УДК 338.242: 658.562.012.7
DOI: 10.24412/2312-6647-2026-248-157-171

ОЦЕНКА ЭФФЕКТИВНОСТИ ПРОЕКТОВ ПО ТЕСТИРОВАНИЮ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ИСПОЛЬЗОВАНИЕМ МЕТОДОЛОГИЙ AGILE И WATERFALL

Юрий Викторович Фролов

Московский городской педагогический университет,
Москва, Россия,
jury_frolov@mail.ru

Дмитрий Дмитриевич Жаворонков

Московский городской педагогический университет,
Москва, Россия,
dima030498@gmail.com

Аннотация. Актуальность исследования связана с необходимостью выбора проектных методологий, используемых в процессе тестирования программного обеспечения, в условиях динамично изменяющихся требований и сроков разработки. В статье выполнено сравнение эффективности методологий Agile и Waterfall на примере работы отделов внедрения программного обеспечения в ИТ-компаниях. Предложены метрики, с помощью которых может выполняться оценка эффективности проектных методологий для программных продуктов разного назначения. Результаты исследования показывают, что Agile-методология позволяет повысить гибкость проектного процесса, ускорить выявление ошибок в программных продуктах и снизить затраты на исправление дефектов. В то же время подход Waterfall обеспечивает более строгий контроль по этапам разработки, но может быть менее эффективным в условиях изменения требований.

Ключевые слова: методологии управления проектами, водопадная модель Waterfall, гибкая модель Agile, тестирование программного обеспечения, метрики эффективности

UDC 338.242: 658.562.012.7
DOI: 10.24412/2312-6647-2026-248-157-171

PROJECT PERFORMANCE ASSESSMENT SOFTWARE TESTING USING AGILE AND WATERFALL METHODOLOGIES

Yuri Viktorovich Frolov

Moscow City University,
Moscow, Russia,
jury_frolov@mail.ru

Dmitry Dmitrievich Zhavoronkov

Moscow City University,
Moscow, Russia,
dima030498@gmail.com

Abstract. The relevance of the study is related to the need to choose design methodologies used in the software testing process, in the context of dynamically changing requirements and development deadlines. The article compares the effectiveness of Agile and Waterfall methodologies using the example of software implementation departments in IT companies. Metrics are proposed, with the help of which assessment of efficiency of design methodologies for software products of various purposes can be performed. The results of the study show that Agile methodology can increase the flexibility of the project process, speed up the identification of errors in software products and reduce the cost of fixing defects. At the same time, Waterfall's approach provides greater control over development phases, but may be less effective in changing requirements.

Keywords: project management methodologies, Waterfall model, Agile model, software testing, performance metrics

Введение

В условиях современных высокотехнологичных рынков, характеризующихся быстрыми изменениями, ИТ-компании сталкиваются с необходимостью обеспечения высокого качества программного обеспечения (ПО) адекватно происходящим изменениям [1]. Тестирование ПО является одним из ключевых этапов разработки, который позволяет выявить и исправить ошибки, а также обеспечить надлежащее качество — соответствие программного продукта функциональным и нефункциональным требованиям заказчика.

Функциональное тестирование — это проверка того, что компьютерная программа выполняет действие так, как ожидают от нее разработчики [2]. Например, кнопка «Сохранить» действительно сохраняет файл, поиск на сайте выдает запрошенные пользователем результаты, а калькулятор правильно складывает числа.

Нефункциональное тестирование — это проверка того, в какой степени программный продукт удовлетворяет требованиям скорости обработки информации, надежности, удобства и безопасности [2]. Например, сайт компании работает, даже если на него зайдут одновременно 10 тыс. пользователей, приложение не расходует заряд батареи телефона в фоновом режиме, а персональные данные пользователя защищены от кражи.

Функциональное и нефункциональное тестирование ПО включает в себя различные методы и подходы, направленные на выявление ошибок и дефектов в коде, а также на обеспечение качества продукта, заданного функциональными и нефункциональными требованиями к ПО.

