

Open Source решения для выбора платформы разработки программного обеспечения

А. С. Беляев¹, Ф. А. Десятириков²

¹Компания Haultmont, Черногория

²Воронежский государственный университет
felixdes@ieee.org

Аннотация. Проведен сравнительный анализ различных подходов к изменению стека технологий ПО в условиях существенных возмущений рынка продуктов разработки. Формализована задача выбора стека технологий в условиях нелинейной среды разработки ПО. На примере платформы Jmix показана необходимость генерации средств разработки с открытым кодом. Показаны преимущества Less-Code технологических решений.

Ключевые слова: стек технологий; Open Source; Jmix; Less-Code; Low-Code

I. ПОСТАНОВКА ЗАДАЧИ

Современное состояние рынка продуктов ИТ разработки характеризуется наличием плохо прогнозируемых возмущений, связанных, главным образом, с социальными и общественными потребностями различных игроков рынка и не обусловлены научно-техническими причинами развития ИТ отрасли. В связи с этим приходится искать уникальные решения по изменению ИТ среды субъектов рынка программного обеспечения в каждой конкретной ситуации. Общим, однако, является тот факт, что цифровизация всех отраслей деятельности в последних десятилетиях как в России, так и во всем мире развивалась без концептуальных ограничений на ресурсы (за исключением финансов) производства программного обеспечения. Естественными причинами смены стека технологий конкретных ИТ проектов были, таким образом, развивающиеся согласно жизненному циклу [1, 2] программные средства, выходящие на рынок в декларируемые и прогнозируемые сроки. Такое эволюционное развитие рынка ПО отражалось на формировании спроса на надежное функциональное продуктовое обеспечение в условиях ценового баланса и качественного покрытия запросов конечного пользователя. Формировалась тенденция к использованию лицензионного ПО известных мировых лидеров, как, например, SAP, Microsoft, Oracle. В свою очередь этот стабильный спрос отражался на качестве доработки программного инструментария под конкретного заказчика, что повышало уровень доверия продукту, стимулировало адаптацию пользователя и повышало его информационную зависимость от вендора. Эта спираль спрос-предложение снижает информационную безопасность потребителя в условиях резких изменений на рынке ПО.

Целью настоящей работы является изучить возможные решения по цифровизации в условиях сильных возмущений рыночной среды высоких технологий. Задачами исследования являются изучение условий разработки программного обеспечения,

устойчивого к колебаниям рынка ПО. Методические аспекты соответствующей миграции стека технологий рассмотрены на примере платформы Jmix [3].

II. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Смена ИТ платформы в процессе разработки ПО, как и вообще при цифровизации, вызывает процессы неспецифической адаптации в системе разработки [4]. Проявлением таких процессов является, в частности, увеличение сопротивления среды, которое вызвано нежеланием изменять действующую технологическую платформу особенно в тех случаях, когда качество конечного продукта находится в допустимых пределах, а замена стека повлечет ещё и значительные финансовые риски. Такое противодействие новациям – это известное проявление субъективного фактора в ИТ проекте, названное Виктором Фадеевым, коммерческим директором платформы Jmix, «психологической ловушкой» для пользователей, которые привыкли к текущему стеку ПО. В [5] предложено рассматривать три основных подхода к решению задачи смены ИТ платформы проекта – коробочное решение, использование LowCode/LessCode платформы и заказная разработка ПО. Каждый подход имеет свои характеристики скорости внедрения, возможности доработки, технологического суверенитета и требует начальной квалификации разработчиков [6].

Коробочное решение представляет собой готовое ПО, разработанное для решения определенного набора отраслевых задач, функционально пригодное для заказчиков из данной отрасли, но требующее адаптации к конкретному пользователю. Внедрение и настройка продукта, может быть произведена вне компетенции разработчика коробочного решения. Так же в текущем режиме деятельности компании проводится доработка и расширение продукта.

