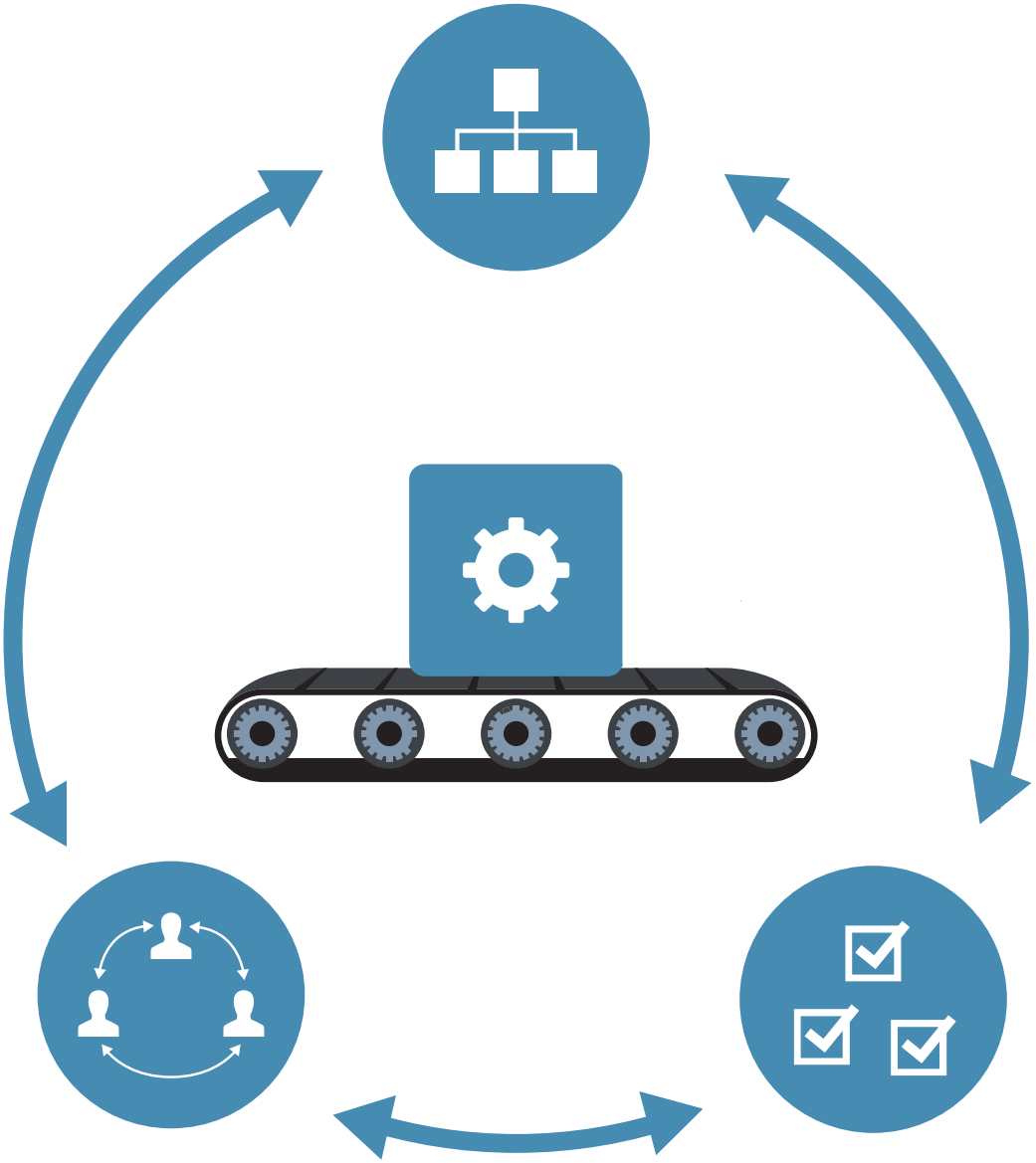
FREEFORM DYNAMICS

In association with

ca technologies



Requirements-driven software development and quality management

Removing the ambiguity from your delivery pipeline

Freeform Dynamics, 2018

Introduction

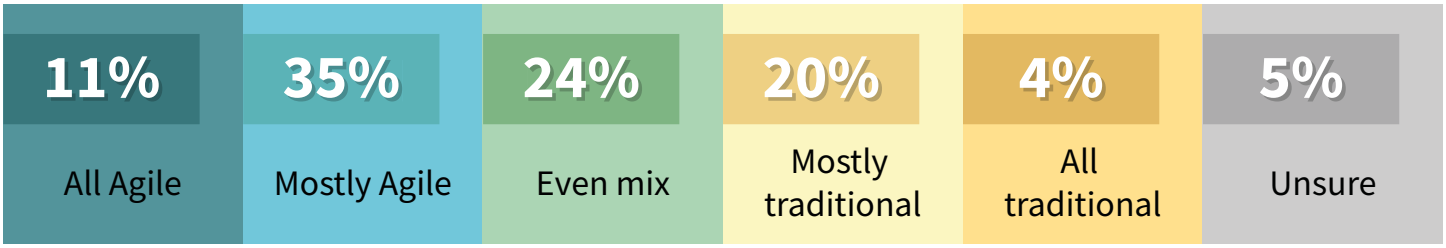
Putting the software quality discussion into perspective

Whatever the word 'digital' means to your organization, there's a good chance that it's on the strategic agenda in one form or another. Even if it isn't, the reality is that pretty much any significant business initiative nowadays needs to be supported by the right software. This invariably puts more pressure on application delivery teams.

Adding to the pressure, is the way in which business requirements and expectations have been evolving in relation to the delivery process itself. The success of many initiatives nowadays increasingly depends not just on a fast time-to-market, but a willingness to experiment, and continually optimize and enhance applications and services once they are in production. This is particularly true of customer engagement systems, but as a result of digital transformation, the pace of delivery and frequency of change is increasing across workforce-facing and B2B solutions as well.

Against this background, a shift is underway in many development teams from traditional delivery models to Agile methods. When implemented well, these support a more responsive and iterative way of working. This shift came through strongly in a recent online survey of 327 IT professionals conducted via The Register (www.theregister.co.uk).

