Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

(Университет ИТМО)

Факультет ПИиКТ

Образовательная программа Веб-технологии

Направление подготовки (специальность) 09.04.04 Программная инженерия

О Т Ч Е Т

по *научно-исследовательской* работе

Тема задания: Непрерывная интеграция и развертывание в разработке веб-приложений. Этап 2. Сравнительный анализ инструментов непрерывной интеграции и развертывания.

Обучающийся Кочетыгов Андрей Владимирович, P41072

Руководитель практики от университета: *Готская И.Б., профессор, доктор педагогических наук*

Практика пройдена с оценкой **отл**

Подписи членов комиссии:

_____ *профессор, д.педагог.наук, Готская И.Б.*

_____ *доцент, к.педагог.наук, Государев И.Б.*

_____ *преподаватель, Флеров А.В.*

16.06.2020

Санкт-Петербург
2020

Оглавление

Введение

В настоящее время непрерывная интеграция и развертывание достаточно распространенные явления в разработке программного обеспечения. Уже существует большое количество разнообразных инструментов, предоставляющих соответствующие возможности. Однако, малое внимание уделяется сравнению данных инструментов среди тех, кто применяет данные практики, что в особенности касается инструментов, обладающих малой популярностью.

Анализ научных источников по данной тематике показывает отсутствие надлежащих результатов, которые бы проводили емкое сравнение какого-либо набора инструментов. В основном подобные сравнения ограничиваются краткими описаниями рассматриваемых инструментов, что дает малое представление о действительном положении вещей. Тоже самое касается различных электронных источников, в которых, кроме всего прочего, зачастую производится только сравнение инструментов, популярных среди сообщества разработчиков.

Все это не позволяет сформировать четкого представления о схожести и различиях подобных инструментов, что обуславливает **актуальность темы научно-исследовательской работы**.

Целью научно-исследовательской работы является сравнительный анализ инструментов непрерывной интеграции и развертывания.

Объектом исследования выступает практика непрерывной интеграции и развертывания. **Предметом исследования** являются результаты сравнительного анализа инструментов непрерывной интеграции и развертывания. Областью исследования является проектирование, разработка и внедрение программного обеспечения.

Для достижения поставленной цели на 2 этапе научно-исследовательской работы необходимо было выполнить следующие задачи:

— выбрать инструменты, подлежащие проведению сравнительного анализа;
— сформировать список источников составляющих основу проводимого исследования и провести его детальный анализ;
— выполнить обзор сравниваемых инструментов;
— подобрать наиболее важные критерии для проведения сравнительного анализа;
— провести непосредственный сравнительный анализ выбранных инструментов;
— выполнить обзор результатов проведенного сравнительного анализа.

Теоретической основой проводимого исследования выступили работы отечественных (Бабаев В. С., Абасова Н. И.) и зарубежных (Gallaba K., Hilton M., Tratt L., Ghaleb T. A., Wikström A., Zhao Y.) авторов, посвященные непрерывной интеграции и развертыванию, а также сравнительному анализу инструментов непрерывной интеграции и развертывания.

Для решения задач, поставленных в рамках исследования, применялся комплекс теоретических **методов** исследования, таких как сравнительный анализ, обобщение, синтез, индукция и дедукция.

Основной **информационной базой** проводимого исследования послужили портал ResearchGate, поисковая система по полным текстам научных работ Google Scholar, техническая исследовательская база данных IEEE Xplore.

Теоретический результат. Проведенный сравнительный анализ инструментов непрерывной интеграции и развертывания позволил:

— выполнить детальный обзор рассматриваемых инструментов;
— выделить наиболее важные критерии для проведения сравнительного анализа инструментов непрерывной интеграции и развертывания;
— рассмотреть преимущества и недостатки выбранных инструментов;
— сопоставить результаты проведенного сравнительного анализа.

1 Обзор сравниваемых инструментов

1.1 Semaphore

Semaphore — это сервис, предоставляющий услуги непрерывной интеграции и развертывания. Данный сервис позиционирует себя как самый быстрый инструмент на рынке, который работает в 2 раза быстрее других лидирующих сервисов. Работает с кодом проектов, размещенных на GitHub и Bitbucket [15].

Модель оплаты услуг включает в себя как бесплатный план для проектов с открытым исходным кодом, так и платные планы с 14-ти дневным пробным периодом [15]:

— 1300 минут за 10$ в месяц;

— с оплатой за используемые ресурсы:

  — $0.000125 за секунду сборки с использованием Linux/Docker/Android;

  — $0.0005 за секунду сборки под macOS/iOS;

— индивидуально рассчитываемый корпоративный план.

Сервис предоставляет как возможность последовательной сборки и тестирования, так и функцию Boosters, которая обеспечивает многоступенчатое распараллеливание и сокращает продолжительность процессов [15, 18].

Сервис поддерживает сборку, тестирование и развертывание приложений для Node.js, Android, iOS и других платформ. Имеется поддержка языков C/C++, Clojure, Elixir, Go, Java, JavaScript, PHP, Python, Ruby, Scala и Rust. Проекты на других языках, требуют ручной настройки [15, 18].

Поддерживаются такие фреймворки тестирования как: test/unit, RSpec, Cucumber, Steak, Capybara Webkit, Jasmine, Karma, Minitest, Poltergeist, PhantomJS [18].

Одной из особенностей является поддержка Docker-образов, которая позволяет ускорить сборку и тестирование, а также развертывание с помощью Kubernetes [15]. При этом для развертывания доступны такие облачные платформы как: AWS, Heroku, Capistrano и Cloud 66 [18].

Для обеспечения безопасности и максимальной надежности сервис предоставляет возможность хранения API ключей и сертификатов в зашифрованном виде, а также комбинировать автоматически шаги с ручными, чтобы можно было самостоятельно контролировать ключевые этапы конвейера. Кроме того, для построения конвейера имеется специальный визуальный инструмент [15].