Таким образом, преимуществом коробочных решений является сравнительно небольшая стоимость внедрения и высокая скорость запуска ПО в среде пользователя. Среди ограничений использования коробочного подхода можно указать следующие. Во-первых, динамическое ценообразование и регулярные платежи за пользование ПО. И даже при высокой стоимости такие решения не покрывают полный необходимый пользователю функционал, в связи с тем, что изначально эти решения разрабатываются в инвариантной парадигме. Последнее ведёт за собой необходимость доработки до нужного функционала с отсечением избыточных для пользователя возможностей ПО. Коробочные решения, как правило, не рассчитаны

на изначальное сопряжение с ИТ средой предприятия пользователя и поэтому требуют адаптации к ресурсной базе, в том числе, к квалификации ИТ-сотрудников. Информационная безопасность пользователя коробки является низкой, так как данные агрегируются на сервере вендора, что частично восполняется ограниченной техподдержкой коробочного решения.

Заказная разработка ПО проводится по индивидуальному техническому заданию с полным описанием цели, задачи, инфраструктуры и требований заказчика. Такая разработка связана с невозможностью удовлетворить потребность заказчика на рынке готового ПО в силу того, что решаются либо узкопрофильные, уникальные, инновационные задачи, либо ресурсы на интеграцию готового ПО существенно превышают бюджет.

К преимуществам заказной разработки ПО относятся следующие его характеристики. Во-первых, такой продукт, будучи разработан по техническому заданию заказчика, является уникальным и полностью удовлетворяет необходимому функционалу. ПО является собственностью заказчика, что снимает вопрос регулярных платежей за его использование и масштабирование. При этом принадлежность ПО заказчику, с одной стороны, предотвращает доступ к ПО и данным компании третьих лиц, а с другой стороны, обеспечивает возможность развития ПО в соответствии с развитием ИТ компании заказчика. Что не исключает последующего взаимодействия с вендором. Так, например, если изменившийся целевой функционал компании повлечет за собой необходимость существенной доработки ПО, то обеспечение качества продукта компании заказчика в новых условиях может потребовать инновационных ИТ решений и может быть выполнено вендором продукта (см., например, [7]). Следующим преимуществом заказного ПО является его максимальная дружелюбность к ИТ среде компании и в отношении интеграции с внешними сервисами.

Есть, однако, и недостатки такого подхода. Во-первых, это высокая начальная стоимость такого решения. Во-вторых, значительный срок разработки, который связан не столько с технической стороной разработки самого ядра ПО, сколько с необходимостью его интеграции в ИТ среду. После коммутации необходимость поддержки ПО полностью делегируется от разработчика заказчику. И, наконец, самый существенный недостаток – это необходимость ознакомить разработчика ПО со всей бизнес-информацией заказчика, от инфраструктуры компании до состава потоков данных в ИТ среде. Это, конечно, не способствует информационной безопасности заказчика.

LowCode/LessCode платформенные решения построены на принципах минимального кодирования. Инструментами разработки ПО в этом случае являются визуальные интерфейсы, которые позволяют изменять код программного продукта без непосредственного использования языков программирования. Скорость разработки при этом многократно возрастает без потери качества продукта. При этом вендор создает платформу, позволяющую экономить ресурсы высококвалифицированных ИТ работников, доверяя создание стандартных ИТ решений бизнес-аналитикам либо продвинутым пользователям на стороне клиента.

В настоящее время тенденции таковы, что переход бизнеса на решения LowCode/LessCode платформы является наиболее предпочтительным [5]. Less-Code платформа обеспечивает независимость от вендора и максимальную свободу развития, представляя следующие преимущества. Во-первых, при существенно меньших затратах, чем в случае кастомной разработки, пользователь получает за ограниченный период эксплуатации платформы ПО требуемого функционала. Это достигается внутрифирменной доработкой ПО под задачи компании самим пользователем, без потери качества конечного ИТ продукта. Снижение затрат связано с отсутствием необходимости в высококвалифицированных ИТ кадровых ресурсах. Во-вторых, отсутствуют риски информационной безопасности, так как вся информация о деятельности компании остается на стороне клиента. И, наконец, такая платформа позволяет проводить мероприятия по гибкому управлению корпоративной средой, что связано с возможностью оперативно перенастраивать ИТ поддержку бизнес-функций под различный целевой и технологический функционал.

