computing
research

RESEARCH
PAPER

DevOps for Executive Management: How the organisation and funding of tech teams drives digital success

January 2021

Sponsored by
puppet

CONTENTS

Introduction

The last five years have seen a majority of enterprises begin to transform into organisations that can compete in the digital marketplace, regardless of the type of physical goods and services on which they are built. This marketplace was once considered the future, but that future has indisputably arrived – in no small part accelerated by the COVID-19 pandemic. Society and commerce have undergone extraordinary changes throughout the course of 2020, and the organisations that had already made significant progress in areas such as transformational software development are the ones which have continued to remain productive and profitable.

The move to more agile practice and the success of DevOps as a driver of software projects involves cultural and organisational change as well as technical. To examine these changes, *Computing* surveyed approximately 150 technical decision makers from organisations employing a minimum of 500 people. The survey sample was drawn from a wide cross-section of UK enterprises, including the financial sector, government, retail, logistics and manufacturing. Whilst the contributors to this research are technical decision makers, the research stands as a message to other executives because it examines how behaviours outside of the IT team are affecting the overall success of DevOps as a driver of these critical software projects.

The research examines the extent to which various factors are holding back DevOps and more agile development practice in organisations and subsequently frustrating the overall success of digital transformation. These factors include: a C-suite with a “command and control” approach to critical software changes; poor understanding of agile methodology; budgeting processes designed for the pre-digital era; and excessive dial up/dial down of contractors to plug skills and headcount gaps.

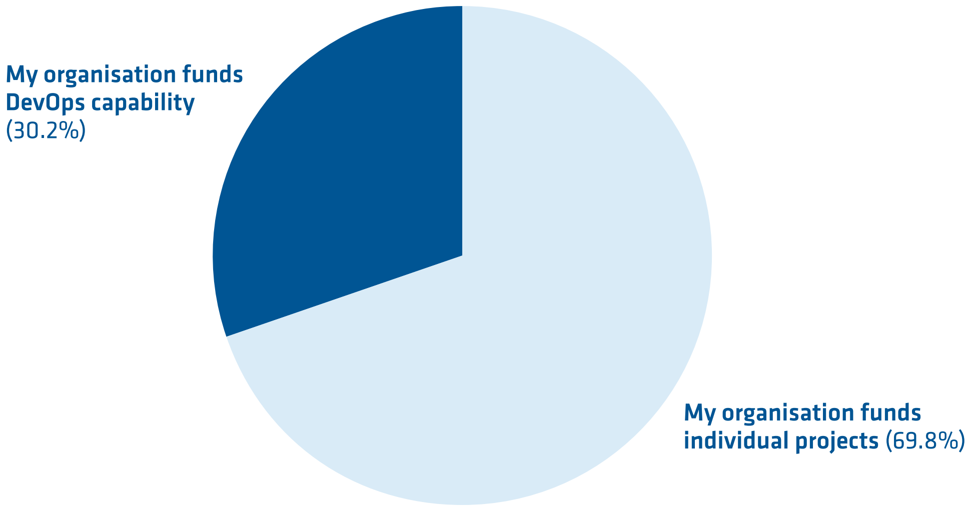
Key Findings Include:

- 70 percent of contributors fund transformative development projects on a project-by-project basis. 30 percent do so by means of funding DevOps capability.
- Only 13 percent stated that their funding model was agile enough to always keep pace with projects.
- Only 14 percent of respondents said that their funding model was “definitely” flexible enough for DevOps to succeed.
- Almost half of all contributors stated that between 20 and 50 percent of their software projects ran over budget, with a further 27 percent stating that between 50 and 100 percent of theirs did. The proportions reporting schedule overruns were broadly similar.
- Scope creep, unforeseen circumstances, and the general changeability of market conditions and software requirements were reported by most contributors as the reason behind late and over-budget projects.
- Only 38 percent of our contributors believed they had the balance between internal and external talent right. 34 percent believed that their organisations were too quick to look outside of the business rather than looking at re-skilling existing employees.
- 70 percent of contributors said that they had experienced problems arising from staff turnover on specific software development projects to at least some extent.
- Only 14 percent of contributors believed that their organisations were very strong on the development of junior technical employees. A majority of 59 percent felt that while their employer’s record was reasonable, it “could do better.”
- The establishment of cross functional DevOps teams has some way to go, with only 36 percent describing these cross-functional teams as well-established in their organisation. In only 9 percent of cases overall are business owners as accountable for project outcomes as their technical counterparts.

