DOI: 10.34020/2073-6495-2020-4-174-183

УДК 004

ПЕРСПЕКТИВНОЕ РАЗВИТИЕ МЕТОДОЛОГИИ DEVOPS

Ермаков А.С.
Российская академия народного хозяйства
и государственной службы при Президенте РФ
E-mail: mirael92@gmail.com

От выбора методологии управления проектами зависит не только модель управления, но и архитектура самого IT-продукта, определяющая его успешность или возможные сложности на более поздних этапах развития. В статье рассмотрены этапы развития методологии управления проектами, начиная с ITIL\ITSM, Waterfall, созданием Agile и DevOps. Исследованы истоки и процесс становления методологии DevOps, которая должна разрешить конфликт между разработчиками и администраторами. За 10 лет своего существования методология DevOps сумела успешно пройти одну трансформацию и в ближайшем будущем ожидается еще одна. Уже сейчас DevOps подразделяется на методологии NoOps, ChatOps, AIOps и др. В статье рассмотрены возможные пути дальнейшего развития методологии, а также ее возможное влияние на IT-сферу в целом.

Ключевые слова: методология, модель, управление проектом, искусственный интеллект, DevOPs.

FUTURE DEVELOPMENT OF DEVOPS METHODOLOGY

Ermakov A.S.
Russian Presidential Academy of National Economy
and Public Administration
E-mail: mirael92@gmail.com

Both the management model and the IT product architecture, defining its success or possible difficulties at the later stages of development, depend on the selection of project management methodology. The article considers the stages of development of project management methodology starting from ITIL\ITSM, Waterfall, creation of Agile and DevOps. The paper studies the origins and process of establishment of the DevOps methodology meant to resolve a conflict between developers and administrators. In a decade of its existence the DevOps methodology successfully went through one transformation and one more is expected in near future. Even now DevOps is divided into NoOps, ChatOps, AIOps and other methodologies. The article considers possible ways of further development of the methodology as well as its possible impact on the IT Field in general.

Keywords: methodology, model, project management, artificial intelligence, DevOps.

Научная новизна. Благодаря быстрому развитию технологий, методики и форматы управления также претерпевают изменения. Так, ввиду своей гибкости и упору на культуру [11] коммуникаций между разными отделами и командами вместо hard-skills [10], т.е. компетенций работы систем, DevOps позволяет проводить разные эксперименты в области управления проектами и вбирать другие успешные практики и методы. Посколь-

ку DevOps допускает большое проведение разнообразных экспериментов, в свое время появились такие направления, как NetOps, GitOps и др.

Проблематика. Главной проблемой, описанной в данной работе, можно считать большую вариативность выбора методологии управления проектом, из-за чего корректно сделать выбор становится сложным и подчас невозможным. Так, от данного выбора может зависеть не только модель управления проектом, но и архитектура самого продукта, которая, в свою очередь, может определить его успешность или возможные сложности на более поздних этапах развития. Для примера можно привести различия состава команды разработки при NoOps, где, как правило, отсутствует постоянный инженер, а команда полностью состоит из разработчиков. В данном случае инженер может быть частью команды в первое время, пока необходимо установить, настроить и провести разнообразные интеграции между используемыми продуктами, или его может не быть вовсе. В таком случае роль инженера на себя частично берут сами разработчики или используются сторонние облачные решения.

Не исследованы до конца возможности искусственного интеллекта в случае самопроектирования и самопрограммирования. Так ИИ, разрабатываемый компанией Facebook, был выключен, потому что через какое-то время используемые ИИ-агенты придумали собственный язык взаимодействия, который не могли понять разработчики [10].

Список ключевых понятий и терминов. DevOps – технология (методология) активного взаимодействия специалистов по разработке со специалистами по информационно-технологическому обслуживанию и взаимной интеграции их рабочих процессов для обеспечения качества продукта.

