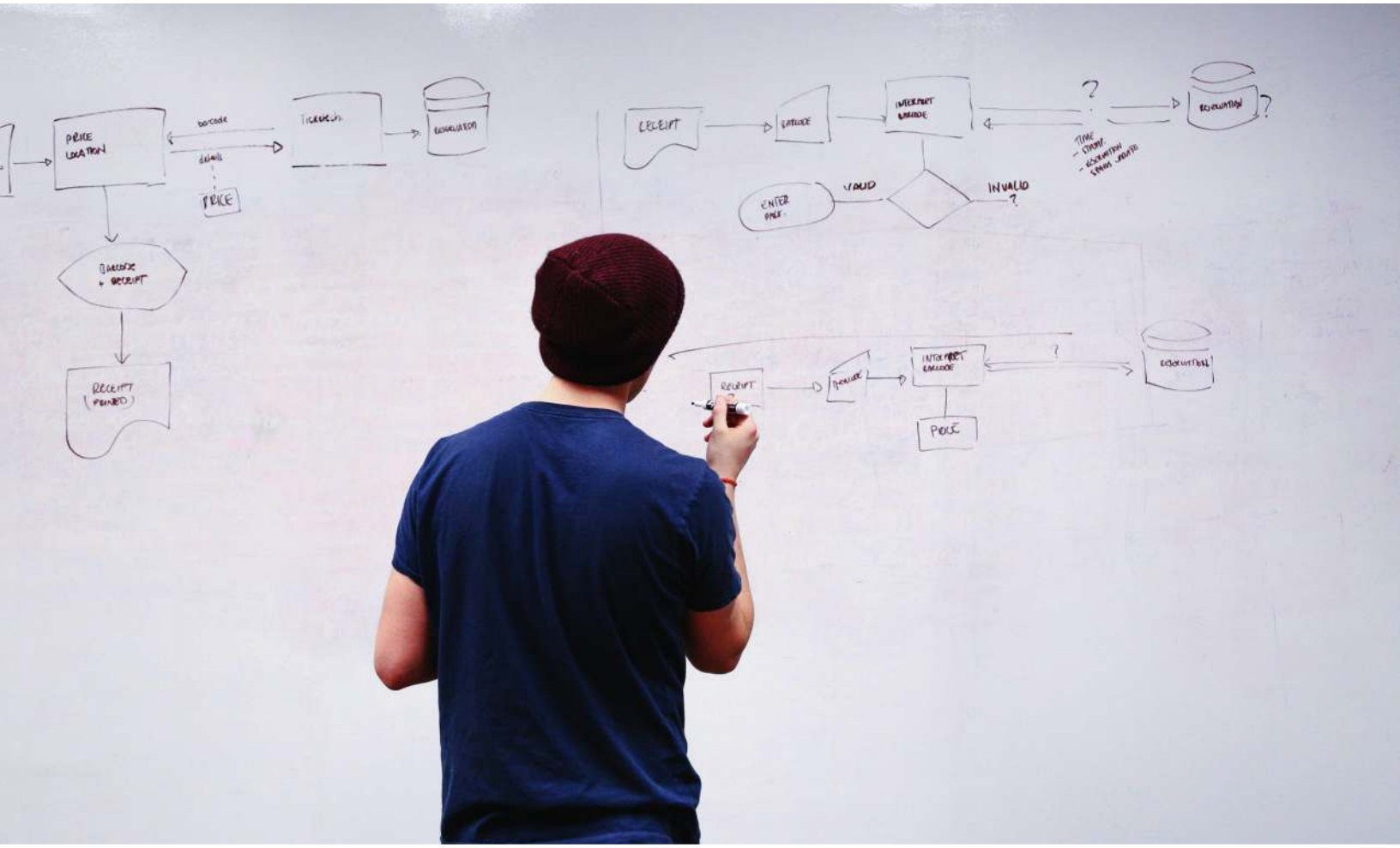
White Paper

Pitfalls of not Transitioning from Waterfall to Agile/DevOps

by Anuradha M



TRIGENT
OVERCOMING LIMITS

In a dynamic business environment, change is the only constant and organizations worldwide accept this fact. To cope with this evolving environment there is the need to constantly view and review existing strategies and policies. Work practices also keep changing to include evolving supplier, buyer, and partner relations and there is the constant need to innovate and reinvent existing business practices. This could involve technology innovations and upgradations. When it comes to technology innovation, Waterfall was considered to be the preferred methodology. This linear approach assumes that a project is finite with a defined beginning and ending and results, i.e. predictable. The fact is, though Waterfall is viewed as a manageable and cleaner approach to project management it may not be able to provide the agility required for today's complex and evolving business environment.

