WHITE PAPER

USING DEVOPS PIPELINE TO GENERATE CONTINUOUS INSIGHTS

HARNESSING DATA TO MAKE YOUR DEVOPS PIPELINES SMARTER AND MORE EFFICIENT

apexon.com

INTRODUCTION

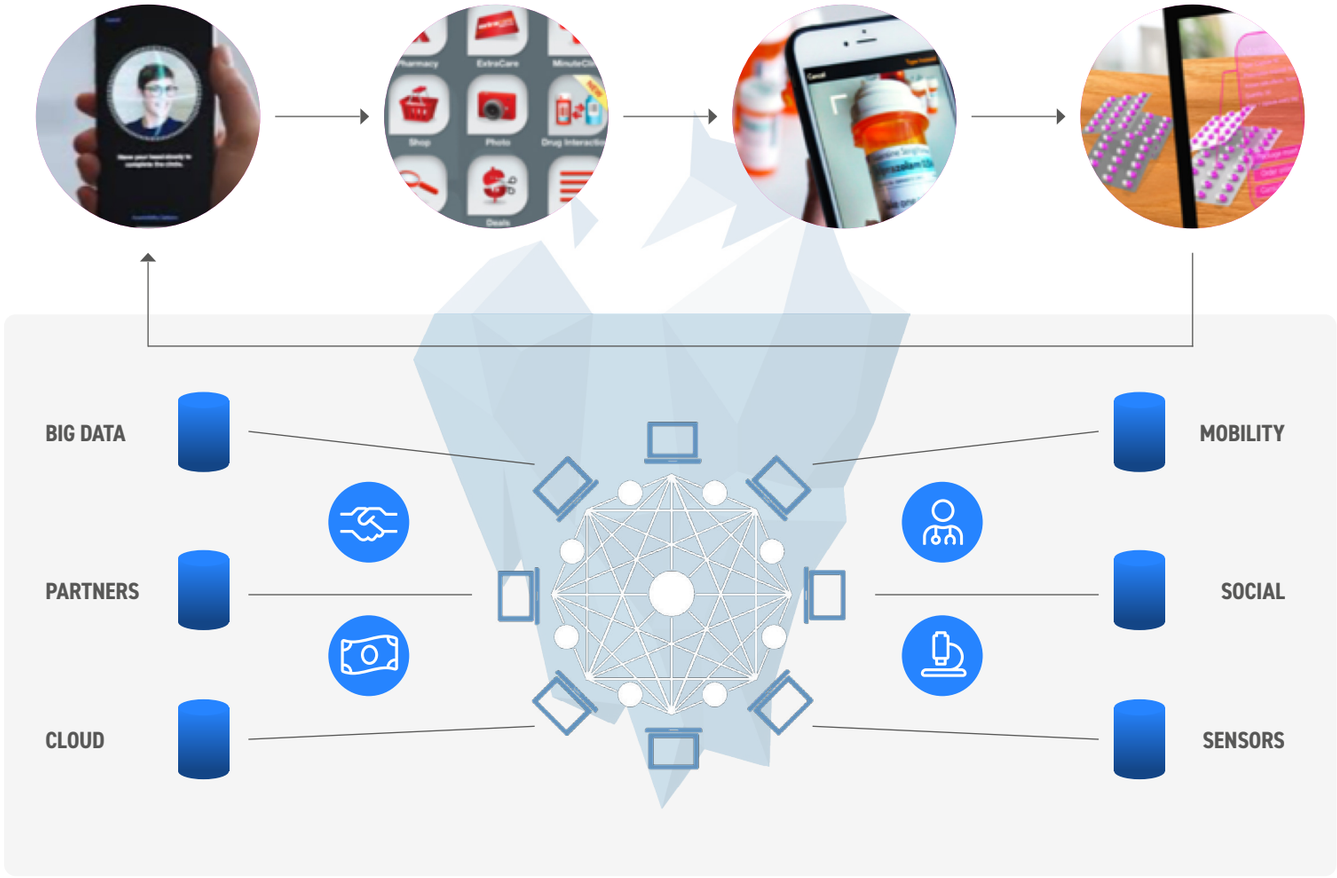
THE AGE OF DIGITAL TRANSFORMATION

As much as we all dislike the term and the hype that goes with it, the reality is that there is substance to digital transformation – and we are living it, especially in the tech profession.

