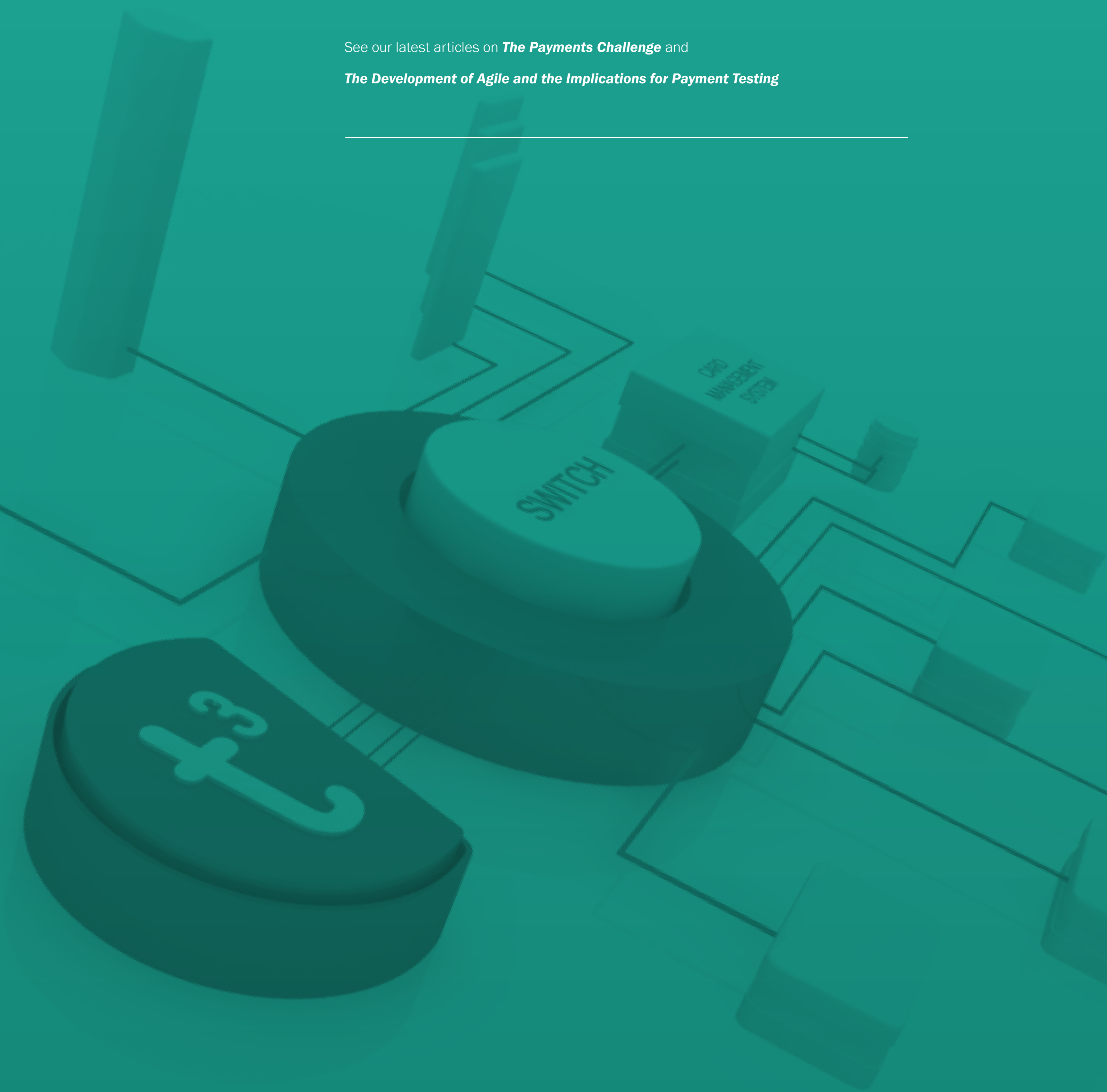
Iliad Research and Analysis

AUGUST 2016

See our latest articles on ***The Payments Challenge*** and

The Development of Agile and the Implications for Payment Testing



The Payments Challenge

In the specialist area of payments testing things have become increasingly more challenging. Ever evolving standards, proprietary interfaces and formats, cryptography, the evolution of new messages and tokens, and the constant external forces of crime and fraud all add immeasurable layers of complexity which need to be managed and controlled. Add to this an increasing move towards Agile development methods (see later in this document) and it's easy to see why testing payments is not quite as simple as it used to be.