Agile – обобщающий термин для целого ряда подходов и практик, основанных на ценностях Манифеста гибкой разработки программного обеспечения и 12 принципах, лежащих в его основе.

Пайплайн – это процесс разработки (подготовки, производства), программный конвейер.

Искусственный интеллект – свойство интеллектуальных систем выполнять творческие функции, которые традиционно считаются прерогативой человека; наука и технология создания интеллектуальных машин, особенно интеллектуальных компьютерных программ.

Машинное обучение – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение в процессе решения множества сходных задач.

ВВЕДЕНИЕ

Как самостоятельная методология DevOps появилась в 2009 г. Основатели данной методологии ставили перед собой одну проблему, которую они хотели решить, – разрушение нисходящей спирали.

Данная проблема, по мнению авторов, была преградой на пути достижения целей компании и создания здорового окружения [2]. Однако с течением времени, когда методологию стало продвигать не только сообщество IT-специалистов, но и такие крупные компании, как Google, Microsoft и Amazon, началось расширение методологии, поиск новых областей ее

применения, а также более активное наполнение другими практиками и методами, например:

– A framework for managing mission needs, compliance, and trust in the DevOps environment [1],

– The role of Network Operations in bringing commercial wireless to tactical networks [9].

В период, когда все возможные варианты имплементаций других практик управления были исчерпаны, начался процесс замены компонентов. В практике NoOps из связки отделов разработчиков и администраторов были убраны администраторы. В случае же с NetOps практики DevOps применяют в своей повседневной работе сетевые инженеры. Примером могут служить анонсы новых возможностей от компании Cisco [6]. Когда центр внимания смещается с людей на технологии, появляется AIOps [15].

ОСНОВНАЯ ЧАСТЬ

Для понимания возможных путей развития методологии необходимо изучить ее истоки, а также процесс становления. В этом процессе нельзя обойтись без поверхностного изучения ее предшественников и понимания, какие проблемы в конечном итоге была призвана решить данная методология.

Так, до появления DevOps основной практикой ведения разработки был Agile [6], а до него ITIL\ITSM [13, 19] и «водопад». При рассмотрении данных практик не стоит забывать, что длительное время среди архетипов разработки главенствовал монолитный подход конечного программного обеспечения, который заключал всю бизнес-логику приложения. Максимум, что было доступно, – это разделение приложения на логические слои. При таком процессе разработки спустя определенное время приложение становилось крупным и неповоротливым. Практикой ведения проекта был «водопад» (рис. 1). В рамках данной методологи идея зарождалась в самом верху и постепенно спускалась до самого низа. Как правило, в случае необходимости вернуть идею обратно на доработку требовалось примерно столько же времени, сколько на ее первоначальную генерацию.

