ERAN KINSBRUNER

A Guide From Key DevOps Industry Experts

Accelerating Software Quality

Machine Learning and Artificial Intelligence in the Age of DevOps

PERFORCE

Chapter 2: How Do AI and ML Testing Tools Fit and Scale in the DevOps Pipeline?

After we've defined some key use cases for using AI and ML methods to improve overall test automation stability and reliability, let's map and integrate new types of testing within the DevOps pipeline. We need to define this because new types of tests often originate from external tool stacks and different personas.

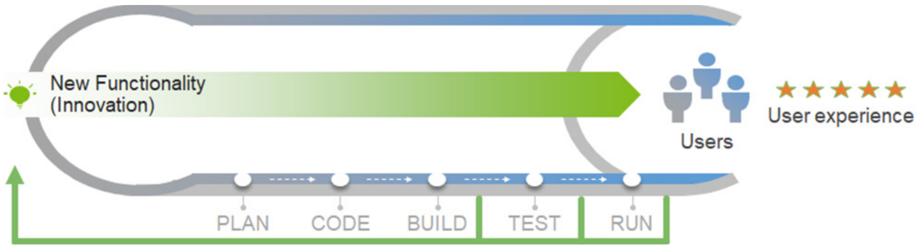


Fig. 23: DevOps Pipeline Illustration (Source: Perfecto)

A typical DevOps pipeline looks like the image above. New requirements, story points, and functionalities are introduced. They then undergo development, building, integration, testing, and deployment. Hopefully, at the end, there are happy customers and a 5-star experience.

More specifically, as seen in the below illustration, the process is being divided into phases by personas. The developer codes based on the product requirements. They perform unit testing against their web and/or mobile apps from their local machine. Once the unit tests pass, they will perform a software build and perform some build acceptance testing upon a commit to the main branch.

Once integrated to the main repository, the entire build is tested more thoroughly through cloud solutions, scalable environments, and more. In this stage, a lot of end-to-end testing, regression testing, new functionality testing, and non functional testing — including security, acceptance, performance, accessibility, and more — are performed until quality level is met. After that, software is deployed to the production environment.

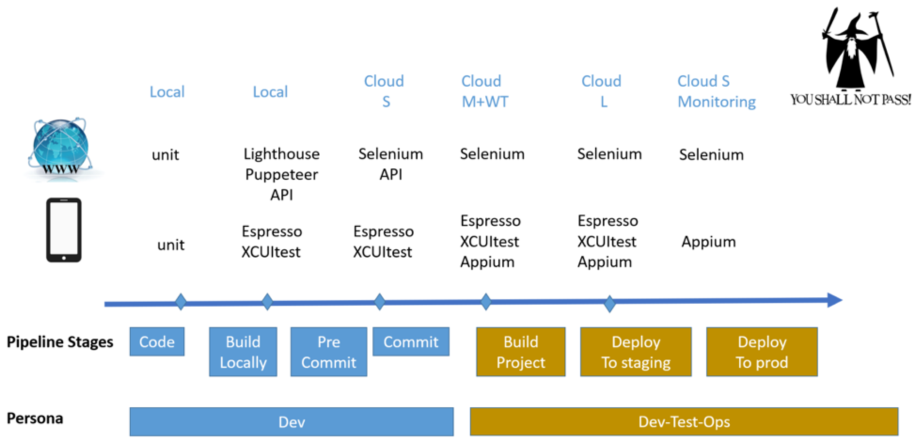
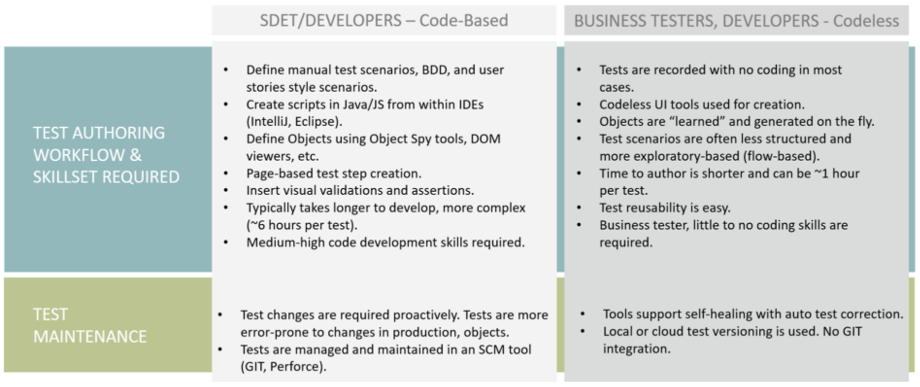


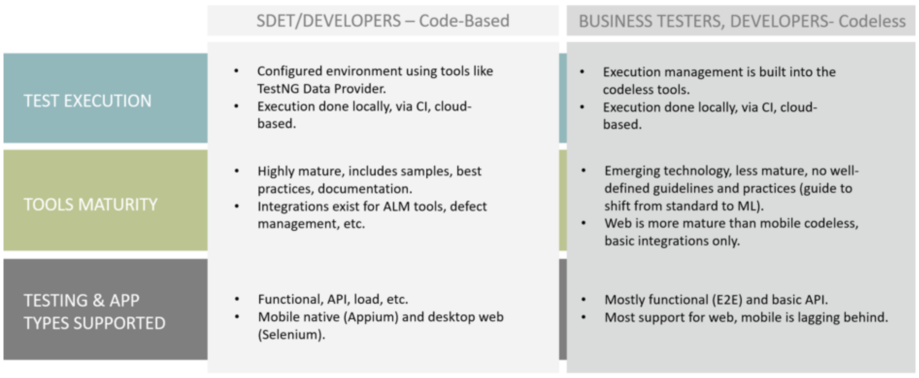
Fig. 24: DevOps Pipeline by Stages Across Phases (Source: Perfecto)

The above process is fine and should not change dramatically by introducing AI and ML tools into various phases of the pipeline. The complexity, as well as the need, in this case is to incorporate and integrate AI and ML tools and techniques into the working processes above. In addition, ML and AI can enable continuous learning and improvements based on gathered data, defect analysis, code analysis, and more.

Before outlining recommendations for AI and ML tools to use alongside non-AI/ML tools, here are some of the main differences between code-based tools (testing only) and codeless (ML-based) tools. The comparison is done from end to end, meaning, from creation through execution, integration, and analysis.



	SDET/DEVELOPERS – Code-Based	BUSINESS TESTERS, DEVELOPERS - Codeless
TEST AUTHORING WORKFLOW & SKILLSET REQUIRED	• Define manual test scenarios, BDD, and user stories style scenarios. • Create scripts in Java/JS from within IDEs (IntelliJ, Eclipse). • Define Objects using Object Spy tools, DOM viewers, etc. • Page-based test step creation. • Insert visual validations and assertions. • Typically takes longer to develop, more complex (~6 hours per test). • Medium-high code development skills required.	• Tests are recorded with no coding in most cases. • Codeless UI tools used for creation. • Objects are "learned" and generated on the fly. • Test scenarios are often less structured and more exploratory-based (flow-based). • Time to author is shorter and can be ~1 hour per test. • Test reusability is easy. • Business tester, little to no coding skills are required.
TEST MAINTENANCE	• Test changes are required proactively. Tests are more error-prone to changes in production, objects. • Tests are managed and maintained in an SCM tool (GIT, Perforce).	• Tools support self-healing with auto test correction. • Local or cloud test versioning is used. No GIT integration.



	SDET/DEVELOPERS – Code-Based	BUSINESS TESTERS, DEVELOPERS- Codeless
TEST EXECUTION	• Configured environment using tools like TestNG Data Provider. • Execution done locally, via CI, cloud-based.	• Execution management is built into the codeless tools. • Execution done locally, via CI, cloud-based.
TOOLS MATURITY	• Highly mature, includes samples, best practices, documentation. • Integrations exist for ALM tools, defect management, etc.	• Emerging technology, less mature, no well-defined guidelines and practices (guide to shift from standard to ML). • Web is more mature than mobile codeless, basic integrations only.
TESTING & APP TYPES SUPPORTED	• Functional, API, load, etc. • Mobile native (Appium) and desktop web (Selenium).	• Mostly functional (E2E) and basic API. • Most support for web, mobile is lagging behind.

Fig. 25: Key Differences in Code-Based Testing and Codeless Testing Approaches (Source: Perfecto)

As outlined in the above tables, there are few material differences between the two approaches. While they complement each other from a test coverage percentage perspective, each requires a different methodology and cross-team discipline. The main consideration is the process, specifically around the following key questions.

How Are New Test Scripts Created Using Both Codeless and Coded Approaches?

As identified above, codeless mostly uses record and playback, while coded is being done in an IDE like IntelliJ or Eclipse. The art in combining both methods involves communication between two different types of personas. Communication prevents duplication of test scenarios. It also ensures that there's a full sync around test coverage, as well as alignment regarding test execution schedules and other project risks.

How Does One Maintain the Two Sets of Test Automation Suites?

Maintaining code should be done as a developer maintains his production code. However, for codeless, maintenance is done through a mix of touch-free self-healing capabilities and overriding pre-recorded scenarios manually.

How Can Management Get Full Visibility Into Quality and Test Results Per Execution?

This is another tricky part since the two approaches report the results into two different dashboards. Leaders should insist on a single pane of glass that captures both results regardless of their origin. While test automation reports prior to codeless originated through API tools, unit testing, CI, and more, and management was able to view it all, codeless should be just one additional origin into the entire pipeline.

Codeless Automation
Code-Based Automation

```
private RemoteWebDriver driver;

@BeforeMethod
void beforeMethod() throws MalformedURLException {

    // For more capabilities and supported platforms, see http://developers.perfectomobile.
    DesiredCapabilities capabilities = new DesiredCapabilities();
    capabilities.setCapability("platformName", "Windows");
    capabilities.setCapability("platformVersion", "10");
    capabilities.setCapability("browserName", "Chrome");
    capabilities.setCapability("browserVersion", "latest");
    capabilities.setCapability("resolution", "1280x1024");
    capabilities.setCapability("securityToken", Utils.Capabilities.getToken());

    //For Regular web (physical devices) remove the /fast at the end of the Url
    URL url = new URL("https://" + Utils.Capabilities.getHost() + "/nexperience/perfectomob

    driver = new RemoteWebDriver(url, capabilities);
    driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
    driver.manage().timeouts().pageLoadTimeout(30, TimeUnit.SECONDS);
}

@Test
public void test() {
    System.out.println("Run started");

    driver.get("http://google.com.com/");

    // complete your test here
```

Fig. 26: Combination of Code and Codeless Test Automation (Source: Perfecto)

How Do I Fit ML and Codeless Testing Within the Other Tools in My Stack?

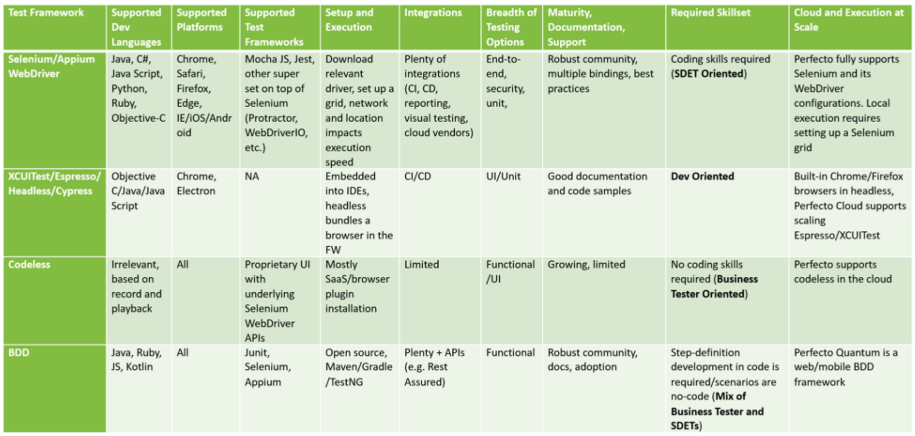
To try answering this question, here is a basic summary table that lists the leading test automation frameworks, divided into various considerations:

- Skills.
- Coverage areas.
- Supported platforms.
- Supported testing types.
- Maturity.

The most important column in this table is the persona matching under the required skillset. It's clear that we want both approaches complementing each other as figure 26 suggests. However, there are personas behind these tools that should drive and own the quality objectives for each. Let's take an example from the table below. Developers that mostly tackle the unit testing challenge will continue to focus on more advanced frameworks that require minimal setup and maintenance like Cypress.io, Puppeteer for web, Flutter, or Espresso/XCUITest for mobile.

On the other hand, test automation engineers that own end-to-end test automation will work mostly with Selenium for web and Appium for mobile to get greater coverage of tests and platforms.

So, where does this leave codeless and BDD (behavior driven development)? For BDD, business testers will own the gherkin test scenario creation, while depending on test automation developers or application developers to implement the custom functions (Java, Python, other). For codeless, business testers can own the entire lifecycle of test creation, execution, maintenance, and reporting since these tools don't require coding skills.