Основной поток работы начинается с выбора проекта и его ветки. Далее сервис определяет конфигурацию проекта, извлекая метаинформацию. Следует отметить, что сервис поддерживает настройку конфигураций сразу для нескольких сред. После конфигурирования, начинается тестирование, результат которого в случае неудачи подсвечивается красным, а в случае успеха — зеленым [15, 18].

1.2 CodeShip

Codeship — это один из популярных инструментов непрерывной интеграции и развертывания, который является быстрой и гибкой SaaS-платформой. Данный сервис позволяет за несколько минут, с помощью простого пользовательского веб-интерфейса организовать процессы CI/CD. В то же время предоставляется возможность переключиться на работу с Docker-контейнерами и ручным написанием конфигураций [14].

Сервис заявляет, что обладает улучшенным дизайном системы безопасности, который поддерживает шифрованное внешнее кэширование Docker-образов, а также позволяет настроить среду таким образом, чтобы собираемый код никогда не записывался на машинах сборки, а размещался

только на частных экземплярах систем управления цепочками поставки, таких как GitHub Enterprise.

Сервис позволяет интегрироваться практически с любым инструментов, службой или облаком, которые могут потребоваться для развертывания, уведомлений, анализа покрытия кода тестами, сканирования системой безопасности и т.д.

Надежная и оптимизированная инфраструктура, поддерживающая умное кэширование и распараллеливание, позволяет быстро и стабильно выполнять сборки, а также предоставляет функционал отладки прямо из среды. Кроме того, присутствует возможность настройки команд и разрешений с помощью центра уведомлений.

Сервис предоставляет как версию Basic, так и Pro. Это позволяет либо получить уже настроенную среду, либо выделенные ресурсы для самостоятельной настройки.

Так Codeship Basic использует быстрые, предварительно настроенные виртуальные машины с предустановленным набором необходимых программных средств, чтобы позволить максимально легко и быстро организовать конвейер CI/CD.

Данная версия предоставляет возможность работы с такими языками программирования как: Dart, Elixir, Go, Java, JavaScript, PHP, Python, Ruby, Rust, а также фреймворки Ionic, Meteor.

Обладает интеграцией с такими ресурсами как: anynines, AWS CodeDeploy, AWS Lambda, AWS S3, IBM CloudFoundry, DigitalOcean, AWS Elastic Beanstalk, Engine Yard, Google App Engine, Heroku, Capistrano, Cloud66, а также позволяет работать с FTP, SFTP, SCP, RSYNC, SSH и создавать собственные сценарии развертывания.

Для обеспечения удобства развертывания предоставляются интерфейсы командной строки от AWS, Azure и gcloud.

Предлагает следующие тарифные планы [14]:

— Free — 1 сборка единовременно, 1 тестовый конвейер за $0.

— Starter — 1 сборка единовременно, 2 тестовых конвейера за $49;

— Essential — 2 сборки единовременно, 2 тестовых конвейера за $99;

— Plus — 3 сборки единовременно, 3 тестовых конвейера за $199;

— Power — 4 сборки единовременно, 4 тестовых конвейера за $399;

— Premium — 6 сборок единовременно, 6 тестовых конвейеров за $999.

Codeship Pro, в свою очередь, использует контейнеры построенные на основе Docker-образов, что позволяет гибко настраивать среду CI/CD. Можно выбрать предоставляемый процессор и размер памяти экземпляров AWS.

Данная версия предоставляет возможность работы с такими языками программирования как: Elixir, Go, Java, JavaScript, PHP, Python, Ruby.

Обладает интеграцией с такими сервисами как: AWS, AWS ECS, AWS EKS, Azure Container Service, Google Cloud Functions, Google Cloud Platform, Google Compute Engine, Google Container Engine, Google Container Registry, IBM Cloud Foundry, IBM Cloud Container Service.

А для обеспечения удобства процесса развертывания поддерживает работу с Kubernetes, Docker Swarm, Terraform.

Предлагает следующие тарифные планы [14]:

— Free — 2 VCPU, 3.75 GB памяти за $0;

— Small — 2 VCPU, 3.75 GB памяти за $75;

— Medium — 4 VCPU, 7.5 GB памяти за $149;

— Big — 8 VCPU, 15 GB памяти за $299;

— Huge — 16 VCPU, 30 GB памяти за $599;

— Massive — 32 VCPU, 60 GB памяти за $1199.

Во всех рассмотренных тарифных планах, как для версии Basic, так и для версии Pro предполагается неограниченное количество сборок и проектов, за исключением тарифного плана Free, который ограничен 100 сборками в месяц.

1.3 Buddy

Buddy — это платформа автоматизации, которая упрощает DevOps-процессы для разработчиков, дизайнеров и команд по контролю качества. Платформа предоставляет умное обнаружение изменений, кэширование, параллелизм и всестороннюю оптимизацию [13].

Все тарифные планы включают в себя 14-ти дневный пробный период. Предлагаются следующие планы [13]:

Free, бесплатно: 5 проектов, 500 MB кэша, 1 GB RAM, 2 vCPU, 120 запусков.

Pro, за $75 в месяц: 20 проектов, 10 GB кэша, 3 GB RAM, 2 vCPU, 1 конвейер,

Hyper, от $99 до $19950 в месяц: неограниченные проекта, 20 GB кэша, от 2 до 8 GB RAM, от 2 до 4 vCPU, от 1 до 10 параллельных конвейеров.

Планы Free и Pro для конвейеров сборки предоставляют: 100 заранее подготовленных сценариев; сборку только необходимых изменений; файловую систему для клонирования git-репозиториев, кэширования, результатов сборки; триггеры для различных событий; шаблоны сред; переменные (простые и шифрованные); управление SSH ключами; подключаемые сервисы (Elastic, MariaDB, Memcached, MongoDB, PostgreSQL, RabbitMQ, Redis, Selenium Chrome и Firefox); кэширование зависимостей, Docker-образов, переменных, сервисов, клонов git-репозиториев, файлов, инструкций; мониторинг процесса в реальном времени и его историю; работу с множеством репозиториев.