What it really means in practice is that those of us in the software development and delivery business are building different types of applications and services; e.g., progressive web apps, mobile apps, IoT, partner ecosystem applications, and more. They may each require their own Agile team and some follow the micro-services roadmap.

Either way, they all are creating many different pipelines. Anyone in the dev, test, delivery world is seeing an increase in the number of pipelines in their work environment.

All of this puts pressure on how these pipelines are managed and how efficiently they are run. We all have finite resources and we have to have a smarter way of managing our software pipelines if we are going to succeed.



DEMANDS FOR MODERN DEVOPS

DIGITAL CREATES NEW DEMANDS FOR DEVOPS TEAMS

The ability to get new digital services to market more quickly while assuring optimal performance, functionality and quality, is a fundamental requirement for almost every business today at some level.

HIGHER QUALITY & GREATER ACCURACY

IMPROVED ANALYTICS

REDUCED COSTS & INCREASED SPEED

GREATER SECURITY & COMPLIANCE

BETTER MANAGEMENT

ADAPTIVE

SELF-HEALING

SCALABILITY WITH SIMPLICITY

As pipelines increase, we all feel increased pressures such as:

- More features
- Faster release cycles
- Reduced cost
- Increased testing scope to accommodate more variations and devices
- Better quality and higher service levels
- Greater security and compliance
- Self-healing systems
- Greater scalability
- Improved analytics and intelligence

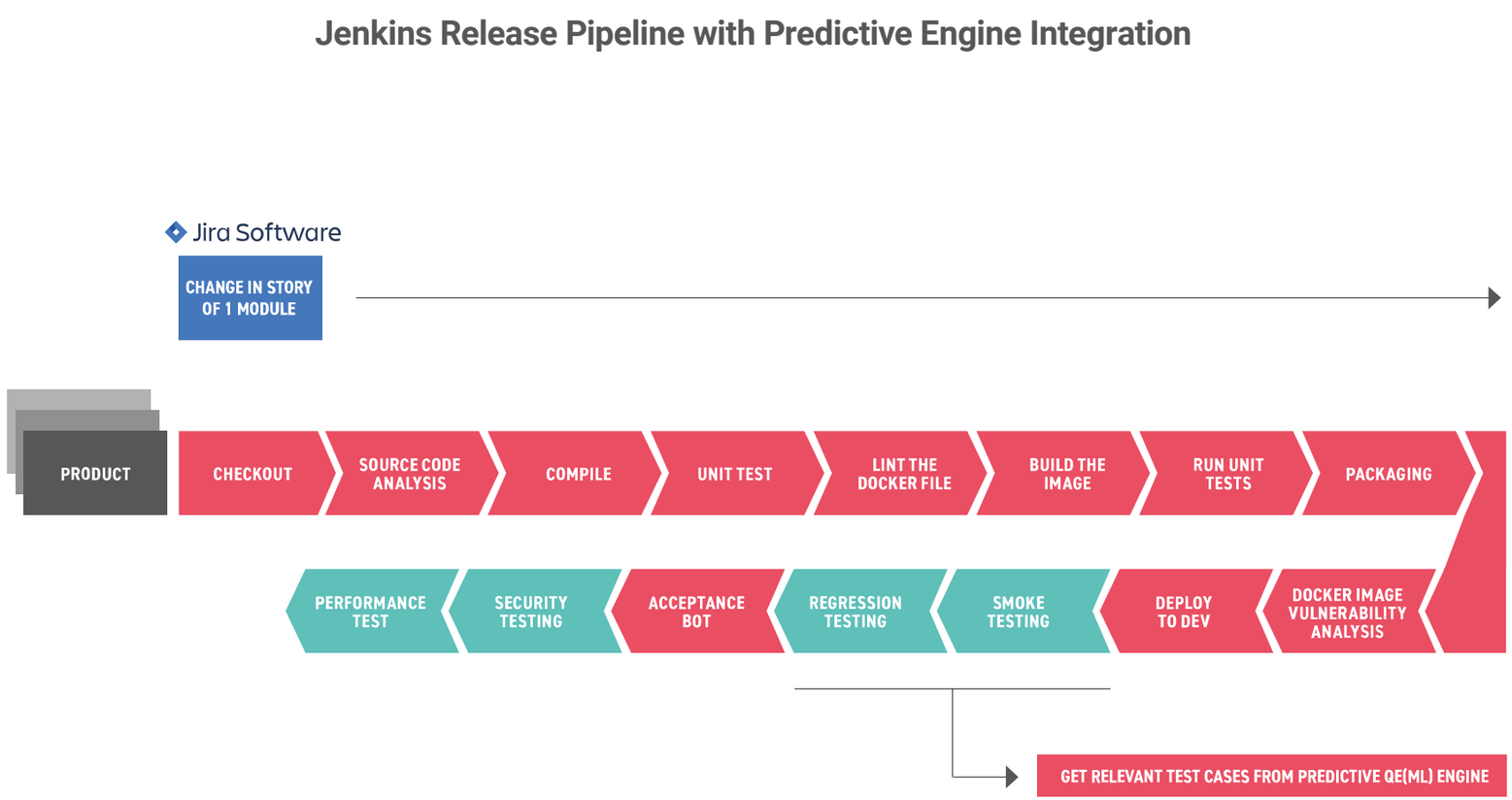
All these things are required for effective DevOps today. And to deliver on these requirements, means it's essential that we make our DevOps pipelines smarter. If not, the workload is just going to keep on growing.