Test Framework	Supported Dev Languages	Supported Platforms	Supported Test Frameworks	Setup and Execution	Integrations	Breadth of Testing Options	Maturity, Documentation, Support	Required Skillset	Cloud and Execution at Scale
Selenium/Appium WebDriver	Java, C#, Java Script, Python, Ruby, Objective-C	Chrome, Safari, Firefox, Edge, IE/iOS/Android	Mocha JS, Jest, other super set on top of Selenium (Protractor, WebDriverIO, etc.)	Download relevant driver, set up a grid, network and location impacts execution speed	Plenty of integrations (CI, CD, reporting, visual testing, cloud vendors)	End-to-end, security, unit,	Robust community, multiple bindings, best practices	Coding skills required (**SDET Oriented**)	Perfecto fully supports Selenium and its WebDriver configurations. Local execution requires setting up a Selenium grid
XCUITest/Espresso/ Headless/Cypress	Objective C/Java/Java Script	Chrome, Electron	NA	Embedded into IDEs, headless bundles a browser in the FW	CI/CD	UI/Unit	Good documentation and code samples	**Dev Oriented**	Built-in Chrome/Firefox browsers in headless, Perfecto Cloud supports scaling Espresso/XCUITest
Codeless	Irrelevant, based on record and playback	All	Proprietary UI with underlying Selenium WebDriver APIs	Mostly SaaS/browser plugin installation	Limited	Functional /UI	Growing, limited	No coding skills required (**Business Tester Oriented**)	Perfecto supports codeless in the cloud
BDD	Java, Ruby, JS, Kotlin	All	Junit, Selenium, Appium	Open source, Maven/Gradle /TestNG	Plenty + APIs (e.g. Rest Assured)	Functional	Robust community, docs, adoption	Step-definition development in code is required/scenarios are no-code (**Mix of Business Tester and SDETs**)	Perfecto Quantum is a web/mobile BDD framework

Fig. 27: Mapping Sample Test Automation Frameworks to Personas, Objectives, and Skillsets (Source: Perfecto)

Note that the pipeline can accommodate more than a single test automation framework and can benefit even more personas contributing to the overall quality based on their skillsets, capabilities, and objectives.

In addition to software testing opportunities for AI and ML to help optimize an ongoing pipeline, there is additional software waste — known as 'TIMWOOD' — that can be enhanced when applying AI models.[25]

Think about eliminating the transfer of sign-offs and approvals between phases. This can be done with AI algorithms that automatically scan a release based on lists of data points and approve the transition to the next phase. Defects and rework

25 Tackle Software Wastes in DevOps - https://enterprisersproject.com/article/2019/10/how-scale-devops-time-wasters activity created by regression issues (covered as “d” in the above acronym) can be eliminated with automated regression suites that are automatically changed and updated based on historical issues.

BOTTOM LINE

Taking all of this into a strategic consolidation of AI, ML, and traditional tools, DevOps teams must work based on actionable and up-to-date data. Such data should cover events and relevant content across all phases of the DevOps pipeline, including production data. Being able to tune software iterations based on real-time production insights automatically is a DevOps dream.

Think about a pipeline that gets production inputs on a real-time defect, or a new release on a specific platform that is buggy and can loop all relevant services into a quick response. In these cases, remediation is a massive productivity and velocity boost.

AI and ML are able to look into historical data analysis, software anomalies, performance analysis, correlation of issues to code and product areas, and much more. Such data can be passed automatically to the right systems and personas for re-assessment, automation of new tasks, and resolution.

An automated end-to-end process for the AI/ML pipeline can accelerate development and drive reproducibility, consistency, and efficiency across AI/ML projects. Since AI and ML models take time to be built into consistent and predictable tools, consider matching such models in the “right” places, and that will be best used over long periods of time as opposed to applying these to one-time model use cases.

As noted in the Continuous Testing for DevOps Professionals book, DevOps is all about bringing together business, development, testing, release, and operational expertise to deliver a valuable solution to customers. Ensure that AI/ML is represented on feature teams and is included throughout the design, development, testing, and operational sessions.

Chapter 9: The New Categories of Software Defects in the Era of AI and ML

Tzvika Shahaf

Tzvika Shahaf *is the VP of Product Management at Perfecto.[92] His experience includes business development, strategy, and investment in technology companies and venture capital firms. His passion is building new, powerful, and effective ways to collaborate with Global 2000 enterprises in order to resolve high-impact business problems using data-driven processes and analytics. Tzvika is partnering with leading DevOps teams to revolutionize the testing space by making it smarter, faster, and cost effective with a clear goal of maturing software delivery lifecycle. Tzvika is keynote speaker at industry leading events, blogger, and a Co-Author of the book, "Continuous Testing for DevOps Professionals: A Practical Guide from Industry Experts."*

INTRODUCTION

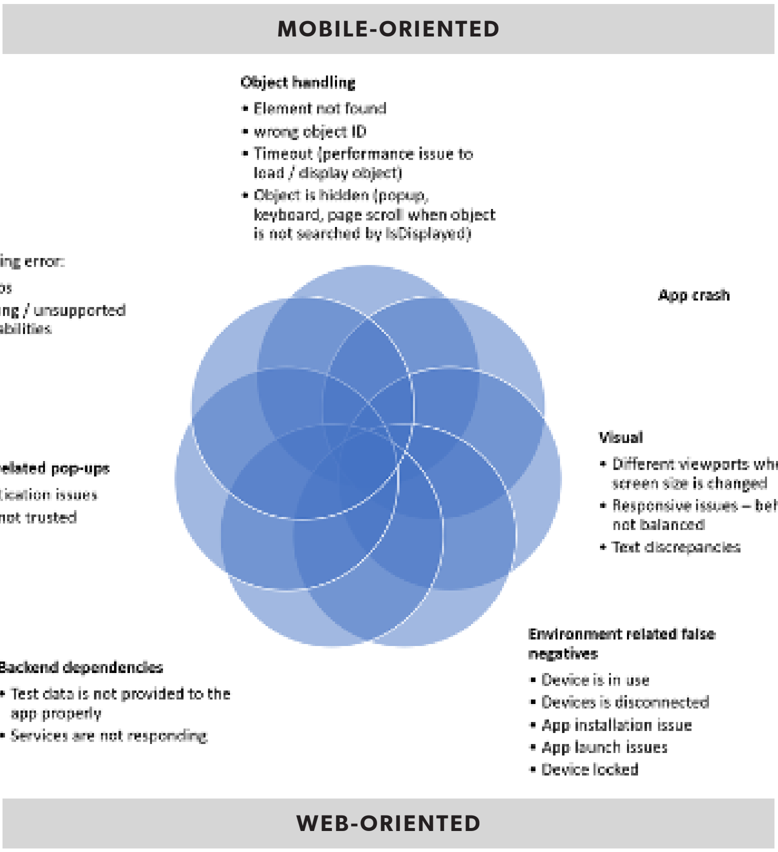
Test automation is an art. But there are fundamental differences between the nature of failure based on application types (mobile native vs. web), coding techniques, and more.

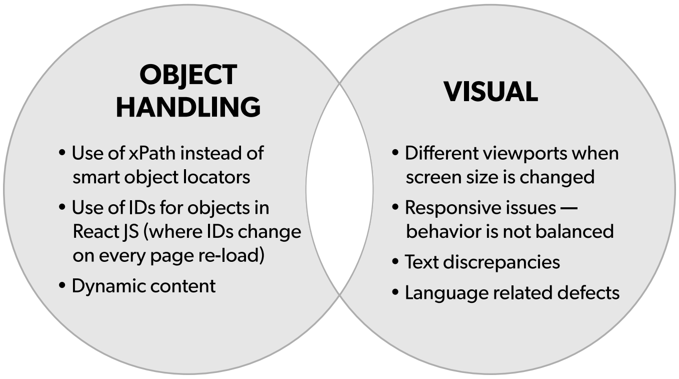
Specifically, there are key differences between a mobile native app and web browser app as it relates to the structure of the UI layer, as well the sustainability of the test automation and debugging process, as described below:

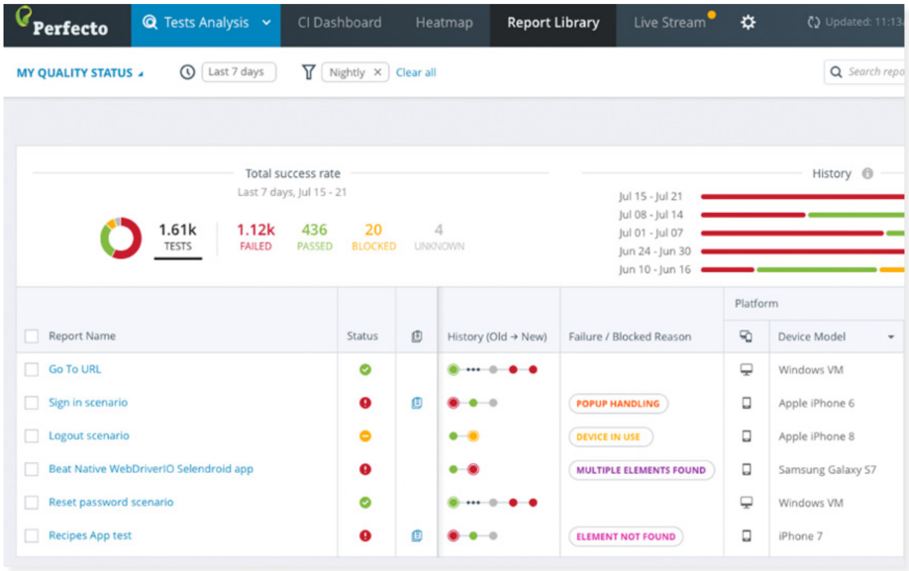
1. Mobile UI tree vs. websites DOM objects tree.
2. Dynamic website content (frequently changed objects) vs. native apps with relatively stable object structure.
3. Flakiness in mobile due to unexpected events like popups, incoming calls, interactions with sensors, app signature, environment-handling activities (file/image upload, installation issues, orchestration, and more).
4. The nature of bug fixes is different and the cycle to release is shorter on web apps:
    a. Web — Push changes easily to validate fix.
    b. Mobile — Build process is complicated and installation of fix requires compilation (either locally as IPA/APK) or longer push to the App Store, etc.

92 http://perfecto.io

With the above-mentioned challenges, it’s encouraging to see that as this book is being developed, existing ML/AI techniques have already made significant progress in their ability to identify (and resolve in some cases) some of the following classified failures.


MOBILE-ORIENTED
Object handling
• Element not found
• wrong object ID
• Timeout (performance issue to load / display object)
• Object is hidden (popup, keyboard, page scroll when object is not searched by IsDisplayed)
Scripting error:
• Typos
• Wrong / unsupported capabilities
App crash
Visual
• Different viewports when screen size is changed
• Responsive issues – behavior is not balanced
• Text discrepancies
Security related pop-ups
• Authentication issues
• User is not trusted
Environment related false negatives
• Device is in use
• Devices is disconnected
• App installation issue
• App launch issues
• Device locked
Backend dependencies
• Test data is not provided to the app properly
• Services are not responding
WEB-ORIENTED


OBJECT HANDLING
• Use of xPath instead of smart object locators
• Use of IDs for objects in React JS (where IDs change on every page re-load)
• Dynamic content
VISUAL
• Different viewports when screen size is changed
• Responsive issues — behavior is not balanced
• Text discrepancies
• Language related defects



Total success rate
Last 7 days, Jul 15 - 21

1.61k	1.12k	436	20	4
TESTS	FAILED	PASSED	BLOCKED	UNKNOWN

History
Jul 15 - Jul 21
Jul 08 - Jul 14
Jul 01 - Jul 07
Jun 24 - Jun 30
Jun 10 - Jun 16

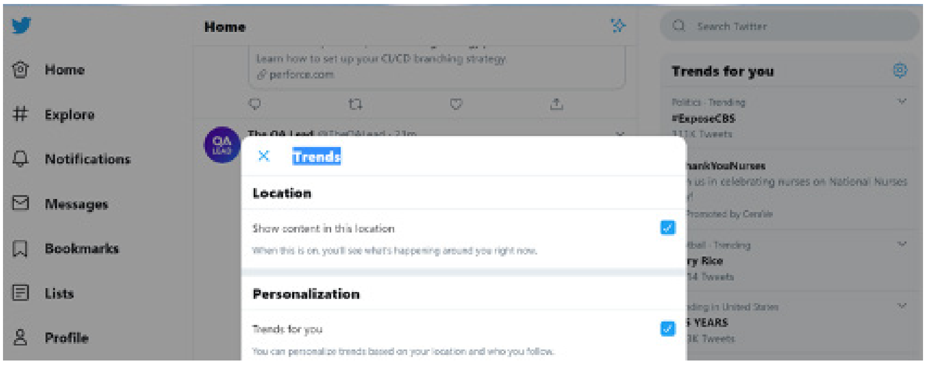
Report Name	Status	History (Old → New)	Failure / Blocked Reason	Platform	Device Model
Go To URL	✓				Windows VM
Sign in scenario	!		POPUP HANDLING		Apple iPhone 6
Logout scenario	−		DEVICE IN USE		Apple iPhone 8
Beat Native WebDriverIO Selendroid app	!		MULTIPLE ELEMENTS FOUND		Samsung Galaxy S7
Reset password scenario	✓				Windows VM
Recipes App test	!		ELEMENT NOT FOUND		iPhone 7

Fig. 60: Error Classification of Traditional Test Automation Failures (Source: Perfecto)

NEW DEFECT CATEGORIES THAT WILL EVOLVE WITH THE RISE OF AI/ML

As part of the need to deal with complex development scenarios, big data challenges, and scale, application and software developers started to embed AI/ML algorithms as part of their development process. Even if the users are not always aware of the underlying advancements, and the UI looks the same on the website or mobile app, in many cases, there are massive algorithms that are running and gathering insights to help serve the clients better.