The Reality of Project Based Funding

Funding for software capability change is typically provided on a project-by-project basis and we can see that for two thirds of contributors to our research this is indeed the case. Software projects are typically funded by the Line of Business (LoB) management or the business unit that will benefit from the change, usually directly from their Profit and Loss as a technology cost – as opposed to from a central budget. The process tends to begin when a business unit decides they need a new IT capability or functionality. They begin dialogue with Business Relation Management (BRM) in IT who scope requirements, and the time, headcount and resources required to produce this functionality. A project plan will be produced with all the relevant milestones, budget, reporting, etc., mapped out.

Fig. 1 : Which of the following statements most closely represents your organisation's approach to funding transformative development projects?

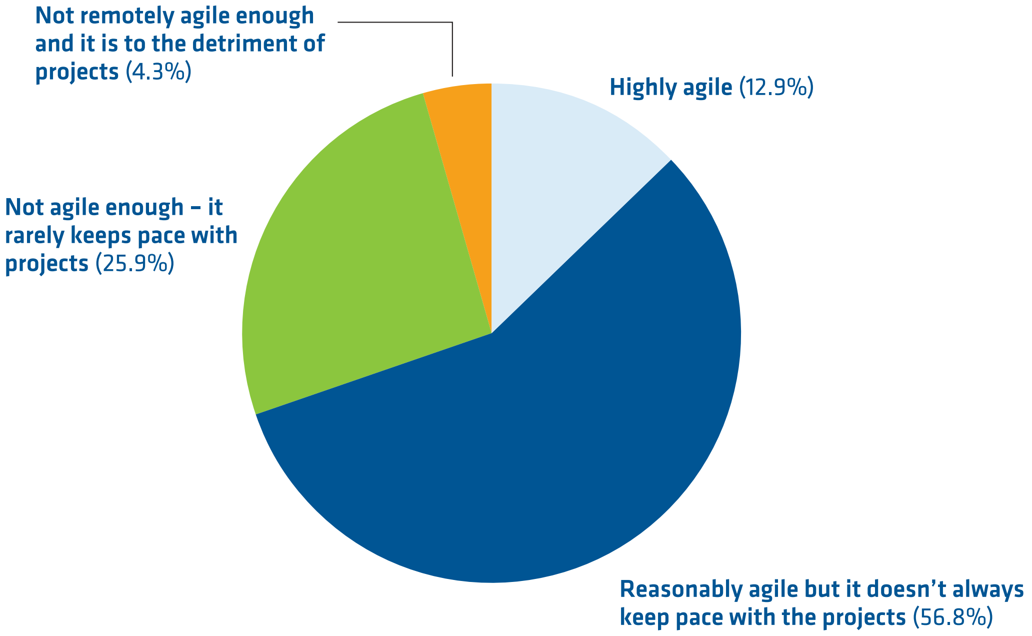


This all sounds pretty straightforward – and it is – until the reality of the business and the wider economic environment make their presence felt. We asked contributors whether the way that their organisation funds software projects is agile enough to accommodate the changeability of the business environment. The essence of transformational software is that it is flexible enough to do just this. That's the whole point. However, only 13 percent of contributors deemed their funding process to be highly agile. For the remainder of respondents, albeit to varying degrees, funding did not keep pace with the project. Stop. Start. Stop. Start. This is the antithesis of agile.