Для процесса развертывания предоставляются: отслеживание изменений, распараллеливание, атомарные релизы, откаты в один клик, развертывание на любом облаке или сервере, мультипоточная передача по FTP/FTPS/SFTP.

Данными планами предоставляется поддержка таких хостингов кода, как: GitHub, GitHub Enterprise, GitLab, Bitbucket. Кроме того, поддерживается размещение кода на встроенном или собственном Git-хранилище.

Кроме того, данные планы предоставляют интеграции с Slack, DigitalOcean, Azure, UpCloud, Heroku, AWS, Shopify, Google Cloud, Pushbullet, Pushover, NewRelic, Rackspace, Cloudflare, Rollbar, Datadog, Honeybadger, Telegram, Laggly, Bugsnag, Discord, Sentry, Vultr и другими сервисами.

Заявляется поддержка Docker-образов; хранилищ Docker-образов Docker Hub (общедоступных и приватных), Amazon ECR, Google GCR и личных; Docker Layer Caching; создание и переиспользование Docker-образов для различных сред [13].

Для обеспечения безопасности предлагается управление разрешениями, шифрованные переменные, отслеживание и аудит изменений в конвейерах сборки, подтверждение запуска конвейера, авторизация с использованием OAuth.

План Hyper отличается наличием расписанием запусков конвейера, специальными условными триггерами, подключением пользовательских сервисов, а также использованием протокола LDAP.

Всеми планами также поддерживаются [13]:

— языки программирования: JavaScript, Java, Haskell, Go, Elixir, Clojure, C/C++, PHP, Python, Ruby, Scala;
— платформы: Node.js, .NET Core, Android;
— фреймворки: Angular, Ember.js, Aurelia, Gatsby, Django, React Native, React.js;
— статические генераторы: Middleman, Jekyll, Hugo.

1.4 GitLab

GitLab — это сервис, который первоначально представлял собой решение для управления исходным кодом и совместной разработки. Позже он

превратился в интегрированное решение для всего жизненного цикла разработки ПО, а затем и всего DevOps-цикла [2, 11].

Данный сервис обладает открытым исходным кодом и предоставляет систему управления хранилищами кода для Git, систему отслеживания ошибок, CI/CD (для данной функции, в январе 2015 года было выпущено приложение — GitLab Runner) и другие функции [11].

Изначально распространялся под лицензией MIT. В июле 2013 года был разделен на версию Community Edition и Enterprise Edition, с сохранением лицензии MIT. В феврале 2014 года был адаптирован под бизнес-модель Open core и версия EE дополнена функциями, отсутствующими в CE. Модель EE была изменена, но исходных код остался общедоступным [11].

Сервис используется более чем 100000 организациями, включая IBM, Alibaba, Sony, Nasa, CERN, Invincea, O'Reilly, LRZ, GNOME [11].

Сервисом заявляются следующие преимущества использования GitLab CI/CD [20]:

— Интегрированность. GitLab CI/CD является частью GitLab, позволяя осуществлять все от планирования до развертывания;
— Открытый исходный код. CI/CD является частью как GitLab Community Edition с открытым исходным кодом, так и запатентованной GitLab Enterprise Edition;
— Бесшовность. Часть единого приложения GitLab, с одним отличным пользовательским интерфейсом;
— Масштабируемость. Тесты выполняются на отдельных машинах, которых можно добавить сколько угодно;
— Более быстрые результаты. Каждая сборка может быть разделена на несколько заданий, которые выполняются параллельно на нескольких машинах;
— Оптимизация развертывания. Несколько этапов, ручное развертывание, среды и переменные.

Сервисом заявляются следующие основные особенности для GitLab CI/CD [20]:

— Сборка на Unix, Windows, macOS и любой другой платформе, поддерживающей Go;
— Сценарии сборки управляются из командной строки и работают с Java, PHP, Ruby, C и любым другим языком;
— Сборки выполняются на отдельной машине;
— GitLab CI/CD разбивает сборки на несколько машин для быстрого выполнения;
— Журналирование сборки в реальном времени;
— Возможность задавать несколько параллельных заданий;
— Запись этапов процесса сборки в файл gitlab-ci.yml для хранения его в репозитории исходного кода;
— Автомасштабирование используемых ресурсов;
— Возможность загружать исполняемые файлы и другие сборки в GitLab, а также просматривать и скачивать их;
— Возможность локального воспроизведения текстов;
— Поддержка Docker и Kubernetes;
— Встроенный реестр для хранения и совместного использования образов контейнеров;
— Защищенные переменные;
— Возможность определения нескольких сред, включая временные приложения Review Apps, а также возможность просмотреть историю развертывания для каждой среды.

Все тарифные планы включают в себя 30-ти дневный пробный период. Предлагаются следующие планы [20]:

— Free, за $0 в месяц для одного пользователя. Предоставляется 2000 минут сборки в месяц, безлимитные приватные и публичные проекты, безлимитное кол-во участников, а также следующие функции: CI/CD,

жалоб проекта, проектной документации, работы с ChatOps, анализа кода с помощью Sourcegraph;

— Bronze, за $4 в месяц для одного пользователя. Предоставляются все особенности и функции тарифа Free, а также: поддержка на следующий рабочий день, многочисленные утверждения при просмотре кода, подтверждение слияний, анализ качества кода;

— Silver, за $19 в месяц для одного пользователя. Предоставляется 10000 минут сборки в месяц, все особенности и функции плана Bronze, а также: доска группировки жалоб, планы развития, приоритет оказания поддержки, много-проектные графы конвейеров, доски поставки, запланированное и ручное поэтапное развертывание;

— Gold, за $99 в месяц для одного пользователя. Предоставляется 50000 минут сборки в месяц, все особенности и функции плана Silver, а также: управление портфолио, гостевые пользователи, оповещения о производительности приложений, доска безопасности, сканирование контейнеров, динамическое тестирование безопасности приложений, мониторинг кластеров Kubernetes.