To give an example, Twitter, Amazon, and the majority of other news websites are leveraging content personalization algorithms based on tastes/preferences and browsing history. It is all is based on a learning system that collects these data points.



Learn how to set up your CI/CD branching strategy
perforce.com

Trends

Location

Show content in this location ✓
When this is on, you'll see what's happening around you right now.

Personalization

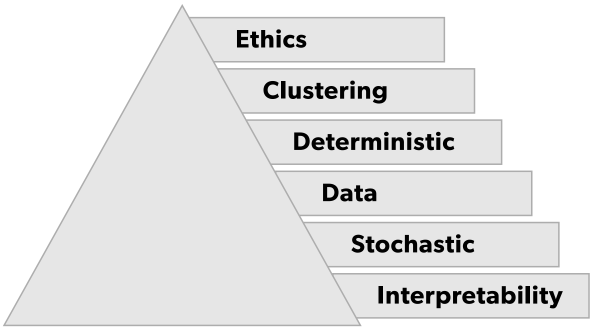
Trends for you ✓
You can personalize trends based on your location and who you follow.

Politics · Trending
#ExposeCBS
111K Tweets

Fig. 61: Twitter AI-Based Trending Algorithms (Source: Twitter App)

The abovementioned reality is going to transform the nature of defects in a significant way and contribute to the creation of a new breed of sophisticated defects. This is set to change the map of defects that DevOps teams use to detect and fix today.

The use of AI/ML in a new type of apps (AIIA[93] — AI Infused Applications) will evolve the following set of new defect categories. These are the top six new categories that should be considered for AI/ML defect classifications.



93 Introduction to AI Infused Applications, Forrester https://go.forrester.com/blogs/no-testing-no-artificial-intelligence/

1. ETHICS

An AI-based algorithm makes predictions based on the user's ability to train its engine. The algorithm will label things based on the data it is trained on. Hence, it will simply ignore the correctness of data. For example, if the algorithm is trained on data that reflects racism or sexism, the prediction will mirror it back instead of correcting it automatically. Therefore, one needs to make sure that the algorithms are fair, especially when it is used by private and corporate individuals

From a developer view, this defect category means that training the AI engine should also include a dedicated set of rules and data that refers to ethics, depending on the target market segments, geographies, and exposure of the app or website.

From a tester's perspective, such a category needs to be included in the test planning and classified upon relevant detection of relevant issues. It will also require the ability to perform all sorts of testing within the lifecycle of the app (unit/APIs/UI/data inputs, etc.).

Fig. 62: Self-Driving Cars (Source: CIO.com[95])

In a recent article by Harvard Magazine, the author gave an example around the use of autonomous cars and people with an arrest warrant against them entering such vehicles.[94] The dilemma here is whether or not the car should drive the suspect directly to the nearest police station without acknowledging a potentially life-threatening emergency that may require a different behavior. In this example, there needs to be various rules and conditions that are part of the AI algorithm that can make a decision that matches the reality as much as possible.

94 Harvard Magazine, AI Limitations https://harvardmagazine.com/2019/01/artificial-intelligence-limitations
95 How Singapore Is Developing Autonomous Cars https://www.cio.com/article/3294207/how-singapore-is-driving-the-development-of-autonomous-vehicles.html

2. CLUSTERING

This challenge may occur when data is not labelled but can be divided into groups based on similarity and other measures of natural structure in the data. An example is the organization of pictures by faces without names, where the human user has to assign names to groups, like iPhoto on Macs. The complexity here is to get the groups right and continuously expand the data in a correct manner. In machine learning, there are various ways of clustering data sets, such as K-means, density-based methods, and others. An additional clustering use case applies to the biology field, where an ML solution can help classify different species of plants and animals.

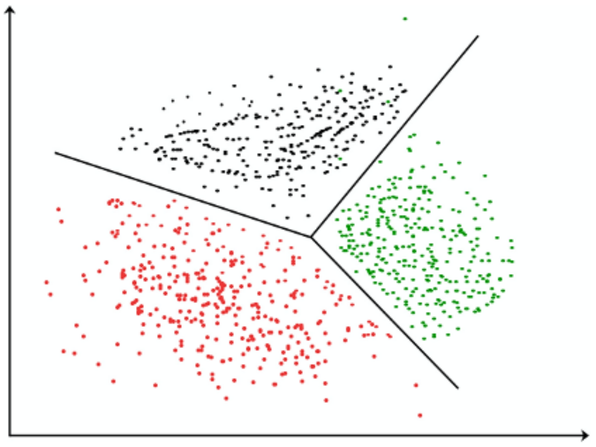


Fig. 63: Clustering of Data in Machine Learning (Source: GeeksforGeeks[96])

Here is an example of the K-means[97] algorithm:

```
Initialize k means with random values
    For a given number of iterations:
        Iterate through items:
            Find the mean closest to the item
            Assign item to mean
            Update mean
```

From a developer's standpoint, the algorithms that are being developed and used must be based on the right characteristics. They must be trained based on large and cohesive data sets.

96 Clustering in Machine Learning https://www.geeksforgeeks.org/clustering-in-machine-learning/
97 K-Means Algorithm Definition https://www.geeksforgeeks.org/k-means-clustering-introduction/

From a test engineering standpoint, in addition to adding a new category to the classified test failures, such a persona must challenge as much as possible. This can be done through testing and parallel data sets to obtain as many outputs as possible in order to build trust in the clusters. In addition, as the product matures, new clusters, as well as data points will be added — this needs to be continuously tested and fed into the testing processes.

3. DETERMINISTIC PROBLEMS

Machine learning and AI algorithms aren't well designed in various cases (e.g. determining weather forecast through analysis of massive data points from satellites and other sensors) to deal with stochastic events. ML can be limited and generate wrong outputs due to the fact that it does not have physical constrains like "real" platforms that are led by humans. As technology evolves, such constrains may be limited. However, this is a category that requires the awareness of developers and testers.

From a developer perspective, they will need to understand the limitations and constrains of the algorithms in the edge cases and situations where things such as the abovementioned examples may occur, and either reroute the app to an alternative source, or avoid using the algorithm altogether.

From a testing perspective, test engineers will need to include the "human" scenarios in such use cases and challenge the apps in various happy and negative paths toward a trustworthy algorithm.

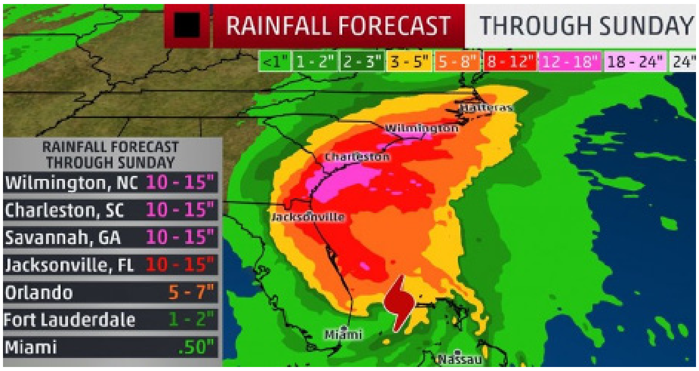


Fig. 64: Weather Forecast Using AI (Source: Interesting Engineering[98])

98 Weather Forecast Using AI https://interestingengineering.com/ai-might-be-the-future-for-weather-forecasting

4. DATA

In this book, this word might be the most repeated one, since data resides at the core of all major AI/ML algorithms, and it is in charge of the success or failure of apps that leverage such algorithms. When thinking of data, there can be a few types of data-related failures, as outlined below:

a. Lack of data.

b. Lack of good data.

From a developer standpoint, the algorithms must be trained with large and accurate sets of data that are relevant to the problems being handled, as well as to be solid enough to cover varying conditions. Such algorithms need to also consider the above and below failure types like ethics, deterministic approaches, stochastics, and more.

From a testing perspective, the entire test plan must include the right level of scenarios that challenge the apps and websites through various data points — good or bad. The test plan must also place proper assertions so that developers can understand the data-specific root cause of failure. Maintaining the tests over time and updating the test data is of course something that must be included in the test planning.

As Google wisely advises (see Fig. 65), there needs to be a proper separation between the model training data set, and the test data sets.

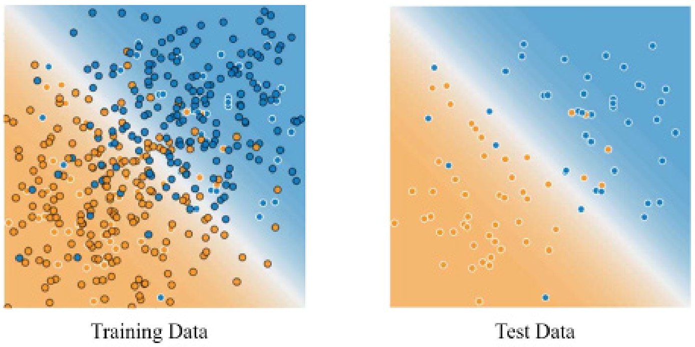


Fig. 65: Training Data Set vs. Test Data (Source: Google Developers[99])

99 Google Recommendations for Training ML based on Data
https://developers.google.com/machine-learning/crash-course/training-and-test-sets/splitting-data

5. STOCHASTIC & SPECULATIVE

This category refers to the validity and quality of the trained engine to handle unexpected results or outputs.[100] This can be a real issue that in many cases can cross-path with the abovementioned categories as well. AI/ML models and algorithms are being created from the beginning to handle such issues. But when they fail, this category needs to be clear and properly reported to developers. The inability for a model to sometimes realize a relationship between two variables may result in a wrong speculation that will obviously then result in a wrong output. Sometimes such an output can cause serious outcomes.

From a developer standpoint, when they develop the algorithm, it must leverage best practices like P-Hacking (data phishing) or scope-analysis to base outputs on mountains of data until a correlation between variables is showing a statistically-consistent result.[101]

From a test engineering perspective, testers must model the applications in a way that they are challenged by multiple variables from various angles to test the reliability of the model, relevancy of the outputs, and the consistency over time and use cases. One of the common failures around ML/AI algorithm as it relates to this category are false positive results. These need to be identified and eliminated in the testing phases. A recommendation to testers is to not test the system after "looking" at the data, but rather test using statistical approaches and pre-registered data, and then analyze the app.

Fig. 66: Stochastic Visualization (Source: Google Web)

100 Stochastic vs. Batch vs Mini-Batch https://mc.ai/batch-vs-mini-batch-vs-stochastic-gradient-descent-with-code-examples/
101 A Primer to P-Hacking https://www.methodspace.com/primer-p-hacking/

6. INTERPRETABILITY

This is a very important failure category that is not only technical, but also quite business related. If the selected model is not interpretable, then it does not serve its purpose and will cause major regressions. Interpretability is a paramount quality that machine learning methods should aim to achieve if they are to be applied in practice. As an example, if a model cannot prompt simple, relevant, and understandable outputs to the clients, they won't be used or accepted by them.

From a developer standpoint, models must translate the algorithm outputs in a meaningful and simple manner back to the users. Once developers can achieve this objective, they will get back relevant feedback from the users, together with growth of usage and system adoption.

From a testing perspective, testers must focus on the business outcomes of such embedded ML/AI algorithms, so the product meets its purpose and drives back happy customers. Testing for unclear strings, outputs of chatbots, translations problems, context-related issues, and others must be covered and reported back to the developers.