How much of your development activity is based on Agile vs traditional methods (e.g. Waterfall)?



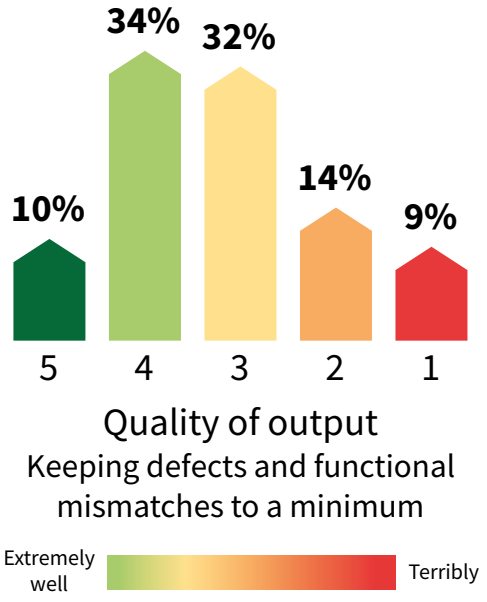
In the drive towards ever faster and more granular delivery cycles, however, it's important to ensure that speed and responsiveness don't come at the expense of quality. Application owners within the business might well take the view "Let's just get the application out there and we'll sort out any problems later". But whatever they say beforehand, if you work in development or delivery, you know it's you or your team that will be held responsible for any negative fallout that impacts the business as a result of a software issue.

Add these factors together - stakeholder impatience, expectations of faster delivery, more frequent changes and enhancements, and new approaches to working such as Agile and DevOps - and the big question is how well QA and testing practices are keeping up.

According to the research, the answer is that when it comes to maintaining quality of output, some are doing a lot better than others. As we can see on the graphic to the right, most report less than optimal performance with regard to keeping defects and functional mismatches to a minimum. Indeed a significant number seem to be struggling quite badly.

Wherever you are on this spectrum, the message is that it's important to pay attention to software QA as you make changes to your delivery processes in response to escalating demands. Even if you are doing well at the moment, there's no room for complacency.

How well does your organization perform in the following area?