Цель исследования заключалась в сравнительном анализе методологий создания программного обеспечения и разработке методов оценки эффективности проектов на примере программных продуктов разного назначения.

Для достижения цели исследования по созданию и использованию методов, которые бы позволили сравнивать технологии управления проектами на примере разработки программного обеспечения, были поставлены следующие задачи:

– изучить особенности разработки ПО на примере ИТ-компаний, продвигающих на рынок корпоративные информационные системы и мобильные приложения;

– проанализировать используемые в ИТ-компаниях проектные методологии, выявить их сильные и слабые стороны;

– получить доступ к статистическим данным, связанным с разработкой и тестированием программного обеспечения разного назначения, выполнить экспериментальные исследования, направленные на сравнительный анализ метрик эффективности процессов тестирования ПО в двух ИТ-компаниях;

– оценить и проанализировать экономический эффект от использования двух проектных методологий в процессах создания и тестирования программных продуктов разного назначения, разработать рекомендации для ИТ-компаний.

Выполнение поставленных задач направлено на повышение эффективности создания программного обеспечения в ИТ-компаниях. В итоге успешное выполнение этих задач позволит ИТ-компаниям на стадии планирования проектов по созданию программного обеспечения оценивать перспективность применения методологий проектирования и используемых подходов к тестированию разрабатываемого ПО. Все это в совокупности даст возможность принимать более обоснованные управленческие решения в сфере разработки и тестирования ПО.

Анализ методологий проектирования программного обеспечения

Методологии разработки программного обеспечения играют ключевую роль в определении того, как будет организован весь процесс разработки ПО, его тестирования и внедрения программных продуктов. Применение разных подходов

влияет на взаимодействие между проектной командой и заказчиком, на сроки разработки, а также на качество и функциональность конечного продукта [2].

Гибкая модель Agile — это методология, которая позволяет адаптировать процесс разработки под изменения в требованиях от заказчика и бизнес-условиях [3–5]. Одним из самых популярных фреймворков Agile является Scrum, который предполагает выполнение работы в рамках коротких циклов (спринтов), регулярные встречи специалистов для оценки хода выполнения проекта и постоянную связь с заказчиком для уточнения требований [4]. Длительность спринта составляет обычно 1–4 недели. По его итогам выпускается продукт, обладающий набором важных для заказчика характеристик. Это позволяет снижать риски, связанные с внедрением в разрабатываемый программный продукт невостребованных заказчиком решений, и оперативно исправлять ошибки в ПО.

В Agile важен фокус на тесное взаимодействие с заказчиком в процессе создания ПО. Заказчик активно участвует в процессе, предоставляет обратную связь и помогает выделить приоритетные задачи. Важный критерий оценки деятельности проектной команды — это своевременный выпуск программного продукта, соответствующего по своей функциональности очередному этапу разработки (спринту). Таким образом, Agile — это методология, ориентированная на быструю доставку заказчику ценности, встроенной в программный продукт. Проектная технология в этом случае базируется на возможности внесения изменений в продукт и на непрерывном улучшении профессионализма членов команды путем получения обратной связи от заказчика[1].

Один спринт (цикл) в Agile применительно к разработке ПО можно описать как последовательность следующих этапов (рис. 1):

1. Анализ требований. В Agile анализ пронизывает весь проект, так как требования могут меняться. На этом этапе анализируются текущие потребности, исследуется, какие изменения должны быть реализованы в программном продукте.

2. План. Команда определяет, что и когда будет сделано. В Agile это происходит в рамках планирования спринта: создается дорожная карта проекта по разработке программного продукта, включая задачи и сроки.

3. Дизайн. Разрабатываются архитектурные решения и интерфейсы ПО. В Agile дизайн также часто итеративен и детали дизайна ПО могут уточняться по мере разработки.

4. Разработка. Выполняется написание программного кода и создается функционал программного продукта.