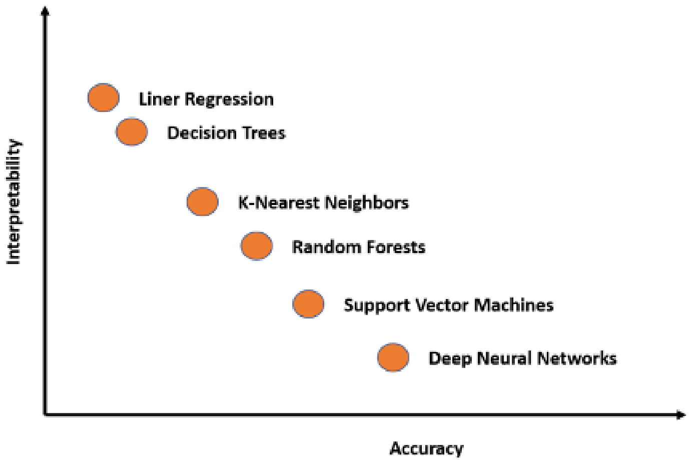


Fig. 67: Interpretable Machine Learning Models (Source: Analytics Vidhya[102])

102 Interpretable ML Models https://www.analyticsvidhya.com/blog/2019/08/decoding-black-box-step-by-step-guide-interpretable-machine-learning-models-python/

SUMMARY AND RECOMMENDATIONS

In this chapter I outlined the upcoming defect types and categories that are expected to pop up as more ML/AI models are embedded into our existing mobile and web applications. While such models aim to add more value to end users, and optimize their content consumption, they are also creating potential quality risks that must be acknowledged and addressed like any other quality risks that are not ML based. The key for success will be to embed the two types of defects into a single defect management system together with proper classification of the defects so the developers can distinguish the root cause and resolve it fast. The time to resolve ML-related defects vs. non-ML defects should not increase, and to ensure that goal, both developers and testers need a modern testing plan and strategy.

Another key here would be to properly divide and segment the two types of tests and validations within a single software iteration — test scoping that covers both the platforms, the functional and non functional tests, as well as the AI-specific cases. This should be the new normal for DevOps teams.

I will wrap up this chapter with a very modern example of AI-based use cases taken from Fintech Circle, that many of us users already enjoy today. Implementing a test-ing plan that takes the above-listed categories and applies them to these use cases is a great recommended practice if you're starting your journey into the AI world.

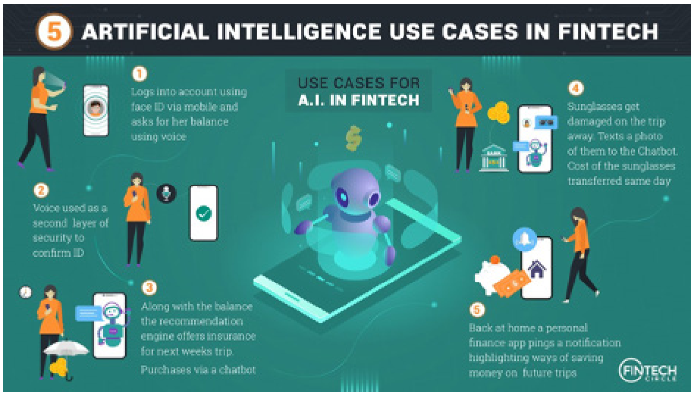

5 ARTIFICIAL INTELLIGENCE USE CASES IN FINTECH
USE CASES FOR A.I. IN FINTECH
1 Logs into account using face ID via mobile and asks for her balance using voice
2 Voice used as a second layer of security to confirm ID
3 Along with the balance the recommendation engine offers insurance for next weeks trip. Purchases via a chatbot
4 Sunglasses get damaged on the trip away. Texts a photo of them to the Chatbot. Cost of the sunglasses transferred same day
5 Back at home a personal finance app pings a notification highlighting ways of saving money on future trips
FINTECH CIRCLE

Fig. 68: AI Use Cases in Fintech (Source: Fintech[103])

103 AI Use Cases in FinTech https://fintechcircle.com/insights/fintech-use-cases-for-ai/

Chapter 11: AI Data Usage — Analyzing the User Experience

AI Appstore

Jennifer Bonine, CEO, AI Appstore

***Jennifer Bonine** is the CEO of AI Appstore, Inc., and was the first female Artificial Intelligence ("AI") platform tech CEO. AI Appstore specializes in custom subscription technology bundles, leveraging an intelligent platform using a personalized "virtual research assistant" to enhance corporate growth. The company exceeds expectations of integration, testing, delivery, and management with a groundbreaking business model that is fully engaged in the sustainable development goals ("SDGs") cultivated by the United Nations. Respected as a gifted speaker, entrepreneur, and philanthropist, Jennifer Bonine addresses the AI industry nationally and internationally, most recently at the World Economic Forum in Davos and for CNNMoney Switzerland. She has held executive level positions leading teams for Oracle and Target and is a founding board member of the United States bid for a Minnesota World Expo 2027. Jennifer is a 2020 Minneapolis St. Paul Business Journal Women in Business honoree, a founding sponsor and member of IVOW AI's Women in History Data Ideation Challenge, and an executive board member of Chad Greenway's Lead the Way foundation. She is a member of Million Dollar Women, member and mentor for TeamWomen, and a council member of DreamTank, an organization designed to champion young entrepreneurs. Recently named one of the Top 30 Leaders to Watch in 2020 by Silicon Review, Jennifer Bonine was featured at the UN's AI for Good summit. Jennifer is also developing a series of books to educate children about the power of AI and machine learning.*

RETHINKING THE CONCEPT OF QUALITY

In a digital landscape, accurate QA testing of functional and non functional device and browser configurations is a significant challenge. Notwithstanding testing and implementing business requirements, app developers must guarantee a seamless user experience across platforms. Apps and websites rely on appealing graphic elements and animations, attractive color schemes, and text that translates accurately in different languages. Ultimately, the goal of developers, testers, product managers, and the various members of the Software Development Lifecycle (SDLC) is to guarantee a high-quality experience for their users.

AI Appstore aims to utilize AI/ML to create "Virtual Research Assistants" designed to relieve human fatigue and empower members of the SDLC to be more productive, with less human error, and reach their goal of creating a superior application. A Virtual Research Assistant bot can take over monotonous tasks so its human counterpart can focus on the strategic work that involves critical thinking and solutioning. AI Appstore has gone one step further to define the differences between humanized and replacement AI/ML with the understanding that AI is an impactful, useful, and necessary part of 21st century existence that must adhere to an inclusive, human-centric AI platform to ensure human interaction is heightened and not diminished by artificial intelligence.

WHAT'S THE REAL PAIN?

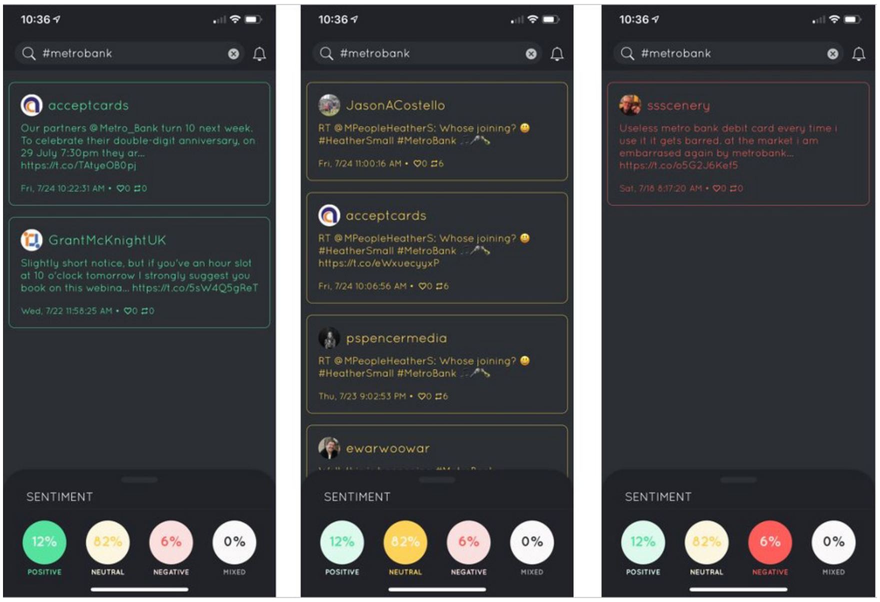
Mobile app developers and any other digital app vendor practicing Agile and DevOps must deal with bi-weekly releases either to the App Store, Google Play, or the web. With such a cadence of releases, making sense of the end user experience and understanding whether value is actually being added or if there are clear regressions is key for business success.

While there are many metrics and methods to track real user experiences, such as RUM (real user monitoring), there is not much time between these releases to act upon all the feedback. This is where AI/ML and other sentiment analysis solutions can bridge the time gap and either report in advance on such issues or sometimes even prevent them proactively.

Consider the real life example of an iOS social media sentiment application that is based on hashtags (#) which can automatically filter all positive, neutral, and negative feedback.[105] It provides a great set of continuous improvement opportunities, A/B feature deployments, and many more benefits.

In the next figure, you can see how a specific brand using such an app (in this case, Metro Bank, UK), provides various sentiments picked out from Twitter via the brand's hashtag. This can help brands make informed decisions.

105 Sentiment iOS App, Apple App Store https://apps.apple.com/us/app/sentiment/id1459910771



acceptcards
Our partners @Metro_Bank turn 10 next week. To celebrate their double-digit anniversary, on 29 July 7:30pm they ar... https://t.co/TAtyeOB0pj
Fri, 7/24 10:22:31 AM • 0 0

GrantMcKnightUK
Slightly short notice, but if you've an hour slot at 10 o'clock tomorrow I strongly suggest you book on this webina... https://t.co/5sW4Q5gReT
Wed, 7/22 11:58:25 AM • 0 0

JasonACostello
RT @MPeopleHeatherS: Whose joining? 😃 #HeatherSmall #MetroBank
Fri, 7/24 11:00:16 AM • 0 6

acceptcards
RT @MPeopleHeatherS: Whose joining? 😃 #HeatherSmall #MetroBank
https://t.co/eWxuecyyxP
Fri, 7/24 10:06:56 AM • 0 6

pspencermedia
RT @MPeopleHeatherS: Whose joining? 😃 #HeatherSmall #MetroBank
Thu, 7/23 9:02:53 PM • 0 6

ewarwoowar

ssscenery
Useless metro bank debit card every time i use it it gets barred. at the market i am embarrased again by metrobank... https://t.co/o5G2J6Kef5
Sat, 7/18 8:17:20 AM • 0 0

Fig. 85: Sentiment Analysis for an App Through Social Media (Source: Sentiment iOS App)

While this is only an example, later on in this chapter I'll expand on more values that AI/ML can add through App Store scanning, app maps, and more.

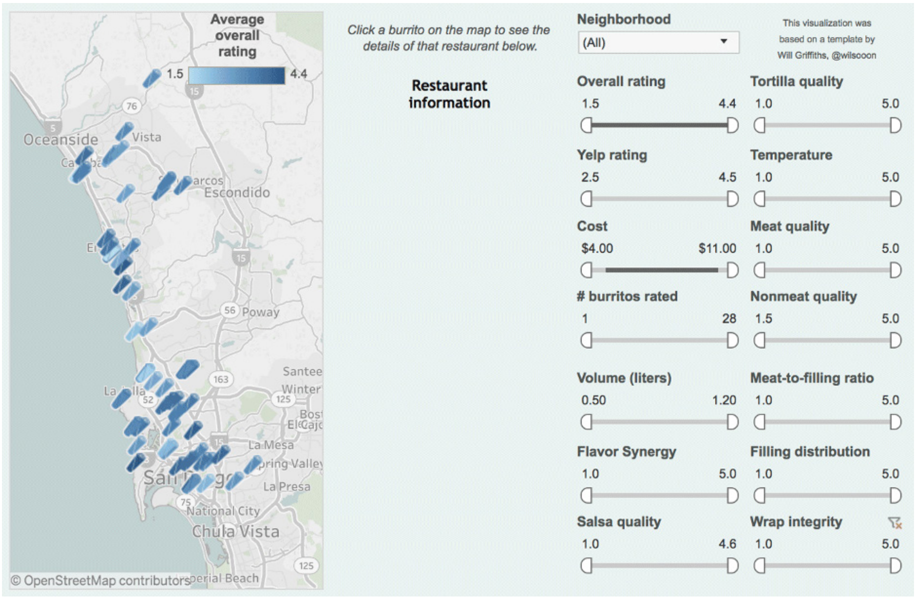
SOLUTION OVERVIEW EXAMPLE

The AI revolution is in its infancy and, as a result, the AI market is difficult to navigate. Enterprises building desktop, mobile, and web applications are seeking to integrate AI solutions to maximize value and support the decision-making process. AI Appstore designs data strategies for enterprises, providing custom AI subscription bundles that will resolve challenges in the software development lifecycle.

There are an overwhelming number of AI tools available on the market and each tool is designed to solve a litany of complicated software development and application challenges. There is one consistency among all of them: they are narrow in scope and created to address and solve a singular problem. In doing so, massive amounts of data are gathered and distributed in order to achieve a perfectly executed solution.