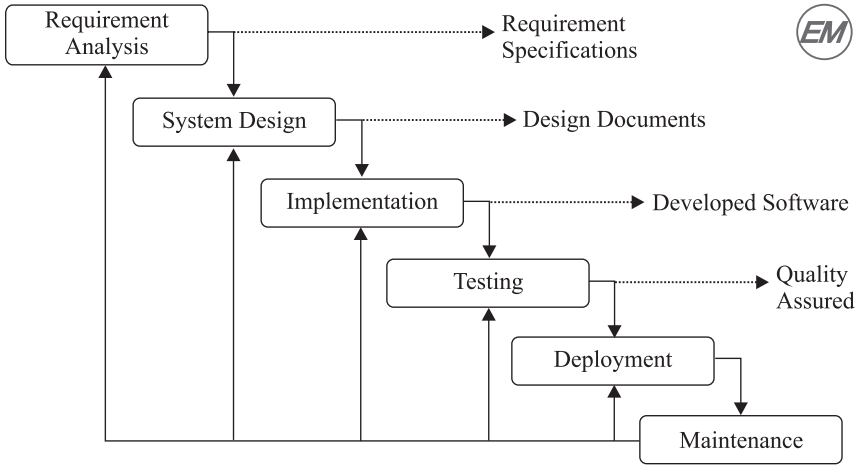

Рис. 1. Waterfall Model SDLC

Проблема данной разработки заключалась в том, что разработчики не передавали друг другу код в режиме реального времени, а перебрасывали его через бюрократические стены. Если код соответствовал внутренним стандартам той или иной команды, он считался успешным и передавался дальше. В случае возникновения каких-либо проблем с разработанным кодом, информация до разработчиков могла дойти только спустя несколько недель, а сама задача по исправлению – через несколько месяцев. Из-за слишком большого времени передачи информации между отделами происходила потеря контекста задачи. Что, как и почему было сделано тем или иным образом, а не как-то иначе, забывалось спустя столь большой промежуток времени. На данную проблему также накладывалось возможное отсутствие какой-либо документации или комментариев в коде, что только усложняло процесс решения проблемы. Однако дальше по «водопаду» уже ушел другой код, который мог учитывать работу предыдущего и зависеть от него. Таким образом изменение кода вышестоящей команды могло привести к изменению кода всеми нижестоящими командами. Таким образом, достаточно весомую часть рабочего времени вместо решения бизнес-задач и удовлетворения пользователей новыми возможностями разработчики тратили на исследование собственного старого кода и исправление тех багов, которые были найдены намного ниже по производственному циклу. Подобная проблема наблюдалась почти в каждой новой версии продукта. Это затягивало сроки сдачи конечного продукта в релиз и, как итог, происходило накапливание функциональных потребностей, что вело сначала к увеличению штата, а затем к поиску новых методик управления проектами. Постоянная необходимость повторно решать задачи и исправлять старый код вместо того чтобы создавать что-то новое, а также все больше сжимающиеся сроки приводили к сильному моральному давлению как на разработчиков, так и на администраторов, которым приходилось поддерживать данный код, из-за чего конфликт между разработчиками и администраторами нарастал. Одной из сторон конфликта хотелось быстрее разрабатывать и быстрее доставлять код, а другой хотелось, чтобы этот код работал стабильнее и не требовал большого количества новой ручной работы. При данной методологии управления проектом разработчики не отвечали за стабильность конечного продукта. Распространенной практикой была проверка кода на своей локальной рабочей станции или на слабых нагрузках в рамках стенда разработки. Однако в продуктовой среде все условия изменяются. Растет нагрузка на систему, требования к ее быстродействию и стабильности и др. Вследствие постоянного морального давления на всех членов разнообразных команд ментальное здоровье сотрудников падало, что приводило к росту уровня стресса у отдельных членов команды, развивались разные неврологические, психосоматические расстройства [17].

На смену ITIL\ITSM с его «водопадом» пришел Agile – методология, которая постаралась ускорить процесс разработки. Теперь цикл разработки выглядел не как водопад, а как бесконечный круг (рис. 2). В рамках данной методологии были разработаны несколько фреймворков, а сама методология основывалась на множестве практик, относящихся к гибкой и бережливой разработке: Scrum, Kanban.

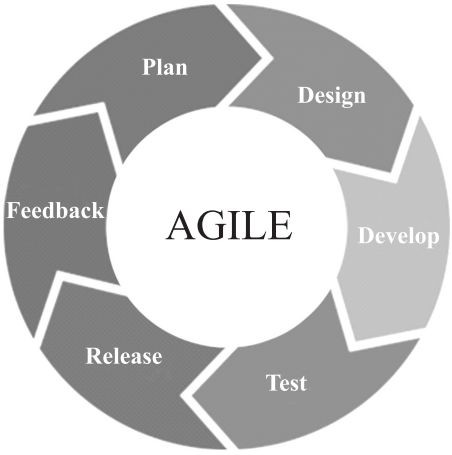

Рис. 2. Agile

На текущий момент почти никто не разрабатывает на чистом Agile или Scrum с Kanban, так как в каждой из данных методологий есть свои плюсы, которые используют почти все. Например, спринты пришли к нам из Scrum, а доски с тремя-четырьмя столбцами статусами пришли из Kanban.

