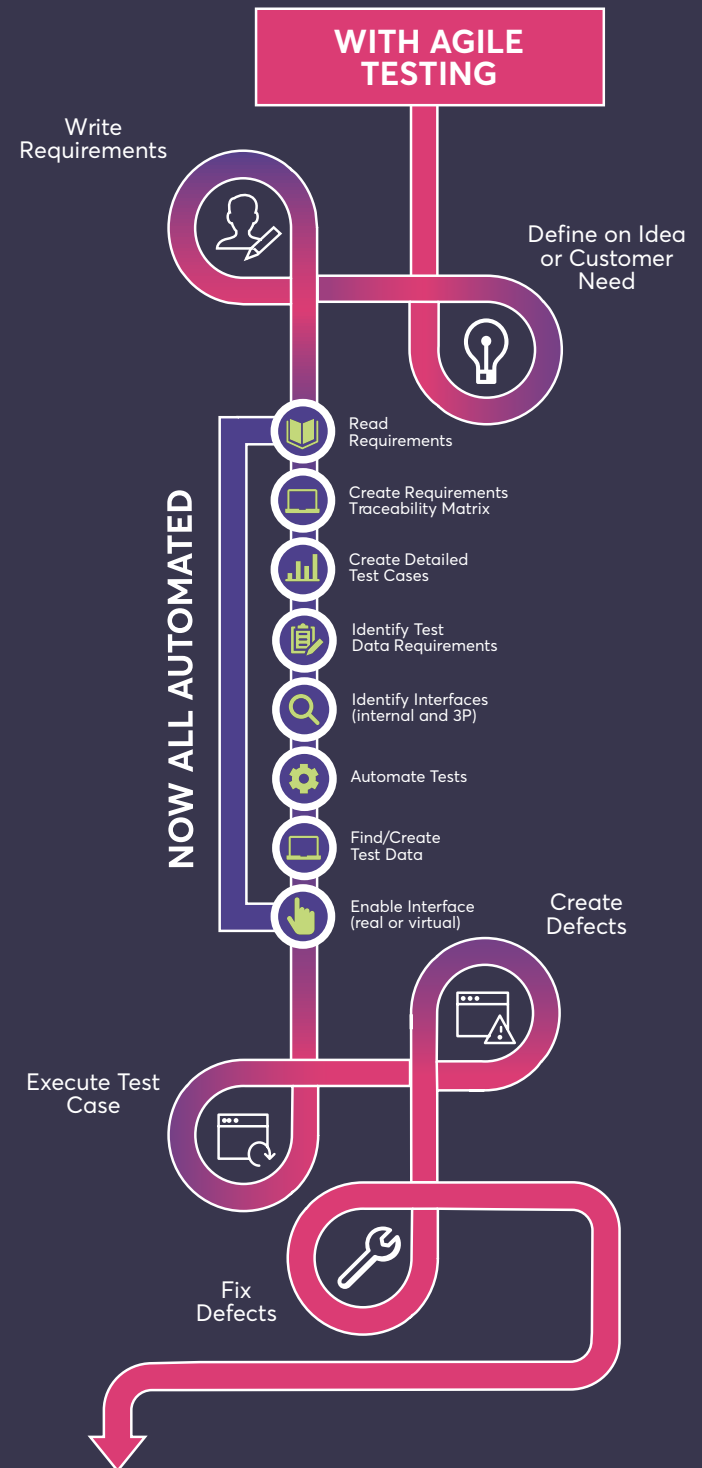
A Fully Integrated, End-To-End Testing Solution

CA Technologies helps you achieve continuous testing, by enabling you to shift left your testing effort into the requirements design phase, automating otherwise time-consuming test asset creation and maintenance tasks thereafter. CA Agile Requirements Designer is an automated testing and modeling solution, enabling organizations to build and deliver high-quality applications to market faster, at less cost. CA Agile Requirements Designer is a key component of continuous testing solutions from CA Technologies, helping organizations build better apps, faster.

CA Testing Solutions

- **CA Test Data Manager.** Creates all the data needed for testing, matched directly to test cases created in Agile Requirements Designer.
- **CA Service Virtualization.** Creates services needed for testing based on virtual end points modeled in CA Test Case Optimizer.
- **CA Agile Central.** Exports requirements to CA Agile Requirements Designer where they are optimized and converted into the right set of test cases.
- **CA BlazeMeter.** Exports test automation scripts from CA Agile Requirements Designer in a Taurus format, running these scripts in CA BlazeMeter for performance and load testing.

Figure 5. Continuous Testing from CA Technologies powers Agile Testing – from requirements to data and environment provisioning, test execution and maintenance.





Quality Software Delivered Earlier and at Less Cost

Together, CA testing solutions enable test teams to work in parallel and achieve rigorous testing within a sprint, even as their requirements are changing. Business analysts can communicate evolving user needs using unambiguous flowcharts, from which they can automatically derive subsequent test assets. This includes optimized test cases, test data, automated tests and virtual data. By eliminating delays created by manual testing and environment constraints and maximizing test coverage, you'll be able to deliver higher-quality applications on time and within budget.

Customer Success

One of the world's largest bank was able to reduce test script creation time by 70%.⁴

Here at CA Technologies, our regression testing went from 40 days to 3 by using CA Agile Requirements Designer.⁶

The CA Technologies solution will help Rabobank increase the efficiency of their business analysts by 10 percent and testers by more than 30 percent over the next three years.⁵

a.s.r., an insurance and financial services firm was able to map a flowchart model to existing requirements in just four hours.³

³ CA Technologies, "Case Study: Agile Requirements Designer at a.s.r.,"

⁴ IT Central Station Review

⁵ CA Technologies Success Story, "Rabobank improves the customer experience with better apps and faster testing founded on CA Agile Requirements Designer,"

⁶ All Day DevOps, <http://cainc.to/hdvHlc>

Learn more.



CA Technologies (NASDAQ: CA) creates software that fuels transformation for companies and enables them to seize the opportunities of the application economy. Software is at the heart of every business, in every industry. From planning to development to management and security, CA is working with companies worldwide to change the way we live, transact and communicate—across mobile, private and public cloud, distributed and mainframe environments. Learn more at ca.com.

Copyright © 2018 CA, Inc. All rights reserved. All trademarks referenced herein belong to their respective companies. This document does not contain any warranties and is provided for informational purposes only. Any functionality descriptions may be unique to the customers depicted herein and actual product performance may vary.

CS200-372668