If a button is detected as being changed, moved, truncated, or otherwise incorrect, the algorithm will immediately report the error. The application is programmed to teach itself, and the neural network is built to continuously learn and grow in confidence with each examination sequence. Whereas human analysis is limited in its capacity to scrutinize and compare data sets, an AI agent or "bot" has infinite capabilities. And, while a human judge *might* test thousands of scenarios over the course of one year, an AI bot *will* accurately execute millions over millions of cases in one week.

The notion is that the bots carry out the "rote" tasks and are taught to run consistently and create a baseline. Pristine testing can only be accomplished when a colossal amount of data has been devoured by an AI agent. AI Appstore is interested in the solutioning possibilities that can be established and re-established by cross pollinating those datasets and creating bundles of information that can be processed and catalogued to support an exceptional user experience.


Average overall rating
1.5
4.4
Oceanside
Vista
Escondido
Poway
Santee
La Jolla
La Mesa
Spring Valley
San Diego
La Presa
National City
Chula Vista
Imperial Beach
© OpenStreetMap contributors
Click a burrito on the map to see the details of that restaurant below.
Restaurant information
This visualization was based on a template by Will Griffiths, @wilsooon

Filter	Min	Max
Neighborhood	(All)	
Overall rating	1.5	4.4
Yelp rating	2.5	4.5
Cost	$4.00	$11.00
# burritos rated	1	28
Volume (liters)	0.50	1.20
Flavor Synergy	1.0	5.0
Salsa quality	1.0	4.6
Tortilla quality	1.0	5.0
Temperature	1.0	5.0
Meat quality	1.0	5.0
Nonmeat quality	1.5	5.0
Meat-to-filling ratio	1.0	5.0
Filling distribution	1.0	5.0
Wrap integrity	1.0	5.0

Fig. 86: 10-Dimensional System for Rating the Burritos in San Diego (Source: Lionbridge.ai)

These new and never-before-utilized datasets are paving the way for companies to move even more efficiently than they might have using a singular AI tool, allowing for faster and more informed decisions across the software development lifecycle. For example, if a company is creating a user-friendly app or website and has the capability of bundling design implementation and usability scoring together with user sentiment, greater understanding is developed to direct whether or not the intention of the site or app is immediately accessible and impactful.

AI Appstore aggregates and parcels existing AI tools to create a custom platform for enterprise app teams, including developers, testers, and product managers. Humans are then called upon to analyze and influence the strategic aspects of the data that a bot does not have the ability to identify, ensuring accessibility, diversity, lack of bias, and other optionality is incorporated.

Where the AI bots operate in an environment that is black and white, humans fill in the grey areas and creatively build out app and website functionality. AI Appstore's innovative new bundles interact in such a way that companies reach consumers with speed and accuracy never before possible.

The solution comes when an AI bot — with an integrated awareness of application infrastructure, visual correctness, and strategic flow — creates an application map. This "app map" moves from one button to the next, mimicking user scenarios as it travels through the application.

The challenge comes when the companies that produce mobile devices, browsers, and apps pioneer innovative features and functionalities that might trigger an interruption in the user experience. AI Appstore collaborates with companies to apply neural networks and AI agents across software platforms, sourcing issues and enhancement opportunities missed by humans and building a solid AI-driven process that will determine and then overcome common challenges across the SDLC.

Enterprise app teams engage as follows:

- Technology team leaders outline needs and pain points.
- ML is deployed to identify bundles of existing AI-tools to meet company needs.

- Technology team leaders approve recommended bundles.
- A custom AI bundle is managed through AI Appstore's single sign on (SSO) platform.

An AI/ML-based approach to navigation intelligently sets forth a squadron of AI tools programmed and managed to streamline identified and unidentified challenges in application design.

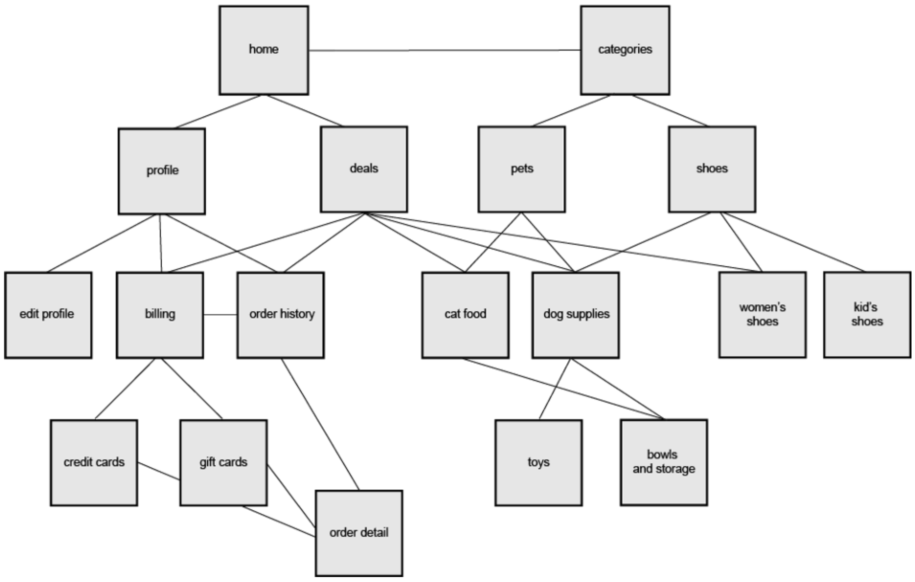


Fig. 87: An "Application Map" AI Bots Use to Understand the Anatomy of an Application (Source: AI Appstore)

QUANTIFYING THE USER EXPERIENCE

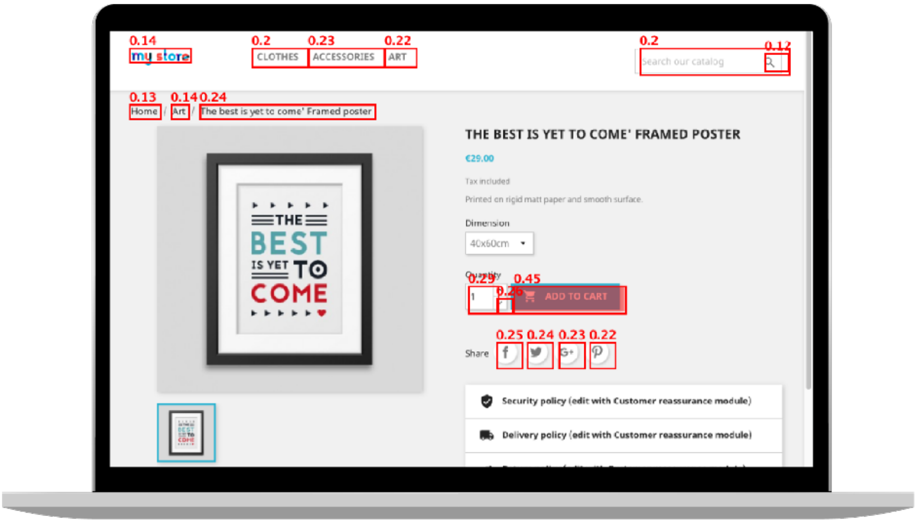
There are companies in the market leveraging AI/ML bots to scan websites and determine the probability that a user will interact with each UI element on the page. These companies use algorithms that train bots to crawl websites and analyze interfaces including buttons, cascading style sheets, HTML, favicons, and other pain points, which then score accessibility, track efficiency, and determine predictability.

These AI bots perform the "black and white" rote tasks to identify a delay or deficiency in quality that might create an issue with the customer. The design team can then "work in the grey" area with empathy and creativity to address the consumer, visualizing what the data doesn't explain and drawing on the elements of usability that are intangible and require human interaction and thought. AI agents identify the broken links and abnormal content that frustrate the user, giving the designer unfettered ability to personalize the graphics and subject matter necessary for an accessible and responsive user experience.

Using AI Appstore's Voice-of-Customer (VOC) application, users can select an app from either the iOS App Store or Google Play store, then filter the reviews by date range, sentiment score, ratings, or keyword phrase. For a minimum viable product, AI Appstore plans to layer in sources of customer feedback as well as apply additional machine learning techniques for optimum accuracy and analysis. The goal of VOC is to leverage customer feedback data and machine learning to help developers, testers, UX/UI designers, and product managers identify actionable opportunities for improvement with regard to bugs, specialized feature requests, and other user experience design-related issues with a company's software application.

AI Appstore recognizes that there is a gold mine of data available to streamline the decision-making process. But we also understand it is a mind-numbing task to structure millions of unstructured datasets. ML-based text classification models can do the job. An AI categorization model can group customer feedback data (e.g. reviews) which then determine whether or not a poor rating is a UX issue or a customer service issue. A second model identifies the actual sentiment of a consumer review and whether it is positive, mixed, negative, or neutral. Yet another model has the ability to amass multiple reviews and provide a three-sentence overview.

When an app or website developer is building a platform, the process can take months, and the creative team is so involved in building the project that overall awareness of user experience becomes an afterthought. Natural Language Processing (NLP) techniques offer a perspective that will open up a huge opportunity for data insights and analysis. The ability to structure data with AI goes beyond customer feedback analysis. AI can be integrated into a DevOps pipeline to automate any kind of analysis. The technology can be used for bug and metric tracking to quickly develop and ship code.



Printed on rigid matt paper and smooth surface.

Security policy (edit with Customer reassurance module)

Delivery policy (edit with Customer reassurance module)

Fig. 88: Sample "Usability Scoring" Analysis for a Website (Source: ReTest)

ENHANCED USER EXPERIENCE WITH ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING — STRATEGIC APPROACH

As stated at the beginning of this chapter, most AI/ML implementations involve a carefully tiered, humanized, and replacement approach. There is a need for solutions that are designed and architected as humanized AI. AI-powered virtual research assistants can be taught to perform tasks that are challenging and time consuming for humans, evaluating millions of data points, recognizing patterns and correlations with remarkable accuracy and speed, and processing information in a way that is relatable to humans navigating the decision-making process.

AI Appstore understands the value that comes from accessing surplus datasets and deploying AI agents to parse through massive amounts of statistics in order to aggregate, bundle, and manage the most efficient AI/ML systems imaginable for companies and corporations to better serve customers and consumers.

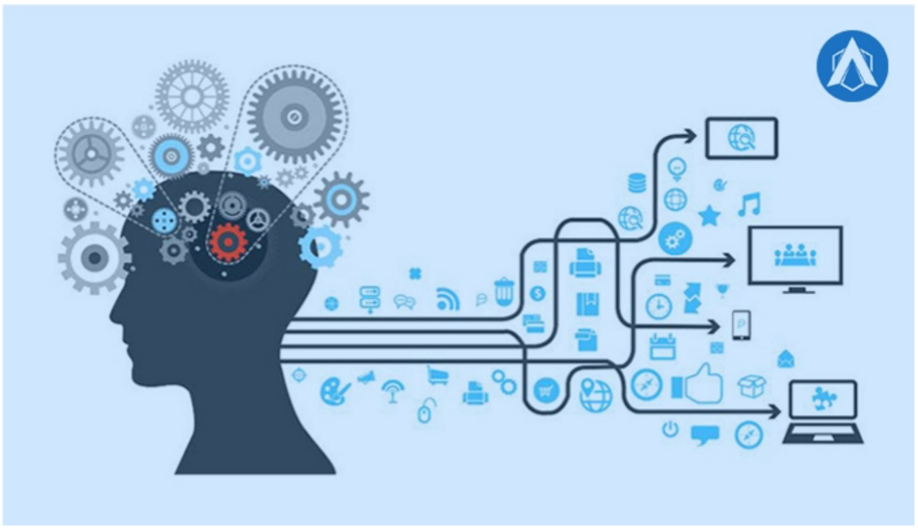


Fig. 89: AI Improving UX Illustration (Source: Appsquadz)

Software application quality using AI Appstore's unique dataset bundles should be applied at the inception of any project to guarantee that brand perception, reputation, and loyalty is accurately represented across all device platforms at the highest level. Application quality is not only about test cases and requirements — it's bigger than that. The quality of experience that a user has with an application is the most important enterprise metric, and AI/ML helps organizations realize a more holistic definition of quality.

AI Appstore is pioneering a method of identifying, aggregating, and deploying AI/ML bundles to reimagine the way companies design, develop, and maintain their applications. By utilizing Virtual Research Assistants and AI/ML to structure vast amounts of unstructured data, AI Appstore curates a custom solution that enterprises can meticulously manage through a single, integrated platform.