5. Тестирование. Осуществляется проверка функционала ПО на соответствие требованиям и на наличие ошибок. В Agile тестирование происходит непрерывно в процессе разработки. Каждая новая предлагаемая заказчиком функция тестируется, после того как она написана разработчиками, чтобы убедиться в ее корректности.

---

[1] Международный стандарт ИСО 9000:2000 Системы менеджмента качества. Фундаментальные принципы и словарь. URL: https://docs.cntd.ru/document/1200015260 (дата обращения: 07.05.2026).

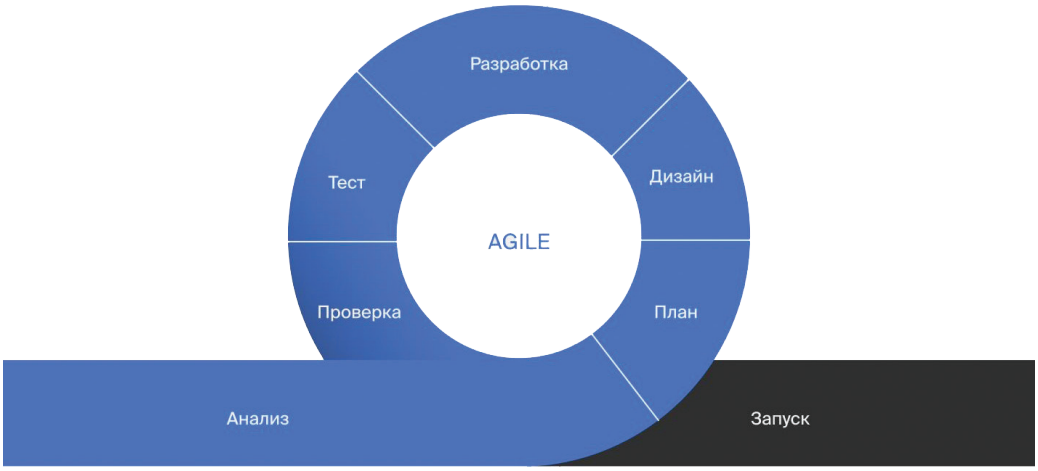

Рис. 1. Методология Agile

6. Проверка. Подтверждается, что продукт соответствует всем требованиям к программному продукту, установленным заказчиком. В Agile проверка происходит в рамках каждого спринта после завершения очередного этапа разработки программного продукта.

7. Запуск. Программный продукт или его части выпускаются в рабочую среду (например, на сервер ИТ-компании или заказчика). В Agile запуск может происходить по мере завершения спринтов, что дает возможность клиентам или пользователям сразу испытывать обновления в ПО.

После этапа запуска программный продукт, созданный по итогам спринта, становится доступен конечным пользователям. В Agile все перечисленные выше этапы тесно связаны и могут повторяться несколько раз в рамках разных спринтов, позволяя постоянно улучшать и адаптировать программный продукт к пожеланиям заказчика. По итогам спринта версия программного продукта готова к использованию в рабочих процессах заказчика, а также к обратной связи и доработкам в следующих спринтах.

Каскадная (водопадная) модель Waterfall является классической методологией, основанной на последовательности следующих этапов: планирование, проектирование, разработка, тестирование, внедрение и обслуживание [5][2]. Эта методология предполагает строгую последовательность выполнения всех этапов: каждый следующий этап начинается только после завершения предыдущего (см. рис. 2). Waterfall идеально подходит для проектов с четко определенными, предварительно заданными требованиями и с минимальными изменениями в процессе разработки ПО.

[2] Также см.: *Кучер А.* Agile или Waterfall. Какой метод организации выбрать для работы с заказчиками? URL: https://habr.com/ru/articles/664300 (дата обращения: 28.01.2025); *Фомина Л.* Битва титанов: Waterfall VS Agile — какую методологию управления проектами выбрать. URL: https://practicum.yandex.ru/blog/vodopadnaya-model-waterfall/ (дата обращения: 28.01.2025); *Скворцов В.* Waterfall — история большой ошибки. URL: https://pmclub.pro/articles/waterfall-istoriya-bolshoj-oshibki/ (дата обращения: 28.01.2025).