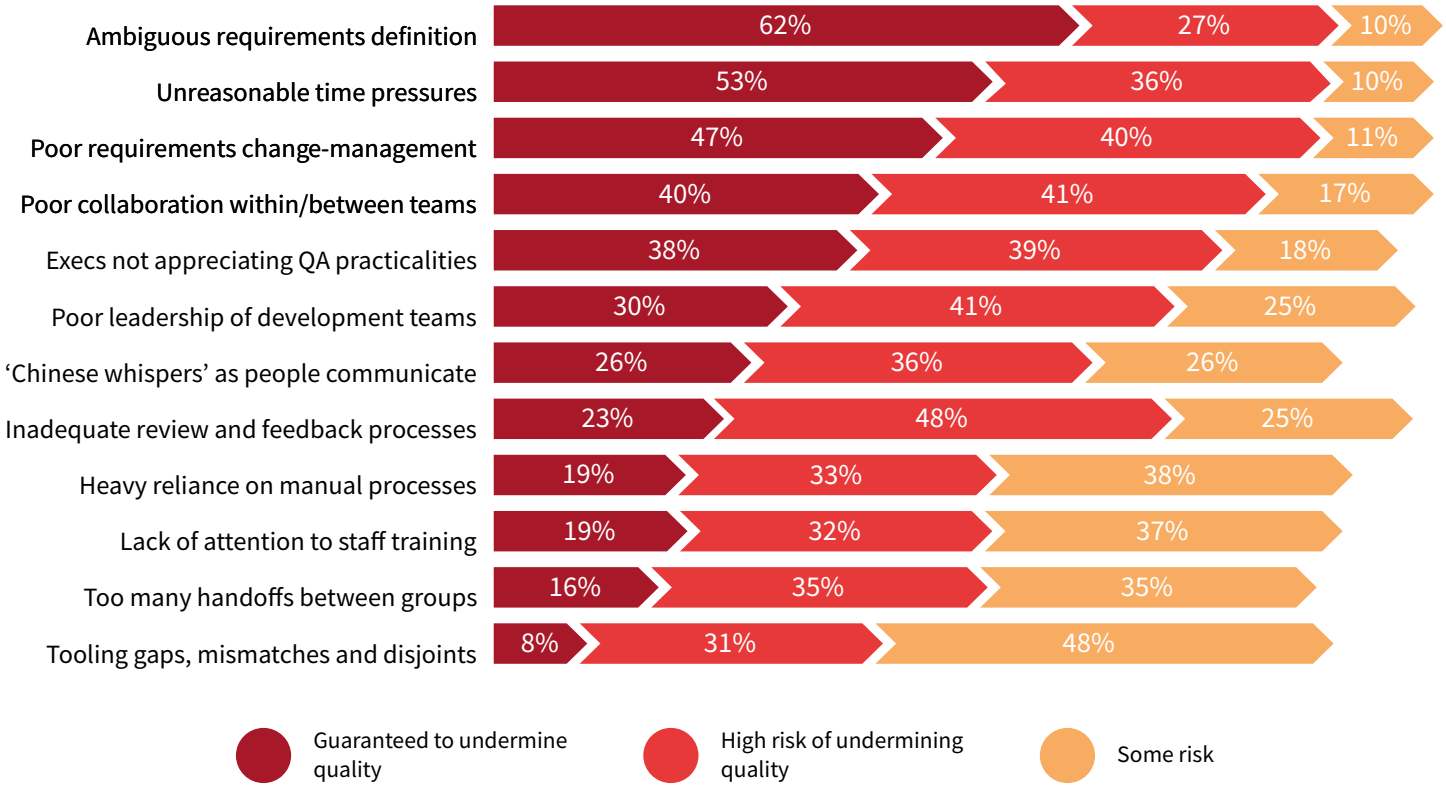


Getting off on the right foot

Quality starts at the requirements definition stage

If you are going to review and optimize your approach to software quality, or reassure yourself that existing processes aren't going to break as they come under increased pressure, you need a clear understanding of the risks you are trying to manage. And if you take a holistic view in this area, it quickly becomes clear that software quality can easily be undermined by a whole range of factors.

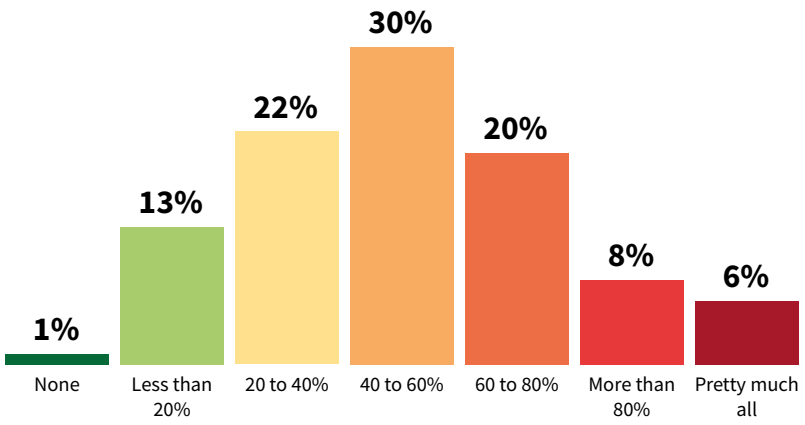
In your general experience over your career (not just in your current organization), how much of a risk are the following to software quality?