Работая по данной методологии, процесс разработки ускорился. Однако противостояние между отделами не уменьшилось, а даже несколько увеличилось. Теперь команда разработчиков работала по спринтам-циклам от одной до двух недель. Они решали только те задачи, которые были взяты в работу в начале спринта. Решив данные задачи, они проводили собственное тестирование и передавали дальше уже не столько исходный код, сколько почти готовый артефакт. Оставалось только провести интеграцию с остальными компонентами системы, проверить на нагрузку и можно выпускать в релиз. В случае каких-либо проблем разработчики не имели права тут же приступать к решению проблемы, так как она не находится в рамках спринта. Таким образом, к решению бизнес-задач примешивались задачи по стабильности продукта, а закрытые спринты не позволяли добавлять новые задачи. Это привело к тому, что бизнес был не очень доволен, его желания хотя и исполнялись быстро, все равно недостаточно быстро, чтобы получить прибыль здесь и сейчас.

Администраторы же были недовольны тем, что релизы и новые версии продуктов стали появляться чаще, из-за чего необходимо было чаще проводить обновление продуктовых машин, что иногда приводило к сверхурочной работе, а разработчики до сих пор не отвечали за стабильность конечного продукта. При этом стоит обратить внимание, что монолитный архетип разработки программного обеспечения стал сменяться в своей массовости микросервисной архитектурой. В данном случае небольшие команды в течение спринта писали программу или решали какие-то ее проблемы, а в конце все команды должны были выпустить новые версии и провести совместное тестирование. И в этот момент все становилось не так хорошо, как предполагалось, то, что хорошо работало по отдельности, не означало, что при сборе воедино всех новых версий сервисов стабильность и эффективность останутся на том же уровне, а не упадут в целочисленных показателях.

В это время зарождается практика DevOps, которая должна разрешить конфликт между разработчиками и администраторами, сблизив их и изменив процесс разработки. Теперь, несмотря на закрытый спринт, проблемы стабильности и работы стали внеочередными. В случае если продуктовый сервис ломается прямо сейчас, разработчики обязаны отложить свои задачи и заняться продуктовой проблемой. Переходу от Agile к DevOps помогло развитие микросервисного подхода к разработке продукта: каждый

элемент системы независим от всей системы и может быть минимально протестирован без всех компонентов инфраструктуры с помощью CI\CD [3] и в случае каких-либо проблем тут же возвращен команде разработки.

Таким образом разрыв между разработчиками и администраторами сократился. Одни стали отвечать за стабильность своего результата труда, а вторые должны обеспечить первых инструментами и окружениями, чтобы код как можно быстрее проходил все этапы пайплайна [16], уходил в продуктовую среду и закрывал бизнес-задачу, которую заложили в этом релизе менеджеры. Теперь процесс разработки напоминает не кольцо, а перевернутую восьмерку или бесконечность (рис. 3).