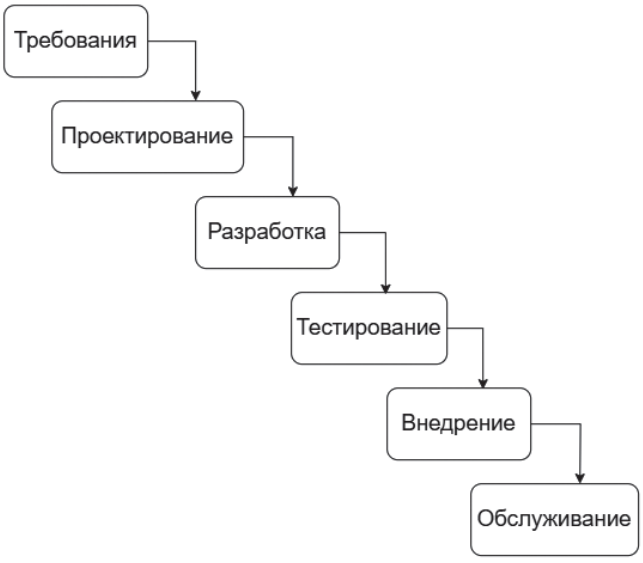

Рис. 2. Методология Waterfall

На первом этапе собираются все требования к программному продукту и тестировщики изучают их, чтобы понять, что именно нужно будет тестировать в процессе его создания.

На этапе проектирования создаются архитектурные решения и технические спецификации. Здесь также определяется, как будет происходить тестирование.

В ходе третьего этапа осуществляется разработка: происходит написание программного кода, а тестировщики начинают подготовку к тестированию, определяя тестовые сценарии на основе спецификаций.

На четвертом этапе выполняется проверка работоспособности создаваемого программного продукта (информационной системы) согласно требуемым спецификациям и функциональности.

На пятом этапе после завершения тестирования созданный программный продукт внедряется на рабочие серверы.

В ходе шестого этапа осуществляется послепродажное обслуживание ПО, а именно дополнительное тестирование для выявления ошибок в программном продукте, допущенных в процессе его создания.

В отличие от итеративных моделей разработки ПО, таких как Agile, в Waterfall тестирование программного продукта начинается только после завершения этапа разработки. Это означает, что тестировщики не могут вносить изменения в код или требования на ранних этапах, а тестируют продукт, который уже полностью разработан. Ошибки, выявленные на поздних этапах, могут потребовать значительных переработок. Следовательно, не исключен возврат в проекте на этапы формулировки требований и проектирования, что увеличивает время и стоимость разработки[3].

[3] *Кучер А.* Agile или Waterfall.

В отличие от Agile, где заказчик участвует в процессе разработки на каждом этапе, в Waterfall заказчик может быть вовлечен в проект только на начальном этапе сбора требований и на финальном этапе в процессе сдачи-приемки продукта заказчиком. Это может привести к тому, что заказчик не будет удовлетворен конечным продуктом, если его требования не были в полной мере учтены на стадии разработки. В Waterfall большое внимание уделяется разработке и выполнению тестов по заранее подготовленным сценариям, которые создаются параллельно с формулировкой требований и проектированием ПО.

Таким образом, Waterfall плохо работает в проектах, где требования могут изменяться в процессе создания ПО. Если заказчик или команда выявляют новые потребности на промежуточных этапах проекта, то это может повлечь за собой необходимость пересмотра всего проекта, что также приведет к увеличению затрат и сроков[4].

Одним из главных преимуществ Waterfall является четкая структура и последовательность этапов. Каждый этап разработки и тестирования описывается с помощью документации, а переход к следующему этапу возможен только после завершения предыдущего этапа. Такая технология разработки позволяет мониторить ход и результаты проекта по созданию ПО.