Understanding Waterfall Methodology

Waterfall follows a straight path and includes the following stages in project development:

- Requirements Gathering
- Designing
- Coding and unit testing
- System testing
- User Acceptance testing (UAT)
- Fixing issues
- Delivering the final product

Waterfall methodology views each of the above steps as a distinct stage and unique from the one before or after. For example, in a development environment using the Waterfall methodology, a requirements document would be put together, which would detail the project's requirement in order of priority. This stage could take from a few days to several weeks. A typical document would include detailed requirements, user scenarios and functionality layouts.

During the next stage, i.e. analysis, the engineering team would ask questions, update the document and launch the design face which would include system design, interface design, mock ups etc.

The actual implementation is the next stage where the engineering team develops functionality and prepares it for testing.

Forrester recently published a research on the rapidly evolving market of tools used to manage software requirements. The report assessed the maturity of ten categories of the technologies within software requirements management tool ecosystem based on user, vendor, industry expert interviews and product demonstrations. According to this report, 'traditional requirements practices are dead'. The report further states that 'Requirements errors are often the root cause of expensive rework'.

To elaborate on this, the efficacy of requirements gathering is always a challenge. Customers can be intimidated by the extent of detail required leading to ambiguity. To ensure that requirement gathering is fool-proof, some development teams put together wireframes and mock-ups but in spite of this, the final product may still fall short of customer expectations. The question which arises is, 'what if some changes are required to be made to the design during the implementation phase?' Since projects during requirements gathering are ambiguous, the chances of the end product not completely matching business requirements cannot be ruled out. Changes, if made, can be costly and difficult, disrupting schedules and pushing the deadline.

Understanding Agile Methodology

Agile management in comparison to Waterfall, begins from the hypothesis that we really don't know everything at the beginning of a project. Even if we are willing to assume that we know a lot, there is still the need to be ready for changes which could happen at any stage of a project's development. Agile, in that sense, safeguards the customer but more importantly it safeguards the developers by getting the customer involved through all stages of development. It helps to identify and squash bugs early and often, bringing down iteration time considerably.

The Agile methodology is iterative in nature and a team-based approach to development. It focuses strongly on the rapid delivery of an application in complete functional components. Instead of creating tasks and schedules, the project is boxed

into sprints which occur in defined durations. Each sprint would ideally have a running list of deliverables which are determined by the client. Agile relies on a very high level of customer involvement throughout the project, especially during reviews.

Agile's strength lies in its strong dependency on customer involvement. As a result, the customer feels a sense of ownership and works extensively with the development team to see the project to closure. The transparent nature of the approach ensures that the customer constantly sees what is going on and has the option to go with functional releases instead of waiting for completion of the full development cycle. Development is user-focused, likely a result of more and frequent direction from the customer.

As a result of its customer-centric approach, Agile makes the planning and designing stages straightforward. Progress is easily measured and the full scope of the work predictable. During development, multiple teams work together. For example, business analysts can learn document requirements on-the-go, and testers can prepare test scripts even while coding is in progress. Since design is completed early in the development lifecycle, Agile lends itself to projects where multiple software components must be designed (sometimes in parallel) for integration with external systems.

Understanding DevOps Methodology

DevOps is an abbreviation of Development and Operations. Its inherent strength lies in its ability to decrease IT operational costs while improving software quality and accelerating time to market. A recent survey commissioned by a leading technology provider reveals an average of 19% revenue increase directly attributed to the adoption of DevOps methodologies. A drill down to the most common financial attributes of DevOps adoption provides more clarity.