Chapter 13: Introduction to Robotic Process Automation (RPA)

Ahmed Datoo, Co-Founder/COO, Mesmer

Ahmed Datoo *is the Co-Founder and COO of Mesmer. His experience in the technology industry spans software engineering, product management, marketing, and strategy. Prior to Mesmer, Ahmed was on the founding team of Zenprise and was the SVP of Product and Marketing. At Zenprise he grew revenues from $0 to $250 million, built an innovative product recognized by Gartner and Forrester, and coined the term and created the MDM (mobile device management) category. Prior to Zenprise he held engineering management and product management positions at Loudcloud/EDS. He started his career as a strategy consultant at Accenture. He's appeared in articles in the WSJ, NYTimes, Economist, Network World, and USA Today, and has spoken at events at Stanford University, Interop, Gartner, and VentureBeat. Ahmed holds an MBA, MA, and BA from Stanford University.*

WHAT IS ROBOTIC PROCESS AUTOMATION?

Robotic Process Automation (RPA) refers to automating routine and repetitive processes. The concept involves use of a software robot, or bot for short, to offload manual and often tedious processes from employees.

Initial use cases involved automating back office processes. Think of a customer who completes an insurance claim online that requires an employee to manually input information into a separate internal claims processing system. A bot can take the information from one system, extract relevant metadata, and then connect to the UI of the claims system and enter the captured data automatically. Most importantly, this work does not require deep technical integration between the systems.

Early iterations of RPA involved screen scraping technologies. This allowed the bots to integrate into systems with ill-defined or nonexistent APIs. Bots would use a variety of technologies, including Optical Character Recognition (OCR) to identify text on a screen, interactable objects, images, etc. RPA vendors would then provide record and replay interfaces to allow employees to capture critical flows within a user interface.

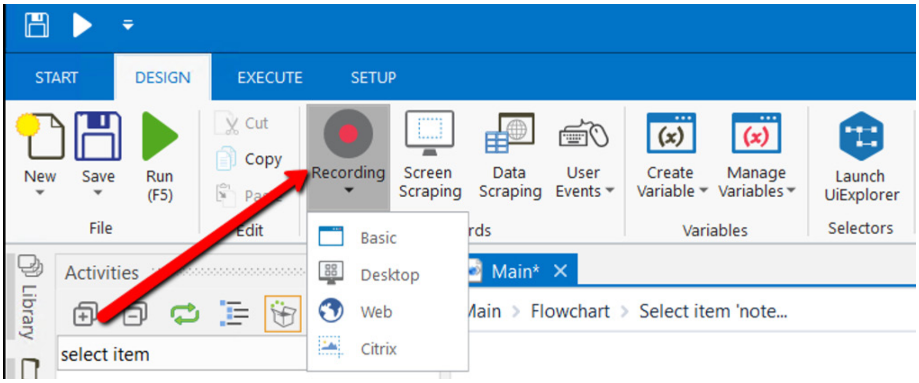
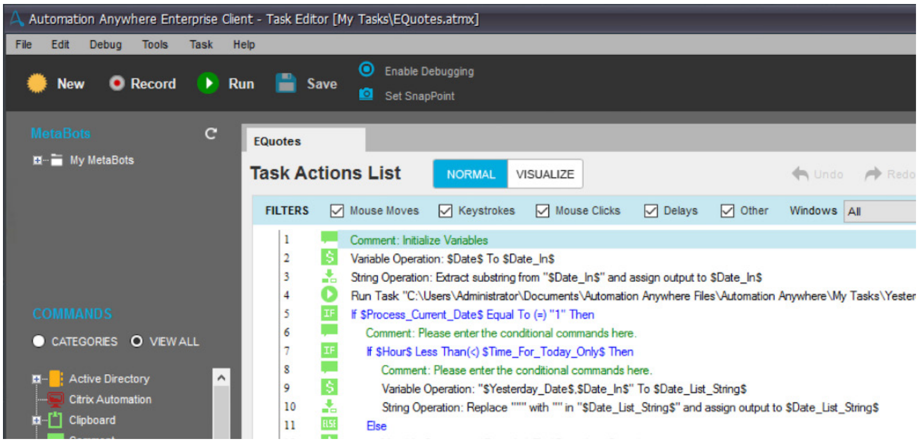


Fig. 94: Screen Scraping Initiation Process (Source: UIPath)

The output of these interfaces is a series of rules with a syntax specific to the vendor. Think of these rules and syntax as a type of program that runs on a schedule to automate the specified process.



```
Comment: Initialize Variables
Variable Operation: $Date$ To $Date_In$
String Operation: Extract substring from "$Date_In$" and assign output to $Date_In$
Run Task "C:\Users\Administrator\Documents\Automation Anywhere Files\Automation Anywhere\My Tasks\Yester
If $Process_Current_Date$ Equal To (=) "1" Then
  Comment: Please enter the conditional commands here.
  If $Hour$ Less Than(<) $Time_For_Today_Only$ Then
    Comment: Please enter the conditional commands here.
    Variable Operation: "$Yesterday_Date$,$Date_In$" To $Date_List_String$
    String Operation: Replace """" with "" in "$Date_List_String$" and assign output to $Date_List_String$
  Else
```

Fig. 95: Rules Generation Example (Source: Automation Anywhere)

For applications with seldom-changing static interfaces, this solution proved more than adequate.

Unfortunately, applications with dynamic interfaces resulted in RPA rules breaking. Consequently, next generation RPA solutions incorporated machine learning to more readily adjust to changes in UI.

As an example, use of computer vision in lieu of OCR allows for more accurate object detection. For context, OCR matches pixels in an image to the pixels of another image. Should an interface change the color, text, or even the look and feel of a button, OCR would fail because pixels no longer match — even though the image is still a button performing the same exact function. OCR failure means scripts break, thereby requiring human intervention to update objects referenced in the script.

Computer vision (CV) is much more resilient to changes. CV relies on deep neural networks to learn about an object. Google in 2012 most famously trained a neural network to identify cats by simply feeding the network frames generated from 10 million videos.[108] The neural net learned attributes of cats by studying what was common across these videos. Even though the cats were different colors, different sizes, in different locations/poses, the neural net accurately detected cats 74.8% of the time.

What's the implication of using CV for object identification in an application? The bot is immune to most common changes that would break OCR. As an example, changes in color of the image have little impact on object detection. Changing a hamburger menu icon from white to green would cause OCR to fail but would not impact CV. Just like cats can be different colors, so too can hamburger menus.

Even changing the icon image from the traditional three stacked horizontal lines to three dot images would not cause CV to fail. The neural network learns that hamburger menus can take different shapes, colors, etc.

RPA IN SOFTWARE TESTING

RPA has the potential to automate the entire testing process, thereby improving engineering productivity. Today, most automation solutions narrowly focus on test creation and execution. But the testing process is much broader than just these functions. A huge amount of time is still spent manually setting up and tearing down test environments. Connecting into CI/CD systems to retrieve latest builds can also be time consuming. Even the process of writing bug reports is manual and tedious, requiring screenshots, videos, and step-by-step instructions on how to reproduce the problem.

108 Google Neural Network Research, 2012
https://static.googleusercontent.com/media/research.google.com/en//archive/unsupervised_icml2012.pdf

Using RPA, bots can set up test environments, auto generate and execute tests, and auto-write bug reports. In essence, the entire testing process can now be automated. Moreover, use of Machine Learning (ML) technologies can eliminate the need for scripting know-how and expertise. Modern RPA solutions generate ML models and not scripts to automate processes.

The implications to the test automation engineer are profound. Automation engineers will increasingly become data scientists, labeling objects in the UI (e.g., this is a login button, this is a checkout button, this is an input field, etc.), and feeding these objects into ML models. The level of technical expertise required will vary depending on use of open source technologies or commercially available solutions. Open source technologies require the creation and maintenance of models specific to your application and will often require deep math expertise to optimize models.

Commercial solutions auto generate models and require significantly less technical expertise to build and maintain. In fact, manual testers can use commercial RPA solutions to automate entire testing processes simply based on being a subject matter expert on their app and their internal processes.

BENEFIT OF RPA IN SOFTWARE TESTING

Many test engineers today are overworked, causing the balance between work and life to increasingly blur. Software teams are particularly short-staffed and under pressure to release faster and better products than more Agile competitors. Managers can try growing the team, but hiring continues to be a challenge, making it feel like there's no relief in sight for overworked workers.

Enter Robotic Process Automation. RPA allows team members to offload tasks that often cause late nights at the office. When you pair an individual with a bot, more work gets done faster. Managers may not be able to hire five more team members, but they can certainly pair someone on their team with a bot. And together they can do the work of five people.

EXAMPLE:
USING RPA TO AUTOMATE CUSTOMER EXPERIENCE (CX) TESTING

Most developers automate unit testing and API testing, but manually perform customer experience testing (CXT). CXT includes UI, end-to-end, functional, integration, and visual testing — the stuff that's fragmented, tool-intensive, and notoriously hard to automate.

Many organizations outsource this testing to third parties who manually validate their application functionality. Or they have a team of QA professionals in-house performing the same function.

But the work is tedious and time-consuming, not to mention slow because it's still manual.

To make sure apps work in the real world, software teams spend time manually tapping buttons in their app on the multiple device types and OS versions used by customers.

But before they can even start manually testing their apps, they have had to spend time designing tests and building the infrastructure to run them. And only once that's done can they write code to fix bugs identified by tests.

This entire process is manual, and ripe for automation via RPA bots.

RPA BOTS IN CX TESTING

Testing teams can use special purpose bots to automate specific portions of their testing process. This next section will explore three types of bots in particular: one that auto builds and tears down testing infrastructure (infrastructure bot), one that auto generates and executes tests (customer bot), and one that auto documents bugs (bug bot).

Infrastructure Bots

These bots automate the tasks involved in setting up testing infrastructure. Let's say the testing team needs the latest iPhone with the current OS to test a new app build. The infrastructure bot connects to Jenkins (or the CI/CD system of record), retrieves the latest build, proceeds to spin up a device in a public or private cloud, and installs and configures the app to point to the appropriate staging/dev/prod environment. There is no need to create custom scripts to do this. And there's no need to wait around while someone in DevOps or IT spins up an environment.

Customer Bots

Customer bots mimic the behavior of end users. These bots, much like end users, will interact with the user interface, tapping on buttons, swiping through pages, inputting data into fields, and performing gestures to unlock new features. The good news is that commercially available tools require no programming or scripts required on your end to make these bots work.

Tools from companies like Mesmer use deep learning and machine learning models to identify and interact with objects in an application (Mesmer RPA https://mesmerhq.com). These bots use a combination of computer vision, natural language processing, and path planning models to accurately detect and interact with UI elements on a screen.

It's the equivalent of a self-driving car — customer bots automatically explore your app with no scripting or coding required.

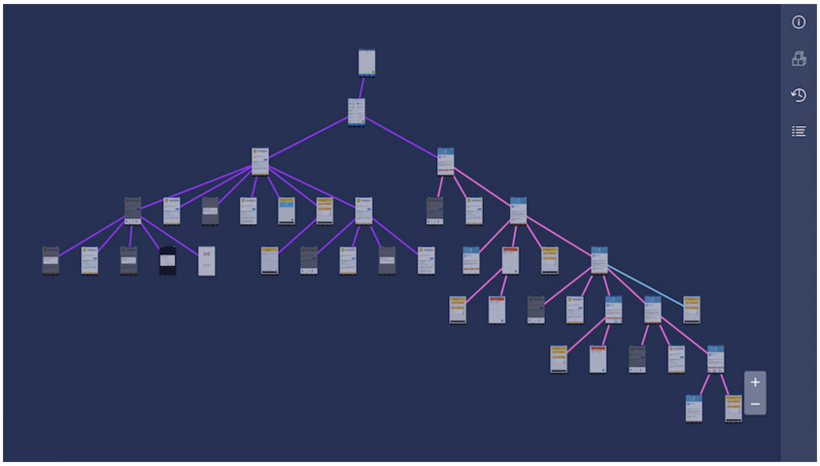


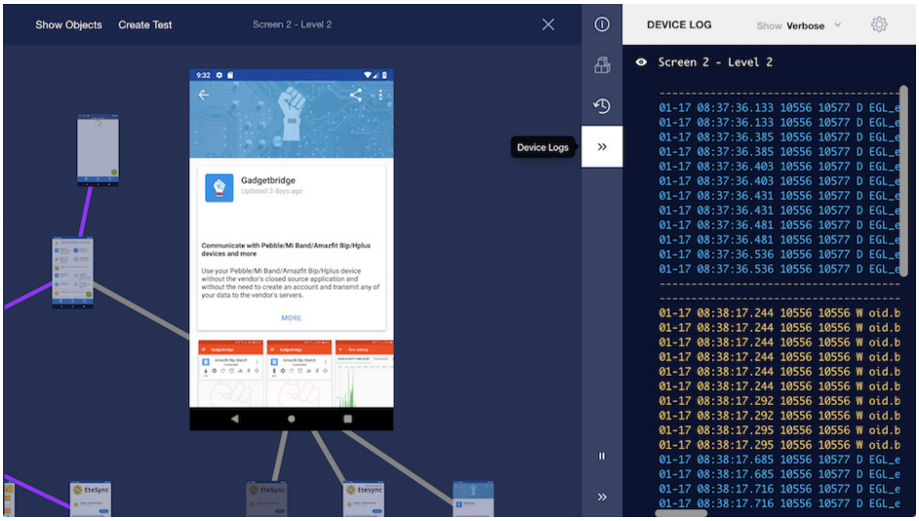
Fig. 96: Output of the Customer Bot Exploring an aApp (Source: Mesmer)

Bug Bots

Bug bots' sole purpose is to look for and report customer experience issues in your pre-released builds. The bug bots run alongside customer bots, looking for defects in the customer journeys explored.