Adding detail to this troubling picture is the fact that when asked if their organisation's budgeting process was flexible enough for DevOps to succeed, only 14 percent of respondents gave an unqualified "definitely." The majority (60 percent) told us that it was "just about" flexible enough but could be better. The remaining 26 percent were considerably less convinced.

Why is this happening? Part of the problem lies in non-technical executives not understanding that long budgeting cycles, or a lack of clarity in the outcomes they want from digital projects, both have the potential to choke their success. There also sometimes exists misunderstanding or incomplete knowledge about the nature and potential power of agile practice. Essentially, historical knowledge about how to manage and control projects and the reality of modern software development are fundamentally at odds with one another.

Fig. 2 : However your organisation funds projects, in your opinion, is the way of funding software projects at your organisation...



We also asked contributors about how often certain scenarios occurred within their organisation with 1 being "not at all" and 7 being "all the time." 52 percent of respondents ranked the statement "additional finding is frequently required to complete projects" as a 5 or above, meaning it's a frequent occurrence for their organisation. 46 percent awarded the same ranking to the statement "engineers are hired on specific contracts for temporary projects."

The proportion of contributors identifying with these statements sheds light on some of the implications of this budgetary inflexibility and illustrates how it can potentially snowball into something much more expensive than a simple delay. If money runs out earlier than planned, it will do so after contractors or outsourcers have been hired. Contractor costs can be high, and enterprises often hire them for a fixed period of time – a duration decided by the original project scope. If new budget takes a long time to be released or projects have to be rescoped, then those contractors naturally tend to move on: they move to other projects or employers. Familiarity with the project is lost and, as outsourcing contracts invariably comes with stringent financial penalties for change or anything outside of the original contractual scope, costs can mount. Controlling contractor costs is highly desirable . . . but there is a balancing act. Over-enthusiastic, or optimistic, best-case scenario cost control can – demonstrably – actually cause costs to rise.

How often are software projects running over budget? Fairly frequently, according to the illustrations below. In 27 percent of organisations contributing to this research, more than half of their projects had done so. 49 percent of users put it at somewhere between a fifth and a half of projects.

Fig. 3a : Please indicate approximately the proportion of software development projects that you believe have run over budget in recent years

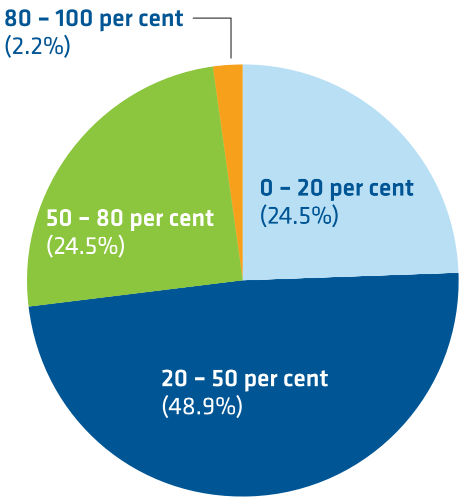
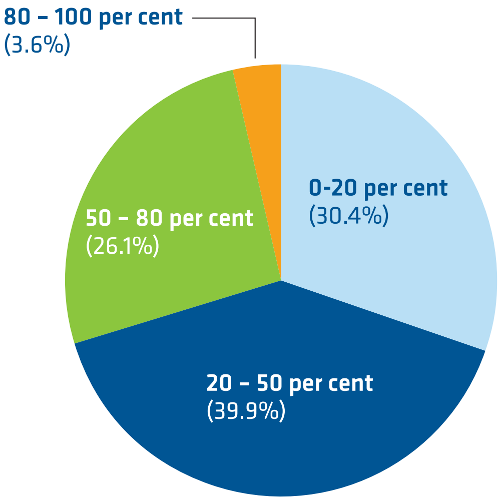


Fig. 3b : Please indicate approximately the proportion of software development projects that you believe have run over schedule in recent years



The proportions who had also run over time broadly mirror those going over budget – a finding which neatly illustrates the cost of project delays.