Система с четким планированием и без изменения требований в ходе разработки позволяет точно прогнозировать сроки выполнения и бюджет проекта. Это особенно важно для крупных проектов с фиксированными сроками и бюджетом, например для государственных заказов или для клиентов, заказывающих создание или модернизацию крупных корпоративных информационных систем.

Сравнение методологий Waterfall и Agile применительно к проектам создания программных продуктов представлено в таблице 1.

Результаты и их обсуждение

Были выполнены сравнительные исследования методологий Waterfall и Agile на примере тестирования создаваемого ПО в двух ИТ-компаниях.

Компания 1 занимается разработкой и применением корпоративных программных решений для клиентов класса ERP (Enterprise Resource Planning). Компания 2 специализируется на создании мобильных приложений для потребительского рынка.

В ходе исследования были предложены метрики, которые позволяют сравнить эффективность двух методологий тестирования программных продуктов разного назначения, создаваемых в этих компаниях (см. табл. 2).

В таблице 2 используются следующие обозначения и показатели.

ФТ — функциональное тестирование.

НФТ — нефункциональное тестирование.

[4] *Фомина Л.* Битва титанов: Waterfall VS Agile...

Таблица 1

Сравнение методологий Waterfall и Agile

Критерий	Waterfall-тестирование	Agile-тестирование
Тестирование как этап проекта	Тестирование проводится после завершения разработки	Тестирование выполняется параллельно с разработкой, что позволяет быстро выявлять и исправлять ошибки
Фокус тестирования	Направлено на проверку соответствия продукта заранее установленным требованиям	Нацелено на обеспечение качества продукта в соответствии с меняющимися ожиданиями заказчика
Роль тестировщика	Выполняет роль инспектора, который ищет ошибки в продукте	Является активным участником команды, который помогает улучшать продукт
Инструменты и методы	Используются традиционные инструменты и методы тестирования	Применяются гибкие инструменты и методы тестирования
Документация	Создается подробная документация по тестам	Объем документации минимален
Внесение изменений	Сложно вносить изменения в процесс разработки ПО и его тестирования после начала проекта	Можно вносить изменения в процесс разработки и тестирования на любом этапе создания ПО

Таблица 2

Метрики эффективности тестирования корпоративной информационной системы и мобильного приложения в двух ИТ-компаниях

Метрики / виды тестирования	Компания 1				Компания 2			
	ФТ		НФТ		ФТ		НФТ	
Методология	Waterfall	Agile	Waterfall	Agile	Waterfall	Agile	Waterfall	Agile
Критические ошибки, шт	30	15	20	10	2	1	1	1
Обычные ошибки, шт	20	15	15	10	2	1	1	5
Количество выполненных тестов, шт	200	300	200	250	15	25	15	20
Время тестирования, часы	100	80	120	80	12	8	10	6
Простои, часы	20	5	25	10	3	1	2	0,5
Эффективность тестирования, шт/ч	0,56	1,08	0,44	0,60	0,68	0,75	0,40	1.0

Эффективность тестирования программного обеспечения.

$$E = \left(\frac{C \times w_C + O \times w_O}{T} \right) \times \left(\frac{T}{H + D} \right) \times \left(1 - \frac{D}{H + D} \right), \quad (1)$$

где E — показатель эффективности тестирования; C — количество выявленных критических ошибок; O — количество выявленных обычных ошибок; W_c, W_o — веса для критических и обычных ошибок; T — количество выполненных тестов; H — время тестирования (в часах); D — время простоев (в часах).

Критичная ошибка — это ошибка, которая полностью блокирует работу программы или делает ее невозможной для использования. Примеры: приложение не запускается; основная функция программы не работает; обнаружены потери или повреждения данных пользователя; ошибка приводит к сбою системы. Последствия критической ошибки: пользователь не может выполнить ключевые задачи, программа становится бесполезной. Такие ошибки должны исправляться в первую очередь. Для них в настоящем исследовании был установлен весовой коэффициент $W_c = 2$.