TESTING PIPELINE

SMARTER DEVOPS PIPELINES

You can apply this thinking to any aspect of the software delivery lifecycle, but let's look at the most time-consuming aspect of the delivery pipeline – testing.

The image to the right depicts a typical type of pipeline:



All these things are required for effective DevOps today. And to deliver on these requirements, means it's essential that we make our DevOps pipelines smarter. If not, the workload is just going to keep on growing.

There is a lot of repetition. Every time you make a change, you are probably still running a lot of things that may not be necessary to run.

Each one of these modules generates data – lots of it. Each and every tool you are using is generating some kind of log. This is data that is available every time you run it.

Are you harnessing that data to get recommendations?

When the modules change, when you release a product, there is generally no such thing as a bug-free product.

So, what is a safe release? What are the criteria in your organization for a safe release?

The changes that are being implemented, how important are they? What are the different dependencies? If we change one module, what are the other things that need to change?

Or might be impacted?

Today, this is being done, but it's typically highly manual, and very time consuming. That naturally limits what can be done.

You are running these pipelines, large volumes of them, with ever faster release cycles. You are likely running some pipeline every hour; every time the developer checks in the code, your CI/CD pipeline is supposed to generate that data. Every time that run is happening, you are generating data that can be used to help improve your pipelines; to make decisions about priorities and where to focus valuable time and resources.

One of the important things to consider here is the resource efficiency.

How are you using your resources? Your Build resources, the Build machines, or the test infrastructure?

You are spinning up some kind of dynamic test infrastructure. Are you really using that efficiently? That's the key. You need data to understand that. If you take into consideration all these competing or different pipelines, and you can start understanding your resource utilization and the associated costs, you will see that making these more efficient by harnessing the available data can produce big benefits.

For example, we worked with a client having a tough time dealing with the rollout of new mobile applications. They had cloud resources that they used to test their mobile apps and they paid every time they ran a test on one of the mobile devices in the cloud. As their pipeline increased, their testing costs increased in a linear fashion. It was simply, not sustainable.

If they could optimize their test pipeline, they could realize some significant savings.