The risks and the costs

"The Economic Impact of Inadequate Software Testing to the US economy is estimated at just under 1% of GDP or about $60 billion."

SOURCE: NIST

When financial institutions get things wrong, high profile exposure in the press and social media damages corporate reputation and share prices. Yet in too many of those institutions, testing remains something that receives insufficient attention until too late in the process. The key reason for this is that the whole process of testing payments is often facilitated by a maze of standalone simulators, in-house tools, stubs, and 'mocks' (ad hoc pieces of code developed by an organisation's employees). The result is an environment that is incredibly difficult for a bank or payments processor to manage.

It is worthy of note that as long ago as 2002 a NIST report entitled 'The Economic Impact of Inadequate Infrastructure for Software Testing' estimated the cost to the US economy to be just under 1% of GDP, or about $60 billion. Though impossible to extrapolate directly from this, it is reasonable to assume that the effect of an inadequate testing environment upon a financial institution is equal to a proportion of its turnover exceeding 1%. Reducing this inefficiency will therefore have a major impact on the industry and those who supply it.

What does this mean?

It is difficult to distil into one phrase the problems of setting up a testing project in a financial institution. But, if one were to attempt this it would probably be 'complex, costly and incredibly difficult to do'. Many organisations have evolved to a state where it almost appears that there has been a deliberate effort to introduce as much complexity and as many discrete components as possible, with no sharing of results, IP or data.

In reality, what has occurred for years is that testing departments have been applying a sticking plaster to a range of immediate crises, ignoring the growing 'wound' beneath. Now, competing pressures are threatening this approach:

- the need for faster changes and to be the first to market with new development.
- the challenge posed by new and nimbler entrants.
- the increase in customer expectations.

These pressures result in the wound being increasingly exposed, and the pain, for many organisations, is becoming unbearable.

Key Issues the Payments Industry needs to tackle.

TESTING SILOS	Standalone solutions, focused on a single element of an environment, limit the learning and provide only part of the answer. For instance, if an ATM test fails due to a VISA response, the separation of the environments and the consequent subset of data available prevent proper analysis. This means that the test delivers inconsistent and questionable results.
ACCESS TO RESOURCES	Physical resources are limited by their very nature. For example, testing on a device or interface locks out other test engineers and usually demands that time is spent on resetting the environment before other tests can take place. Where this extends to ownership outside an institution (such as a scheme environment) major delays ensue and further costs can be incurred.
MULTIPLE SKILL SETS	It is common for financial institutions to have eight or more simulators in a payment environment; each of these is likely to be supported by two to three people. Maintaining multiple skill sets adds cost and limits the ability to manage the peaks and troughs of resourcing requirements.
HUMAN WHITE SPACE	There is a huge amount of waiting around in testing. Waiting for the system to be set up, waiting for access, waiting for physical resources, waiting for re-sets and many others. This human white space is expensive, and can often eat unnoticed into the margins of a project.
REGRESSION ISSUES	Isolation of components and tests mean that full regression is often hard to achieve. Changes made in one area of the system may be tested with success, but could trigger issues latent in another, untested part. This, in turn, is exacerbated by manual testing
MANUAL TESTING	Expensive and often inconsistent, manual testing is highly inefficient. Pressure on margins often forces down the quality and knowledge of test engineers, which further exposes the process in every sense. In short, today's payments testing environment is riddled with inefficiency, cost and risk. If senior managers understood the state of a typical test environment, business heads would struggle to rest easily. Past the point of being unfit for purpose, it is approaching the stage where it has the potential to do real damage to a financial institution.