Much of what we see here will not come as a surprise, but the list provides a useful reminder of what you need to have covered if you are going to strengthen your footing in this area and drive sustainable improvement.

Most pertinent to our discussion in this paper, however, is the ranking of "Ambiguous requirements definition" at the top of the list, with "Poor requirements change-management" in third place. This tallies with the numbers on the right, which provide a feel for what this means in terms of impact. It also prompts us to take a closer look at the area frequently referred to as 'requirements engineering'.

What percentage of the software defects or functional mismatches are caused or aggravated by ambiguous or poorly managed requirements definition?

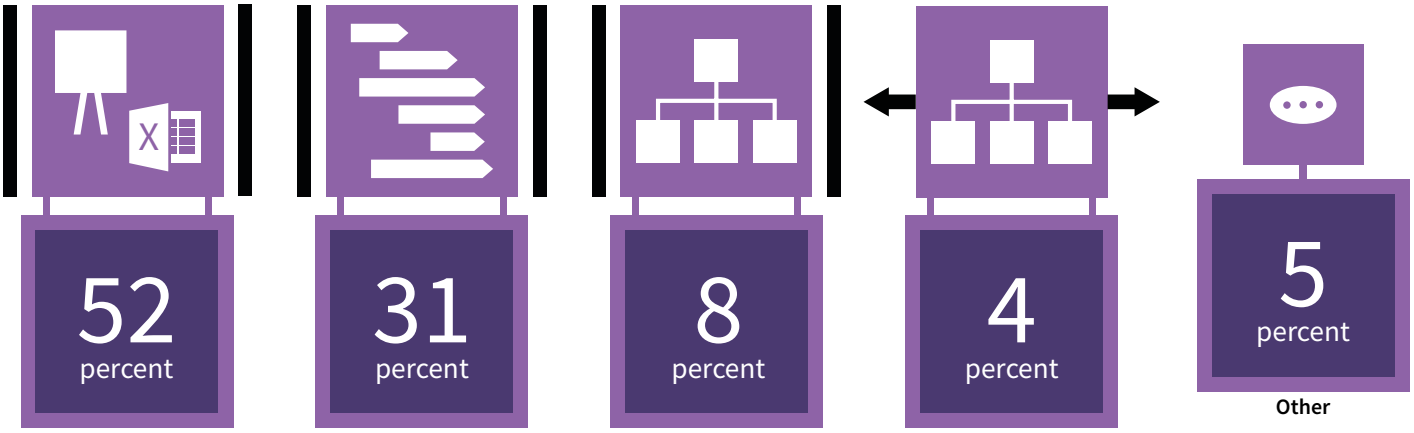


A closer look at requirements engineering

Eliminating ambiguity from the delivery process

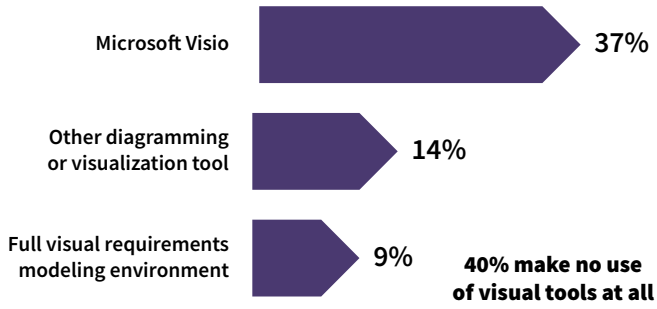
Requirements-ambiguity is one of those problems that lurks behind the scenes, continuously creating quality issues that often don't have an obvious cause. Once you recognize this, you can start to take appropriate action. In an ideal world, this would be done by defining and communicating requirements clearly up front, incorporating them into all aspects of delivery, and making sure subsequent changes are propagated across all relevant lifecycle activities. The trouble is that most don't have the necessary integrated model-driven approach in place to achieve this.

How would you sum up your approach to requirements gathering, analysis and engineering?