IN PERSPECTIVE

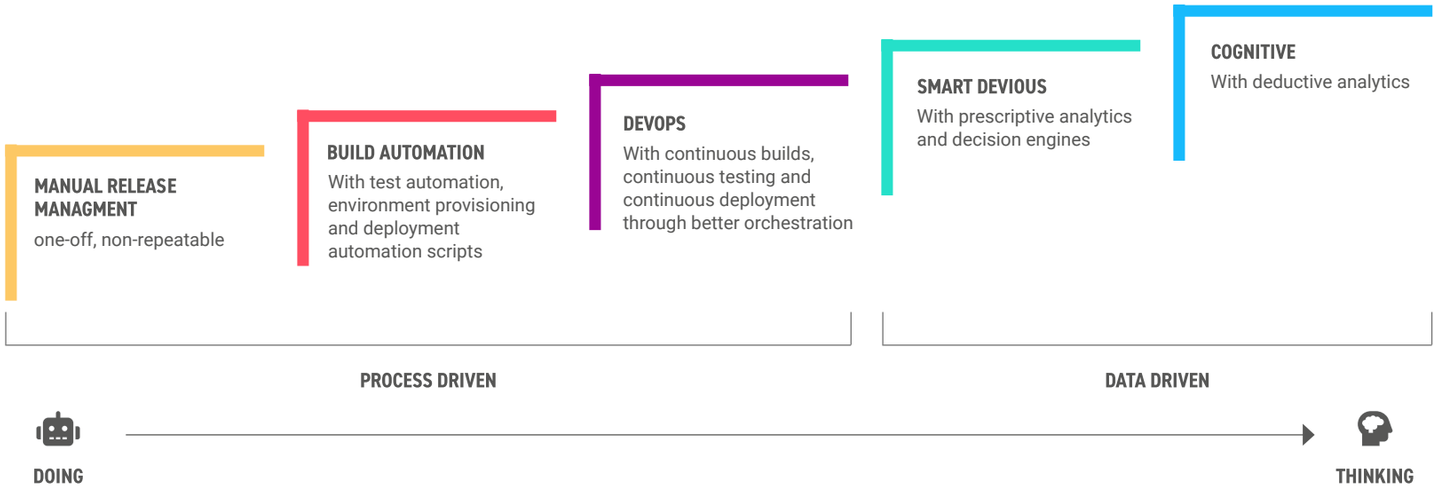
PROCESS DRIVEN VS. DATA DRIVEN

We all started doing one-off, manual release management. But then we started moving to build automation and then to DevOps. And now, the future is about accelerating the pace of our DevOps efforts by moving towards Smart DevOps that incorporates prescriptive analytics.

And the holy grail is Cognitive with deduced logic that promise to make our entire process smarter.

It's a movement from more process-driven (and manual doing) to more data-driven "thinking things" so we can delegate more routine tasks to digital workers or software robots.

An Evolution of Software Delivery



DATA TESTING

MAKING DEVOPS PIPELINES SMARTER

The key to moving from process-driven to data driven is liberating all the data in our DevOps pipelines.

Let's consider an example from a testing perspective:

In a testing environment, we have the opportunity to collect data at multiple points and tools in the process:

- Requirement management tools
- Check-in
- Status analysis
- Tools that you are using
- Test data management

If you can start harnessing the data from each of these steps, you can start drawing good insights to decide which Smoke tests you need to run, which Regressions you need to run, when you need to do Security and Performance testing, etc.

And the holy grail is cognitive with deduced logic that promise to make our entire process smarter.

- Leveraging Defect and Project data to predict the critical modules for the next version.
- Performing what-if analyses and providing early indications of schedule overruns.
- Reducing testing effort and improving the likelihood of success.
- Drawing insights from Build logs (e.g. Jenkins) on the number of Build failures and related trends, and then utilize it to design more effective test use cases.
- Examining source code check-in and check-out logs to design optimized test suites.
- Detecting quality deficiencies at an early stage so corrective steps can be taken in advance.