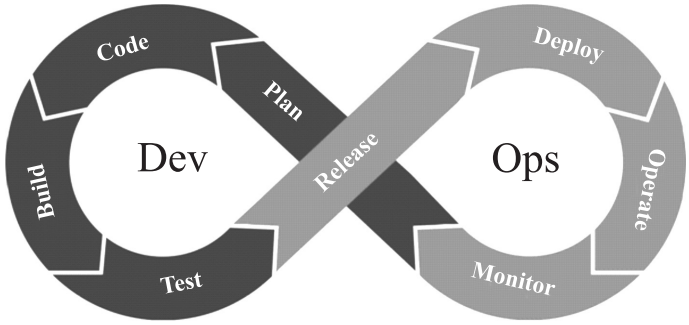

Рис. 3. DevOps pipeline

Если в системе что-то не так, но явных ошибок нет, например, высокое потребление ресурсов центрального процессора, администраторы заводят заявку и один из разработчиков переключается на ее решение со своей текущей задачи. Это помогает поддерживать продукт в стабильном состоянии, что в свою очередь ускорило процесс разработки, так как теперь разработчик находит ошибку в продуктовой системе, исправляет ее, тут же проверяет исправление и отправляет исправленную часть кода в общий репозиторий. Таким образом, возврата на прошлую стабильную версию продукта не происходит и бизнес-задачи остаются покрытыми текущим релизом программного обеспечения. Не стоит забывать, что DevOps – это не просто процесс разработки, а взаимодействие между отделами. Сама философия данной методологии говорит нам об этом:

– культурное, профессиональное движение с мировосприятием и ценностями;

– ответная реакция на плохую коммуникацию между отделами [12].

Однако стоит обратить внимание, что речь постоянно идет про разработчиков и администраторов. А что с тестировщиками и сотрудниками отдела информационной безопасности? Последние несколько лет набирает обороты тренд включения тестировщиков в процесс создания продукта, чтобы они писали автоматические интеграционные тесты и разгрузили разработчиков от этих обязанностей. Что касается сотрудников отдела безопасности, спрос на их услуги тоже изменился. Теперь им обязательно не только знать какие-то теоретические подходы к защите инфраструктуры, но необходимо уметь внедрять и настраивать специализированный софт, а также интегрировать его с процессом разработки, дабы разрабатываемый

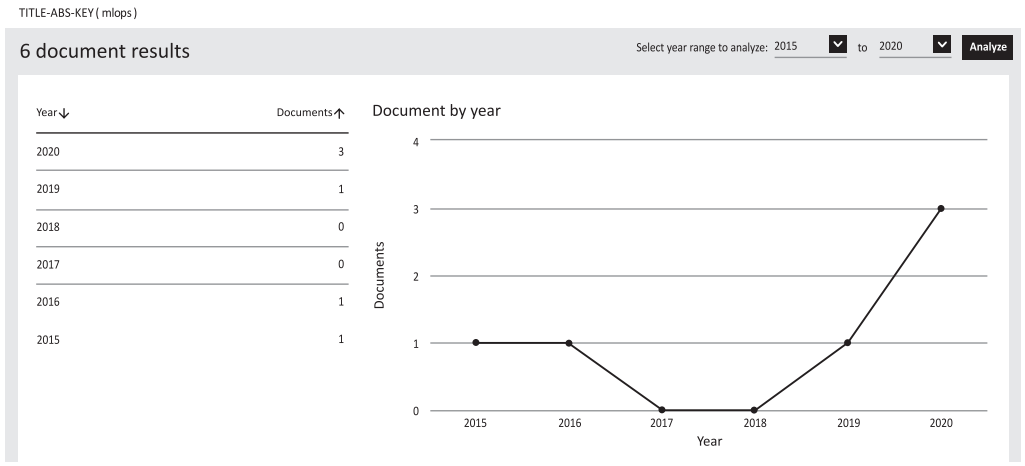

Рис. 4. Динамика написания статей на тему MLOps [18]

код сразу проверялся и в случае проблем возвращался разработчику на доработку. Это привело к появлению нового направления в рамках DevOps под названием DevSecOps.

Таким образом, к 2020 г. помимо логов касательно работы приложения добавились логи сборки, большие этапы тестирования и логи безопасности. Например, Netflix заявляет, что одна неуспешная сборка приложения может породить до нескольких миллионов строк логов [14]. Из-за такого огромного количества поступающей информации, которую невозможно обработать с помощью человеческого мозга за приемлемое время, для решения данной проблемы были подключены специалисты по машинному обучению. Для выявления проблемной информации в рамках такого огромного количества поступающей информации в системы логирования стали встраиваться подсистемы машинного обучения, что, в свою очередь, привело к очередному изменению и появлению нового течения в рамках DevOps методологии под названием MLOps. Данное направление помимо специалистов прикладного уровня начали изучать и в научных кругах (рис. 4). Так, по данным международной базы Scopus наблюдается повышенный интерес к данной тематике.