1.5 CircleCI

CircleCI — это гибкая платформа непрерывной интеграции и доставки, которая помогает разработчикам ПО быстро и уверенно выпускать код, автоматизируя процесс сборки, тестирования и развертывания. Данная платформа работает с репозиториями кода GitHub, GitHub Enterprise и Bitbucket. Сборка и тестирование осуществляются в чистых контейнерах или виртуальных машинах. Предоставляется возможность получения уведомлений о проблемах в процессе сборки, при интеграции с соответствующими сервисами. Успешные сборки развертываются в требуемых средах [16].

Сервис предоставляет возможности определения и упорядочения выполняемых заданий, запуска любого Docker-образа, возможность создания Docker-образова, доступ к Docker Layer Caching, возможность настройки используемых ресурсов (CPU/RAM), возможность отладки по SSH или запуска заданий в локальной среде для быстрого устранения проблем, LDAP для управления пользователями, ведение журналов аудита, полная изоляция виртуальных машин и многое другое.

Также сервис обеспечивает мощное кэширование, которое ускоряет работу конвейеров за счет кэширования изображений, исходного кода, зависимостей и пользовательских видов кэширования. Предоставляет доступ к состоянию, продолжительности процессов и информации о потребляемых ресурсах.

Заявляется поддержка любого языка программирования, который может быть построен на Linux, Windows или macOS, включая C++, JavaScript, PHP, Python и Ruby.

Дается возможность собирать проекты с использованием предоставляемого облака, либо с использованием частной инфраструктуры:

— Облако. Установка, защита и поддержка вашего CI экземпляра. Мгновенный доступ к новейшим функциям. Автоматические обновления, устраняющие необходимость в обслуживании. Быстрая авторизация с помощью GitHub или Bitbucket. Поддержка построения в среде Docker, Linux, Android, Windows, macOS, а также репозиториев кода GitHub, Bitbucket.

— Сервер. Установка CircleCI на частный сервер. Самостоятельное управление обслуживанием для повышения безопасности. Полный доступ к возможностям администрирования системы. Обновления, настраиваемые в соответствии с графиком технического обслуживания сервера. Поддержка построения в среде Docker, Linux, Android, Windows, macOS, а также репозиториев кода GitHub, GitHub Enterprise.

Имеются интеграции с сервисами Amazon, Azure, Google Cloud, JFrog, Kublr, packagecloud, Sonatype, Codecov, CodeScene, ConfigCat, Coveralls, datree. Packtracker, realMethods, Jira и множеством других платформ и сервисов.

Предлагаются следующие тарифные планы [16]:

— Free, бесплатно: 2500 кредитов в неделю, единовременный запуск 1-ой работы, сборка на Linux и Windows;

— Performance, от $30 в месяц ($15 за одного пользователя). Включает в себя все, что и тариф Free, а также: 25000 кредитов за каждые $15, автоматическую масштабируемость для удовлетворения спроса на ресурсы, построение на macOS, доступ к большим мощностям, Docker Layer Caching;

— Custom, за индивидуально подбираемую цену. Включает в себя все, что и тариф Performance, а также: вычисления на GPU, доступ к круглосуточной поддержке, настраиваемая годовая оплата;

— Self-hosted, за $35 в месяц для одного пользователя. Требует минимум 20 пользователей. Расчет индивидуальной оплаты доступен для команд численностью более 100 человек.

Кредиты, используемые в планах, являются валютой, которая потребляется по разным ставкам в зависимости от того, какая конфигурацию сборки используется.

2 Подбор критериев для сравнительного анализа

2.1 Поддерживаемые языки программирования

Данный критерий является существенным при проведении сравнительного анализа инструментов, поскольку, на данный момент, в мире существует огромное множество различных языков программирования, которые в той или иной степени применяются при решении реальных задач. В частности, данный критерий важен в отношении динамически типизированных языков, так как наибольший процент применения непрерывной интеграции относится именно к проектам, в основе которых лежат языки программирования с динамической типизацией [4, 7]. Высокий процент использования, в данном случае, связан с тем, что в языках с динамической типизацией имеется высокий риск возникновения ошибок во время исполнения программы, так как на стадии разработки отсутствуют инструменты статического анализа кода, который бы позволил выявить проблемные участки еще до запуска программного продукта в производственной среде [8].

2.2 Поддерживаемые хранилища кода

Поддержка различных хранилищ исходного кода, с системой контроля версий, является одним из важных критериев для инструментов непрерывной интеграции и доставки, поскольку они являются неотъемлемой составной частью любого конвейера сборки, тестирования и доставки. Именно из данных хранилищ на конвейер поступает исходных код, который нужно обработать [6, 9, 10].

Помимо поддержки традиционных сервисов, вроде GitHub или Bitbucket, немаловажной является также поддержка хранилищ, развернутых с использованием частной инфраструктуры, так как многие пользователи

озадачиваются вопросами безопасности хранения и обработки исходных кодов своих программных продуктов.

2.3 Скорость сборки и тестирования

И наконец последний, но наиболее важный критерий. Высокая важность данного критерия обусловлена тем, что при низких показателях скорости сборки по факту теряется большая часть смысла в осуществлении непрерывной интеграции и доставки, так как данный процесс по своей сути требует частых итераций, чего невозможно достичь без оптимизации скорости сборки.

Процесс сборки, тестирования и доставки должен занимать не более нескольких минут. В частности, верхним пределом являются 10 минут. В то время как 5 минут считает неплохим временем для объемного проекта с обширной кодовой базой [3].

Исследования же показывают, что продолжительность процесса интеграции составляет от 8 до 90 минут (средняя продолжительность около 20 минут). При этом 84% сборок выполняется свыше приемлемого предела в 10 минут, 40% сборок выполняются более 30 минут [5].

