

18+

Андрей Николаевич Трушкин

АРХИТЕКТУРА ЦИФРОВЫХ ПЛАТФОРМ

От настоящего к будущему



Андрей Трушкин

**Архитектура цифровых платформ.
От настоящего к будущему**

«Издательские решения»

Трушкин А. Н.

Архитектура цифровых платформ. От настоящего к будущему /
А. Н. Трушкин — «Издательские решения»,

ISBN 978-5-00-645813-0

В настоящей книге изложены основы построения современных и устремленных в будущее цифровых платформ, рассматриваются характерные элементы, обязательные для любой платформы. Поднимаемые в книге вопросы затрагивают все этапы жизненного цикла платформ и продуктов, создаваемых на их основе. Особое внимание уделено архитектуре платформ, рискам, возникающим при их создании и развитии, а также способам преодоления этих рисков. Книга будет полезна как ИТ-специалистам, так и руководителям организаций.

ISBN 978-5-00-645813-0

© Трушкин А. Н.
© Издательские решения

Содержание

Введение	6
Проблематика современных цифровых платформ	9
Цифровые платформы и ключевые тренды развития архитектуры	20
Еще раз о ключевых трендах развития архитектуры	20
Цифровые платформы и открытый код	24
Цифровые платформы и распределенность	33
Цифровые платформы и бизнес-процессы	42
Конец ознакомительного фрагмента.	44

Архитектура цифровых платформ От настоящего к будущему Андрей Николаевич Трушкин

© Андрей Николаевич Трушкин, 2024

ISBN 978-5-0064-5813-0

Создано в интеллектуальной издательской системе Ridero

Введение

Данная книга является логическим продолжением предыдущего труда автора – «Архитектура цифрового мира». Бытует мнение, что каждая книга (или серия книг) должна быть логически завершенной, позволять осмыслить изложенное в ней в целом. Подобное мнение имеет право на существование. Однако жизнь не стоит на месте и дарит пищу для размышлений с учетом как прочитанного, так и ранее изложенного. А потому мы идем дальше и стараемся расширить и углубить те мысли, которые были представлены в предыдущей книге, на нее же будем регулярно ссылаться по ходу настоящего изложения.

Ранее мы достаточно подробно останавливались на вопросах цифровых платформ, им посвящен отдельный раздел в «Архитектуре цифрового мира». Вместе с тем роль цифровых платформ в современном цифровом же мире, их позиционирование, взаимовлияние исключительно важны: платформы (а в дальнейшем мы, в отсутствие дополнительных уточнений, будем подразумевать под платформами именно цифровые платформы) привлекают внимание не только ИТ-специалистов, но и ученых, философов, политиков, историков и многих других специалистов. Можно сказать, что платформы являются самостоятельным направлением развития цифрового мира. И потому крайне важно рассмотреть подобное направление, так как его значение с течением времени будет только возрастать. Именно указанному рассмотрению и посвящена настоящая книга.

Опираясь на базис, сформированный в предыдущем труде, мы принимаем за основу, что платформа является фреймворком создания и исполнения ИТ-решений организации. При этом платформы могут выходить за рамки конкретной организации, объединяя в структуре экосистемы целые отрасли человеческой деятельности. Примером может служить инструментальная реализация концепции *banking-as-a-platform* (BaaP), при которой различные организации схожих областей деятельности объединяются на одной платформе. Аналогичным образом сложные холдинговые структуры могут объединять собственные продукты и процессы на общих кросс-платформах. Также значимым фактором современной жизни являются социальные платформы, объединяющие людей в различных странах и на разных континентах. Кроме того, нельзя забывать те характеристики платформ, которые являются следствием потенциальных ментальных ловушек, связанных с платформенными проектированием и реализацией:

- Платформа не является информационной системой.
- Платформа не является обособленным программным комплексом.
- Платформа должна быть открытой, и изменения в нее могут вносить любые команды, которые должны брать на себя ответственность за вносимые изменения.
- Платформа не должна быть замкнутой.

Для полноты восприятия вкратце раскроем вышеперечисленные тезисы, напомнив их читателю.

Тот факт, что платформа не является информационной системой, обусловлен отсутствием самостоятельной ценности, создаваемой и приносимой пользователям и клиентам, со стороны платформы. Ценность создается продуктами, реализуемыми приложениями, которые в свою очередь создаются и исполняются при помощи платформы, то есть ценность платформы опосредованная. Таким образом, платформа не содержит смысла самостоятельного овеществления в ИТ-ландшафте организации.

Обособление (не путаем с овеществлением, характерным для информационных систем), предполагающее независимое технологическое позиционирование платформы, приводит к необходимости сложной синхронизации платформенных (содержащих технологические функции) и прикладных (содержащих функции, реализация которых несет непосредственную

ценность клиентам) бэклогов, существенному усложнению релизных циклов и т. д. Кроме того, говоря о необходимости избегать обособления платформ в ИТ-ландшафте, мы подчеркивали необходимость обеспечения их встраиваемости в приложения, в рамках реализации которых и создается непосредственная пользовательская ценность.