Последнее становится определяющим, когда на рынке ИТ продуктов происходят критичные изменения. Такие как, например, в настоящее время. Существенное резкое снижение доступных ИТ продуктов является нелинейным возмущением ИТ среды, что для обеспечения устойчивости развития требует компенсирующих воздействий со стороны игроков рынка, в том числе и государства [4]. Политика импортозамещения в области ИТ приводит к необходимости рассмотреть ещё один аспект предметной области. А именно, при выборе ИТ решения необходимо определиться с тем, будет ПО проприетарным или разработанным на базе Open Source стека технологий. В первом случае исходный программный код является собственностью вендора и неизвестен конечному потребителю ПО. Во втором случае исходный код ПО известен и свободно доступен для использования.

В случае проприетарного ИТ решения ПО вендора используется на основании лицензии вендора. Но эта лицензия не дает право ознакомления с кодом. Нет также возможности копировать, распространять и модифицировать код ПО. Все эти права есть только у собственника ПО, которым может быть разработчик или организация, выпустившая ПО. За эти лицензионные ограничения пользователь, однако, получает компенсирующий сервис: стабильность функционала лицензионного ПО, регулярные обновления в соответствии с циклом управления выпуском ПО, уникальный функционал, который и был основной причиной закрытия кода, т.к. требовал дорогого ресурсного обеспечения при реализации ПО. В результате пользователь имеет дорогое и эффективное ПО, которое со временем становится только лучше, т.к. повышается его стандарт, расширяются его возможности, растёт скорость и производительность.

Основой Open Source программного обеспечения является открытый и доступный в Интернете исходный код [8]. Такой код иногда также поставляется с лицензией, которая существенно отличается от предыдущего случая, так как предоставляет возможность изучать, копировать, модифицировать,

распространять и использовать код без каких-либо ограничений. Такой информационный подход не является ни новым (ОС UNIX, 1973, [9]) ни уникальным (в частности, его можно транслировать на любые информационные продукты, см., например, [10]). Однако, именно в последнее время он приобрёл особую актуальность и признается экспертами наиболее перспективным [5].

III. ФОРМАЛИЗАЦИЯ РЕШЕНИЯ ЗАДАЧИ

Проведенный анализ предметной области исследования показывает, что задача выбора ПО на ИТ рынке сводится к нахождению оптимального решения и подразделяется на две подзадачи. Во-первых, необходимо выработать формализм, описывающий решение задачи с позиций вклада используемого заказчиком ПО в качество конечного продукта, а во-вторых, необходимо учесть нелинейность внешней ИТ среды разработки и использования ПО.

Переменными модели первой задачи являются все критичные характеристики использования ПО, главными из которых являются скорость разработки и внедрения (Т), эффективность/производительность (Е), стоимость разработки и поддержки использования (С). Эти же взаимозависимые показатели можно считать ресурсами повышения качества использования ПО. Таким образом, оптимальность решения подзадачи выбора подхода к разработке ПО может быть оценена по агрегированному сложному показателю качества (К):

$$K(F^*) = \max \sum \sum \sum [E(f_{ect}(F)) + T(f_{ect}(F)) - C(f_{ect}(F))] \quad (1)$$

Оптимальное решение находится как наилучшая комбинаторная альтернатива (функции распределения):

$$F^* : f_{ect} \rightarrow \{f \in F(e \in E, \tau \in T, c \in C)\}$$

которая при заданном уровне наличия ресурсов каждого вида обеспечивает экстремум глобального сложного показателя $K(F^*)$.

Решение второй подзадачи, связанной с наличием нелинейных возмущений среды разработки возможно провести в терминах классической теории управления нелинейными системами [11]. Такие задачи решаются путем приведения дифференциального уравнения состояния ИТ-проекта к виду (в обозначениях, использованных выше):

$$\begin{cases} \frac{dc}{dt} = f_{ect}(e, \tau, c) \\ \frac{d\tau}{dt} = f_{ect}(e, \tau, c) \\ \frac{de}{dt} = f_{ect}(e, \tau, c) \end{cases}$$

где координаты e, τ, c являются кусочно-непрерывными функциями. Учет в задаче менее значимых параметров, влияющих на принятие решения о платформе разработки ПО, повышает её размерность, что определенно увеличивает вычислительные трудности её решения [12].

