Politecnico di Torino

Department of Management and Production Engineering

Master's Degree: Management Engineering

Academic year 2021/2022

Master Thesis:

A comparison between agile(incremental) , non agile(predictive) and DevOps methods in global software development

Politecnico di Torino

PROFESSOR:

DEMAGISTRIS PAOLO EUGENIO

GRADUATE STUDENT:

JAKHONGIR SALOKHIDINOV

Table of Content

Abstract

Nowadays more and more software engineering related projects are being carried out using a global software development model in order to take advantage of the global talent pool at a cheaper cost and deliver a product in a short period of time. To this end, project managers are trying to adopt different methods to achieve their goals. Incremental, predictive and DevOps methodologies have been proven to deliver great success but, in distributed development model, some challenges as well. The aim of this study is to present a comparison between the predictive, which is the pioneer method and “father” of all kinds of project management methodologies, incremental, which emerged as an alternative to situations where traditional plan-driven methods are insufficient and have gained a very important place today, and DevOps ,a method that incorporates agile principles and practices but also emphasises improved collaboration between development and operation teams. This study intends to identify the project management characteristics that are associated with greater project success in global software development settings.

1.Introduction

In GSD(Global Software Development) model the development of products is carried out in globally distributed settings in various geographical locations. Predictive, incremental and DevOps methods are used to manage such projects. In global software development projects the main issues are communication barrier between the onsite and offsite teams and failure of requirement gathering which results in the low-quality output.

Adopting the proper method for the the project implementation will be done by the project manager.

Predictive methods have long been in use and proven to be a reliable form of project management. However, they demonstrate a low performance in projects where adaptation to changing market needs and customer demands is key. And at present day in software development market without adaptability failure is guaranteed.

Since the advantages of the agile methodologies in the situations above are well-known, application of them also in the global software development projects can have fewer investment costs and time.

As agile methods became a de facto industry standard, however, it became clear that agile practices leave out operations teams that deploy and manage the product. This gave rise to DevOps, an approach that aligned development and operations teams.

This thesis intends to demonstrate advantages and disadvantages of using agile and non agile and DevOps methodologies and offer suggestions which methods are better to be used in distributed development.

The thesis will be developed using scholarly articles, research papers and data obtained through survey. Project managers, HR managers and companies that want to deliver successful projects in the GSD model might benefit from this paper. Main stakeholders in this paper are individuals working in GSD with background in project management, software engineering and quality assurance.

2.Literature

In this section a brief overview of project management, software development, agile, non-agile and DevOps methods will be discussed.

2.1 Project Management

Project management has been practiced for thousands of years since the Egyptian era, however, it has been about half a century ago that organizations started applying systematic project management tools and techniques to complex projects. In the 1950s, Navy employed modern project management methodologies in their Polaris project. During the 1960s and 1970s, the Department of Defense, NASA, and large engineering and construction companies utilized project management principles and tools to manage large budget, schedule-driven projects. In the 1980s, manufacturing and software development sectors started to adopt and implement sophisticated project management practices. By the 1990s, the project management theories, tools, and techniques were widely received by different industries and organizations (Carayannis, 2003). As numerous project-oriented organizations began to form, a demand for project management as a profession emerged. At the same time professional associations in the field of project management, particularly the Project Management Institute, PMI, which was established in 1969 with the mission to share professionalism and establish a body of knowledge in the field of project management.

PMI defines a project as a temporary endeavor undertaken to create a unique product, service, or result (PMI, 2017).

Alternatively, a project can be defined as a series of tasks that need to be completed to reach a specific outcome. A project can also be defined as a set of inputs and outputs required to achieve a particular goal. Projects can range from simple to complex and can be managed by one person or a hundred. Projects are often described and delegated by a manager or executive. They go over their expectations and goals, and it's up to the team to manage logistics and execute the project on time. Sometimes deadlines can be given. For good project productivity, some teams break it up into individual tasks to manage accountability and utilize team strengths (Wrike, What Is a Project in Project Management?, 2022).

All projects are a temporary effort to create value through a unique product, service or result. All projects have a beginning and an end. They have a team, a budget, a schedule and a set of expectations the team needs to meet. Each project is unique and differs from routine operations—the ongoing activities of an organization—because projects reach a conclusion once the goal is achieved. Project management is the use of specific knowledge, skills, tools and techniques to deliver something of value to people. The development of software for an improved business process, the construction of a building, the relief effort after a natural disaster, the expansion of sales into a new geographic market—these are all examples of projects (PMI, What is Project Management? , 2022)

Many types of project management have been developed to meet the specific needs of certain industries or types of projects. They include Waterfall project management, Agile project management, Lean project management.

2.2 Software Development

Project management has seen a great adaptation and growth in the field of software development. A goal of any software project management is to develop and maintain a software product by applying good project management principles as well as software engineering principles so that the software project is delivered at minimum cost, within minimum time, and with good product quality (Ahmed, 2012).

Software development refers to a set of computer science activities dedicated to the process of creating, designing, deploying and supporting software. Software itself is the set of instructions or programs that tell a computer what to do. It is independent of hardware and makes computers programmable (IBM, 2022).There are several different types of software development. They can be grouped into four basic categories:

1.Application development that provides functionality for users to perform tasks. Examples include office productivity suites, media players, social media tools, and booking systems. Applications can run on the user's own personal computing equipment or on servers hosted in the cloud or by an internal IT department. Media streaming development is one example of application development for the cloud.

2.System software development to provide the core functions such as operating systems, storage systems, databases, networks, and hardware management.

3.Development tools that provide software developers with the tools to do their job, including code editors, compilers, linkers, debuggers, and test harnesses.

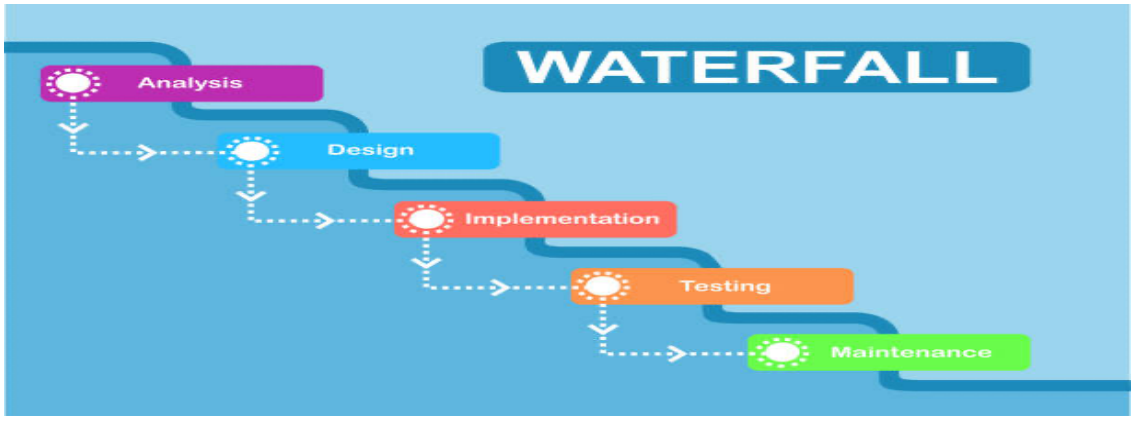
4.Embedded software development that creates the software used to control machines and devices, including automobiles, phones, and robots. The software that is developed can remain proprietary within the organization, or be sold to others. It can also be made freely and widely available to anybody that wants to use it; this is called open source development. (ITchronicles, 2022). In software