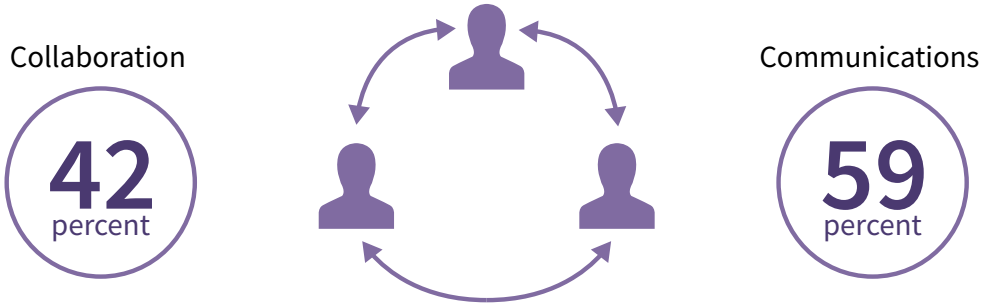


The unstructured, static and/or disconnected approaches you see cited here by over 80% of research participants are fine for traditional waterfall projects with a discrete up-front requirements phase, but they don't sit well with the shift to fast-moving Agile methods we saw earlier. Neither does the absence of visualization capability or the heavy reliance on static diagramming tools that's evident from the chart on the left. Again a big problem is managing change. It's hard to keep visual representations of requirements up-to-date, which in turn hampers the communication needed to support effective collaboration. With this in mind, many acknowledge that full visual modeling tools can help.

Do you currently use any visualization or diagramming tools to define/specify requirements in lieu of describing the requirements in text form?



Does visual modeling offer significant advantages in these areas?

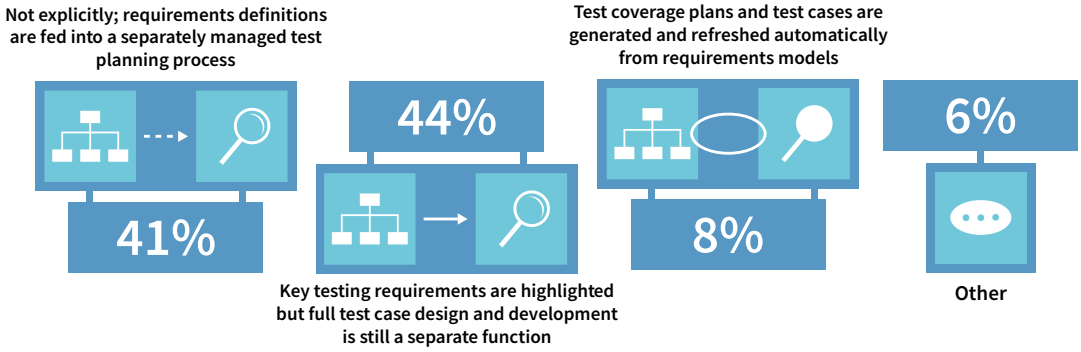


The link between requirements and testing

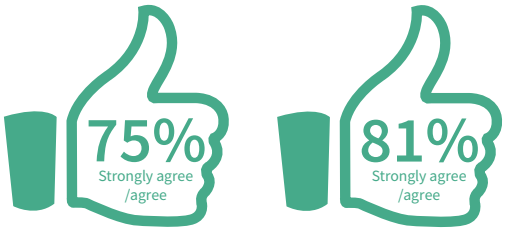
A question of coverage, automation and effective iteration

Apart from constraining communication and collaboration, static and disconnected approaches to requirements management also limit the degree to which you can integrate and automate across the delivery cycle. Coming back to our core theme of software quality, the obvious touch point relevant here is between requirements engineering and software testing. Ideally, requirements would flow through automatically to test case definition, with round-trip management to handle inevitable changes down the line. As we can see from the graphic on the right, however, this level of integration and automation is rarely in place. The overhead of manually translating requirements into test plans is clear, as is the scope for errors and inaccuracies to creep in.

To what degree is testing generally considered and/or incorporated into requirements definition and change-management?



The test case coverage objective

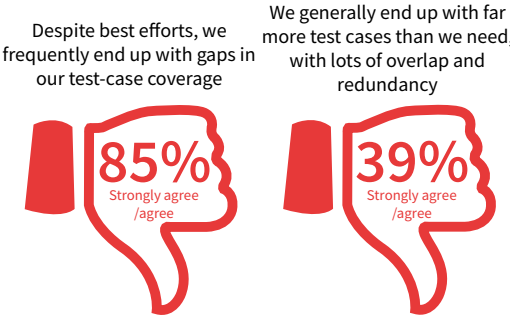


Another consideration when requirements engineering and the QA aspects of software delivery are handled separately is the challenge of optimizing test case coverage. The majority agree that for both efficiency and quality purposes, the aim is to achieve the maximum coverage with the minimum number of test cases.