Automating the software delivery process benefits the organization financially in multiple value areas:

- Revenue gains from enhanced developers productivity and reduction of IT headcount waste
- Revenue gains from accelerated time to market of new functionality
- Gains from cost reduction of application failures resulting from increased quality
- Gains from flexibility in the IT environment

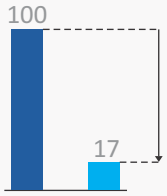
The value of adopting Devops can be significant

Indexed to 100

Pretransformation Posttransformation

Improvement in time to market

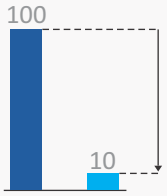
Average number of days from code completion to live production



- ✓ Eliminate rework through integrated change management and automated deployment and testing

Reduction in cycle time

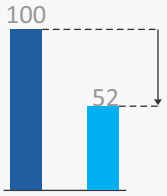
Number of days to update servers and the IT environment



- ✓ Eliminate wait time and rework through standardized processes
- ✓ Eliminate no-value-added work through automation

Improvement in productivity

Average number of DevOps handoffs per processing activity



- ✓ Eliminate wait time and rework through improved development and operations communication

Source: McKinsey analysis

To understand how DevOps benefits an organization requires us to delve a little deeper into the subject.

As we all know, applications are a key driver to the growth and success of today's businesses. Visibility into application performance and the efficiency of the software development lifecycle reaches far beyond IT and well into business stakeholders. Applications are complex and fast evolving, and with the expansion of mobile and web consumption demanding flexibility to handle constant requirement changes effectively, businesses are forced to reassess their application delivery strategies. Coping with the need for change requires more than a strategy! It calls for an ever-expanding budget without clarity on the financial benefits of adopting agile innovations.

New software development methodologies have emerged to address the need for agility starting from development practices to full automation of the software release process. This collection of best practices has matured into the continuous delivery (CD) process, which applies industrialization concepts to software. Designed to streamline and accelerate software delivery while ensuring that reliable software is released, CD creates an alignment between the application and changing business needs.

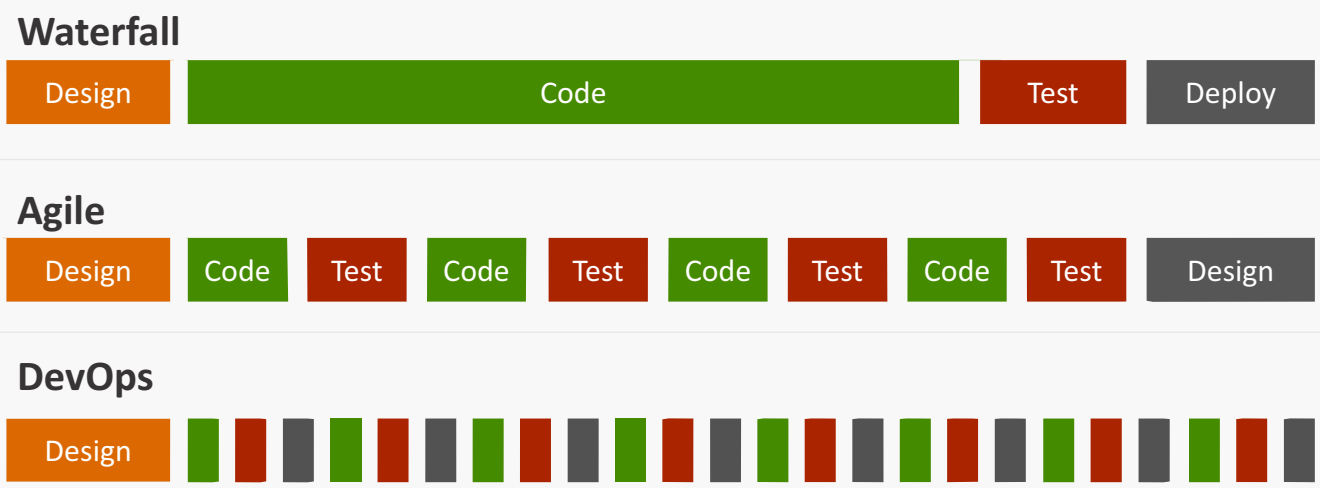
The principles of CD find their roots in the DevOps movement, established to bridge traditional gaps between software development and the operations team running applications. DevOps automates manual testing and release processes by replacing them with scripted procedures. It extends Agile development in building applications incrementally to include the full integration, testing, and validation phases. By streamlining application delivery across the entire development lifecycle, applications can be iteratively developed, automatically packaged and tested, and then released to production in a continuous, rapid, consistent manner. The CD archetype establishes the notion of an ever-evolving production-ready version of the application in a functional, deployable state throughout its lifetime.