DATA IS THE KEY

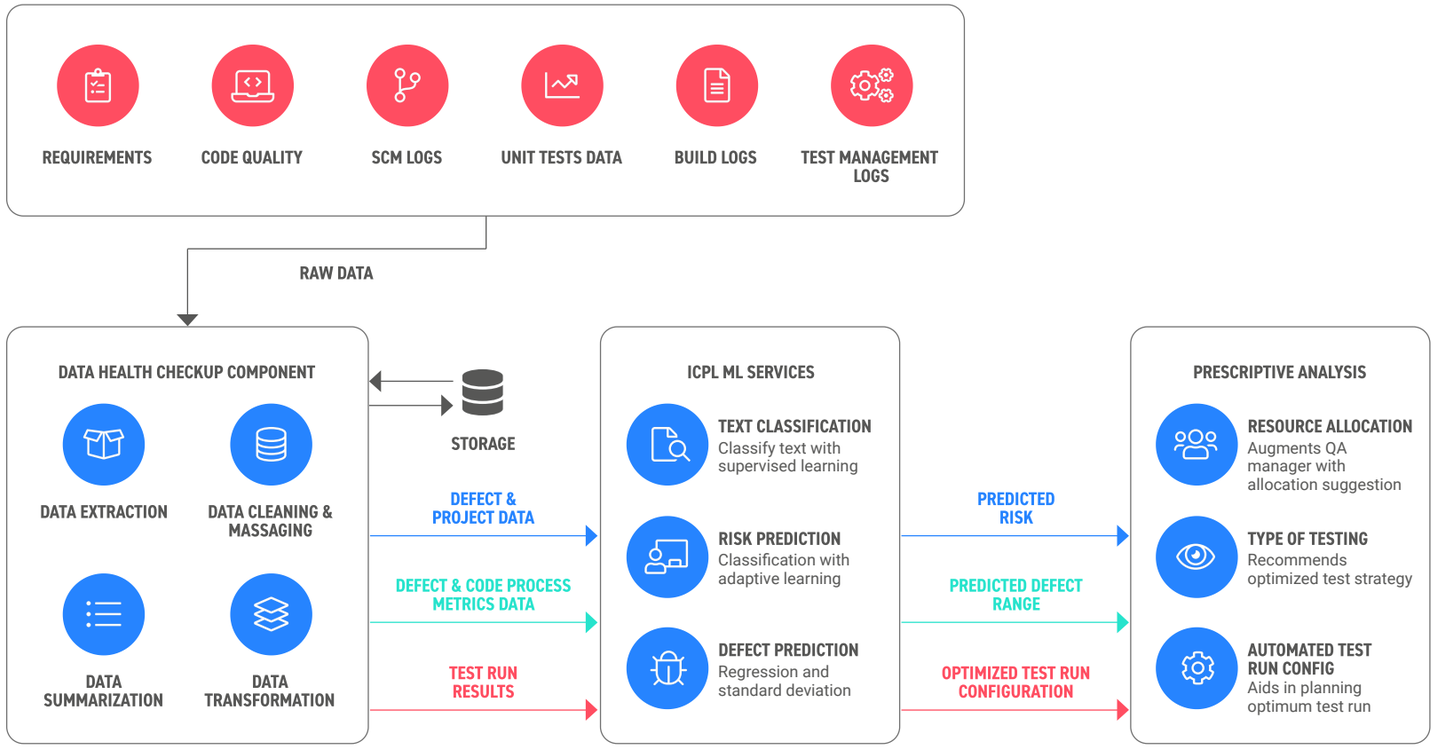
HOW DO WE MAKE OUR DEVOPS PIPELINES SMARTER? DATA.

You can gather data from various sources; e.g., requirements management tools, code repositories, code quality tools, unit test data logs, build logs, and test management data.

All this data is plugged into the engine – some of it is pulled in; some comes through CSBs by dumping data logs into a particular directory, etc.

Before we can use the data to train your machine learning model, you need to create a unified way of representing that data. Some of that data will be structured, some will be semi-structured, and some will be completely unstructured, especially logs. So you need to standardize the data and then balance it. This is very important. Every different tool has a different interpretation of how it looks and labels elements of different versions and components. We need to normalize the data so we can take advantage of it.

Harnessing Data in a DevOps Pipeline



Let's take two examples:

1. VERSIONING

This data might be available in some logs and other logs. We can find the data ranges we are working with, and which particular version is getting affected.

2. COMPONENT

This is obviously critical information to have so we know which component is being impacted. We can use text classification to identify whether this is a login module, a shopping cart, etc.

The key is we start classifying our data and labelling it. Then we can start to use that data more effectively to define the risks and prioritize our testing requirements accordingly.

Balancing is also important. In the past, before Agile, etc., if you looked at your past data manually, you'd immediately notice things like the size of different release cycles; e.g., some are a month, others are as long as three months. We'd see a similar skew in terms of the numbers of bugs in which modules or components. So, it's important to balance that data out so it doesn't skew our results.