2.4 Выводы по разделу

Таким образом, в ходе подбора критериев для проведения сравнительного анализа инструментов непрерывной интеграции были выделены следующие основные критерии:

1. Поддерживаемые языки программирования;
2. Поддерживаемые хранилища кода;
3. Скорость сборки и тестирования.

3 Сравнительный анализ инструментов

3.1 Поддерживаемые языки программирования

В рамках проведенного ранее обзора инструментов было установлено, что инструмент Semaphore заявляет встроенную поддержку языков программирования C/C++, Clojure, Elixir, Go, Java, JavaScript, PHP, Python, Ruby, Scala и Rust. Однако проекты на других языках требуют самостоятельной настройки.

Сервис CodeShip заявляет о поддержке таких языков программирования как: Dart, Elixir, Go, Java, JavaScript, PHP, Python, Ruby и Rust.

Сервис Buddy заявляет о поддержке: JavaScript, Java, Haskell, Go, Elixir, Clojure, C/C++, PHP, Python, Ruby, Scala.

Сервисы GitLab и CircleCI заявляют о поддержке любого языка программирования, который может быть собран на платформах Linux, Windows или macOS.

Поскольку, как было обозначено ранее, непрерывная интеграция чаще всего применяется в проектах с динамически типизированными языками, то для сравнения по данному критерию было решено выбрать только 5 наиболее популярных, согласно индексу TIOBE, языков с динамической типизацией.

Индекс TIOBE представляет собой рейтинг популярности языков программирования, работающий на основе подсчета результатов поисковых запросов, содержащих название языка [1].

Пятью наиболее популярными языками с динамической типизацией являются Python (3 место), JavaScript (7 место), PHP (8 место), Ruby (15 место) и Perl (18 место), по состоянию на май 2020 года [17]. Сравнение поддержки данных языков представлено в таблице 1.

Таблица 1 — Поддержка языков инструментами непрерывной интеграции

	Semaphore	**CodeShip**	**Buddy**	**GitLab**	**CircleCI**
Python	+	+	+	+	+
JavaScript	+	+	+	+	+
PHP	+	+	+	+	+
Ruby	+	+	+	+	+
Perl	-	-	-	+	+

Данное сравнение дает понять, что рассматриваемые инструменты предоставляют возможность работы с основными популярными динамически типизированными языками. Однако в то же время можно заметить, что не все инструменты поддерживают работу с языком Perl, что, скорее всего, связано с серьезным падением популярности языка по сравнению, например, с ситуацией на 2016 год (с 9 на 18 место) [12].

3.2 Поддерживаемые хранилища кода

В современной разработке программного обеспечения, когда встает вопрос о выборе хранилища исходного кода, то наиболее подходящими решениями являются GitHub, GitLab и Bitbucket. При этом, если остро стоит вопрос безопасности, то это может быть частное хранилище исходного кода. Также возможно использование GitHub Enterprise, в рамках которого предлагается пользоваться возможностями GitHub на частной инфраструктуре.

Сравнение поддержки данных хранилищ исходного кода представлено в таблице 2.

Таблица 2 — Поддержка хранилищ кода инструментами непрерывной интеграции

	Semaphore	**CodeShip**	**Buddy**	**GitLab**	**CircleCI**
GitHub	+	+	+	+	+
GitLab	-	+	+	+	+
GitHub Enterprise	-	+	+	+	+
Bitbucket	+	+	+	+	+
Частные	-	+	+	+	+

Как видно из данного сравнения — большинство из представленных инструментов обладают возможностью интеграции с рассматриваемыми вариантами хранилищ. Однако из общей картины выбивается только Semaphore, который не поддерживает GitLab, GitHub Enterprise, BitBucket.

3.3 Скорость сборки и тестирования

Для проведения сравнительного анализа скорости сборки и тестирования проектов было принято решения использовать кодовую базу одного из уже существующих программных продуктов, с открытым исходным кодом. При этом выбираемый программный продукт должен быть достаточно развитым и популярным среди сообщества, что с большой долей вероятности гарантирует значимое покрытие его исходного кода всевозможными тестами. Наличие большого количества тестов, в свою очередь, позволит обеспечить максимально реалистичную и близкую к производственной среде нагрузку на инструменты непрерывной интеграции.

В качестве такого программного продукта было принято решение использовать JavaScript-фреймворк Svelte, который представляет собой компилятор, преобразующий декларативные компоненты в эффективный JavaScript код, который с высокой точностью обновляет только необходимые участки DOM [19]. Данный фреймворк набирает популярность среди

сообщества и обладает набором тестов, который обширно покрывает исходный код. Он также является частью платформы Node.js.

Поскольку, как описывалось ранее, хранилища кода с поддержкой контроля версий являются неотъемлемой частью процесса непрерывной интеграции и тестирования, то для размещения исходного кода выбранного программного продукта необходимо выбрать вариант подобного размещения, который бы позволил без каких-либо проблем интегрировать его со всеми сравниваемыми инструментами непрерывной интеграции.

Двумя наиболее простыми вариантами, в данном случае, являются GitHub и Bitbucket. Данные хранилища поддерживаются всеми представленными для сравнения инструментами и позволяют не усложнять процесс анализа разворачиваем хранилища в частной инфраструктуре. В конечном итоге, было решено использовать хранилище GitHub.

В процессе подготовки инструментов непрерывной интеграции было решено использовать уже готовые конфигурации для Node.js, который предлагаются самими сервисами. Для упрощения конфигурирования инструментов непрерывной интеграции из тестов были убраны элементы, требующие запуска браузера. Для замеров времени сборки и тестирования было решено использовать встроенные метрики соответствующих инструментов.

Было решено выполнить серию из 6 запусков конвейера сборки и тестирования. Первый запуск рассматривается отдельно, в целях исключения из процесса механизмов кэширования. Результаты последующих запусков представлены в виде усредненных значений времени сборки и тестирования, измеряемых в секундах.

Первый запуск конвейера для инструмента Buddy дал результат равный 58 секундам. Результат успешной сборки и тестирования, с помощью данного инструмента, представлен на рисунке 1.

