УДК 004.042

5.2.2. Математические, статистические и инструментальные методы экономики (физико-математические науки, экономические науки)

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ТРАДИЦИОННОЙ И LOW-CODE РАЗРАБОТКИ: ЭФФЕКТИВНОСТЬ, ГИБКОСТЬ, ПЕРСПЕКТИВЫ

Кушнир Надежда Владимировна
старший преподаватель кафедры информационных систем и программирования
РИНЦ-SCIENCE INDEX SPIN-код=6951-4012
kushnir.06@mail.ru
ФГБОУ ВО "Кубанский государственный технологический университет", 350020, улица Московская, 2, Краснодар, Россия

Урбанович Мария Владимировна
студентка 3 курса
mariaurbanviolin@gmail.com
ФГБОУ ВО "Кубанский государственный технологический университет", Краснодар, Россия, 350020, улица Московская, 2, Краснодар, Россия

В данной статье исследуются различия между традиционной и low-code разработкой программного обеспечения, их эффективность, гибкость и перспективы применения. Актуальность темы обусловлена ростом популярности low-code платформ, упрощающих создание программных продуктов и позволяет ускорить цифровую реализацию бизнеса. Однако остается открытым вопрос о границах применимости данного подхода и его влиянии на качество и масштабируемость ПО. Результат исследования авторов заключается в формировании рекомендаций по выбору оптимальной стратегии разработки с учетом специфики бизнеса и требований к программному обеспечению. Результаты исследования могут быть полезны тимлидам, разработчикам, руководителям проектов и бизнес-аналитикам, принимающим решения о внедрении новых технологий в разработку ПО

Ключевые слова: LOW-CODE, РАЗРАБОТКА, РЕАЛИЗАЦИЯ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, АНАЛИЗ

http://dx.doi.org/10.21515/1990-4665-208-058

UDC 004.042

5.2.2. Mathematical, statistical and instrumental methods of economics (physical and mathematical sciences, economic sciences)

COMPARATIVE ANALYSIS OF TRADITIONAL AND LOW-CODE DEVELOPMENT: EFFICIENCY, FLEXIBILITY, PROSPECTS

Kushnir Nadezhda Vladimirovna
Senior Lecturer at the Department of Information Systems and Programming
RSCI-SCIENCE INDEX. SPIN code=6951-4012
kushnir.06@mail.ru
Kuban State Technological University, Krasnodar, Russia, 350020, Moskovskaya, 2

Urbanovich Maria Vladimirovna
3rd year student
mariaurbanviolin@gmail.com
FGBOU VO "Kuban State Technological University", Krasnodar, Russia, 350020, Moskovskaya, 2

The current article investigates the differences between traditional and low-code software development, their efficiency, flexibility and application prospects. The relevance of the topic is due to the growing popularity of low-code platforms that simplify the creation of software products and allow to accelerate the digital realization of business. However, the question about the limits of applicability of this approach and its influence on the quality and scalability of software remains open. The result of the authors' research is the formation of recommendations for choosing the optimal development strategy taking into account the specifics of business and software requirements. The results of the study can be useful for team leaders, developers, project managers and business analysts who make decisions on the introduction of new technologies in software development

Keywords: LOW-CODE, DEVELOPMENT, IMPLEMENTATION, SOFTWARE, ANALYSIS

Введение.

Мир становится всё более цифровым, и компании стремятся оставаться максимально гибкими и конкурентоспособными. Но для этого, прежде всего, необходимо оптимизировать собственные ресурсы, чтобы быстрее внедрять соответствующие проекты и решения. А поскольку сегодня скорость важна в любом процессе разработки программного обеспечения, организации спешат воспользоваться возможностью ускорить его разработку. В результате выбора у них остается два варианта: low-code и традиционная разработка.

Чтобы сделать выбор между этими двумя вариантами, технические директора, ИТ-директора, руководители групп разработчиков и другие руководители должны учитывать различные критерии, такие как скорость, стоимость, гибкость бизнеса и т.д. И неудивительно, что благодаря всем преимуществам, которые предлагает low-code, он получил значительное распространение. По некоторой статистике, можно спрогнозировать, что в 2025 году 70% новых приложений, разрабатываемых организациями, будут использовать технологии low-code или no-code, по сравнению с менее чем 25% в 2020 году. И все же, несмотря на такую популярность, это решение должно быть подкреплено практическими финансовыми и стратегическими соображениями [1].