Таким образом, для случая нелинейной внешней среды задача миграции стека технологий ИТ проекта эквивалентна задаче экстремального регулирования, в которой имеется статическая характеристика с экстремумом, положение и величина которого точно неизвестна, а поиск решения осуществляют по показателю качества, достигающему экстремум, обеспечивая удержание системы вблизи экстремального значения в каждый текущий момент времени. Классификация задачи миграции стека как задачи экстремального регулирования обусловлена тем, что существует физически ясный показатель качества, экстремальное значение которого соответствует оптимизации принятого решения о выборе платформы разработки ПО в условиях неопределенных значений значимых координат системы в последующие моменты времени. Понижению размерности задачи способствует доопределение характеристик показателя качества, которое может быть проведено в условиях Open Source стека технологий.

IV. РЕШЕНИЯ ЗАДАЧИ ВЫБОРА СТЕКА РАЗРАБОТКИ ПО

Рассмотрим возможности Open Source Less-Code платформы разработки ПО на примере платформа Jmix [3]. В табл. 1 показаны преимущества Less-Code по сравнению с Low-Code.

ТАБЛИЦА 1. СРАВНИТЕЛЬНЫЕ ХАРАКТЕРИСТИКИ OPEN SOURCE ПЛАТФОРМЫ

Low-Code	Less-Code
Проприетарная платформа. Требуется квалифицированных ИТ работников для адаптации к задачам компании.	Jmix Less-Code предоставляет как базу знаний, так и технологии для расширения возможностей ПО без привлечения ИТ работников.
Low-Code гарантирует резервную копию на стороне вендора.	Jmix Less-Code обеспечивает возможность резервного копирования на стороне клиента.
Low-Code обеспечивает поддержку вендора.	Jmix Less-Code дает доступ к опыту сообщества Open Source.
Low-Code предлагает набор готовых инструментов модульной генерации ПО для ускорения процесса разработки.	Jmix Less-Code ускоряет разработку, предоставляя специальные инструменты и опыт десятков тысяч разработчиков сообщества Open Source.
Low-Code разрабатывается вендором.	Jmix Less-Code разрабатывается внутри компании на основе изучения методического сопровождения платформы.
Расходы Low-Code зависят от количества пользователей.	Расходы Jmix Less-Code зависят от количества разработчиков

Важным показателем для выбора Open Source решения является гибкость пользовательского интерфейса. В отличие от Low-Code платформы, которая предлагает выбор между готовыми шаблонами, Less-Code позволяет настроить интерфейс под запросы пользователя, что является существенным преимуществом, позволяющим снизить психологическую нагрузку при смене стека [4]. По сравнению с Low-Code платформа Less-Code обеспечивает большую гибкость бизнес-логики за счет возможности сгенерировать сложные модели данных, механизмы бизнес-процессов и рабочих процессов. Независимость Less-Code платформы от вендора позволяет реализовать ПО в той среде, которую выбирает сам пользователь, предлагая набор

инструментов интеграции API открытой платформы и развертывания [13].

Средняя по вендорам Low-Code (ProcessMaker, Creatio, QuickBase, Mendix, Outsystems, Retool, WaveMaker) оценка стоимости высокопроизводительной платформы разработки приложений Jmix без учета облачных сервисов в сравнении с аналогичными по функционалу продуктами приведена в [3] в зависимости от времени разработки (Т), числа разработчиков (N), числа работников компании, поддерживающих платформу в актуальном состоянии (n), количества конечных пользователей (U), стоимости лицензии (L).

ТАБЛИЦА II. НАЧАЛЬНЫЕ УСЛОВИЯ ОЦЕНКИ СРАВНИТЕЛЬНОЙ СТОИМОСТИ OPEN SOURCE РЕШЕНИЙ

Показатель	Значение				
	A	B	C	D	E
Т(мес.)	6	6	10	10	10
N	4	6	4	4	5
n	1	1	2	2	1
U	100	100	100	200	150
L (\$/мес.)	10	10	15	5	10