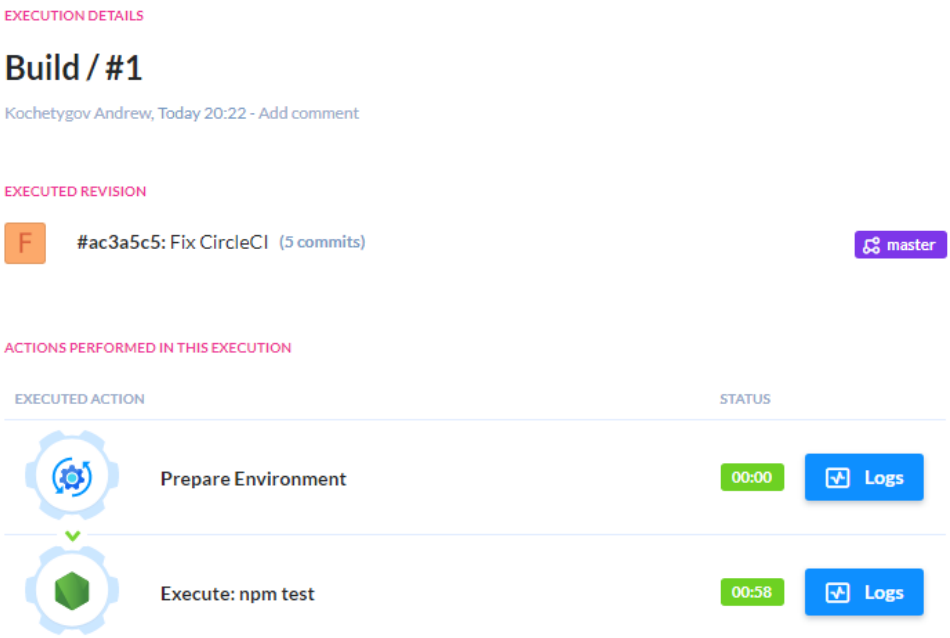

Рисунок 1 — Результат успешной сборки и тестирования, с помощью Buddy

Первый запуск конвейера для инструмента CircleCI дал результат равный 69 секундам. Результат успешной сборки и тестирования, с помощью данного инструмента, представлен на рисунке 2.

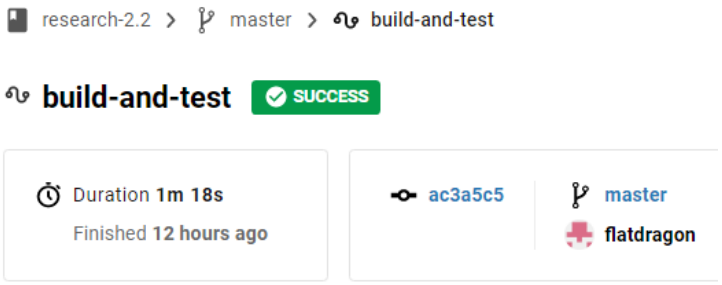

Рисунок 2 — Результат успешной сборки и тестирования, с помощью CircleCI

Первый запуск конвейера для инструмента Semaphore дал результат равный 131 секунде. Результат успешной сборки и тестирования, с помощью данного инструмента, представлен на рисунке 3.

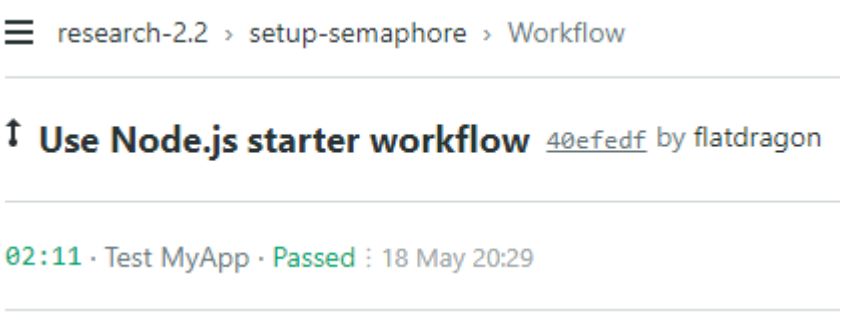

Рисунок 3 — Результат успешной сборки и тестирования, с помощью Semaphore

Первый запуск конвейера для инструмента GitLab дал результат равный 316 секундам. Результат успешной сборки и тестирования, с помощью данного инструмента, представлен на рисунке 4.

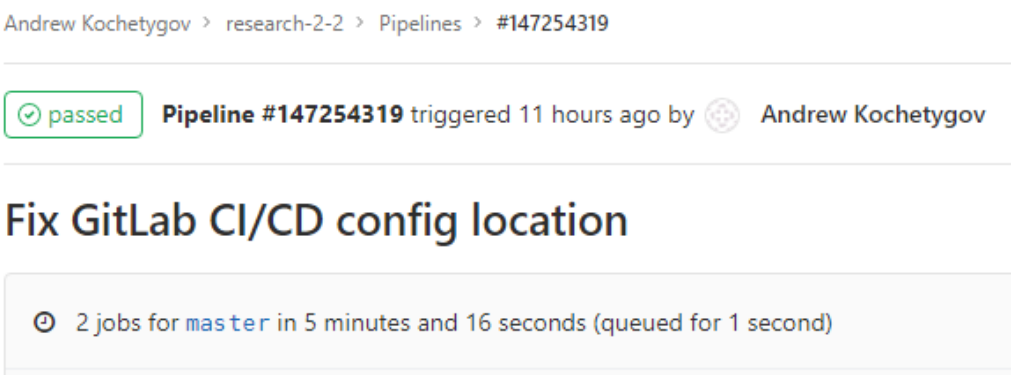

Рисунок 4 — Результат успешной сборки и тестирования, с помощью GitLab

Первый запуск конвейера для инструмента CodeShip дал результат равный 210 секундам. Результат успешной сборки и тестирования, с помощью данного инструмента, представлен на рисунке 5.