We asked contributors to briefly describe in their own words the reasons for these overruns. The message came back loud and clear, and some examples are as follows:

"Poor scoping and poor understanding of existing data and systems. Vendors over-promising."

"Operational pressures, with team members not available at key times to commit to projects."

"Too many changes throughout the life of the project. Also, most users/managers do not know what they want and do not put in the relevant planning time."

"Scope creep, objectives and requirements not adequately defined."

"Changes to core functionality received from customer during project."

"Because of the things you don't think of, and future patches that break custom work."

"Lack of upper management understanding."

"Requirements Change Issues encountered. No contingency. Expertise lost mid project."

"Lack of clarity of business requirements."

"Poor user requirements, lack of agility, time for development."

While the pandemic was mentioned a few times as well as a lack of automation in code control, regression testing and code-promotion to unit/integration testing, the vast majority of the given reasons boil down to often vague requirements from internal customers and management, which then lead to changing requirements mid-project. Planning is both poor quality and too ambitious and, crucially, is not flexible enough to adapt to changing requirements. Scope creep and unforeseen circumstances were raised time and time again. Or, as one contributor pithily surmised:

> *"Expectations versus reality."*

It is impossible to discuss the propensity of the project funding model to increase project costs without also discussing its impact on risk. The traditional approach of rigid documentation of project costs by breaking everything down into its component parts and putting it into a spreadsheet was supposed to give rigid control of cost and a clear understanding of risk. But the process is demonstrably failing because it fails to consider changing conditions.

Original project plans for software development effectively act as a snapshot of market conditions at that given moment. What came through in the responses to the questions above were also difficulties in communication. Technical teams do try and communicate the challenges experienced in terms of changing requirements to the relevant lines of business, executives, etc., but this communication seems to be failing in quite a few cases.

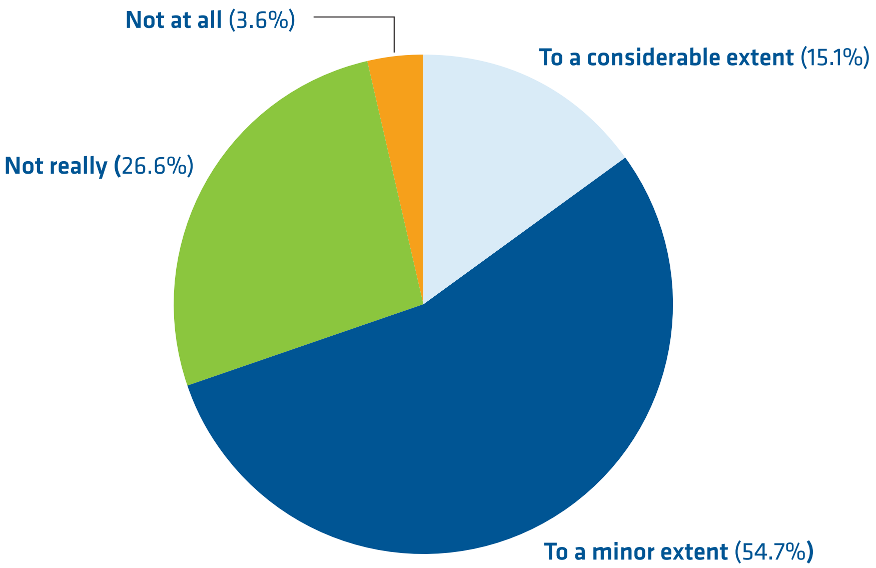
The project-based funding model means that when changing circumstances dictate a changed project, there are a number of process steps that must be completed in order for change to be signed off. This affects head count and other costs. These steps are there to ensure compliance, reduce risks and keep on top of costs, but the delays end up increasing both risks and cost.
It is clear that project-by-project funding was not designed for the business climate in which we find ourselves. The frequency of budget overruns and the use of external contractors suggests that, rather than controlling the risk of overspending, project-based funding actually entrenches it into the process. It is meant to control risk but often ends up doing the exact opposite, primarily because it is too slow to adapt to changing circumstances.