You may also have user feedback which you want to factor in. You can decide which releases you want to consider and which you don't based on their validity. You want to avoid skewing your results one way or another.

With your data massaged and balanced, you are now ready to get into two important things:

1. RISK PREDICTION

Looking at your data, we can start to predict which modules are higher risk and which are lower. By evaluating bug data and the severity of bugs of a particular label, we can start to see trends around specific labels; e.g.:

- **Bugs reported**
- **Check-ins – what percentage failed, what specific test has failed**
- **History of specific code – who checked it in, history of that developer, etc.**

All these things are important to building our understanding of what is high risk, low risk; more or less likely to happen. Once we understand what it is, we can add in historical data for computation.

So now, every time you move forward with a DevOps pipeline, you are putting more and more data in, making your pipeline continuously smarter for each successive release. This will improve the predictions are you making and reduce your risk.

2. REGRESSION

The other aspect of risk predictions takes into account more qualitative factors such as the complexity of the changes that are going on in development, the stability of the module, value points; i.e., the importance from the customer's or developer's perspective, and how many high severity bugs were found.

Once you start understanding that data, you can start running your regression modules and you will have a pretty high degree of confidence about how many bugs you can expect in any particular module.

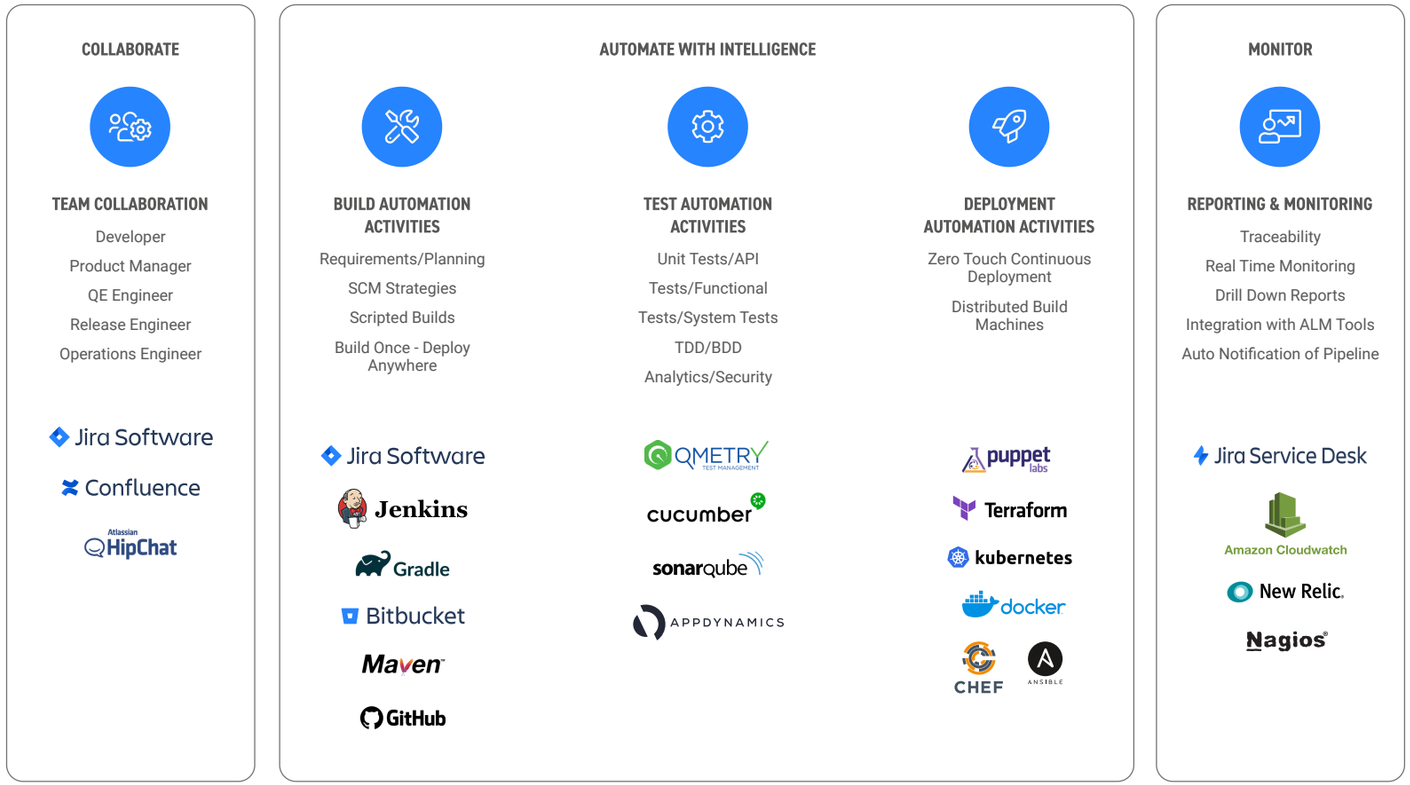
With a good understanding of your risk factor with any specific module, we can start using that knowledge to start making predictions. This will dictate your test configuration; e.g., tests you need to run the most, those you may not need to run, where you may not want to waste resources, etc.