Addressing the Payments Testing Challenge

There are specific ways to address the payments testing challenge.

VIRTUALISATION

Virtualisation delivers great benefits through the removal of resource constraints, the consolidation of skills, and a testing approach that creates a more consistent, standardised environment. Being able to virtualise an environment is, in most cases, sufficient to produce greater efficiencies. Creating multiple virtual images of the system under test enables entire testing streams to be run in parallel. Each engineer, or group of engineers, has access to their own virtual system, without affecting the testing of others. Virtualisation enables the simulation of a real world environment, without the impossible costs of operating for testing purposes multiple mirror images set up end to end. In the payments space though, lacking support for the relevant standards, message flows and cryptography means that standard generic platforms can only hope to scratch the surface of what is possible.

TESTING THE PAYMENT PROCESS – END-TO-END TESTING

End-to-end testing is a procedure which tests whether the flow of an application is performing as designed from start to finish. In payments, it should ensure that the integrated components of the application function as expected. The entire application is tested in a real-world scenario such as communicating with the database, network, hardware and other applications.

It is no longer sufficient to test elements of the payment process flow individually and in silos. The payments cycle incorporates all aspects of the payment process, from initiation all the way through to settlement and reconciliation, and the test environment should have the capacity and capability to test this as if in the real world.

TEST AUTOMATION

To many people the benefits of automation are obvious. Tests are consistent, can be run faster, and run over and over again with fewer overheads. As more automated tests are added to the test environment more can be run each time. Manual testing never entirely goes away, but efforts can now be focused on more rigorous and consistent tests.

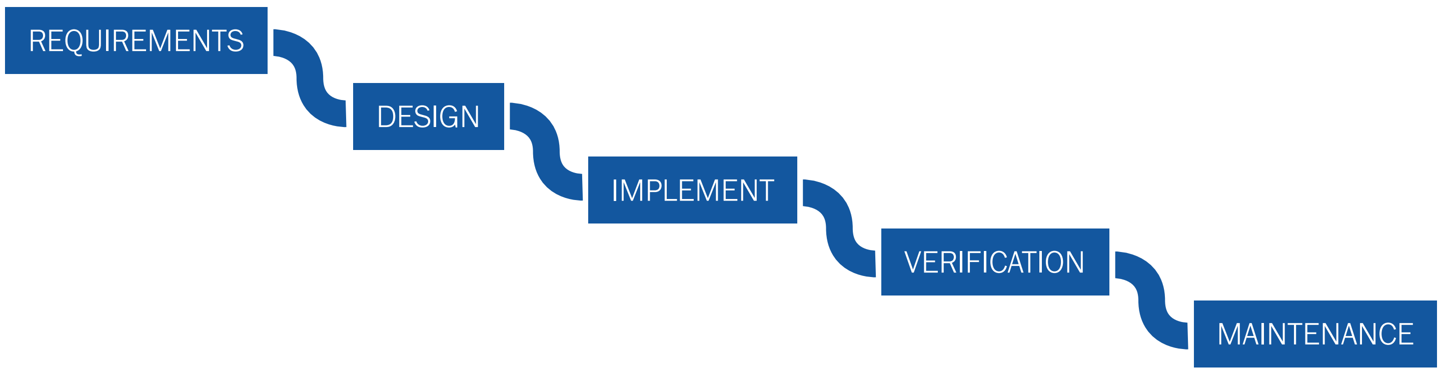
Developing a test automation strategy is very important in mapping out what's to be automated, how it's going to be done, how the test assets will be maintained and what the expected costs and benefits will be. Just as every testing effort should have a testing strategy or plan, so should there be a 'plan' for test automation. Sometimes there's the perception that automation is easier than testing manually, but automated testing does not replace good planning or the writing of test cases.