With little or no integration, however, it's no surprise that many say they are struggling here, with most reporting coverage gaps. There's then the opposite problem of wasting resources on overlaps and test case redundancy.

The challenge

The outcome



And the impact of this lack of integration is also evident further downstream. Beyond the results we presented above, 53% of study participants said that at least 6 out of 10 test cases were still executed manually, with few (less than 2 in 10) saying they had pushed their reliance on manual testing below the 20% level.

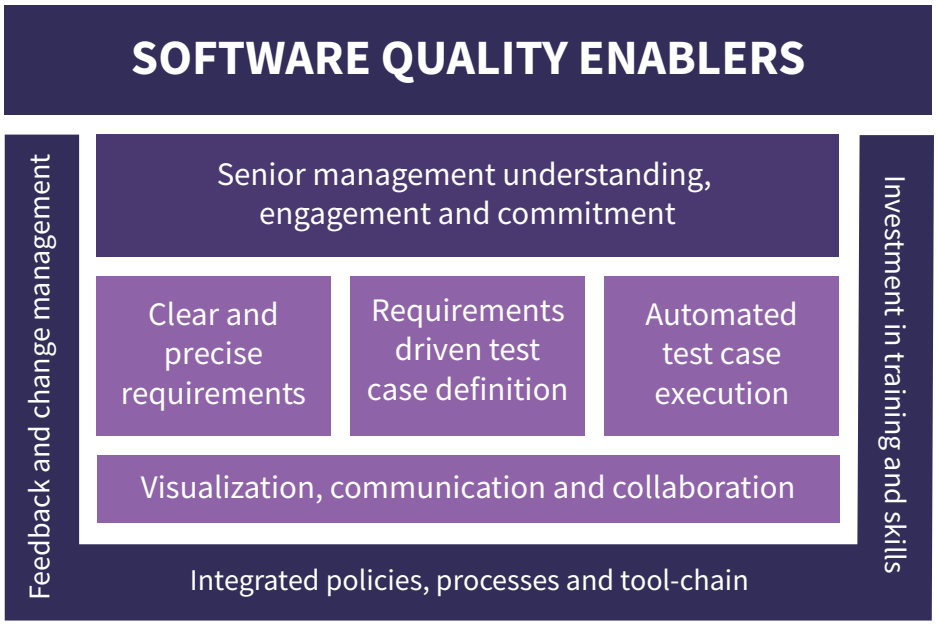
Driving improvements

Lessons from the research and further thoughts

So, there's clearly work to be done by most development and delivery teams to free themselves from the challenges of requirements ambiguity, and more generally to create a more responsive, effective delivery environment that can keep up with demands while maintaining quality. But what should you initially prioritize?

To a degree that depends on your environment, current situation, and the outcomes you are looking for. But when we asked an open question on this during the research, we found the same things being mentioned over and over again. Here's a quick summary of what we heard.

If you have been working in and around software and applications for any length of time, you might sigh at the fact that so much of what we see here is so familiar - they are the kind of things that have dogged delivery teams for many years. The trouble is that in a fast-moving Agile and/or DevOps environment, the impact of many of them is amplified significantly. If continuous change is not just a fact of life that you accept, but something you actively embrace and encourage as part of an experimental, fail-fast culture, then you had better be prepared for it. Fundamental to this is effective communication and understanding from top to bottom, and this is really the context in which requirements engineering needs to be considered today. Given this, and the results of the study, we therefore make no apology for the reminder about some basics in the graphic to the right. So, as you look to drive improvements in your own environment, remember that while modern tools and best-practices can make a big difference, success depends just as much on dealing with the human dimension.



About the Research

The research upon which this report is based was designed, executed and interpreted by Freeform Dynamics Ltd. Data was collected from 327 experienced IT and business professionals via an online survey hosted on a popular IT news site (www.theregister.co.uk). Respondents were from a cross section of industries and organization sizes, with a geographical focus on the UK and USA. The research was sponsored by CA.

About Freeform Dynamics

Freeform Dynamics is an IT industry analyst firm. Through our research and insights, we aim to help busy IT and business professionals get up to speed on the latest technology developments, and make better-informed investment decisions.

For more information, and access to our library of free research, visit www.freeformdynamics.com.

About CA Technologies

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

Learn more at www.ca.com.