development, Software Development Life Cycle is the application of standard business practices to building software applications. It's typically divided into six to eight steps: Planning, Requirements, Design, Build, Document, Test, Deploy, Maintain. Some project managers will combine, split, or omit steps, depending on the project's scope. These are the core components recommended for all software development projects (Jevtic, 2022).

2.3 Waterfall

Waterfall project management is one of the more traditional software development methodologies. It follows a linear, sequential design approach where progress flows downwards in one direction, like a waterfall. A project is delivered through a set of ordered stages and until all activity within the current stage has been completed and approved, advancing to the next stage or any later stages is not possible. The Waterfall methodology has its origins within the manufacturing and construction industries, to which you could ascribe its stringent process. Due to the structured physical environments, the process leaves little room for changes as any changes made would result in high costs. It was first formally introduced as a method for software development in an article written by Winston W. Royce in 1970, however, the term "Waterfall" wasn't used. It wouldn't be until 1976 when a paper was written by T.E. Bell and T.A. Thayer that the term may have been first used. (Asmo, 2019)

The Waterfall methodology follows a chronological process and works based on fixed dates, requirements, and outcomes. With this method, the individual execution teams aren't required to be in constant communication and, unless specific integrations are required, are usually self-contained. Team members also tend to work independently and aren't expected to provide status reports as often as with the Agile approach. Usually, one phase doesn't begin until the previous one is finished. (Workfront, 2022)



Although the Waterfall methodology was initially developed for hardware manufacturing, it has proven to be one of the most consistently popular methodologies for software development projects. This method is a progress-based method that develops sequential steps to complete a particular project. By using the Waterfall method, teams will see a project to completion in several different phases – starting from the ideation phase to the implementation and maintenance. Here are **several** pros of Waterfall Project Management for remote teams:

- The Waterfall method is a good option for remote teams because of its insistence on clear documentation. Since communication is one of the major challenges of remote tech teams, documentation can go a long way in ensuring that everyone is on the same page.

- The Waterfall method is easy for remote tech teams to follow because it lays down a straightforward and logical sequence for every project that the team undertakes.

- Every member of the remote team must provide each other with status updates and reviews at the end of each phase.

- The predictable timelines of this project management method make it extremely simple for the remote team to stay on top of deadlines and other requirements.

However, there are also several major cons of Waterfall Project Management:

- One of the biggest cons of Waterfall project management is that it is relatively inflexible compared to other methods. It means that the team can't incorporate any changes to the project that might be necessary for the future.

- The Waterfall method also does not involve enough client participation. It usually incorporates only two meetings with the client – one at the beginning of the project and the other at the end.

- This project management method is also much slower in terms of delivery times. The team absolutely cannot move forward to the next phase without completing the previous one.

- Mistakes and changes can prove to be a costly affair in projects that have been developed using the Waterfall method, mainly because they require the team to go back several steps to fix the problem (Waterfall Vs. Agile: Which Is the Best Choice For the Remote Tech Team? , 2021).

2.4 Agile

Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. Agile methods or Agile processes generally promote a disciplined

project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals. Agile development refers to any development process that is aligned with the concepts of the Agile Manifesto (CPRIME, 2022).

Agile was formally launched in 2001, when 17 technologists drafted the Agile Manifesto. They wrote four major principles for agile project management, intended to guide teams on developing better software:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

What all agile frameworks have in common is that they continuously iterate on the work process itself and aim to deliver value to customers quickly and frequently. But there are now countless frameworks and each one offers its own flavor of agile. (Parabol, 2022)

Examples for Agile frameworks for teams are Scrum, Kanban, Scrumban, Extreme Programming (XP).

There are also Scaled Agile frameworks for organizations, for example, SAFe (Scaled Agile Framework), LeSS (Large Scale Scrum), DA (Disciplined Agile), Scrum of Scrums (Scrum @ Scale), Spotify Model. Every party using agile can benefit from it immensely.

- Customers find that the vendor is more responsive to development requests. High-value features are developed and delivered more quickly with short cycles, than with the longer cycles favored by classic “waterfall” processes.

- Vendors reduce wastage by focusing development effort on high-value features, and reduce time-to-market relative to waterfall processes due to decreased overhead and increased efficiency. Improved customer satisfaction translates to better customer retention and more positive customer references.
- Team members enjoy development work, and like to see their work used and valued. Agile benefits Team members by reducing non-productive work (e.g., writing specifications or other artifacts that no one uses), and giving them more time to do the work they enjoy. Team members also know their work is valued, because requirements are chosen to maximize value to customers.
- Product Managers, who typically fill the Product Owner role, are responsible for making customers happy by ensuring that development work is aligned with customer needs. Agile makes this alignment easier by providing frequent opportunities to re-prioritize work, to ensure maximum delivery of value (CPRIME, 2022).

Agile development was originally imagined for clustered teams, or teams physically located together in the same office. In keeping with the idea that "the most efficient and effective method of conveying information to and within a development team is face-to-face conversation", early agile teams were meant to work together in close proximity.

But today most businesses have a few–or several–distributed teams. This isn't just a trend; it makes good sense. Distributed teams can work on projects around the clock, and strong talent can be found in less competitive markets. But the benefits of distributed teams aren't without some trade-offs. For many distributed teams, it's difficult to adopt the agile practice of face-to-face interactions .

Other challenges that arise for distributed software teams:

- Coordinating across time zones

- Building rapport when everyone is not in the same office
- Collaborating among different development cultures
- Scheduling meetings or informal conversations when both teams are online at the same time for only a few hours

Good software architecture dictates modular design, so team structuring should be the same way. Every office should be self-sufficient in developing a single piece of technology, which minimizes the amount of collaboration required with teams in other time zones and makes them generally autonomous. When a project does require teams in different locations to pitch in, they can focus on their integration points and APIs (Radigan, 2022).