Making the Choice between Agile, Waterfall and DevOps

1

More often than not, when discussing methodologies, companies focus on the advantages and disadvantages of the available options. However, what is required is a shift in the thinking process, a change in culture, and a new approach based on business success rather than methodology. A forward-thinking organization will not really see the shift from Waterfall to Agile and Agile to DevOps as a dramatic shift.

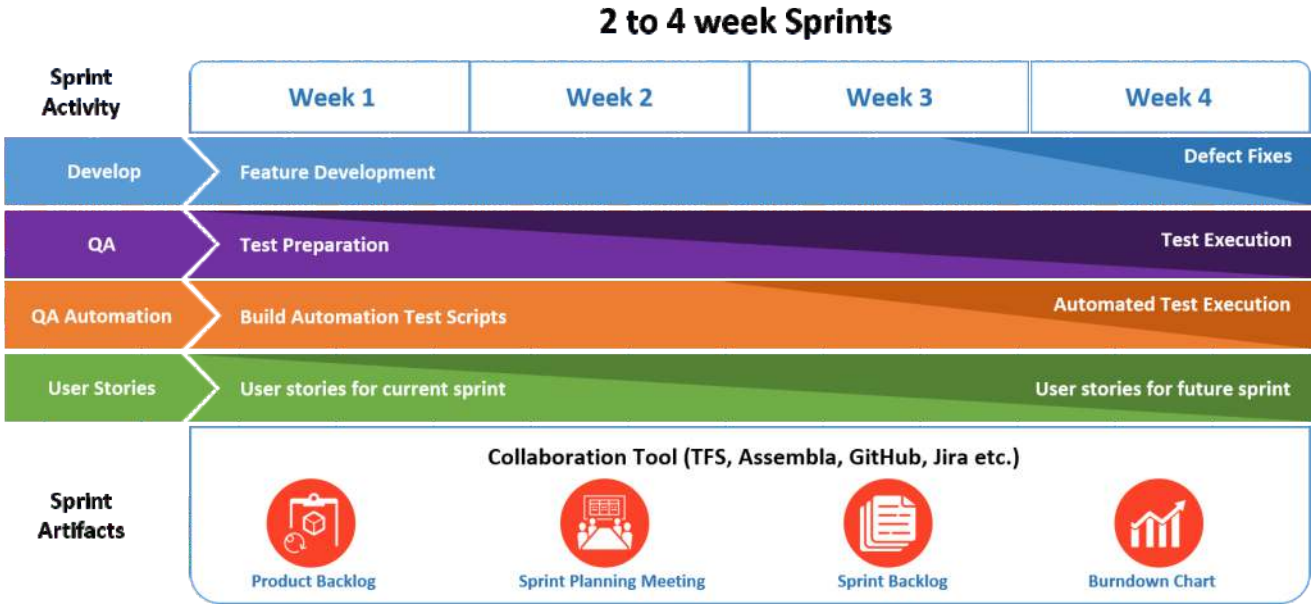
Project Execution Methodologies - The Change



It would be a natural outcome of a cohesive environment where multiple stakeholders work together with an end goal, i.e. successful product launch, as their only aim. It does away with looking for reasons for failure and focuses on what works best to get the work done fast. This kind of cultural change is what defines the methodology. To put this differently, in a culturally evolved organization, even if the team is aware of the challenges associated with the Waterfall approach, they might prefer it because it could be the best option. They might choose this approach simply because the customer does not want to be involved during the development phase of a project for various reasons. There could be several other reasons but what is important is the cultural makeup of an organization. Even mature organizations can run into problems with projects because of their inflexible approach to projects. Overall, we can comfortably say that the culture of agile embraces the aspect of continuous delivery and integration. Continuous delivery accelerates the process of coding into building and into testing and deployment.

2

Users and customers are demanding improved software solutions at a pace that is equal to many of the cloud services that they are familiar outside their work. This strong desire for frequent updates at increased speed has forced enterprises and software companies to adapt to Agile and DevOps. Many mobile and cloud development platforms (PaaS) makes Agile/DevOps methodology as the preferred way of working.