Основные понятия

Low-code и традиционные подходы к программированию имеют некоторые явные различия: у каждого есть свои преимущества и недостатки, но они часто пересекаются в рамках одних и тех же рабочих процессов и могут быть интегрированы в одну эффективную и действенную стратегию разработки.

Разработка с low-code – это подход, при котором разработчик реализует некоторые элементы приложения с помощью предварительно созданных программных модулей, часто выбираемых через интерфейс

перетаскивания элементов, для создания желаемой функциональности. При low-code подходе разработчики обычно пишут меньше строк кода, чем при традиционной разработке приложений, хотя обычно с некоторым минимальным уровнем кодирования для настройки модулей.

Традиционная разработка – данный подход подразумевает создание веб-сайтов и приложений посредством ручного кодирования с нуля. Эти приложения создаются на заказ и масштабируются, проектируются и разрабатываются в соответствии с поставленными конкретными бизнес-целями и их обновлениями. Каждая часть будущего веб-сайта или приложения требует индивидуального, уникального кода, и команде разработчиков необходимо будет проектировать его с нуля.

Сравнительный анализ

Главное и самое очевидное преимущество разработки low-code заключается в том, что это быстро. Готовые модули сокращают время на реализацию функциональности приложения, поэтому команды разработчиков могут сосредоточиться на задачах, требующих большей оригинальности или имеющих более высокий приоритет для бизнеса. Разработка low-code также может помочь разработчикам интегрировать приложение с внешней платформой, не изучая все тонкости этой внешней платформы. Например, согласно исследованию, проведенному компанией Forrester, платформы с низким кодом ускоряют разработку примерно в пять-десять раз, что является очень значительным показателем [2].

Ускоренному проектированию способствуют некоторые аспекты low-code платформ, такие как заранее запрограммированные компоненты приложений и встроенные инструменты моделей данных, которые облегчают создание и управление базами данных, поскольку несколько микросервисов обрабатываются без вмешательства пользователя.

Использование low-code платформы позволяет запустить продукт за 3 месяца, тогда как традиционная разработка заняла бы от 6 месяцев до года.

Из этого следует прямолинейное влияние выбора стратегии разработки на количество разработчиков, а как следствие, на издержки и траты его обслуживания.

Техническое обслуживание – важный аспект, который следует рассматривать, поскольку после запуска приложения его следует регулярно обслуживать и обновлять. Безопасность, обслуживание и обновления приложений выполняемые платформой с низким кодом позволяет разработчикам тратить минимальные ресурсы на регулярное обслуживание [3].

Традиционные методы разработки требуют специальную команду, которая выполняет регулярные обновления и поддерживает индивидуальный веб-сайт или приложение, и компании нужно иметь сотрудника, который отвечает за контроль приложения, чтобы гарантировать бесперебойность и безопасность работы.

Например: компания-разработчик создает банковское ПО, где критически важны безопасность, высокая нагрузка и сложные бизнес-процессы. Использование традиционной разработки позволяет создать архитектуру, обеспечивающую надежность, масштабируемость и возможность точной настройки под нужды банка.

Обслуживание, как следствие, влияет на расходы. Поэтому, другим важным фактором является экономическая эффективность. Решения с малым кодом обычно более экономически эффективны по сравнению с традиционными разработками, но все сводится к потребностям и целям бизнеса. Следует учесть стоимость покупки конструктора приложений, готовых шаблонов, а также дополнительных плагинов или лицензий.

Для оптимизации денежных ресурсов и получения быстрого результата, можно использовать гибридный подход: low-code платформа отвечает за интерфейс клиентских сервисов, а бэкенд реализован традиционными методами, что позволяет поддерживать высокую нагрузку и сложные бизнес-процессы.

Несмотря на то, что решения low-code часто обновляются, чтобы предоставить команде разработчиков свои лучшие возможности разработки с помощью широкого спектра уже существующих шаблонов, функций и возможностей, но данные улучшения работают исключительно в рамках данной платформы. Расширяться и настраивать приложение сверх возможностей платформы – не может быть осуществимым.

На рисунке 1 представлена Диаграмма Венна, показывающая общие черты и различия между подходами, основанными на моделировании, прикладными платформами с низким уровнем кода и разработкой программного обеспечения с низким уровнем кода. На данной диаграмме можно заметить, что взаимосвязь между традиционным принципом разработки и low-code.