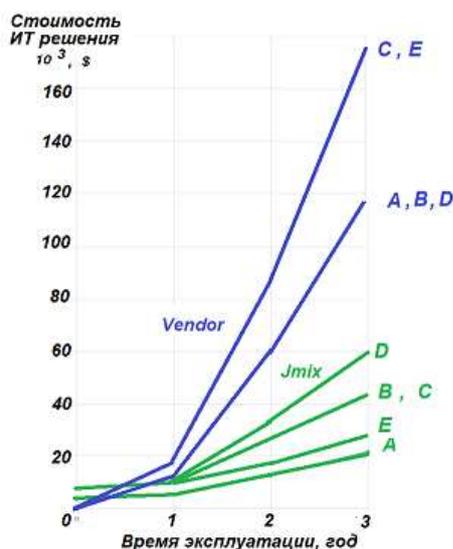


Рис. 1. Сравнительная характеристика стоимости ПО

В табл. 2 и на рис. 1 приведены условия и зависимость стоимости Open Source Low/Less-Code решений от данных показателей. Из приведенных данных можно заключить следующее. Open Source бюджеты на ИТ решения в рамках Low-Code и Less-Code сопоставимы по агрегированному показателю качества в течение первого года эксплуатации. Причем, в начале этого периода наблюдается большая эффективность решения от вендора. Однако к концу периода, если для кривых B и D бюджеты равны, то при других начальных условиях (табл. 2) Less-Code решения

становятся менее затратными. В дальнейшем при увеличении срока эксплуатации ИТ платформы разработки ПО решения и на длительных сроках эксплуатации ПО Less-Code решения многократно превышают по агрегированному показателю качества (1) решения Low-Code вендоров. И это справедливо для всех параметров агрегированного показателя качества. Поэтому очевидным является оптимальность выбора ИТ решения в пользу Open Source Less-Code платформы разработки ПО.

V. ЗАКЛЮЧЕНИЕ

Показано, что в условиях слабо прогнозируемой ситуации на рынке ПО задача выбора платформы для разработки ПО относится к классу задач экстремального регулирования. Наилучшим является решение о выборе платформы на основе Open Source стека технологий. Преимущества полученных решений показаны на примере платформы Jmix компании Haulmont.

СПИСОК ЛИТЕРАТУРЫ

- [1] Boehm B.W. A spiral model of software development and enhancement // ACM SIGSOFT Software Engineering Notes Volume 11 Issue 4 August 1986, pp. 14-24. <https://doi.org/10.1145/12944.12948>
- [2] International Standart 14764-2022 - ISO/IEC/IEEE International Standard - Software engineering - Software life cycle processes – Maintenance.Third Edition, 2022-01. <https://standards.iteh.ai/catalog/standards/sist/70aa2449-ac83-46d7-84c2-fe73a5a2efcc/iso-iec-ieee-14764-2022>
- [3] Jmix. URL: <https://www.jmix.io>
- [4] Volkova V.N., Loginova A.V., Desyatirikova E.N. “Simulation Modeling of a Technological Breakthrough in the Economy” // Proc. of 2018 IEEE EIConRus. DOI: 10.1109/EIConRus.2018.8317332
- [5] CNews Conference «Импортозамещение 2022: первые итоги» Обзор, Фадеев В. «Плюсы и минусы разных подходов», URL: https://events.cnews.ru/articles/2022-10-17_promezhzhochnye_itogi_tsifrovizatsii
- [6] Беляев А.С. «Современные тенденции развития методологии ведения проектов в ИТ» // Информатика: проблемы, методология, технологии: Мат. X межд. науч.-метод.конф. 11-12.02.2010, Воронежский госуд. университет, Воронеж, 2010, т.1., с. 3-6.
- [7] Леднев А.М. «Генетический алгоритм в проекте Haulmont». Блог компании Haulmont, URL: <https://www.haulmont.ru/blog/dec-14-2018/913/?ysclid=lg96lzkxor413917139>
- [8] Open Source Initiative, The Open Source Definition. <https://opensource.org/osd/>
- [9] The Open Group UNIX System Homepage. <https://web.archive.org/web/20080705220928/http://www.unix-systems.org/>
- [10] Houlmont Technology Limited, Jmix Website Terms of Use. <https://www.jmix.io/terms-of-use/>
- [11] Волков В.Д., Шашкин А.И., Смольянинов А, Десятирикова Е.Н., Теория автоматического управления. Воронеж: «Научная книга», 2015. 745 с.
- [12] Курбатов В.Г., Кургалин С.Д., Понижение размерности модели динамической системы методом подпространства Крылова // Вестник Воронежского госуд. университета. Серия: Системный анализ и информационные технологии. 2013, № 1, с. 46-54.
- [13] Беляев А.С. Jmix – будущее CUBA Platform. <https://habr.com/ru/companies/haulmont/articles/536224>