С ростом интереса в научных и коммерческих кругах на свет появляются синергетические компании, которые помогают ученым обрабатывать их огромные массивы данных для последующего создания обученных моделей. Для примера можно привести медицинские компании с техническим уклоном [7], которые разрабатывают облачные платформы, куда ученые из медицинского сектора могут загружать собственные наборы данных, а уже компьютер ведет полный обсчет, строит возможные вектора развития и ищет успешные химические соединения. Данный бизнес помогает сократить издержки при производстве лекарств, а также ускорить развитие фармакологической отрасли в целом, снижая издержки при производстве и экономя время самих ученых. Теперь им нет никакой необходимости заниматься ручным смешиванием необходимых ингредиентов: за них это может сделать машина. Рост и разнообразие входных данных ускорит обучение

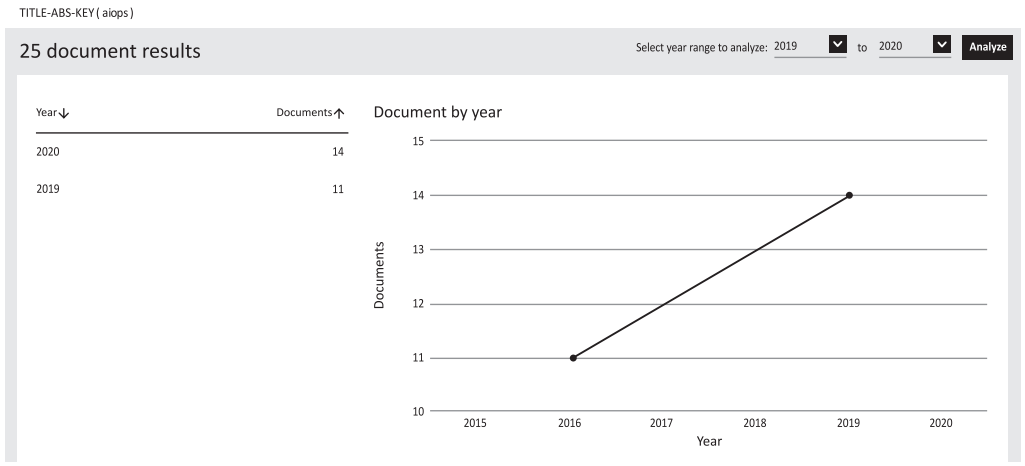

Рис. 5. Динамика написания статей на тему AIOps [18]

модели, что в дальнейшем может привести к еще большему сокращению трудозатрат и материальных ресурсов.

Также стоит обратить внимание, что MLOps дает толчок к росту и становлению другого направления в рамках DevOps, а именно – AIOps. Интерес к данному направлению по данным Scopus с 2019 г. начали проявлять ученые (рис. 5) и коммерческие компании.

Так, новозеландская компания Soul Machines занимается разработками в области создания полноценного искусственного интеллекта. Для своих целей они собирают огромные массивы данных (фото, видео, аудио) с использованием современных технологий, а также стараются применять все современные наработки по изучению человеческого мозга, чтобы создать не просто реплику человека, а полноценный искусственный интеллект с кибернетическим мозгом.

Также со счетов не стоит сбрасывать компанию Google, которая с использованием машинного обучения ведет проект по системе интеллектуального перевода в реальном времени [20]. По задумке авторов, данная технология должна помочь людям с нарушением речи лучше коммуницировать и социализироваться в обществе.