Обычная ошибка — это ошибка, не затрагивающая функции системы, от которых зависит ее работоспособность. Такие ошибки вызывают неудобства или небольшие проблемы в процессе эксплуатации информационной системы. Примеры: неправильное отображение текста или интерфейса; незначительное замедление работы программы. Последствия обычной ошибки: пользователь может продолжать использовать программный продукт, но испытывает при этом дискомфорт. Обычные ошибки исправляются после фиксации и устранения критичных ошибок. Для них в исследовании был установлен весовой коэффициент $W_o = 1$.

Предложенная формула для расчета показателя эффективности тестирования программных продуктов включает следующие компоненты.

С помощью отношения количества выявленных в ходе тестирования ошибок к общему количеству выполненных тестов оценивается индекс качества тестирования:

$$\left(\frac{C \times w_C + O \times w_O}{T} \right).$$

Чем больше ошибок было выявлено по итогам тестирования программного продукта, тем выше показатель эффективности.

Второй компонент формулы показателя эффективности тестирования указывает на его производительность (количество выполненных тестов за единицу времени):

$$\left(\frac{T}{H + D} \right).$$

Чем больше производительность тестирования, тем выше показатель эффективности.

Следующий компонент предложенной формулы эффективности сигнализирует о влиянии простоев на процесс тестирования:

$$\left(1-\frac{D}{H+D}\right).$$

Это выражение можно интерпретировать как коэффициент полезного использования времени, который изменяется в диапазоне от 0 до 1. Например, если время простоев D растет и в сумме общего времени на тестирование уменьшается доля времени непосредственно на само тестирование H, то отношение D / (H + D) будет стремиться к единице, а значение коэффициента полезного использования времени будет приближаться к нулю. Как следствие, эффективность тестирования будет уменьшаться.

Единицей измерения предложенного показателя эффективности является количество тестов в час. При расчете данного показателя учитываются также метрики качества (количество выявленных ошибок в программном продукте), важность найденных ошибок (степень их критичности, влияющая на эксплуатацию ПО), коэффициент использования времени на тестирование, сигнализирующий, в частности, об использовании средств автоматизации тестирования.

Необходимо отметить, что предложенный показатель эффективности позволяет сравнивать проекты по разработке, выполняемые для программных продуктов разного назначения и с использованием разных методологий управления проектами.

Чтобы привести показатель эффективности тестирования к оценкам в денежном выражении, необходимо иметь примерное значение стоимости нормо-часа тестирования.

Проведем оценочный расчет. Анализ, выполненный на базе компаний 1 и 2 (см. выше), показывает, что стоимость одного часа работы тестировщика (S) составляет примерно 1 500 руб/час.

Тогда общие затраты на проведение тестирования (включая простои) составят (H + D) S.

Среднюю стоимость выявления одной ошибки в разработанном ПО можно подсчитать по формуле:

$$\text{Стоимость одной ошибки} = \frac{(H+D)\times S}{C\times w_C + O\times w_O}. \qquad (2)$$

Таблица 3 иллюстрирует полученные расчетные результаты по сравнению метрик эффективности тестирования в денежном выражении.

На основе выполненных исследований по анализу эффективности тестирования программных продуктов разного назначения в ИТ-компаниях после их перехода с технологии Waterfall на использование проектной методологии Agile были получены следующие результаты.

Таблица 3

Результаты сравнительного экономического анализа тестирования ПО

Метрики \ виды тестирования	Компания 1				Компания 2			
	ФТ		НФТ		ФТ		НФТ	
Методология	Waterfall	Agile	Waterfall	Agile	Waterfall	Agile	Waterfall	Agile
Эффективность тестирования, шт/ч	0,56	1,08	0,44	0,60	0,68	0,75	0,40	1,0
Затраты на проведение тестирования, рубли	180 000	127 000	217 500	135 000	22 500	13 000	18 000	9 750
Стоимость нахождения одной ошибки, рубли/шт	2 250	2 833	3 955	4 500	3 750	4 500	6 000	1 393

Эффективность функционального тестирования (ФТ) в Компании 1, занимающейся разработкой систем класса ERP, выросла на 93 % с 0,56 до 1,08. В Компании 2, которая занимается созданием мобильных приложений, рост показателя эффективности функционального тестирования составил 10 % (увеличился с 0,68 до 0,75).