Contracting a Problem

One aspect of the overall approach of project-based funding, as hinted above, is that of contract labour. We now turn to this in more detail. Anyone even remotely connected to the technology industry is aware of the acute shortage of critical skill sets – software development and cloud native skills being particularly pertinent here. These shortages mean that individuals who are experienced in these areas are well aware of their worth and those who specialise in transformational projects often, quite understandably, choose to work as contractors to maximise their earning potential and flexibility.

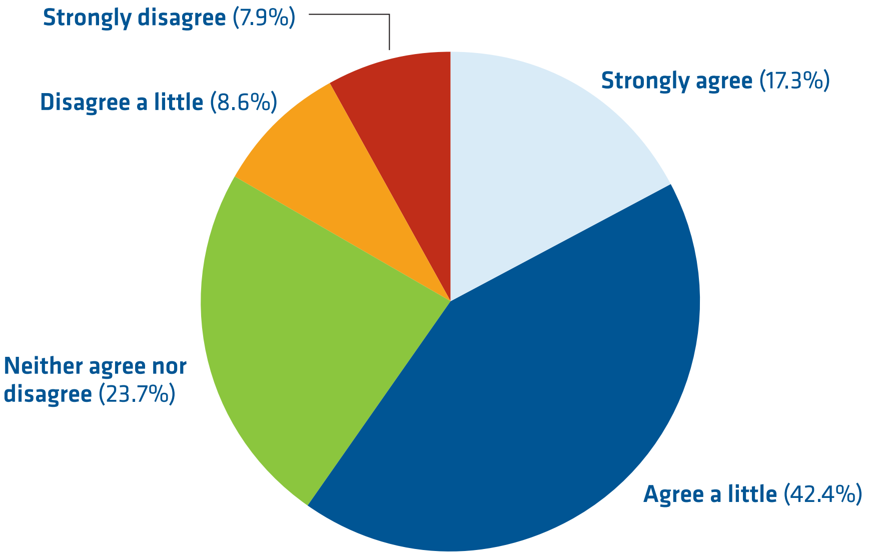
Most organisations use contractors to fill the skills gaps in software development projects. These could be developers, consultants, service providers or other outsourcers. The blend of contract and in-house skills isn't easy to get right and is of course a highly changeable business. Tellingly, only 38 percent of our contributors believed they had the balance right. 34 percent believed their organisations were too quick to look outside of the business rather than looking at reskilling existing employees, and 24 percent said they hired contractors for too long – a very expensive mistake. Furthermore, 70 percent of contributors said they had experienced problems arising from staff turnover on specific software development projects to at least some extent.

Fig. 4 : Has your organisations experienced problems arising from staff turnover on specific software development projects?



Perhaps more worrying are the answers illustrated in the diagram below, which shows just how many organisations are having to deal with the issue of skills and knowledge of their infrastructure walking out of the door. 60 percent had experienced this problem to some degree.

Fig. 5 : Please indicate whether you agree with the following statement: “Our organisation has experienced issues with skills retention and knowledge of infrastructure overall because of contractors and outsourcing practice.”



The high turnover of contractors is mirrored by an underdevelopment of more junior technical staff. Only 14 percent of contributors believed their organisations were very strong on this. 59 percent felt that while their employer's record was reasonable, it "could do better." The remaining 27 percent felt that not enough thought was going into staff development.