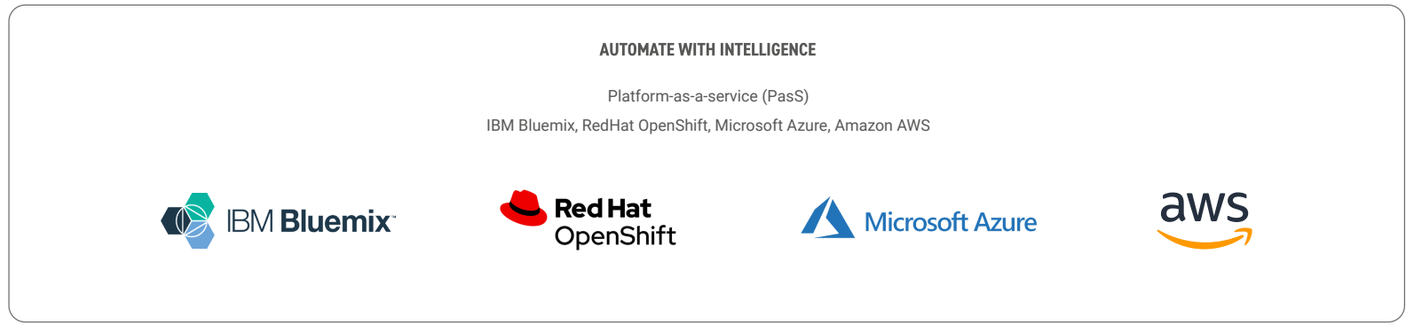
This can be extremely important when running very large volumes, with lots of applications, and different devices. You now know what to run, when to run it, and can release product with very high confidence.

APEXON DEVOPS

AN INTEGRATED EXPERIENCE

Apexon provides an integrated DevOps experience that covers from requirements management all the way through deployment and monitoring.





HARNESS THE DATA

MOVE YOUR DIGITAL INITIATIVES FORWARD

Our team can provide the expertise and execution you need to harness the data in your DevOps pipeline to move your digital initiatives forward faster and with more confidence.

We've worked with leading enterprises to help them automate their build process, incorporate sound design principles, enable cross-functional visibility, and identify the right infrastructure and tools for their environment and team.

As a result, they have seen substantial benefits:

Increased speed, stability and reliability in software delivery

Better software quality and reduced QA time and costs

Greater flexibility in development

The ability to identify and resolve code problems earlier in the development process, and correct them before they become larger, more costly issues downstream

Reduced long-term development and business costs

Apexon

APEXON IS A PURE-PLAY DIGITAL ENGINEERING SERVICES FIRM FOCUSED ON HELPING COMPANIES ACCELERATE THEIR DIGITAL INITIATIVES FROM STRATEGY AND PLANNING THROUGH EXECUTION.

We leverage deep technical expertise, Agile methodologies and data-driven intelligence to modernize systems of engagement and simplify human/tech interaction.

We deliver custom solutions that meet customers' technology needs wherever they are in their digital lifecycle. Backed by Goldman Sachs and Everstone Capital, Apexon works with both large enterprises and emerging innovators — putting digital to work to enable new products and business models, engage with customers in new ways, and create sustainable competitive differentiation.

info@apexon.com

www.apexon.com

FEELING SOCIAL?