Эффективность нефункционального тестирования (НФТ) в Компании 1 увеличилась на 36 % — с 0,44 до 0,60, а в Компании 2 рост показателя эффективности составил 2,5 раза (с 0,40 до 1,0).

Одновременно с повышением эффективности проектов по созданию ПО переход на методологию Agile позволяет снизить затраты на проведение тестирования. Так, при проведении функционального тестирования Компания 1 сократила затраты на 29 %, с 180 000 до 127 000 руб., а Компания 2 — на 42 %, с 22 500 до 13 000 руб. Аналогичным образом, как показывают результаты, представленные в таблице 3, сокращаются расходы и на проведение нефункционального тестирования.

Таким образом, результаты экспериментального исследования подтвердили преимущества методологии Agile, которая позволяет проводить тестирование параллельно с разработкой, что сокращает время на поиск и исправление ошибок.

В то же время следует отметить, что результаты исследования указывают на увеличение стоимости нахождения одной ошибки при переходе ИТ-компаний на проектную методологию Agile. Это связано с тем, что применение методологии Agile за счет встроенных в процесс создания ПО проверок содействует, с одной стороны, увеличению затрат времени на тестирование, но с другой стороны, способствует повышению качества создаваемых программных продуктов, а следовательно, приносит выгоды компаниям в процессе и по итогам их взаимодействия с клиентами.

Заключение

Выполнены исследования по сравнению эффективности методологий Waterfall и Agile на примере создания и тестирования программного обеспечения в двух ИТ-компаниях, занимающихся разработкой и продвижением корпоративных информационных систем и мобильных приложений.

Предложена формула, позволяющая оценивать эффективность тестирования программного обеспечения применительно к созданию программных продуктов разного назначения с применением двух технологий управления проектами. Установлено, что переход ИТ-компаний на проектную методологию Agile позволяет улучшить качество программных продуктов и ускорить процесс разработки путем встраивания тестирования в спринты. При этом ошибки в программных продуктах выявляются на ранних этапах выполнения проектов.

Благодаря возможности тестирования в рамках каждого спринта Agile, ИТ-компании снизили затраты на исправление ошибок, а также стали быстрее реагировать на изменения требований заказчика. Это, в свою очередь,

повысило удовлетворенность заказчиков, так как программный продукт в процессе создания развивался и улучшался с учетом их пожеланий.

Результаты выполненного исследования показывают, что в целом переход ИТ-компаний на проектную методологию Agile оправдан, но требует оптимизации процесса тестирования для снижения стоимости нахождения ошибки. В целях повышения эффективности функционального и нефункционального тестирования в Agile необходимо внедрять методы непрерывной интеграции (CI/CD) и инструменты автоматизации тестирования [10, 11], чтобы снизить стоимость нахождения ошибок. Также исследование показало, что методология Waterfall остается эффективной для проектов с заранее заданными и фиксированными требованиями заказчика, в которых планирование тестирования программных продуктов осуществляется на этапе формирования требований к проекту.

Список источников

1. Современная {цифровая} дидактика / Р. Х. Абдюханов [и др.]. М.: А-Приор, 2023. 140 с.

2. Святенко А. С. Подходы к управлению распределенными командами тестирования и их экономическая эффективность // Наука и бизнес: пути развития. 2020. № 9 (111). С. 44–47.

3. Аппело Ю. Agile-менеджмент: Лидерство и управление командами / пер. с англ. А. Олейник; под ред. А. Обуховой. М.: Альпина Паблишер, 2021. 534 с.

4. Сазерленд Д. Scrum. Революционный метод управления проектами / пер. с англ. А. Гескиной. М.: Манн, Иванов и Фербер, 2016. 288 с.

5. Чернов Ю. Искусство Agile-тестирования. М.: БХВ, 2024. 208 с.