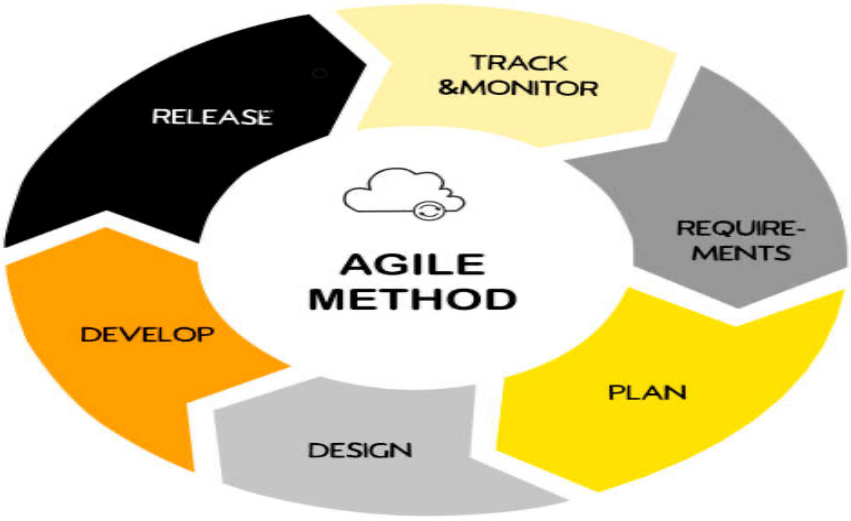


Figure 2. Agile Methodology (Volchenkova, 2022)

2.5 DevOps

DevOps is a set of practices, tools, and a cultural philosophy that automate and integrate the processes between software development and IT teams. It emphasizes team empowerment, cross-team communication and collaboration, and technology automation. The DevOps movement began around 2007 when the software development and IT operations communities raised concerns about the traditional software development model, where developers who wrote code worked apart from operations who deployed and supported the code. The term DevOps, a combination of the words development and operations, reflects the process of integrating these disciplines into one, continuous process.

A DevOps team includes developers and IT operations working collaboratively throughout the product lifecycle, in order to increase the speed and quality of software deployment. It's a new way of working, a cultural shift, that has significant implications for teams and the organizations they work for.Under a DevOps model, development and operations teams are no longer "siloed." Sometimes, these two teams merge into a single team where the engineers work across the entire application lifecycle — from development and test to deployment and operations — and have a range of multidisciplinary skills. DevOps teams use tools to automate and accelerate processes, which helps to increase reliability (Atlassian, 2022). DevOps consists of various stages such as continuous development, continuous integration, continuous testing, continuous deployment, and continuous monitoring (Arvind, 2022).

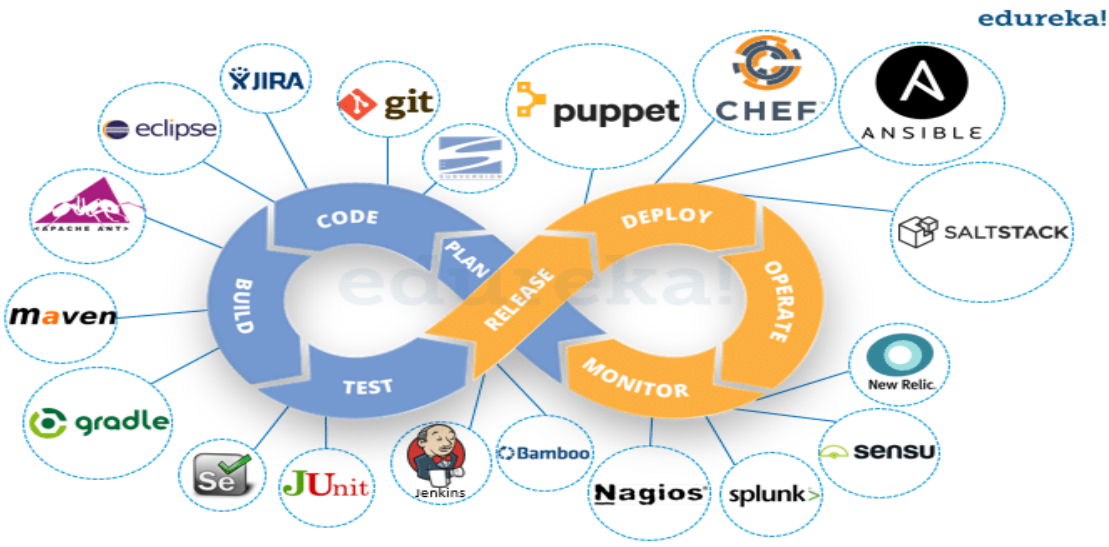


Figure 3. DevOps (Edureka, 2022)

2.5.1 Continuous Development

This is the phase that involves ‘planning‘ and ‘coding‘ of the software. The vision of the project is decided during the planning phase and the developers begin developing the code for the application. There are no DevOps tools that are required for planning, but there are a number of tools for maintaining the code.

The code can be written in any language, but it is maintained by using Version Control tools. Maintaining the code is referred to as Source Code Management. The most popular tools used are Git, SVN, Mercurial, CVS, and JIRA. Also tools like Ant, Maven, Gradle can be used in this phase for building/ packaging the code into an executable file that can be forwarded to any of the next phases.

2.5.2 Continuous Testing

Continuous Testing is a software testing type in which the product is evaluated early, often, and throughout the entire Continuous Delivery (CD) process. Continuous testing uses automated tests to ensure teams receive immediate feedback to quickly mitigate as many risks as possible throughout the software development lifecycle. Moreover, team members are able to continuously learn about their product and what can be done to increase quality and reliability (Katalon, 2022).

Following are the benefits of Continuous Testing:

- Find errors: Ensure as many errors are found before being released to production
- Test early and often: Tested throughout the development, delivery, testing, and deployment cycles
- Accelerate testing: Run parallel performance tests to increase testing execution speed
- Earn customer loyalty: Accomplish continuous improvement and quality
- Automation: Automate your test cases to decrease time spent testing
- Increase release rate: Speed up delivery to production and release faster
- Reduce business risks: Assess potential problems before they become an actual problem
- DevOps: Incorporates into your DevOps processes smoothly
- Communication transparency: Eliminate silos between the development, testing, and operations teams (Katalon, 2022)

For Continuous testing, automation testing tools like **Selenium**, **TestNG**, **JUnit**, etc are used. These tools allow QAs to test multiple code bases thoroughly in parallel to ensure that there are no flaws in the functionality. Moreover, Docker Containers can be used for simulating the test environment (Arvind, 2022).

2.5.3 Continuous Integration

Continuous integration (CI) is a software development practice in which developers merge their changes to the main branch many times per day. Each merge triggers an automated code build and test sequence, which ideally runs in less than 10 minutes. A successful CI build may lead to further stages of continuous delivery.