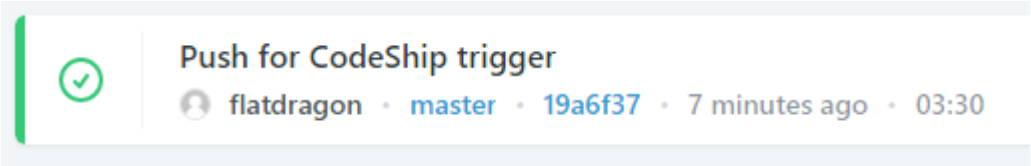

Рисунок 5 — Результат успешной сборки и тестирования, с помощью CodeShip

Таким образом, в рамках первого запуска конвейера сборки и тестирования, были получены результаты, представленные в таблице 3, а их визуализация на рисунке 6.

Таблица 3 — Продолжительность сборки и тестирования при первом запуске конвейера

	Semaphore	**CodeShip**	**Buddy**	**GitLab**	**CircleCI**
Продолжительность (секунды)	131	210	58	316	69

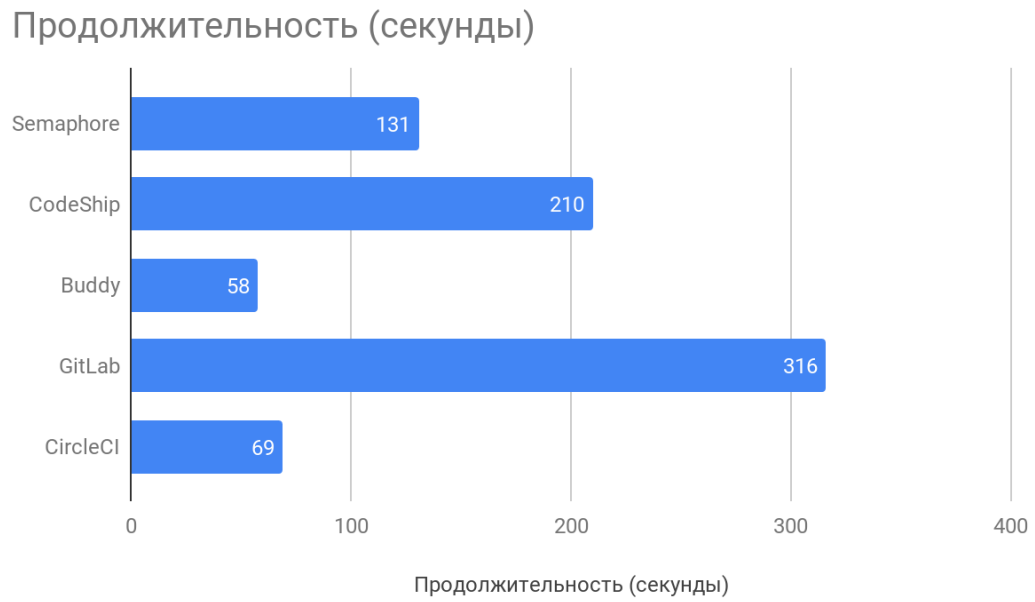

Рисунок 6 — Продолжительность сборки и тестирования, с помощью разных инструментов, измеряемая в секундах

Из представленной таблицы видно, что, по результатам первого запуска, лидирующие позиции в скорости сборки и тестирования занимают инструменты Buddy и GitLab.

В серии последующих запусков были получены результаты, представленные в таблице 4, а визуализация времени средней продолжительности сборки и тестирования представлена на рисунке 7.

Таблица 4 — Скорость сборки и тестирования при первом запуске конвейера

№ запуска	Semaphore, сек.	CodeShip, сек.	Buddy, сек.	GitLab, сек.	CircleCI, сек.
1	101	148	48	293	68
2	77	212	49	281	77
3	83	199	50	303	114
4	73	165	48	293	91
5	77	254	48	290	101
В среднем	82	196	49	292	90

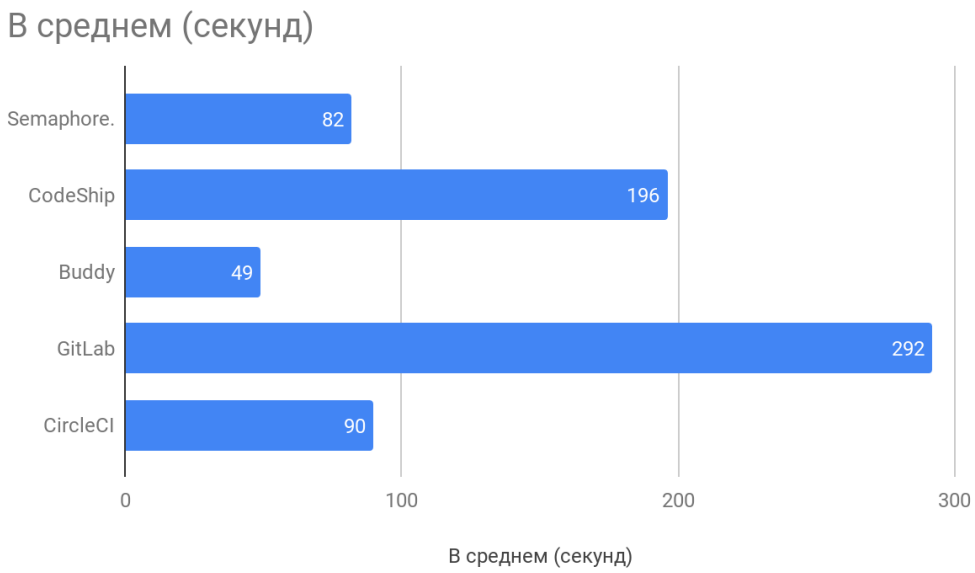

Рисунок 7 — Средняя продолжительность сборки и тестирования, с помощью разных инструментов, измеряемая в секундах

Из представленной таблицы видно, что, по результатам серии из 5 запусков, лидирующие позиции в скорости сборки и тестирования, по-прежнему, занимают инструменты Buddy и GitLab. При этом инструменты Buddy и Codeship заявляют о наличии умного кэширования. Однако инструмент Buddy выглядит самым производительным на фоне других инструментов.