RE-USING AND MIGRATING LEGACY TEST ASSET

When considering migrating to a next generation testing platform, consideration must be given to the use and re-use of existing test assets and whether they can be used in the future. No platform is really complete without the option of such a migration path. Without it, migration can be a very tedious and cumbersome process, and in practice this is one of the biggest stumbling blocks in migration projects. It should be possible to redeploy to the new platform existing testing assets developed over the years, without starting from scratch.

The development of Agile and the implications for payment testing

Since the late 1970s, the computing industry has relied on waterfall development patterns to provide the structure around which software is created and deployed. The waterfall is a robust framework in which each element in the sequence largely depends upon the completion of the one before. A typical waterfall programme would look like this:

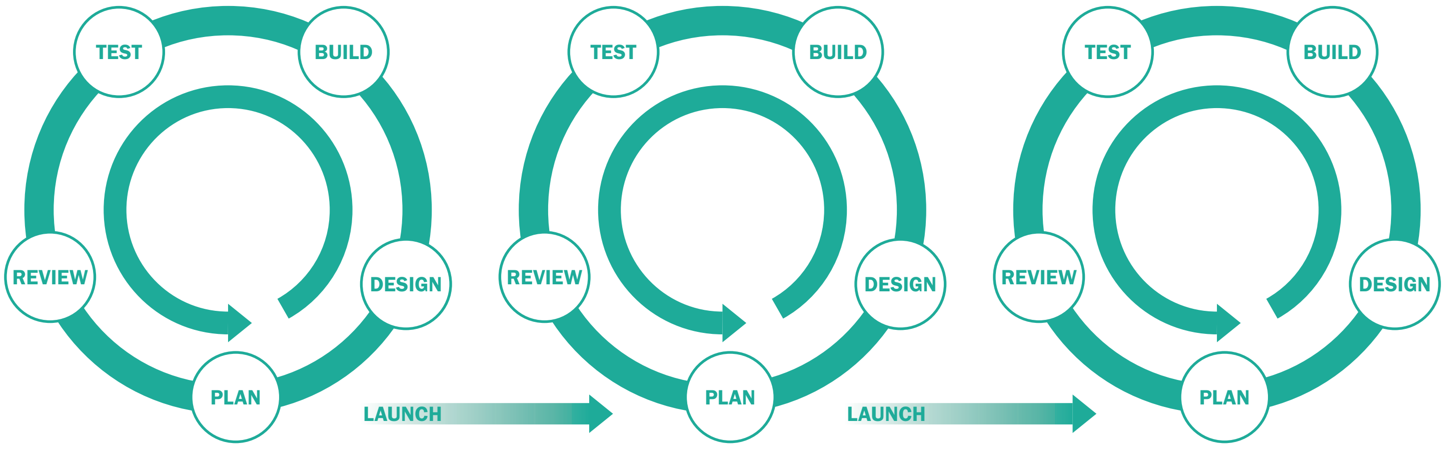


An entire industry sprang up around waterfall, and various mnemonics were applied to the stages through which a software or programme development had to pass. Perhaps the most generally accepted was: Define, Design, Develop and Deliver.

As computing power increased along with the demands on development departments, born out of aggressive commercial change across all industries, a view developed that Waterfall was not responsive enough for 21st century businesses. Instead, the computing industry began to use a methodology called Agile. In essence, this was more a mind-set and philosophy than a methodology, offering a new, integrated approach to developing and implementing software that closely engaged all elements of an organisation from inception to completion.
Agile was established on a simple set of principles, published as the Agile Manifesto, which sought to lay out the values upon which it would be based:

1. Customer satisfaction from early and continuous delivery of valuable software
2. An acceptance of changing requirements, even in late development
3. The speedy delivery of working software (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. The building of projects around trusted and motivated individuals.
6. Face-to-face conversation (co-location) as the best form of communication
7. Working software as the principal measure of progress
8. A constant pace of sustainable development
9. Continuous attention to technical excellence and good design
10. The importance of simplicity—the art of maximizing the amount of work not done
11. Best architectures, requirements and designs, emerging from self-organizing teams
12. Regular team reflection on how to become more effective, and appropriate adjustment

These principles dictated an iterative, progressive approach to development rather than one bound by heavy duty, inflexible steps. This iterative process focused on reducing the size of cycles and creating a fluid programme through which software elements could be created rapidly.