If a build fails, the CI system blocks it from progressing to further stages. The team receives a report and repairs the build quickly, typically within minutes. All competitive technology companies today practice continuous integration. By working in small iterations, the software development process becomes predictable and reliable. Developers can iteratively build new features. Product managers can bring the right products to market, faster. Developers can fix bugs quickly and usually discover them before they even reach users.

Continuous integration requires all developers who work on a project to commit to it. Results need to be transparently available to all team members and build status reported to developers when they are changing the code. In case the main code branch fails to build or pass tests, an alert usually goes out to the entire development team who should take immediate action to get it back to a “green” state (Anastasov, 2022).

The following are the benefits of adopting Continuous Integration:

- Lead time: Early feedback and build/test automation help decrease the time it takes to go from code committed to code successfully running in production.
- Deployment frequency: Automated tests and builds are a pre-requisite to automated deploy.

- Time to restore service: Automated pipelines enable fixes to be deployed to production faster reducing Mean Time to Resolution (MTTR).
- Change failure rate: Early automated testing greatly reduced the number of defects that make their way out to production (Gitlab, 2022).

Many third-party tools exist to aid in CI management and installation. Some popular options are Codeship, Bitbucket Pipelines, SemaphoreCI, CircleCI, Jenkins, Bamboo, Teamcity, and many others. These tools have their own in-depth setup guides and documentation to help get started (Rehkopf, 2022).

2.5.4 Continuous deployment

Continuous deployment is a strategy or methodology for software releases where any new code update or change that makes it through the rigorous automated test process is deployed directly into the live production environment where it will be visible to customers. Continuous deployment adds an element of risk to the software release process, as developers are frequently committing unproven code which could potentially contain bugs to the live environment. Organizations that implement continuous deployment must therefore develop real-time monitoring capabilities of the live environment to rapidly discover and address any technical issues that occur after a new release (Continuous Deployment, 2022). The most significant advantage of continuous deployment is speed. Teams that practice continuous delivery routinely move code from developer commit to production in just a few days and often in just a few hours or minutes. Speed, in turn, unlocks greater safety and innovation.While it may seem paradoxical or counterintuitive that a faster process without a manual review and approval step would be safer, DevOps Research and Assessment studies have repeatedly shown that teams with a short cycle time from commit to production have significantly lower incident rates and dramatically shorter

time to resolve issues. They also achieve better business outcomes, which leads to the next advantage: innovation. Continuous deployment shifts significant amounts of power and responsibility towards the team that is writing code. This leads to another paradox: Instead of leading to reckless, unchecked behavior and decisions that don't reflect business requirements, continuous delivery places developers closer to end-users, and that naturally leads to greater empathy, greater pride of ownership, and a more effective feedback loop when iterating towards desired outcomes (Split.io, 2022).

2.5.5 Continuous Monitoring

Continuous monitoring (CM), also referred to continuous control monitoring (CCM), is an automated process that allows DevOps teams to detect compliance and security threats in their software development lifecycle and infrastructure. Traditionally, businesses have relied on periodic manual or computer-assisted assessments to provide snapshots of the overall health of their IT environment. This method often provides information that's too outdated to be useful and can result in undetected security threats, exposing the business to liability or compliance fines. Continuous monitoring, on the other hand, helps DevOps teams identify and track key risks in real time. Besides providing support for critical processes like threat detection, continuous monitoring is a valuable tool for ensuring that your business remains compliant (Ruck, 2021). This practice involves the participation of the Operations team who will monitor the user activity for bugs or any improper behavior of the system. The popular tools used for this are Splunk, ELK Stack, Nagios, NewRelic and Sensu. These tools help you monitor the application's performance and the servers closely and also enable you to check the health of the system proactively. They can also

improve productivity and increase the reliability of the systems, which in turn reduces IT support costs. Any major issues if found are reported to the development team so that it can be fixed in the continuous development phase. This leads to a faster resolution of the problems (Arvind, 2022) .

2.6 Purpose of the present study

Companies are more and more turning to new techniques to make the most of the smaller development teams and cope with more complex, distributed applications. As Agile has been widely adopted in software development, IT teams had a hard time keeping up with the output and create a secure environment for the applications being produced. One obvious solution would be to suggest IT teams to adopt Agile practices. However, it was not as simple as thought. IT teams work differently to software development teams, and an Agile methodology couldn't be applied in the same way. There is a different culture with varying roles and priorities. For example, Agile developers focus on speed and frequency, whereas security is the number one priority for IT professionals. To connect the two, the concept of DevOps was created (Wrike, What Is Agile Operations?, 2022). Many studies have been done on the Agile, non-Agile and DevOps. However, there has not been a research of comparison among these methods from global software development perspective. Moreover, much of the previous study in this area is comprised of qualitative studies focused on individual methodologies separately; Despite being useful this research is not able to provide any predictive insight related to which project characteristics are associated with project success among GSD projects. This study is about understanding more which project management characteristics are associated with global software development project success. Especially how Agile and Non-Agile and DevOps methods perform in GSD. Hence, the following research questions were investigated in this study:

1. Does DevOps software development method in GSD projects achieve greater project success than Agile and Non-Agile software development methods?
2. Which project management characteristics in global software development projects are associated with greater project success?

3.Design

The thesis is a quantitative study that uses survey data to find answer to the questions mentioned below. Convenience sampling technique will be used to contact people and gather data. Contacted individuals are engaged in GSD projects with backgrounds in project management, software development, quality assurance and DevOps.

A convenience sampling method was used to contact groups of individuals for this research. The people targeted for this research were employed in various industries with experience working on GSD projects, including software development, project management, quality assurance and DevOps. To be precise, potential study participants were contacted through Telegram groups, channels and private messages. In particular, three project management groups, 10 software development groups, and three DevOps engineers group were contacted.

A total of thirty-three responses were collected. Out of these, seventeen responses met the eligibility criteria, which included having worked on global projects and having complete survey data on the required variables. Sixteen participants responded as never having worked on global software development projects. Four participants did not finish the survey or left some questions unanswered.

3.1 Data Collection

Data for this research was collected using an online survey. Online surveys are a great option for people and organisations who would like to conduct their own research – they are less time consuming, they are cheaper, you get the results faster, and you can transfer and use the data in various applications to answer important questions (SmartSurvey, 2022).

3.2 Survey Instrument

The data collection tool for this study was a web based survey that consisted of 19 questions, including nine multiple choice questions, three short answer questions, and seven mixed type of questions. The initial survey was first run among three individuals to test for question clarity. Based on feedback from the pilot test the survey was revised. The survey was conducted in 2 languages: English and Russian .

3.3 Variables

Company size. The participants were asked “How many employees work in your company (if you work for a company)?” in order to understand the company size. The options given were Less than 30 31-60 61-100 100 – more.