ВЫВОДЫ

Принимая во внимание все вышеперечисленное, можно сделать вывод, что в ближайшие 5 лет будет продолжаться развитие направления DevSecOps из-за возросшего количества необработанной информации, что даст толчок к росту MLOps. Таким образом, уже в 2022 г. данное направление выйдет на плато изучения, когда большинство трудов уже будет написано, и начнется прямая имплементация данного направления в производственные циклы. Бурный рост машинного обучения, в свою очередь, подтолкнет развитие AIOps, которая уже к 2030 г. будет находиться в начале своего пути имплементации в существующую инфраструктуру, а не являться уделом крупных компаний, составляющих FAANG [3], или отдельных компаний, которые создают продукт будущего.

Литература

1. *Farroha B.S., Farroha D.L.* A framework for managing mission needs, compliance, and trust in the DevOps environment. Institute of Electrical and Electronics Engineers Inc, 2014.
2. *Kim G., Debois P., Willis J., Humble J., Allspaw J.* The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. IT Revolution Press, 2016. 480 p.
3. Что такое непрерывная интеграция? [Электронный ресурс]. URL: https://aws.amazon.com/ru/devops/continuous-integration/
4. Accelerate: State of DevOps 2019 URL:DORA & Google Cloud. 2019. [Электронный ресурс]. URL: https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling
5. Acronyms Dictionary. [Электронный ресурс]. URL: https://www.dictionary.com/e/acronyms/faang/
6. Agile Alliance. [сайт] URL: https://www.agilealliance.org/agile101
7. Atome Wise. [Электронный ресурс]. URL: https://www.atomwise.com
8. Cisco Blogs. [Электронный ресурс]. URL: https://blogs.cisco.com/tag/netops
9. *Elmasry G., Welsh R., Jain M., Hoe B., Kim Jakubowskib, Whittaker K., Oddo G.* 2011. The role of Network Operations in bringing commercial wireless to tactical networks. [Электронный ресурс]. URL: https://ieeexplore.ieee.org/document/6127504
10. *Griffin A.* Facebook`s artificial intelligence robots shut down after they start talking to each other in their own language. 2017. [Электронный ресурс]. URL: https://www.independent.co.uk/life-style/facebook-artificial-intelligence-ai-chatbot-new-language-research-openai-google-a7869706.html
11. *Heery E., Noon M.* A Dictionary of Human Resource Management. 3 ed. Oxford University Press, 2017. eISBN: 9780191827822. [Электронный ресурс]. URL: https://www.oxfordreference.com/view/10.1093/acref/9780191827822.001.0001/acref-9780191827822
12. *Humble J.* DevOps Manifesto. [Электронный ресурс]. URL: https://sites.google.com/a/jezhumble.net/devops-manifesto/
13. IT service manager [сайт]. URL: https://www.gov.uk/guidance/it-service-manager
14. *Kirdey S., High W.* 2020. Machine Learning for a Better Developer Experience. [Электронный ресурс]. URL: https://netflixtechblog.com/machine-learning-for-a-better-developer-experience-1e600c69f36c
15. *Lerner A.* AIOps Platforms. 2017. [Электронный ресурс]. URL: https://blogs.gartner.com/andrew-lerner/2017/08/09/aiops-platforms/
16. Lexico. [Электронный ресурс]. URL: https://www.lexico.com/definition/pipeline
17. *Maldonado M.* How Stress Affects Mental Health. 2018. [Электронный ресурс]. URL: https://psychcentral.com/blog/how-stress-affects-mental-health/
18. Scopus. [Электронный ресурс]. URL: https://www-scopus-com.ezproxy.ranepa.ru:2443/term/analyzer.uri?sid=2ee86a84790e019df3a476da7a2d639b&origin=resultslist&src=s&s=TITLE-ABS-KEY%28MLOps%29&sort=plf-f&sdt=b&sot=b&sl=20&count=6&analyzeResults=Analyze+results&txGid=efd45a3c301977e0df5a2bc50c3a0a96
19. Service management good practice [сайт]. URL: https://www.gov.uk/government/publications/public-services-network-psn-service-management-good-practice/service-management-good-practice
20. Text-to-Speech. [Электронный ресурс]. URL: https://cloud.google.com/text-to-speech

Bibliography

1. *Farroha B.S., Farroha D.L.* A framework for managing mission needs, compliance, and trust in the DevOps environment. Institute of Electrical and Electronics Engineers Inc, 2014.