6. Фролов Ю. В., Босенко Т. М., Жаворонков Д. Д. Автоматизация процесса тестирования конфигураций «1С:ERP Управление предприятием» в компании-интеграторе // Международный научно-исследовательский журнал. 2024. № 12 (150). URL: https://research-journal.org/archive/12-150-2024-december/10.60797/IRJ.2024.150.160 (дата обращения: 18.12.2024). https://doi.org/10.60797/IRJ.2024.150.160

7. Босенко Т. М., Фролов Ю. В. Применение облачных платформ глубокого и машинного обучения студентами в условиях дистанционного образования // Актуальные проблемы теории и практики обучения физико-математическим и техническим дисциплинам в современном образовательном пространстве: IV Всероссийская (с международным участием) научно-практическая конференция, посвященная 75-летию факультета физики, математики, информатики Курского государственного университета, Курск, 16–17 декабря 2020 года. Курск: Курский гос. ун-т, 2020. С. 414–417.

References

1. Sovremennaya {cifrovaya} didaktika / R. X. Abdyuxanov [i dr.]. M.: A-Prior, 2023. 140 s.

2. Svyatenko A. S. Podxody` k upravleniyu raspredelenny`mi komandami testirovaniya i ix e`konomicheskaya e`ffektivnost` // Nauka i biznes: puti razvitiya. 2020. № 9 (111). S. 44–47.

3. Appelo Yu. Agile-menedzhment: Liderstvo i upravlenie komandami / per. s angl. A. Olejnik; pod red. A. Obuxovoj. M.: Al`pina Pablisher, 2021. 534 s.

4. Sazerlend D. Scrum. Revolyucionny`j metod upravleniya proektami / per. s angl. A. Geskinoj. M.: Mann, Ivanov i Ferber, 2016. 288 s.

5. Chernov Yu. Iskusstvo Agile-testirovaniya. M.: BXV, 2024. 208 s.

6. Frolov Yu. V., Bosenko T. M., Zhavoronkov D. D. Avtomatizaciya processa testirovaniya konfiguracij “1S:ERP Upravlenie predpriyatiem” v kompanii-integratore // Mezhdunarodny`j nauchno-issledovatel`skij zhurnal. 2024. № 12 (150). URL: https://research-journal.org/archive/12-150-2024-december/10.60797/IRJ.2024.150.160 (data obrashheniya: 18.12.2024). https://doi.org/10.60797/IRJ.2024.150.160

7. Bosenko T. M., Frolov Yu. V. Primenenie oblachny`x platform glubokogo i mashinnogo obucheniya studentami v usloviyax distancionnogo obrazovaniya // Aktual`ny`e problemy` teorii i praktiki obucheniya fiziko-matematicheskim i texnicheskim disciplinam v sovremennom obrazovatel`nom prostranstve: IV Vserossijskaya (s mezhdunarodny`m uchastiem) nauchno-prakticheskaya konferenciya, posvyashhennaya 75-letiyu fakul`teta fiziki, matematiki, informatiki Kurskogo gosudarstvennogo universiteta, Kursk, 16–17 dekabrya 2020 goda. Kursk: Kurskij gos. un-t, 2020. S. 414–417.

Информация об авторах / Information about the authors

Юрий Викторович Фролов — кандидат технических наук, доктор экономических наук, профессор, профессор департамента информатики, управления и технологий Института цифрового образования, Московский городской педагогический университет, Москва, Россия.

Yuri Viktorovich Frolov — PhD in Technology, Doctor of Economics, Professor, Professor of the Department of Informatics, Management and Technology at the Institute of Digital Education, Moscow City University, Moscow, Russia.

jury_frolov@mail.ru

Дмитрий Дмитриевич Жаворонков — аспирант Института цифрового образования, Московский городской педагогический университет, Москва, Россия.

Dmitry Dmitrievich Zhavoronkov — Postgraduate Student of the Institute of Digital Education, Moscow City University, Moscow, Russia.

dima030498@gmail.com