Independent variables

Experience. The participants were asked “How many years of experience do you have in your current role? (1,2,3... years)” in order to understand how experienced they are in general. The respondents had to type their answer in the box in the form of short answer text. In addition, they were asked “Have you worked on global projects (team members in different countries working on the same project)?” in order to understand whether they have worked on a global project and, if yes, how recent it was. Options to choose were: 1. No, I have never worked on a global project 2.I am currently involved in a global project 3.I worked on a global project within the last 2 years 4.I worked on a global project in the last 3-5 years 5.I worked on a global project in the last 6-10 years 6.I worked on a global project over 10 years ago.

Requirements. The participants were asked “How often do the requirements for the tasks that you are responsible for change?” in order to find out how often the requirements change in participants’ projects. The following response categories were available to choose from: 1. Always 2. Often 3. Sometimes 4. Rarely 5. Never.

Management method. The participants were asked “What type of management do you (or your manager) use?” in order to determine what method is used in the projects of participants. Options available to choose were: 1. Agile management (Scrum, Kanban, Extreme Programming (XP) etc.) 2. Waterfall (Linear: Plan, Design, Develop, Test, Release) 3. Do not know. 4. “Other” text box to insert short text answer.

DevOps adoption. The participants were asked “Who manages the deployment process in your projects?” in order to determine who deals with deployment of the project. Response categories were 1. Development team 2. Operations team. 3. “Other” text box for short text answer. Moreover, they were asked “Do you use CI/CD tools for deployment?” in order to determine the if pipelines are used in their projects. Available response options were 1. Yes, all the time 2. Not always, it depends on the project size 3. No, never 4. Other” text box for short text answer. Another question asked to understand the level of DevOps adoption was

“How often are the changes made by development team deployed?“. The available options were 1. Once a week 2. 1-2 times a month 3. 1-2 times in three months 4. Once in six months 5. Once a year. 6. Other” text box for short text answer. Finally, the participants were asked “How often are maintenance and monitoring of application performance included in your project goals?” in order

to investigate if the projects they worked for used DevOps method throughout the project life cycle.

Project success. In order to measure a project success 3 questions, concerning delivery time, targeted project deliverables and customer satisfaction, were asked. The first one concerning project deliverable was “How many of the global projects that you have worked have met their planned project deliverables?”. The available options to choose from were 1. All 2. 75% - 99%. 3. 50% - 74% 4. 25% - 49% 5. 0% - 24%. Then, the participants were asked “How many of the global projects that you have worked have met their planned project schedule?” in order to find out how many of the projects met their planned schedule. In the end, the participants were asked “How satisfied are customers of the global projects you have worked on?”

3.4 Data Analysis

This study used a quantitative methodology. The average level of project success for participants who used Agile software development project management methods and those who used Non-Agile software development management methods was first compared using t-tests to see if there were any statistical differences. OLS regression analysis was conducted to determine the impact of independent variables on the dependent variables (GSD success). 5% level of significance was considered as significance of the variables.

4.1 Descriptive statistics

In this study 67.5% of the survey participants, members of the local teams, were located in Asia, 20% were from the Europe, and 5% were from the North America.

Table 1.1

Continent		Percent of local team survey participants	
Location	Freq.	Percent	Cum.
Asia	27	67.5	67.5
Europe	8	20	87.5
North America	5	12.5	100
Total	40	100	

Status of the employee:

The question of the survey about the working status, freelancer or work for a company. The table 1.2 shows the employee status, 11 employees were doing both (freelancer and work for company). Six employees were working as freelancers and 23 were working for the company.

Table 1.2

Are you a freelancer or work for a company?	Percent of local team survey participants		
	Freq.	Percent	Cum.
Both	11	27.5	27.5
I am a freelancer	6	15	42.5
I work for a company	23	57.5	100
Total	40	100	

How many employees work in your company?

Table 1.3 represent the how many employees were working in the company. The table shows that there were 45% participant which have less than 30 employees and 35% company employees have 100 or more employees. 31-60 workers company have 5% while 61-100 workers have 15% employees that work in company.

Table 1.3

How many employees work in your company (if you work for a company)?		Percent of local team survey participants	
	Freq	Percent	Cum.
100 - more	14	35	35
31-60	2	5	40
61-100	6	15	55
Less than 30	18	45	100

How many years of experience employee have in current role?
The table 1.4 shows the working experience of workers in years. The table shows the mostly workers have 1 years' experience with 22.5%. Number of years of experience like 12, 2, 41 and 8 years have only one employees in the list. Six employees have 3 years of experience.

Table 1.4

Number of years	Freq.	Percent	Cum.
1	9	22.5	22.5
12	1	2.5	25
15	2	5	30
2	7	17.5	47.5
2 years	1	2.5	50
2+ years	1	2.5	52.5
3	6	15	67.5
4	4	10	77.5
41	1	2.5	80
5	4	10	90
8	1	2.5	92.5
less than a year	2	5	97.5
more than a year	1	2.5	100
Total	40	100	

Have you worked on global projects (team members in different countries working on the same project)?
Question related to global projects participation shows in the table 1.4. Table shows that worker which were involve in the global projects were 12 make 30%. Only two employee were with more than 10 years' experience. There were 17 employees in the survey which were never involved in the global projects.

Table 1.5

Working status	Freq.	Percent	Cum.

I am currently involved in a global project	12	30	30
I worked on a global project over 10 years	2	5	35
I worked on a global project within the last 2 years	1	2.5	37.5
I worked on a global project in the last 3-5 years	8	20	57.5
No, I have never worked on a global project	17	42.5	100
	Total	40	100

In which countries are the other project team members located? (List if more than one)

Table 1.6 shows employees status about the project team members' location. There were 2 members which have members from the Belarus, Kazakhstan, Latvia, USA. Three employee have members from the Germany. Four employees have members from the Russia. The table shows that seven employees have team members from the Uzbekistan.

Table 1.6

	Freq.	Percent	Cum.
Belarus, Kazakhstan, Latvia, USA	2	5.41	5.41
Belgium, Ukraine, Poland	1	2.7	8.11
Belgium, Ukraine, Poland, Belarus, Italy	1	2.7	10.81
France, India, Romania	1	2.7	13.51
Germany	3	8.11	21.62
India	1	2.7	24.32
Ireland, India	1	2.7	27.03
Italy	1	2.7	29.73
Italy, Japan and Belarus	1	2.7	32.43
Italy, Uzbekistan	1	2.7	35.14
None	2	5.41	40.54
Pakistan	1	2.7	43.24
Russia	4	10.81	54.05
Russia, Belarus	1	2.7	56.76
Russia, UAE	1	2.7	59.46
Russia, USA, Ukraine	1	2.7	62.16
South Korea	1	2.7	64.86
UAE	1	2.7	67.57
USA	2	5.41	72.97
USA, UK, Russia, India, Ukraine, Uzbeki	1	2.7	75.68
USA, Uzbekistan	1	2.7	78.38
Ukraine, Russia, Kazakhstan, Australia	1	2.7	81.08
Uzbekistan	7	18.92	100

What type of management do you (or your manager) use?
Table 1.7 shows that type of management of the employees. There are maximum have 18 members have Agile management (Scrum, Kanban, Extreme Programming (XP) etc.), and 12 members have Waterfall (Linear: Plan, Design, Develop, Test, Release). Nine members do not know the management type.