As an example, the customer bot clicks a button that causes your app to crash. The bug bot immediately analyzes the log file looking for any crash exceptions or network issues that could cause the app to experience the fatal exception.

Bug bots look for much more than crashes in your log file. They can surface warnings, developer-specific assertions written to logs, or other important exceptions testers would otherwise need to look for in logs. Testing teams can even map specific log errors to specific pages traversed by the customer bot. No more needing to offload device log files to log aggregators for reporting and manually mapping back to customer journeys.



Communicate with Pebble/Mi Band/Amazfit Bip/Hplus devices and more

Use your Pebble/Mi Band/Amazfit Bip/Hplus device without the vendor's closed source application and without the need to create an account and transmit any of your data to the vendor's servers.

```
Screen 2 - Level 2

------------------------------------------
01-17 08:37:36.133 10556 10577 D EGL_e
01-17 08:37:36.133 10556 10577 D EGL_e
01-17 08:37:36.385 10556 10577 D EGL_e
01-17 08:37:36.385 10556 10577 D EGL_e
01-17 08:37:36.403 10556 10577 D EGL_e
01-17 08:37:36.403 10556 10577 D EGL_e
01-17 08:37:36.431 10556 10577 D EGL_e
01-17 08:37:36.431 10556 10577 D EGL_e
01-17 08:37:36.481 10556 10577 D EGL_e
01-17 08:37:36.481 10556 10577 D EGL_e
01-17 08:37:36.536 10556 10577 D EGL_e
01-17 08:37:36.536 10556 10577 D EGL_e
------------------------------------------
------------------------------------------
01-17 08:38:17.244 10556 10556 W oid.b
01-17 08:38:17.244 10556 10556 W oid.b
01-17 08:38:17.244 10556 10556 W oid.b
01-17 08:38:17.244 10556 10556 W oid.b
01-17 08:38:17.244 10556 10556 W oid.b
01-17 08:38:17.244 10556 10556 W oid.b
01-17 08:38:17.292 10556 10556 W oid.b
01-17 08:38:17.292 10556 10556 W oid.b
01-17 08:38:17.295 10556 10556 W oid.b
01-17 08:38:17.295 10556 10556 W oid.b
01-17 08:38:17.685 10556 10577 D EGL_e
01-17 08:38:17.685 10556 10577 D EGL_e
01-17 08:38:17.716 10556 10577 D EGL_e
01-17 08:38:17.716 10556 10577 D EGL_e
```

Fig. 97: Mesmer Solution Snapshot (Source: Mesmer)

Bug bots can also analyze screenshots, looking for design-related bugs. Incorrect icons, logos, fonts, spacing, and CSS style-related inconsistencies between builds are reported by the bug bots. The bots use computer vision to recognize all the objects on a particular page, and then compare these objects to the last known good build to determine whether anything has materially changed. Any such changes are reported back to the development team for remediation.

Bug bots can additionally identify functional errors in your application. The bot can look for specific text expected in designated areas of a page, look for certain strings, verify that images appear in the appropriate places, and even ensure that numeric values on the screen are correct and that date formats render appropriately.

Chapter 24: How Does AIOps Benefit DevOps Pipeline and Software Quality?

While the majority of this book covers topics focused on the left side of the DevOps pipeline — like functional, unit, API, and other processes related to automation that benefit AI and ML — this chapter is all about the right side of the DevOps process.

As this book is being developed, there is already an advanced approach by IT leaders toward a more efficient operation that aims to address inefficiencies and bottlenecks standing in the way of automated production and operation environment management.

Things like noise reduction for false production alerts through clustering and pattern-matching algorithms, identifying root causes of incidents in production and pre-production through smart correlation of the issues with user journeys, detection of abnormal conditions and product behavior, business impact analysis traced to issues, defect trends that may result in production outages, and generic triaging of problems are among the key objectives behind ops managers looking into AI and ML as possible solutions.

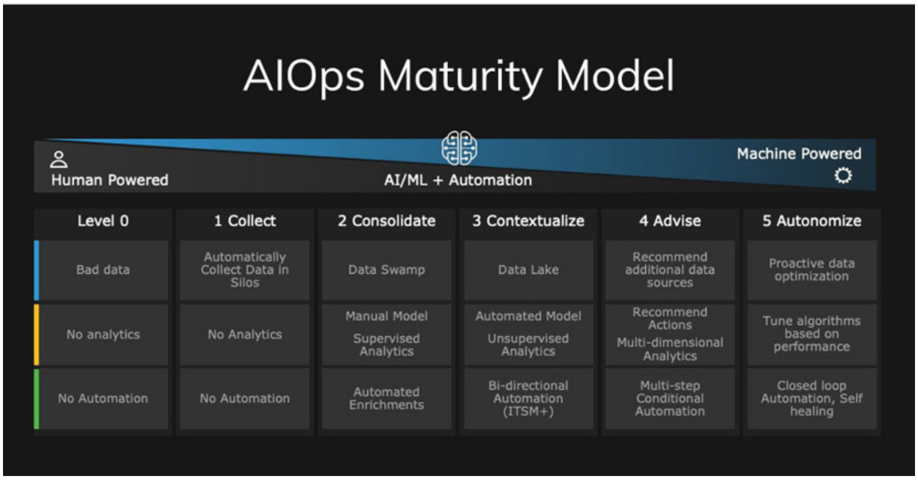
When armed with advanced abilities that make an AIOps portfolio valuable, IT managers can make an impact on the entire software delivery cycle, production quality, and more. Abilities as mentioned above can translate specifically into anomalies detection, automated pattern discovery and predictions, software topological analysis for greater accuracy in issues classification as well as impacted dependent areas and resolution, and statistical data analysis that can result in clustering of issues, correlations of issues, and better decision making within the entire product life cycle.

All of the above is an introduction to and breakdown of the modern term called AIOps. IBM defines this term accordingly: "AIOps, (for artificial intelligence for IT operations) is the application of artificial intelligence (AI) to enhance IT operations. Specifically, AIOps uses big data, analytics, and machine learning capabilities to do the following:

- Collect and aggregate the huge and ever-increasing volumes of operations data generated by multiple IT infrastructure components, applications, and performance-monitoring tools.
- Intelligently sift 'signals' out of the 'noise' to identify significant events and patterns related to system performance and availability issues.
- Diagnose root causes and report them to IT for rapid response and remediation — or, in some cases, automatically resolve these issues without human intervention."[203]

An AIOps complete solution does not only cover smart **APM** (application performance monitoring) solutions, but also leverages **ITIM** (IT Infrastructure Monitoring) and **ITSM** (IT Service Monitoring) to build a comprehensive layer of production and operational insights analysis that can run on big data and against advanced modern software architecture (micro-services, cloud, etc.).

With the power of AI-based operations, teams can better focus on determining the service heath of their applications, and gain control and visibility over their production data. With that, DevOps teams can expedite their MTTR (mean time to resolution) using automated incident management in real time and quickly.



AIOps Maturity Model

Human Powered — AI/ML + Automation — Machine Powered

Level 0	1 Collect	2 Consolidate	3 Contextualize	4 Advise	5 Autonomize
Bad data	Automatically Collect Data in Silos	Data Swamp	Data Lake	Recommend additional data sources	Proactive data optimization
No analytics	No Analytics	Manual Model Supervised Analytics	Automated Model Unsupervised Analytics	Recommend Actions Multi-dimensional Analytics	Tune algorithms based on performance
No Automation	No Automation	Automated Enrichments	Bi-directional Automation (ITSM+)	Multi-step Conditional Automation	Closed loop Automation, Self healing

Fig. 191: AIOps Maturity Model (Source: ScienceLogic[204])

203 IBM Definition of AIOps https://www.ibm.com/cloud/learn/aiops
204 Science Logic Definition of AIOps and its Maturity Model https://sciencelogic.com/solutions/aiops

Looking into the above suggested maturity model for AIOps within a DevOps organization, we can learn what good looks like, and the necessary path and metrics needed to get there or position your own maturity in such a model.

The path starts from a fully human-oriented approach (level 0) where it is reactive to issues that are happening, trying to reduce noise and false negatives and only filter and cluster real issues, toward a more automated and smart approach. The path goes through collection and analysis of big production data in an automated fashion (**level 1**), through data consolidation and contextualization (**levels 2 and 3**) that fits into a smart model of IT service monitoring (ITSM) and an unsupervised model, and then to the higher maturity of automation of data (**levels 4 and 5**) through predictions, recommendations, and advisories to IT managers on how to proactively optimize their data usage and autonomous operations.

A different maturity model suggested by Gartner also looks at five levels toward autonomous IT Service Management. It goes from a state of manually and inefficiently detecting issues and trying to map them to patterns and clusters to reduce noise and increase MTTR.[205]

Such a model evolves through issues prevention using proactive alerting and determining what Gartner calls "**causality.**" The final stage in the proposed model looks at service management automation that can analyze tickets through chatbots, change risk analysis, and more.

All of the abovementioned benefits take the business from a state of being **reactive** to **proactive** to **predictive**.

What AIOps Consists Of and Landscape Examples

As mentioned earlier, to build an advanced AIOps model requires a combination of tools that can analyze big data and deliver a relevant and up-to-date analysis of issues. To succeed in this very complex task, an AIOps tool stack must be able to "run" on sets of data like the below (obviously there are more):

205 Market Guide for AIOps Platforms https://www.gartner.com/en/documents/3971186/market-guide-for-aiops-platforms

- Historical analysis of issues (e.g. service tickets).
- Current and older performance analysis.
- High-risk product anomalies that were detected.
- Outages and other service-related issues.
- Key user journey analysis.
- Pattern discovery and correlation.
- Market-specific KPIs and analytics (benchmarks).
- Product specific inputs.

The end goal should be continuous insight to IT managers/ops around their service quality, product quality, and other anomalies in the product that occur in real time and can prevent a high-severity outage in an autonomous manner.

To embed AIOps into a DevOps workflow requires an E2E process change. It starts with developers who can identify potential risks in their code and alert it to the DevOps engineer to “plug” it into the AI system as a potential and categorized risk, so when something similar occurs or is reported through a ticket, the time to address it will be either zero or as short as possible. The logging of an incident done by the DevOps engineer should then be translated into an automated test scenario that runs against a pre-production and of course production monitoring system (APM) so the loop is being closed throughout the entire SDLC continuously.

The AIOps landscape is evolving and will continue to evolve throughout the main categories that were mentioned above (APM, ITIM, ITSM). These players collectively are trying to help in the automation of detection and clustering of production issues and anomalies, predictability of issues and prevention of these issues, and for specific issues that do rich production — an automated RCA (root cause analysis) for faster MTTR.

If to mention a few players in the industry for each of these categories, I would start by looking into what the following vendors offer:

APM

- New Relic[206]
- Dynatrace[207]
- Cisco AppDynamics[208]

ITIM

- Logz.io[209] (also covered in Chapter 22)
- Elastic Search[210]
- Splunk[211]
- DataDog[212]

ITSM

- BMC Predictive IT Service Management Solution[213]
- ServiceNow[214]

Mapping or visualizing AIOps as a continuous process was very well defined and explained in an insightful post on Medium.[215]

206 NewRelic APM https://newrelic.com/
207 Dynatrace Monitoring https://www.dynatrace.com/
208 AppDynamics https://www.appdynamics.com/
209 Logz.io https://logz.io/
210 Elastic Search ITIM https://www.elastic.co/
211 Splunk Monitoring and ITIM solutions https://www.splunk.com/
212 DataDog Monitoring and Analysis https://www.datadoghq.com/
213 BMC Heilx ISTM https://www.bmc.com/it-solutions/bmc-helix-itsm.html
214 Service Now Solutions https://www.servicenow.com/
215 AIOps, The new member in the DevOps Family https://medium.com/faun/aiops-the-new-member-in-the-devops-family-d76bab14c98e

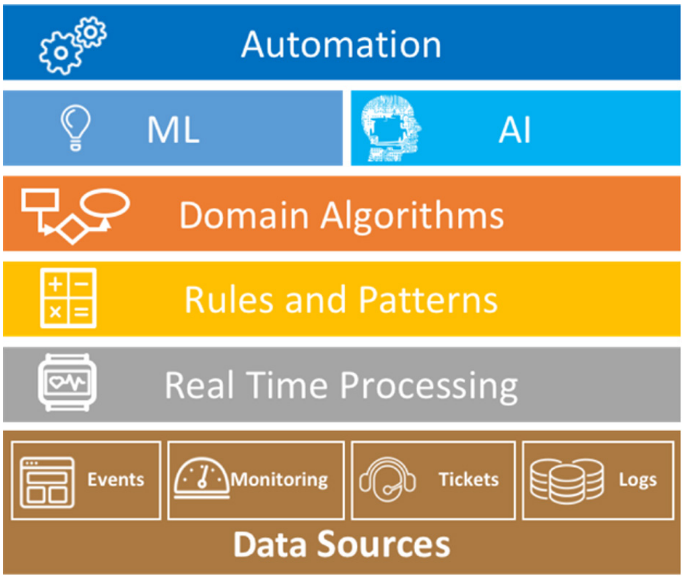


Fig. 192: AIOps Process Explained (Source: Medium)

The above architecture is being mapped into a complete DevOps workflow that again, was very well defined in the abovementioned Medium article.