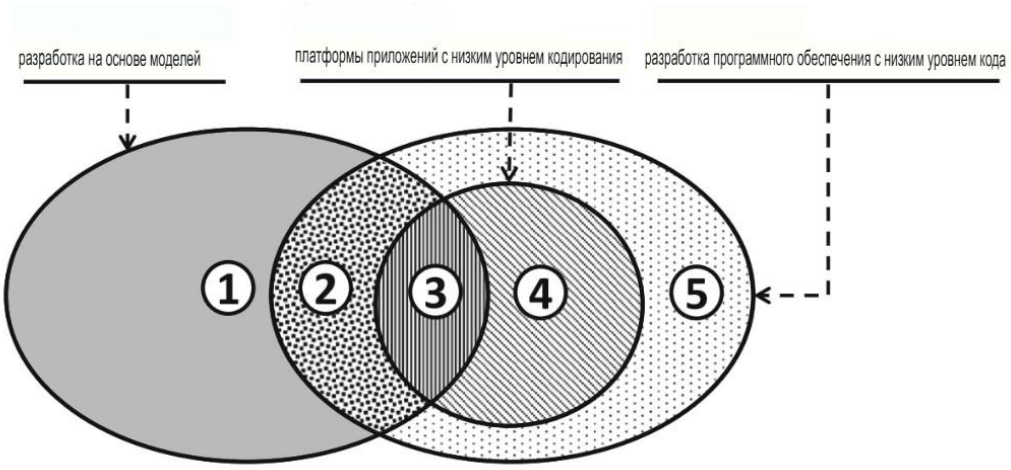

Рисунок 1 – Диаграмма Венна для low-code принципов

Принцип реализации гибридного подхода

Наиболее эффективный способ воспользоваться преимуществами low-code – применить преимущества двух стратегий разработки и определить, где целесообразно включение low-code подходов в рабочие процессы.

Большинство корпоративных разработчиков давно внедрили методы low-code в определенные части своих рабочих процессов. Например, если IDE автоматически завершает строки кода или автоматически заполняет имена переменных, то эта функциональность приближается к low-code разработке, хотя традиционные поставщик инструментов разработки редко продвигают ее таким образом. Внешние модули или сторонние API, которые интегрируют функциональность в приложение, также сродни low-code программированию [4].

Для большинства стратегий разработки наиболее эффективно сочетание традиционной разработки и подходов с низким кодом. Несмотря на то, что на рынке представлены различные платформы, специально предназначенные для разработки с низким кодом, сами по себе они, скорее всего, не смогут полностью удовлетворить потребности организации в разработке.

ЛИТЕРАТУРА

1. А.С. Бок и Ю.Франк, “Low-Code платформа”, стр. 733–740, декабрь 2021. Ссылка: https://doi.org/10.1007/s12599-021-00726-8

2. Д. Русцио, Д. Коловос, Ж. де Лара, А. Пьерантонио, М. Тиси, и М.Виммер, “Low-code разработка и проектирование на основе моделей: две стороны одной медали?” стр. 437–446, апрель 2022. Ссылка: https://doi.org/10.1007/s10270-021-00970-2

3. Р.Важковски, “Low-code платформа для автоматизации бизнес-процессов на производстве” в IFAC-PapersOnLine, 2019, стр. 376–381. Ссылка: https://doi.org/10.1016/j.ifacol.2019.10.060

4. Г. Холбарт, “Low-Code, No-Code. Что под капотом?” IT Проффесионал, стр. 4–7, 2021. Ссылка: https://doi.org/10.1109/MITP.2021.3123415

REFERENCES

1. A.C. Bok i Ju.Frank, “Low-Code platforma”, str. 733–740, dekabr' 2021. Ssylka: https://doi.org/10.1007/s12599-021-00726-8

2. D. Ruscio, D. Kolovos, Zh. de Lara, A. P'erantonio, M. Tisi, i M.Vimmer, “Low-code razrabotka i proektirovanie na osnove modelej: dve storony odnoj medali?” str. 437–446, aprel' 2022. Ssylka: https://doi.org/10.1007/s10270-021-00970-2

3. R.Vazhkovski, “Low-code platforma dlja avtomatizacii biznes-processov na proizvodstve” v IFAC-PapersOnLine, 2019, str. 376–381. Ssylka: https://doi.org/10.1016/j.ifacol.2019.10.060

4. G. Holbart, “Low-Code, No-Code. Chto pod kapotom?” IT Proffesional, str. 4–7, 2021. Ssylka: https://doi.org/10.1109/MITP.2021.3123415