Table 1.7

	Freq.	Percent	Cum.
Agile management (Scrum, Kanban, Extreme Programming (XP) etc.)	18	45	47.5
Do not know	9	22.5	70
Waterfall (Linear: Plan, Design, Develop, Test, Release)	12	30	100
Total	40	100	

What is the primary way you get notified of the project tasks you are responsible for?
Table 1.8 shows about the primary way you get notified of the project tasks you are responsible. The 7 employees from the survey were notified by informal discussions. 18 from the respondent were notified by meeting and one 14 by project tracking software.

Table 1.8

	Freq.	Percent	Cum.
By informal discussions	7	17.5	17.5
By meeting	18	45	62.5
By phone	1	2.5	65
By project tracking software	14	35	100
Total	40	100	

How many of the global projects that you have worked have met their planned project deliverables?
The project success score percentage are represented in table 1.9. 7 people answered that all of their projects have met planned project deliverables.

Table 1.9

	Freq.	Percent	Cum.
0%-24%	8	20	20
25%-49%	1	2.5	22.5
50%-74%	12	30	52.5
75%-99%	12	30	82.5
All	7	17.5	100
Total	40	100	

T-test:

The results of the t-test comparing participants' perceptions of project success based on how satisfied are customer and what type of management do they adopt. With a standard deviation of 0.175, the project success factor for projects using the Agile methods has a mean of 2.72, whereas for non agile project management methods mean value is 2.8, it is standard deviation of 0.911.

Table 1.11

Variable	Obs	Mean	Std. Err	Std. Dev.
How satisfied are customer	40	2.725	0.175366	1.109111
What type of management do you use?	40	2.8	0.144115	0.911465
combined	80	2.7625	0.112851	1.009371
diff		-0.075	0.226986	

Ha: diff < 0 Ha: diff != 0 Ha: diff > 0

Pr(T < t) = 0.3710 Pr(T > t) = 0.7420 Pr(T > t) = 0.6290

The t-test shows that we cannot reject the null hypothesis, it mean that the difference in mean between “how satisfied are customer” and “what type of management do you” is zero or there is no difference in the mean of these variables.

Table 1.12 represent the t-test mean comparison between customer satisfaction and deployment management process.

The results of the t-test comparing participants' perceptions of project success based on how satisfied are customer and who manage the deployment process. With a standard deviation of 1.109111, the project success factor for projects using the Agile technique has a mean of 2.72, whereas who manages the deployment process mean value is 1.15, with a standard deviation of 1.14.

Table 1.12

Variable	Obs	Mean	Std. Err.	Std. Dev.
How satisfied are customers	40	2.725	0.175366	1.109111
Who manages the employment process	40	1.15	0.057177	0.36162
combined	80	1.9375	0.127468	1.140106
diff		1.575	0.184452	

Ha: diff < 0 Ha: diff !=0 Ha: diff > 0
Pr(T < t) = 1.0000 Pr(T > t) =0.0000 Pr(T > t) = 0.0000

Table 1.14 represent the t-test mean comparison between Do you use CI/CD tools for deployment and how satisfied are customers of the global project.

Table 1.14

Variable	Obs	Mean	Std. Err.	Std. Dev.
Do you use CI/CD tools for deployment	40	2.2	0.130089	0.822753
how satisfied are customer	40	2.725	0.175366	1.109111
combined	80	2.4625	0.11243	1.005601
diff		-0.525	0.218349	

Ha: diff < 0 Ha: diff != 0 Ha: diff > 0
Pr(T < t) = 0.0093 Pr(|T| > |t|) = 0.0186 Pr(T > t) = 0.9907
t = -2.4044

The results of the t-test mean comparison between "Do you use CI/CD tools for deployment" and how satisfied are customers of the global project. With a standard deviation of 0.822753, Do you use CI/CD tools for deployment has a mean of 2.2, whereas for how satisfied are customers of the global project mean value is 2.725, it is with a standard deviation of 1.109111. We have sufficient evidence in favor of rejection of null hypothesis, meaning that there is difference in the mean between use of CI/CD tools for deployment and how satisfied are customers of the global project.

Table 1.15 represent the t-test mean comparison between how often does your development team release update and how satisfied are customers of the global project.

Table 1.15

Variable	Obs	Mean	Std. Err.	Std. Dev.

how often does your development team release update	40	2.85	0.230801	1.459715
how satisfied are customers of the global project	40	2.725	0.175366	1.109111
combined	80	2.7875	0.144185	1.289625
diff		0.125	0.289866	

Ha: diff < 0 Ha: diff != 0 Ha: diff > 0
Pr(T < t) = 0.6663 Pr(|T| > |t|) = 0.6675 Pr(T > t) = 0.3337
t = 0.4312

The results of the t-test mean comparison between "how often does your development team release update" and how satisfied are customers of the global project. With a standard deviation of 1.459715, "how often does your development team release update" has a mean of 2.85, whereas for how satisfied are customers of the global project mean value is 2.725, it is with a standard deviation of 1.109111. Here we do not sufficient evidence in favor of rejection of null hypothesis, meaning that there is no difference in the mean between "how often does your development team release update "and how satisfied are customers of the global project.

Regression analysis:

Table 1.16

How satisfied are customers	Coef.	Std. Err.	t	P>t
How many of the global projects	0.348657	0.142554	2.45	0.02**
Are you a freelancer or work for a company	0.203438	0.222036	0.92	0.36
How many employees work in your company	-0.0828	0.099396	-0.83	0.41
What do you primarily do in your current role	-0.17567	0.188402	-0.93	0.35
How many years of experience do you have	-0.04656	0.0442	-1.05	0.30
Have you worked on global projects	0.031754	0.099632	0.32	0.75
In which countries are the other projects	-0.01424	0.023555	-0.6	0.55
What type of management do you	0.435513	0.196195	2.22	0.03**
_cons	1.144473	1.19823	0.96	0.34

** shows the 5% level of significance

Number of obs	37
F(8, 28)	2.64
Prob > F	0.02**
R-squared	0.43
Adj R-squared	0.26
Root MSE	0.93

According to multivariate regression analysis, several aspects of project management methods were linked to successful projects. The coefficient of

determinant shows that 43 percent variation in the project success is caused by all these independent variables. Multiple regression table 1.17 shows that we have 2 significant variables. The coefficient of the “How many of the global projects that you have worked have met their planned project deliverables?” shows that rise in percentage of the “How many of the global projects that you have worked have met their planned project deliverables?” lead to increase the project success by 0.3486 percent points. The coefficient of the “What type of management do you (or your manager) use?” as when manager shift from the agile to waterfall the project success raised by 0.4355. Overall model’s independent variables explain theory supportive relationship with project success.