An over-reliance on contractors is clearly having some unhappy consequences in many businesses. It's also a huge issue in the public realm and appears to have been exacerbated or at least highlighted in all its expensive glory by the pandemic. In addition to the sheer expense of contract hires – and potentially consultants to figure out where the problems lie – a tendency to look outside of an organisation to plug skills gaps can damage communication between sometimes resentful technical teams and senior executives. As with general cost and risk, the thing that businesses are doing to try to solve a problem – in this case, skills shortages – are actually making the problem worse because the expensively procured skills are walking out the door, and existing staff will not have been equipped with enough knowledge of their own infrastructure. The vicious cycle continues.

Organisations which invest in training and recruiting more of their own people tend to see better collaboration during projects, as retention is easier when employees have a longer-term career path in an organisation. This increases their incentive to make projects work and reduces the chance of having to pay out for expensive consultants after something has gone wrong.

It is also the case that a more flexible, DevOps based approach to project funding and a more agile approach to software development is likely to attract exactly the kind of digital native skills that businesses need. The autonomy inherent in agile is an attractive proposition for developers and enterprises who can sell this to potential candidates.

Conclusion: Enterprises Need to Build an Effective DevOps Culture

Our research has shown that the project-by-project funding model for software development is no longer fit for purpose. The process is inflexible and slow and in a majority of organisations, funding is not keeping pace with projects – nor is the process flexible enough for DevOps to succeed. Because the traditional finding model relies on taking what is effectively a snapshot of market conditions when software projects are scoped, as soon as those conditions change – which is highly likely for a number of reasons – the model struggles to adapt.

Effectively, enterprises are using the "waterfall[1]" model to budget for software, whereas best practice in software development long ago mandated a shift to agile. Agile is an incremental and iterative approach that can flex to deal with changes in scope and requirements; waterfall is linear and sequential – and cannot.

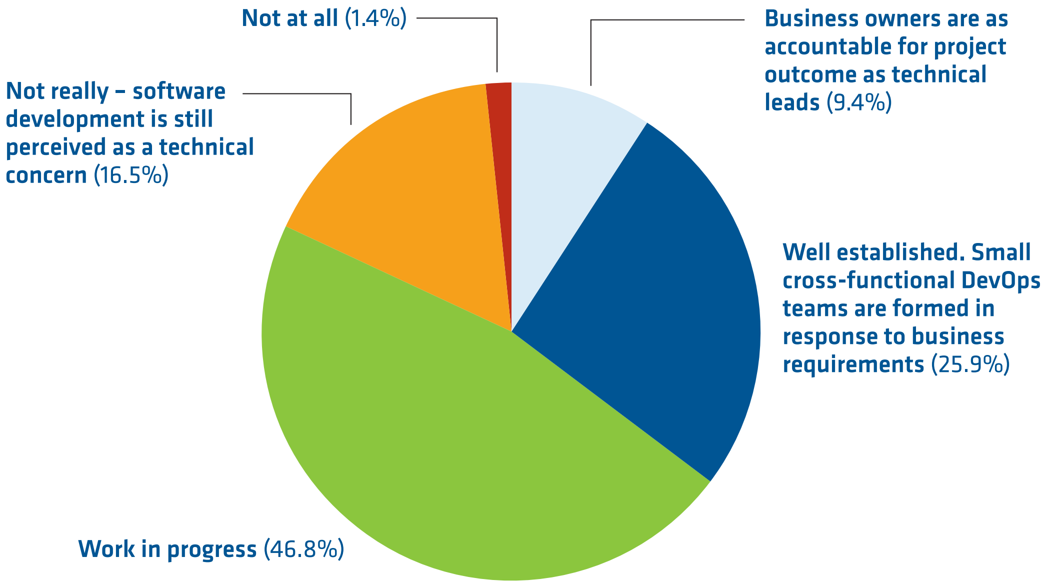
Applied to budgeting, the waterfall model struggles to adapt to constant, necessary and desirable change. Waterfall budgeting is actually increasing the overall risks and costs of software projects: a financial approach designed to control costs and mitigate risk is perversely entrenching both. The over-reliance on contractors is also ingraining the very skills shortages that it's been deployed to resolve. High value skills can walk off the job, taking valuable infrastructure knowledge with them. Existing staff are left demoralised and undervalued.