Agile, executed properly, is a disciplined approach to system development that delivers a genuinely business-centred methodology, uniting business and IT personnel in the rapid construction of software solutions. Executed badly, the term Agile can be a licence to adopt a sloppy approach to development, with compressed schedules, dismissed documentation and coding continued right up to the point of implementation.
The challenge for all Agile teams is to ensure that the methodology's benefits are maximised and that it does not become an excuse for ill-discipline and poor quality delivery.

Testing in Agile – ensuring success

Central to ensuring success in the execution of an Agile project is testing. The principles of Agile demand that testing is a core part of the whole process, not simply a random addition or something that merely checks a piece of code is working. Teams must adopt a new mentality, where testing is seen as central to information management, improved communication across teams, and a fluid assurance element supporting all stakeholders in the project. It is not just a developer of code, or a means of policing a project's results.

If no clear testing strategy exists, or testing becomes a mere component level exercise, even the most well intended Agile project will fail. One example quoted is that of a developer exclaiming to a project manager that nobody had told him that the software had to be able to handle corrupt data. He had simply tested to ensure that the code worked, which it did in his development environment, but wouldn't ever in a real deployment. Proper testing ensures that the real world state can be protected at all times throughout the process.

What this example demonstrates is that fragmented testing is equally dangerous, whatever the development methodology. Whilst the importance of testing is fundamental to Agile, success requires a well-executed test strategy focused on extended collaborative working with simultaneous learning about the software, and designing and executing tests. Continuous integration and testing automation require an environment that is designed to expose learning, share intelligence and support an evolving real world state. Component simulation cannot deliver this as it isolates learning, behaviour and individuals, and makes test automation impossible to achieve effectively.

Delivering the testing environment Agile needs

Service virtualisation enables the end to end modelling of a complete environment for testing purposes by simulating the behaviour of software components and removing dependencies and constraints on development and testing teams. Such constraints occur in complex, interdependent environments when a component connected to the application under test is:

- Not yet completed
- Still evolving
- Controlled by a third-party or partner
- Available for testing only in limited capacity or at inconvenient times
- Difficult to configure in a test environment
- Needed for simultaneous access by different teams with varied test data setup and other requirements

Whilst every environment faces different challenges, the above list defines pretty comprehensively the conditions experienced in an Agile environment. The methodology dictates evolution, and implies that components of different sizes will be at various stages of incompletion through much of the development process; in addition, lack of access to physical resources is an issue in almost every payments software project.
Commercial, off-the-shelf service virtualisation technologies deliver the power to address these challenges in a generic application environment. However, the idiosyncrasies and specifics of payments make it close to impossible to employ them in the payments environment.

Iliad Solutions

In a world experiencing unprecedented technological change, businesses need to be confident that they can keep pace and rapidly evolve. Iliad has over 25 years of experience in payments. We have been at the forefront of building, implementing and supporting major payment solutions. Our experience has led us to develop the most comprehensive and resilient test solutions available in the world today.
Iliad's global customer base trusts us to take the risk out of payment testing. With Iliad solutions:

- You can automate payment testing which significantly reduces your project costs.
- You can enable your teams to work rapidly, with more control and more visible results.
- You can forensically analyse test results which saves you time and increases the effectiveness of test teams.
- You can decrease risk through end to end testing in new deployments, which helps to protect your brand from high profile failures.

We regularly publish articles on all the latest developments in the world of payments. You can access all our regular publications and analysis at www.iliad-solution.com