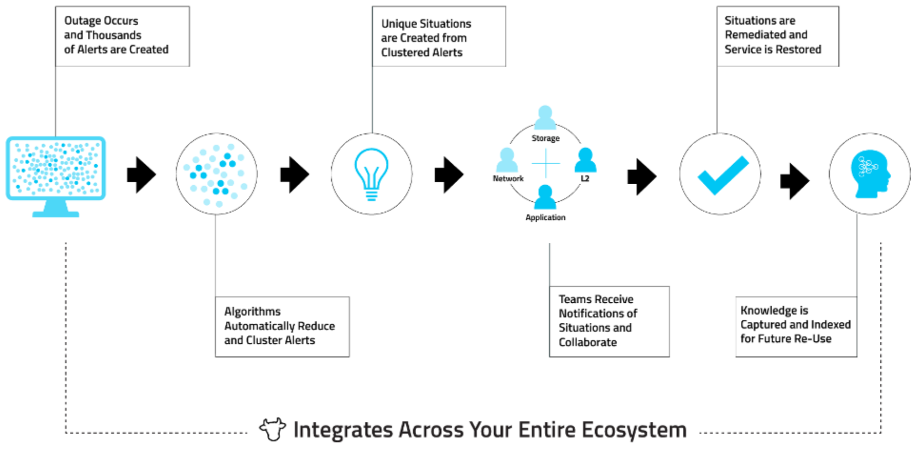

Outage Occurs and Thousands of Alerts are Created
Algorithms Automatically Reduce and Cluster Alerts
Unique Situations are Created from Clustered Alerts
Storage
Network
L2
Application
Teams Receive Notifications of Situations and Collaborate
Situations are Remediated and Service is Restored
Knowledge is Captured and Indexed for Future Re-Use
Integrates Across Your Entire Ecosystem

Fig. 193: AIOps Process Explained (Source: Medium)

BOTTOM LINE

To optimize a full DevOps pipeline takes a village. And that means entire teams play a role in it. It is often stated that maximum automation enables CI/CD, and this is correct. However, in the context of modern DevOps, with optimized software delivery, this also requires a smart assistant in the shape of AI and ML agents.

Such assistive technologies must also capture the software after its release and ensure that noise is eliminated. Real issues must be addressed quickly or prevented through predictive analysis, smart monitoring solutions, and autonomous software corrections across the various clusters that the algorithms defined.

The future of DevOps relies on E2E AI and ML from the early development phases, through code quality, defects resolution, deployment to production, and post-production issues remediation.

Chapter 26: What's Next for AI/ML Testing Tools?

This book covered in-depth the main use cases of AI and ML in testing as well as development activities, targeting specific challenges, pains, inefficiencies, and more.

Some of the topics that were covered in the book are already available today and can be used either through commercial or open source solutions. Some use cases are still being considered and being developed as solid AI/ML solutions to these problems. In addition, there are even more advanced topics in software development and test automation that are not even being worked on. The future will tell when, how, and in which use cases we will first see these solutions introduced.

Specifically, around test automation, as it relates to AI and ML, this book provides solid coverage of the abilities that exists today. However, even these abilities are not 100% fully leveraging smart algorithms.

In some of today's cases, AI and ML are helping through the self-healing of scripts by adjusting element locators upon changes to the product. In other cases, AI and ML are being used for automated visual testing and root-cause analysis classification. These are all stand-alone solutions for specific challenges. The future of test automation, however, holds an even greater functionality that many developers and vendors are starting to consider as this book is being authored.

Fully autonomous testing may seem like an impossible thing to accomplish. However, based on the following visual that shows different levels of test automation, level 5 is where the market aims to be.

Simply pointing a mobile or web application to an AI/ML autonomous testing solution will process and model the app, then generate a full set of screens, user flows, and key business cases. Based on these artifacts, it will generate proper testing scenarios. Maintenance of such scripts will not be a huge obstacle for these tools, since the creation of the tests will be based on an end-to-end journey through the app or website. Whenever there is a new element, screen, or business flow that disrupts the already-learned app, the machine learning solution will regenerate the proper testing scenarios.

The future of testing will not be as simple and easy as I describe above, since there will need to be process-related matching between the testing artifacts and the CI/CD workflows. In addition, defect management and test management will also need to be addressed by next-generation tools in order to be adopted by existing DevOps teams who have built solid working processes over the years.

In addition, specific verticals as well as app-specific abilities (mobile vs. web) will need to be very well addressed and supported by the next wave of AI/ML that aims to be completely autonomous.

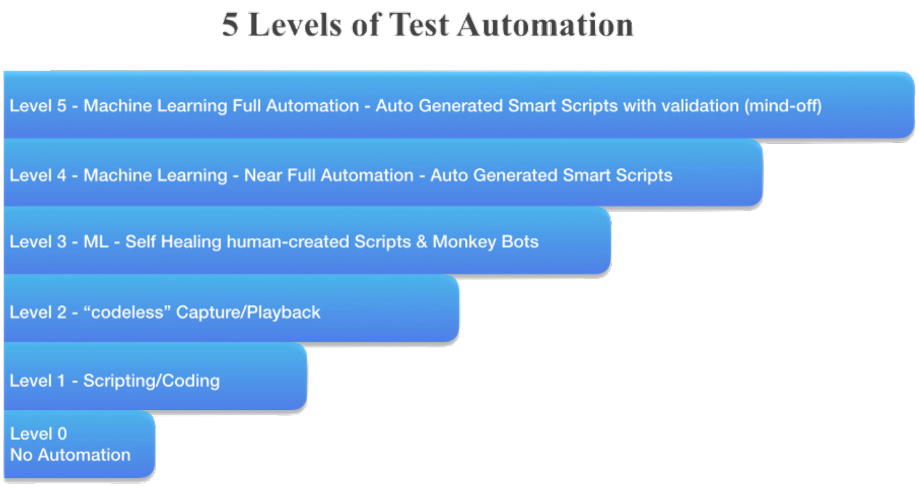

5 Levels of Test Automation
Level 5 - Machine Learning Full Automation - Auto Generated Smart Scripts with validation (mind-off)
Level 4 - Machine Learning - Near Full Automation - Auto Generated Smart Scripts
Level 3 - ML - Self Healing human-created Scripts & Monkey Bots
Level 2 - “codeless” Capture/Playback
Level 1 - Scripting/Coding
Level 0
No Automation

Fig. 204: Microsoft Azure DevOps, VSTest TIA (Source: Microsoft Documentation[224])

NEXT-GEN TEST AUTOMATION TOOLS: CHALLENGES AND ROADBLOCKS

Some of the expected challenges for the next wave of autonomous AI/ML tools are also relevant to existing AI/ML tools evolving in the market already.

The above challenges are not only the responsibility of the tool providers, but also a checklist for teams that will be evaluating such tools in the future. Teams should assess these tools to ensure they match their DevOps reality at the time of evaluation and for the future.

224 Appvance 5 Levels of Test Automation https://www.appvance.ai/5-levels-ai-testing-autonomy

- Autonomous test creation and maintenance for mobile and web.
- Test execution, orchestration, and scalability.
- Reporting and analysis of test results.
- DevOps process fit: integration with tools, complementing code-based solutions.
- Keeping up with evolving technologies.

Let's expand a bit on each of the above challenges.

Autonomous Test Creation and Maintenance for Mobile and Web

Autonomous testing tools must be platform agnostic and support advanced capabilities of the mobile and web platforms. This means that supporting gestures, sensors, events, alerts, and other platform specifics must be at the core of these tools. Test automation by nature is something that runs continuously in parallel across platforms, and it's triggered by code changes. Having hiccups in the testing flow will fail such tool adoptions.

All of the complexities mentioned above also relate to the maintenance and lifecycle of test cases. If the tools promise zero-touch creation and maintenance, they must stand behind the promise.

Test Execution, Orchestration, & Scalability

The creation phase is great, but in DevOps and Agile processes, execution at scale has equally the same importance. Execution speed and scalability contribute to fast feedback upon code changes, and to fast resolution of defects. The generated tests also need to run seamlessly across different platforms, including mobile and web. This by itself is a huge challenge, since these platforms are very much different across screens, form factors, OS versions, and more.

Consider running autonomous scripts on various Android, iOS, and desktop browsers that are different in many ways — Android custom vs. stock OS versions, screen sizes and resolution, supported capabilities per iOS/Android OS families, and more.

Reporting & Analysis of Test Results

At the end of each test execution, developers and test automation engineers must have a reliable, fully detailed set of test reports that can give them sufficient data to base their decision and analysis on. In addition, the executions as mentioned above are large in scale and continuous, and that means dealing with an ongoing large test report. Slicing and dicing abilities of test data as traditional test automation tools offer today (Perfecto, for example) must be also part of autonomous testing tools.

If teams have to spend hours analyzing test results that are generated by autonomous tools, the cost effectiveness of these tools will decrease in a significant way. The idea behind autonomous testing should be that since the tools "learn" all the business flows of the apps and create scripts accordingly, the tools can quickly spot anomalies and defects. It is nice in theory and must be proven in reality and across platforms. Generating too many false negatives will cause a loss of trust by both developers and testers — hence, this needs to be well designed and baked into the tools.

DevOps Process Fit

Autonomous tools must "play" nice with existing tools. Testing is done throughout the pipeline upon code changes, prior to build acceptance and version releases. Depending on the app, the vertical, and the practitioner, the DevOps process uses a wide range of tools — from performance and monitoring, through CI servers, deployment, environment virtualization, third-party tools, APIs, and many more. The expectation as well as challenge for autonomous testing is that their use will be seamless, and not cause a separate, whole new process.

Keeping Up With Evolving Technologies

Last, but definitely not least, is technology and innovation. Creating a set of working autonomous scenarios for product version X is nice and promising. However, both the product and its features evolve, and the market constantly shifts.

Any new feature that is introduced for desktop browsers or mobile platforms (OS, devices) must be supported quickly by autonomous tools, or else the work-around will be to revert back to automating via code-based tools. In today's reality, communities like Selenium and Appium are chasing each and every iOS and Android release and making many efforts to support test automation developers. The same approach should be taken by the autonomous testing vendors.

While I kept this summary focused on autonomous testing, and less on no-code development and non-functional activities like code reviews, security audits, and other software-related use cases, the market will clearly invest in parallel with the test creation domain, while also aiming to solve various development challenges.

BOTTOM LINE

The future of test automation is brighter than ever. Many practitioners, tool vendors, and analysts are placing their bets on AI and ML to finally boost test automation coverage, reliability, and success toward a true DevOps reality.

To deliver on this hype, pitfalls from previous years and the considerations listed in this book must be well thought out so that organizations will have a solid foundation to rely on for automating both development activities and testing.

It may take few iterations of autonomous tools to get closer to this vision. But when this happens, teams will achieve higher productivity than ever.

About the Author

Eran Kinsbruner is Chief Evangelist and Product Manager at Perfecto by Perforce. He is also the author of the 2016 Amazon bestseller, "The Digital Quality Handbook," and "Continuous Testing for DevOps Professionals," which was named one of the Best New Software Testing Books by BookAuthority. He is a development and testing professional with over 20 years of experience at companies such as Sun Microsystems, Neustar, Texas Instruments, General Electric, and more. He holds various industry certifications such as ISTQB, CMMI, and others. Eran is a patent holding inventor (test exclusion automated mechanism for mobile J2ME testing). He is active in the community and can be found all over social media (Facebook, Twitter @ek121268, LinkedIn), and on his professional blog: http://continuoustesting.blog

About Perforce

Perforce powers innovation at unrivaled scale. With a portfolio of scalable DevOps solutions, we help modern enterprises overcome complex product development challenges by improving productivity, visibility, and security throughout the product lifecycle. Our portfolio includes solutions for Agile planning & ALM, API management, automated mobile & web testing, embeddable analytics, open source support, repository management, static & dynamic code analysis, version control, and more. With over 20,000 customers, Perforce is trusted by the world's leading brands to drive their business critical technology development. For more information, visit www.perforce.com