[1] https://en.wikipedia.org/wiki/Waterfall_model

The alternative is to fund by value rather than projects. If an enterprise wants digital transformation, they would be well-advised to fund that capability, rather than simply taking it project by project.

However, if DevOps and, by extension, a greater level of agility in software development and re-architecture is to succeed, this funding model has to be supported by a corporate culture which understands the power of DevOps and the necessity of autonomy in DevOps teams.

Fig. 6 : How well established are cross-functional teams for software development projects at your organisation?



The first aspect of this is the formation of cross-functional teams aligned to business value streams. This resolves one of the most significant structural barriers to digital progress which is the fact that in many organisations, non-technical executives and other employees tend to consider technical teams as solely responsible for the successful delivery of digital business propositions.

However, the best way to deliver transformational business outcomes is to create a cross-functional team and hold them directly accountable for the outcome. The product owner should be on the hook for exactly the same degree of responsibility as their technical counterparts. The existence of these teams, their stability (as opposed to heavy use of external labour which entrenches instability), and the autonomy of these teams is a helpful indicator of digital maturity. The autonomy of these teams is important because it makes it easier to recruit and retain the necessary skills. The further a business is into its digital transition cycle, the more likely it is to be attracting the professionals it needs to continue.

However, the illustration in figure six shows that the majority of businesses still have some way to go to achieve this maturity. Only 36 percent describe these cross functional teams as well established in their organisation, and only in 9 percent of cases overall are business owners as accountable for project outcomes as their technical counterparts.

Part of a more agile enabling culture also involves looking again at KPIs to measure the progression of a project. Long lists of metrics are profoundly unhelpful, and very much part of the spreadsheet focused, project-based funding model and an overarching "command and control" culture. All of this is death to agility. However, here, the picture among our contributors was not entirely positive. 45 percent agreed to at least some extent that "in my organisation, KPIs/approval steps for development projects act as an impediment to genuine progress and agility." Only 26 percent actively disagreed.

A better approach is the use of compound metrics focusing on quality, cadence and delivery. An example of this is a team that plans 70 or 80 strong points in a two-week sprint. In the first two weeks, they might only hit 20 but soon ramp up. Another useful metric might be the average time between a developer committing code and that feature being on the production site. In the more digitally mature organisations, the time between commit and production is very short.

What these cultural and organisational changes amount to can be described as BusDevOps. In order to instil greater agility in transformational software projects, the barriers between technical teams and the business as a whole need to come down. At present, in many organisations these barriers are being actively shored up by the process demands of traditional financial and project accounting methods – methods which are increasing risks, costs and staff attrition.

The measurement of digital progress with yardsticks designed for a different era leads to poor results. It takes time to see payback. Our world is a digital native one and the organisations who have a fast cadence of release cycles for their applications and software are the ones currently chipping away at more established products. If these companies are iterating every fortnight, it won't take them long to catch up with their more established competitors. Time is running out for those who have yet to make the change.

About the Sponsor, Puppet

Puppet makes infrastructure actionable, scalable and intelligent. From the data center to the cloud, Puppet helps enterprises modernize and manage their infrastructure to deliver innovation and efficiency through continuous automation. More than 40,000 organizations – including more than 80 percent of the Global 5000 – have benefited from Puppet's open source and commercial solutions to ensure business continuity, optimize costs, boost compliance and ensure security - all while accelerating the adoption of DevOps practices and delivery of self-service. Headquartered in Portland, Oregon, Puppet is a privately held company with offices in London, Belfast, Singapore, Sydney and Timișoara.

To learn more:

Visit: www.puppet.com

puppet