One of the top DevOps adoption drivers is the need to increase quality, followed by the need to improve the customer experience, reduce complexity, and overall IT costs.

In order for Agile/DevOps to be successful, quality initiative must be made an integral part of the methodology. Specifically, QA roles need to be embedded into the Agile/DevOps team from the get-go; QA automation must be aligned with build/deployment automation. Continuous Integration (CI) and Continuous Delivery (CD) practices strive to make small changes, integrate them frequently and ensure fast feedback. This results in fewer failures and faster recovery from failures. But mature teams go beyond CI/CD, to a Continuous QA through automated testing. Continuous QA and automated testing begin with a comprehensive framework that defines the process, scope, and tools.

3 In the traditional method, releasing a product into production can be a high risk proposition. It could bring about unexpected issues, escalations and non-stop fire-fighting. It is in fact not uncommon to see IT set up war rooms for weeks before and after a software release. DevOps, converts high risk event into a series of non-events. By daily integration of code, automation testing and ensuring that everything is in sync, reduces risk drastically. The only code which is promoted from one stage to the next is the one which the team is confident that it can go into production. This ensures drama-free events.

4 Both types of organizations generate and share a ton of data. The difference lies in how the teams uses data. In traditional IT, information is generated by specialists (e.g. operations team), bundled together with other data into a massive report, goes through management approval, is then shared with other managers, who then sends it out to their specialist (testers, developers etc.). In most cases the report goes unread, simply because there is too much data, not timely enough, and/or not the right data. In other words, information is shared but poorly consumed, and rarely used to take any actions.

5 Within a DevOps organization, it's the team cell that gather and creates the data. Since the data is to be consumed locally they only collect the data (also automated) that the team deems is necessary. And since the data is processed within the team, it eliminates the time lag of creating lengthy reports, manager approvals, and queue time (sitting in some ones email box). As a result, the team is quickly able to read and react to the data – resulting in faster feedback time.

6 Traditional organizations hate risk and the CIO's first priority is to protect the organization from any kind of harm. This is also one of the reasons why IT departments follow some old-fashioned processes of approvals etc. All this is to prevent failure and yet despite all the precautions, IT's track record speaks volumes. Over 30% of new projects are delivered late, and 50% of all new enhancements are rolled back due to quality issues. Over 40% projects are delayed by infrastructure issues. DevOps organizations also hate risk but they understand that it is difficult to eliminate risk. They take small bites ensuring that failures are also small and recovery is fast.

7 Traditional IT structures are created around silos and tied to cost and capacity. IT is measured on how much is done and how cost effective the approach was. Cutting costs has been its mantra all along. In the new age, applications along with cost focus on the element of time. DevOps organizations understand this paradigm shift and they include ‘flow’ as an additional metric. Flow forces an organization to take a look at its end to end cycle time, identify areas of waste, calculate true productive time, quantify quality, and focus on activities that add the most value.