2. *Kim G., Debois P., Willis J., Humble J., Allspaw J*. The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations. IT Revolution Press, 2016. 480 p.
3. Chto takoe nepreryvnaja integracija? [Jelektronnyj resurs]. URL: https://aws.amazon.com/ru/devops/continuous-integration/
4. Accelerate: State of DevOps, 2019. URL:DORA & Google Cloud. 2019. [Jelektronnyj resurs]. URL: https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling
5. Acronyms Dictionary. [Jelektronnyj resurs]. URL: https://www.dictionary.com/e/acronyms/faang/
6. Agile Alliance [sajt]. URL: https://www.agilealliance.org/agile101
7. Atome Wise. [Jelektronnyj resurs]. URL: https://www.atomwise.com
8. Cisco Blogs. [Jelektronnyj resurs]. URL: https://blogs.cisco.com/tag/netops
9. *Elmasry G., Welsh R., Jain M., Hoe B., Kim Jakubowskib, Whittaker K., Oddo G*. 2011. The role of Network Operations in bringing commercial wireless to tactical networks. [Jelektronnyj resurs]. URL: https://ieeexplore.ieee.org/document/6127504
10. *Griffin A*. Facebook`s artificial intelligence robots shut down after they start talking to each other in their own language. 2017. [Jelektronnyj resurs]. URL: https://www.independent.co.uk/life-style/facebook-artificial-intelligence-ai-chatbot-new-language-research-openai-google-a7869706.html
11. *Heery E., Noon M*. A Dictionary of Human Resource Management. 3 ed. Oxford University Press, 2017. eISBN: 9780191827822. [Jelektronnyj resurs]. URL: https://www.oxfordreference.com/view/10.1093/acref/9780191827822.001.0001/acref-9780191827822
12. *Humble J*. DevOps Manifesto. [Jelektronnyj resurs]. URL: https://sites.google.com/a/jezhumble.net/devops-manifesto/
13. IT service manager [sajt]. URL: https://www.gov.uk/guidance/it-service-manager
14. *Kirdey S., High W*. 2020. Machine Learning for a Better Developer Experience. [Jelektronnyj resurs]. URL: https://netflixtechblog.com/machine-learning-for-a-better-developer-experience-1e600c69f36c
15. *Lerner A*. AIOps Platforms. 2017. [Jelektronnyj resurs]. URL: https://blogs.gartner.com/andrew-lerner/2017/08/09/aiops-platforms/
16. Lexico. [Jelektronnyj resurs]. URL: https://www.lexico.com/definition/pipeline
17. *Maldonado M*. How Stress Affects Mental Health. 2018. [Jelektronnyj resurs] URL: https://psychcentral.com/blog/how-stress-affects-mental-health/
18. Scopus. [Jelektronnyj resurs]. URL: https://www-scopus-com.ezproxy.ranepa.ru:2443/term/analyzer.uri?sid=2ee86a84790e019df3a476da7a2d639b&origin=resultslist&src=s&s=TITLE-ABS-KEY%28MLOps%29&sort=plf-f&sdt=b&sot=b&sl=20&count=6&analyzeResults=Analyze+results&txGid=efd45a3c301977e0df5a2bc50c3a0a96
19. Service management good practice [sajt]. URL: https://www.gov.uk/government/publications/public-services-network-psn-service-management-good-practice/service-management-good-practice
20. Text-to-Speech. [Jelektronnyj resurs]. URL: https://cloud.google.com/text-to-speech