Заключение

В результат проведенного исследования был выполнен обзор инструментов непрерывной интеграции и доставки. Были освещены поддерживаемые данными инструментами платформы и языки программирования, а также поддерживаемые интеграции с другими сервисами. Кроме того, были представлены тарифные планы и услуги, которые в них входят.

Далее были изучены работы авторов, которые относятся к теме непрерывной интеграции и выделены наиболее значимые, с точки зрения подобных инструментов, критерии для проведения сравнительного анализа выбранных инструментов.

Сравнительный анализ выбранных инструментов непрерывной интеграции и развертывания показал, что большинство рассматриваемых инструментов поддерживают интеграцию с основными вариантами хранилищ исходных кодов, а также поддерживают наиболее популярные языки с динамической типизацией.

По результатам проверки на продолжительность сборки и тестирования, со стандартной конфигурацией для Node.js, предлагаемой инструментами, было установлено, что данный процесс с использованием Semaphore занимает в среднем 82 секунды, с использованием CodeShip в среднем 196 секунд, с использованием Buddy в среднем 49 секунд, с использованием GitLab в среднем 292 секунд, с использованием CircleCI в среднем 90 секунд. Ни один из инструментов не превысил порога в 10 минут.

В дальнейшем предполагается разработка приложение непрерывной интеграции и доставки, которое раскроет практическую значимость данной исследовательской работы.

Список литературы

1. Абасова, Н. И. Анализ рейтингов языков программирования при их выборе / Н. И. Абасова, Н. А. Лаврухина // Информационные технологии и проблемы математического моделирования сложных систем. — Иркутск : ИИТМ ИрГУПС, 2009. — Вып. 7 — С. 5–13.
2. Бабаев, В. С. Исследование современных CI/CD инструментов и внедрение их в крупные web-проекты на примере Jenkins / В. С. Бабаев, А. В. Гунько // «Научное сообщество студентов XXI столетия. Технические науки»: Электронный сборник статей по материалам LXXIV студенческой международной научно-практической конференции. — Новосибирск : АНС «СибАК». — 2019. — № 2(73). — С. 148–150.
3. Хамбл, Д. Непрерывное развертывание ПО: автоматизация процессов сборки, тестирования и внедрения новых версий программ / Д. Хамбл, Д. Фарли // Вильямс. — Москва, 2011. — 432 с.
4. Gallaba, K. Use and Misuse of Continuous Integration Features / K. Gallaba // Institute of Electrical and Electronics Engineers (IEEE). — 2020. — Vol. 46. — P. 33–50.
5. Ghaleb, T. A. An Empirical Study of the Long Duration of Continuous Integration Builds / T. A. Ghaleb, D. A. da Costa, Y. Zou // Empirical Software Engineering. — 2019. — Vol. 24. — № 4. — P. 2102–2139.
6. Hilton, M. Continuous integration (CI) needs and wishes for developers of proprietary code / M. Hilton, N. Nelson, D. Dig, T. Tunnell, D. Marinov // Oregon State University, Dept. of Computer Science, Tech. Rep. — 2016. — 11 p.
7. Hilton, M. Usage, Costs, and Benefits of Continuous Integration in Open-Source Projects / M. Hilton, T. Tunnell, K. Huang, D. Marinov, D. Dig // Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering. — IEEE, 2016. — P. 426–437.

8. Tratt, L. Dynamically typed languages / L. Tratt // Advances in Computers. — 2019. — Vol. 46. — P. 149–184.
9. Wikström, A. Benefits and challenges of Continuous Integration and Delivery — A Case Study / A. Wikström // University of Helsinki. — Helsinki, 2019. — 33 p.
10. Zhao, Y. The impact of continuous integration on other software development practices: a large-scale empirical study / Y. Zhao, A. Serebrenik, Y. Zhou, V. Filkov, and B. Vasilescu // In Proceedings 32nd International Conference on Automated Software Engineering. — IEEE, 2017. — P. 60–71.
11. GitLab — Википедия [Электронный ресурс]. — URL: https://ru.wikipedia.org/wiki/GitLab (Дата обращения: 16.05.2020).
12. История языков программирования: Perl — необычный язык, созданный лингвистом для программистов / Хабр [Электронный ресурс]. — URL: https://habr.com/ru/post/305402/ (Дата обращения: 19.05.2020).
13. Buddy: The DevOps Automation Platform [Электронный ресурс]. — URL: https://buddy.works/ (Дата обращения: 16.05.2020).
14. Continuous Integration, Deployment & Delivery with Codeship [Электронный ресурс]. — URL: https://codeship.com/ (Дата обращения: 15.05.2020).
15. Continuous Integration & Delivery — Semaphore [Электронный ресурс]. — URL: https://semaphoreci.com/ (Дата обращения: 14.05.2020).
16. Continuous Integration and Delivery — CircleCI [Электронный ресурс]. — URL: https://circleci.com/ (Дата обращения: 16.05.2020).
17. index | TIOBE - The Software Quality Company [Электронный ресурс]. — URL: https://tiobe.com/tiobe-index/ (Дата обращения: 19.05.2020).
18. Semaphore (Software) [Электронный ресурс]. — URL: https://en.wikipedia.org/wiki/Semaphore_(software) (Дата обращения: 14.05.2020).
19. sveltejs/svelte: Cybernetically enhanced web apps [Электронный ресурс]. — URL: https://github.com/sveltejs/svelte (Дата обращения: 19.05.2020).

20. The first single application for the entire DevOps lifecycle — GitLab [Электронный ресурс]. — URL: https://about.gitlab.com/ (Дата обращения: 16.05.2020).