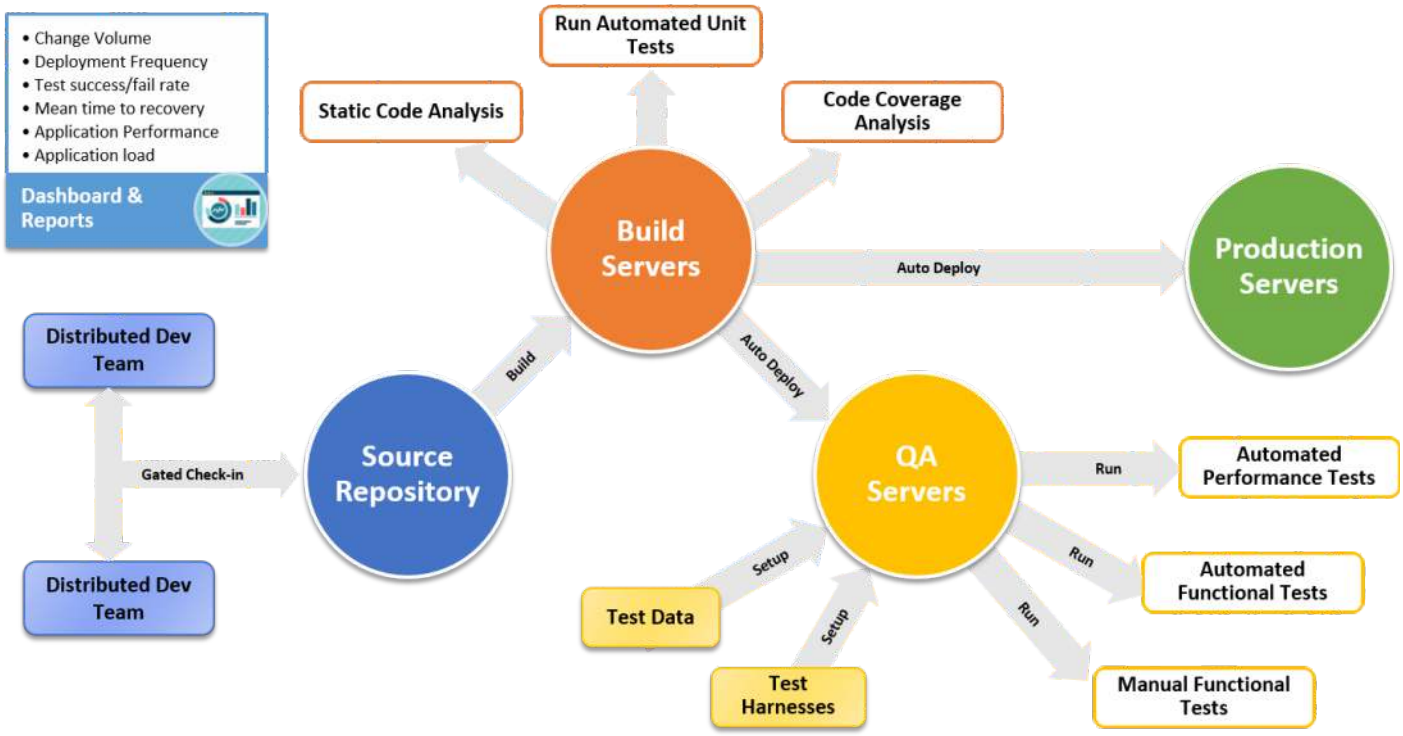
To summarize, it is best to define the process, with prototyping to provide the customer a better view of the finished product early in the cycle. This helps to improve requirements and customer communication. After this, the iterative approach offered by Agile or DevOps will ensure that the overall project has no unpleasant surprises at the end.

How Trigent can Help

Trigent's DevOps solution accelerates the software delivery process by providing a comprehensive Continuous Delivery environment. It leverages your existing ALM tool investments, uses industry best practices and provides real-time dashboards. It offers a self-service mechanism for Dev - Test provisioning either in cloud or hybrid environment.

Our DevOps and Continuous Delivery teams can help you:

- Improve speed and quality of software delivery
- Provide constant visibility during the development cycle
- Reduce operational risks
- Support you with automation, tools, and processes for a lean, integrated, predictive and automated process
- Build a culture and mindset for collaboration between developers and operations that supports DevOps



About Trigent

Trigent is a technology solutions company that provides comprehensive solutions for business problems via outsourced software product and applications design, development and quality assurance. Trigent serves customers like Independent Software Vendors (ISVs), enterprises and SMBs in the High Tech, Healthcare, Education, Ecommerce and Manufacturing areas. Trigent's solutions help clients overcome budget, schedule and resource constraints.
To learn more about Trigent visit www.trigent.com

Our Headquarters

Trigent Software Inc.
2 Willow Street, Suite #201
Southborough, MA 01745
+1 (877) 387-4436
sales@trigent.com

Other Location

Trigent Software Ltd.
Khanija Bhavan, 1st Floor
#49, West Wing, Race Course Road
Bangalore - 560001, India
+91 (80) 2226-3000

CERTIFIED ISO 9001:2008 COMPANY

Microsoft Partner
Gold Application Development
Gold Collaboration and Content

amazon web services | Partner Network
CONSULTING PARTNER

2016 SOFTWARE 500 COMPANIES

2017 IAOP The Global Outsourcing 100

TOP 10 SiCLOUD SOLUTION PROVIDERS - 2017

TRIGENT
OVERCOMING LIMITS