Table 1.17

	Coef.	Std. Err.	t	P>t
How many of the global projects	1.00772	0.38699	2.6	0.01**
Are you a freelancer or work for a company	1.203593	0.587093	2.05	0.05*
How many employees work in your company	-0.01953	0.098889	-0.2	0.845
What do you primarily do in your curre1	-0.14612	0.183291	-0.8	0.432
How many years of experience do you have	-0.04985	0.042626	-1.17	0.252
Have you worked on global projects	0.098525	0.102785	0.96	0.346
In which countries are the other projects	-0.03333	0.024726	-1.35	0.189
What type of management do you use	0.433392	0.189101	2.29	0.03**
How many of the global projects *Are you a freelancer or work for a company	-0.26287	0.145855	-1.8	0.08*
constant	-1.63585	1.916988	-0.85	0.401

** shows the 5% level of significance

Number of obs	37
F(9, 27)	2.86
Prob > F	0.01**
R-squared	0.48
Adj R-squared	0.31
Root MSE	0.900

** shows the 5% level of significance

The interaction term regression analysis table 1.17 shows that coefficient of determinant improved with the interaction term between “How many of the global projects” and “Are you a freelancer or work for a company”. The coefficient of determinant shows that 48 percent variation in the project success is caused by all these independent variables. Multiple regression with interaction term in table 1.14 shows that we have 4 significant variables. The coefficient of the “How many of the global projects that you have worked have met their planned project deliverables?” shows that rise in percentage of the “How many of the global projects that you have worked have met their planned

project deliverables?" lead to increase the project success by 1.00 percent points. The coefficient of the "Are you a freelancer or work for a company" indicate that the worker who for company have 1.203 percent point more success in project as compare to worker who work as freelancer. The coefficient of the "What type of management do you (or your manager) use?" as when manager shift from the agile manage to waterfall the project success raised by 0.433. The interaction term of "How many of the global projects" and "Are you a freelancer or work for a company" shows the both variable interaction has negative impact on the project success by 0.262 percent points. Overall model independent variables explain theory supportive relationship with project success.

5. Discussion

This study offers helpful information about the qualities of successful project management and GSD project outcomes. The findings of this study show that GSD projects managed using Agile software development project management methods and GSD projects managed using Non-Agile software development project management methods both experience somewhat the same degrees of project success.

It also shows that DevOps adoption is still low among the companies as both deployment and monitoring are still carried out by people who developed the project not by dedicated DevOps team and deployment is managed manually. Contrary to popular belief that Non-agile project management methods are not in use much these days, they are still used by many project managers during the development of software. Additionally smaller teams seem to show a higher project success level regardless of the project management methods used.

During the interviews managers and developers also indicated that clear requirements were vital for achieveing successful GSD. As it can be seen from the survey results projects with less frequent changing requirements were subject to higher degree of meeting planned project deliverables project and customer satisfaction.

5.1 Implications for Practice

Managers of GSD projects should take precautions to reduce project risk brought on by shifting needs. Implementing iterative knowledge acquisition and sharing methods can help all stakeholders in GSD projects have a shared understanding of the client's needs and the technology capabilities required to satisfy them. This makes it possible for all stakeholders to communicate project information effectively. Additionally, GSD projects profit from iterative development methodologies that identify integration problems earlier in a project's lifespan, while project deliverables are still in the early phases of development and easier to fix. Delays will result from holding off on initial integration of work created by remote teams.GSD initiatives need to promote informal contact across remote teams to aid iterative development cycles.

As it was the similar case in earlier studies, there is a direct correlation between stable project requirements and project success. This study offers proof that, whether GSD project teams employ Agile or Non-Agile management techniques, their success will be increased by lowering the frequency of

requirement changes throughout the course of a project. The project team members spend time implementing the modified requirements as a result of frequent requirements revisions rather than going forward with the previously intended work, which causes project rework. Often, this rework causes project delays which in turn leads to decreasing of the project success.

5.2 Limitations

Because a convenience sample rather than a random sample was chosen, the generalizability of the study's findings is constrained. Due to the lack of public access to entire lists of user groups and members of different user groups, random selection was not possible. The use of a one-item indicator rather than a sequence of questions allowing for classification into either of these groups was a second study constraint for measuring the use of Agile or Non-Agile software development project management approaches. The survey has a third drawback in that it has not been tested; future researchers in this field should think about creating a validated instrument to further explore the ideas that stood out as being significant in this research. Another point that should be mentioned is that the lack of DevOps participants the resulted in conclusion that it is not largely adopted by companies working in the GSD.

5.3 Future Research

Future research in this area should take into account the kinds of tools that are crucial for the success of GSD projects in addition to using a larger, more representative sample, validating the survey tool, and investigating various types of Agile software development project management techniques. Future research should also take the size of international project teams into consideration. Future research should also focus on surveying more of DevOps team members.

6.Conclusion

The success of Agile management techniques in comparison to Non-Agile management techniques in GSD projects was examined in this quantitative study, along with the influence of a particular set of project management characteristics on project success. It offers preliminary understanding of what facets of project management techniques and project manager traits are linked to improved GSD project success. In this study, it was not discovered that participants who indicated utilizing Agile management techniques on GSD projects had noticeably more successful projects than those who claimed using Non-Agile techniques. However, GSD projects with fewer requirement modifications, better communication tools for project team members connections had improved project success. By ensuring that there is efficient communication among all stakeholders, project managers can minimize risks related to requirement changes. Additionally, having the right project communication and collaboration technologies at their disposal can help project team members cooperate remotely while reducing the risks related to requirement changes. Even if DevOps tools have been partially adopted in GSD projects, more steps should be taken to fully embrace DevOps philosophy.

Bibliography

Asmo. (2019). *Waterfall Project Management: An Overview*. From zenkit.com: https://zenkit.com/en/blog/waterfall-project-management-an-overview/#:~:text=The%20Waterfall%20methodology%20has%20its,would%20result%20in%20high%20costs.

Workfront. (2022). From https://www.workfront.com: https://www.workfront.com/project-management/methodologies/waterfall

Parabol. (2022). *Agile Frameworks: A Complete Overview*. From https://www.parabol.co: https://www.parabol.co/resources/agile-frameworks-guide/

ITchronicles. (2022). *What is Software Development?* From https://itchronicles.com: https://itchronicles.com/what-is-software-development/

Jevtic, G. (2022). *What is SDLC? Phases of Software Development, Models, & Best Practices*. From https://phoenixnap.com: https://phoenixnap.com/blog/software-development-life-cycle#:~:text=Software%20Development%20Life%20Cycle%20is,%2C%20Test%2C%20Deploy%2C%20Maintain.

CPRIME. (2022). *WHAT IS AGILE? WHAT IS SCRUM?* From https://www.cprime.com: https://www.cprime.com/resources/what-is-agile-what-is-scrum/

Waterfall Vs. Agile: Which Is the Best Choice For the Remote Tech Team? . (2021). From https://www.zenbusiness.com: https://www.zenbusiness.com/blog/waterfall-vs-agile-which-best-choice-remote-tech-team/#:~:text=Waterfall%20Project%20Management&text=This%20method%20is%20a%20progress,to%20the%20implementation%20and%20maintenance.

Radigan, D. (2022). *Think globally, code locally: the secret to remote teams*. From https://www.atlassian.com: https://www.atlassian.com/agile/teams/remote-teams

Atlassian. (2022). *What Is DevOps?* . From https://www.atlassian.com: https://www.atlassian.com/devops

Arvind. (2022). *DevOps Life cycle: Everything You Need To Know About DevOps Life cycle Phases*. From https://www.edureka.co: https://www.edureka.co/blog/devops-lifecycle/

Katalon. (2022). *Introduction to Continuous Testing | Definition, Tools & How to Perform*. From https://katalon.com: https://katalon.com/resources-center/blog/continuous-testing-introduction

Anastasov, M. (2022). *Continuous Integration (CI) Explained* . From https://semaphoreci.com: https://semaphoreci.com/continuous-integration

Gitlab. (2022). *What is Continuous Integration (CI)?* From https://about.gitlab.com: https://about.gitlab.com/topics/ci-cd/benefits-continuous-integration/

Rehkopf, M. (2022). *What is continuous integration?* . From https://www.atlassian.com: https://www.atlassian.com/continuous-delivery/continuous-integration

Continuous Deployment. (2022). From https://www.sumologic.com: https://www.sumologic.com/glossary/continuous-deployment/

Split.io. (2022). *Continuous Deployment*. From https://www.split.io: https://www.split.io/glossary/continuous-deployment/

Ruck, D. (2021). *Continuous Monitoring: What Is It and How Is it Impacting DevOps Today?* From https://lightrun.com: https://lightrun.com/best-practices/continuous-monitoring-what-is-it-and-how-is-it-impacting-devops-today/

Edureka. (2022). *DevOps Life cycle: Everything You Need To Know About DevOps Life cycle Phases*. From https://www.edureka.co: https://www.edureka.co/blog/devops-lifecycle/

Volchenkova, V. (2022). From https://www.istockphoto.com/: https://www.istockphoto.com/

Wrike. (2022). *What Is Agile Operations?* From https://www.wrike.com: https://www.wrike.com/agile-guide/agile-operations-it/

SmartSurvey. (2022). *10 Advantages of Online Surveys*. From https://www.smartsurvey.co.uk: https://www.smartsurvey.co.uk/articles/10-advantages-of-online-surveys

Carayannis, K. a. (2003). *The Story of Managing Projects.* Quorum Books.

PMI. (2017). *A Guide to the PROJECT MANAGEMENT BODY OF KNOWLEDGE.* Project Management Institute, Inc.

Wrike. (2022). *What Is a Project in Project Management?* From https://www.wrike.com: https://www.wrike.com/project-management-guide/faq/what-is-a-project-in-project-management/

PMI. (2022). *What is Project Management?* . From https://www.pmi.org: https://www.pmi.org/about/learn-about-pmi/what-is-project-management

Ahmed, A. (2012). *Software Project Management : A Process-Driven Approach* . CRC Press.

IBM. (2022). *What is software development?* From https://www.ibm.com: https://www.ibm.com/topics/software-development

Survey questions:

1.Where do you live?

- open text response

2. Are you a freelancer or work for a company?

- I work for a company
- I am a freelancer
- Both

3. How many employees work in your company(if you work for a company)?

- Less than 30
- 31-60
- 61-100
- 100 - more

4. What do you primarily do in your current role?

- Manage
- Develop
- Design
- Quality Assurance
- DevOps engineer
- Other (please specify)

5. How many years of experience do you have in your current role?(1,2,3... years)

- open text response

6. Have you worked on global projects (team members in different countries working on the same project)?

- No, I have never worked on a global project
- I am currently involved in a global project
- I worked on a global project within the last 2 years
- I worked on a global project in the last 3-5 years
- I worked on a global project over 10 years

7. In which countries are the other project team members located? (List if more than one)

- Open text response

8. What type of management do you (or your manager) use?

- Agile management (Scrum, Kanban, Extreme Programming (XP) etc.)
- Waterfall (Linear: Plan, Design, Develop, Test, Release)
- Do not know
- Other(Please specify)

9. What is the primary way you get notified of the project tasks you are responsible for?

- By informal discussions
- By meeting
- By project tracking software
- By phone

10. How often do the requirements for the tasks that you are responsible for change?

- Always
- Often
- Sometimes
- Rarely
- Never

11. How often do you interact with your project team members when working on a global project?

- Every day
- 2-3 times a week
- Once a week

- Once a month
- Never

12. Who manages the deployment process in your projects?

- Operations team
- Development team

13. Do you use CI/CD tools for deployment?

- Yes, all the time
- Not always, it depends on the project size
- No, never

14. How often does your development team release updates to their applications?

- Once a week
- 1-2 times a month
- 1-2 times in three months
- Depends on project

15. How often are the changes made by development team deployed?

- Once a week
- 1-2 times a month
- 1-2 times in three months
- Depends on project

16. How often are maintenance and monitoring of application performance included in your project goals?

- Always
- Often
- Sometimes
- Rarely
- Never

17. How many of the global projects that you have worked have met their planned project deliverables?

- 0% - 24%
- 25% - 49%
- 50% - 74%
- 75% - 99%
- All

18. How many of the global projects that you have worked have met their planned project schedule?

- 0% - 24%
- 25% - 49%
- 50% - 74%
- 75% - 99%
- All

19. How satisfied are customers of the global projects you have worked on?

- Totally Satisfied
- Partially Satisfied
- Partially unsatisfied
- Totally unsatisfied
- Do not know

Professional Online User Groups Contacted	Platform
Agile Water Cooler	Discord
DevOps Italia	Telegram
Scrum Master discussions	Telegram
iOS developers group	Telegram
Kanban Talks	Telegram
xCoding.it	Telegram
Jenkins	Telegram

Survey Participants Location
Russia
USA
Uzbekistan
India
South Korea
Italy

Other team members location
Russia
USA
Uzbekistan
India
South Korea
Italy
Belgium
Germany
Belarus
UAE
Kazakhstan
Latvia
Ukraine
Turkey
Ireland
Poland
France
Japan