Открытость платформ предполагает обеспечение их прозрачности для команд разработки, создающих ценность для клиентов и партнеров организации. Команды должны иметь возможность дополнять функциональность платформ, при этом дополнения становятся достоянием всех работающих с платформой команд, то есть платформы подчиняются принципам проектов с открытым исходным кодом (возможно, в ограниченном формате, при котором платформенное сообщество формируется в масштабах развития организации).

Отсутствие замкнутости платформ предполагает их постоянное развитие в технологическом плане, открытость новым технологическим трендам, изменениям лицензионных соглашений и т. д. По сути, требования к платформе быть открытой и не быть замкнутой являются взаимно дополняющими друг друга. Тем самым обеспечивается возможность интенсивного развития платформы, платформенных приложений и сервисов, самой организации, применяющей платформенный подход, достижение необходимого уровня архитектурного mindset.

Перечисленные свойства платформ не предполагают, что они (платформы) являются некими «черными дырами» ИТ-ландшафта, засасывающими финансовые, временные, трудовые и иные ресурсы организаций, но при этом не имеют четких границ и областей применения. Наоборот, платформы должны быть наиболее прозрачными (пусть и не обособленными) компонентами приложения усилий, ведь их опосредованное участие в создании ценности становится ценностным мультипликатором организаций, применяющих платформенный подход, обеспечивающим интенсивное развитие, препятствующим росту энтропии, риск которого является существенным при современных требованиях к цифровизации. Если мы говорим о том, что современная компания является ИТ-компанией, предоставляющей услуги самого широкого спектра деятельности (как основного профиля, так и смежных и вспомогательных), то скорость цифровизации, предполагающая кардинальное изменение мышления, переход к продуктовому подходу от классических проектного или процессного, вполне может способствовать росту энтропии. И сдержать его может помочь платформенный подход, о чем мы и поговорим в настоящей книге.

Разумеется, это не означает, что платформа, создающая опосредованную ценность, а также платформенные приложения «скованы одной цепью», то есть являются неразрывным целым с некой жестко определенной структурой. Возможно, побудительные мотивы выстраивания подобной структуры в организациях и могут возникать, но они являются очередной ментальной ловушкой, подстерегающей нас на пути следования интенсивному развитию. Такие структуры оказываются хрупкими, а потому либо являются недолговечными в быстро меняющемся цифровом мире и водовороте новых требований, либо приводят организации к застою и деградации, за которыми следует потеря конкурентоспособности. Платформы должны иметь распределенную сервисную структуру, предоставлять платформенные сервисы, а платформенные приложения должны бесшовным образом как исполняться на платформе, так и взаимодействовать между собой. Вопросы связи платформ, платформенных сервисов и платформенных приложений будут рассмотрены в соответствующей главе.

За время, прошедшее с написания предыдущей книги автора, подчеркнутые в ней ключевые тенденции развития архитектуры не только не угасли, но и дополнительно укрепились. Как архитектура в целом, так и платформы обязаны им следовать. Перечислим данные тенденции:

- Открытый код, на котором основываются ключевые технологические решения современности.

- Распределенность (при этом в данное понятие входит как технологическая, так и организационная составляющая).
- Бизнес-процессы, автоматизация которых несет конечную ценность для клиентов и пользователей.
- Данные, являющиеся основным богатством организаций, а потому принимаемые в качестве краеугольного камня цифровизации.
- Искусственный интеллект, проделавший существенный качественный рост (пусть негативно настроенные авторы и утверждают, что виной тому эффект низкой базы).

Успешные платформы современности (например, развиваемые технологическими гигантами) следуют вышеперечисленным тенденциям. Но следование им не является прерогативой только технологических гигантов – это веление времени, а потому любая организация, стремящаяся сохранить конкурентоспособность на рынке, должна учитывать их, творчески адаптировать, а также синхронизировать платформенный подход (обязательный к применению для обеспечения интенсивного развития) с указанным перечнем. Вопросы применения и адаптации приведенных ключевых тенденций развития архитектуры в сочетании с платформенным подходом, соответствия платформ данным тенденциям будут рассмотрены в соответствующей главе.

При этом автор не ставит своей целью описать единственное верное построение цифровой платформы, тем более что это и невозможно. Современное развитие технологий, связанное с ним изменение лицензионных соглашений, проработка новых топологий развертывания, формирование новых архитектурных подходов исключают не просто решение, но и саму постановку подобной задачи. При обосновании использования, проектировании, реализации и развитии платформ необходимо следовать ключевым тенденциям развития архитектуры, учитывать представленные ранее свойства платформ, а также принимать во внимание необходимость перманентного интенсивного развития организации, являющегося ключом к конкурентоспособности. Только такой mindset позволит обеспечить реализацию и использование и постоянное развитие платформ, обеспечивающих указанную конкурентоспособность. И только такой mindset приемлем для архитектора, являющегося лидером технологических изменений. Также мы не будем обсуждать вопросы сайзинга оборудования, применяемого для развертывания платформ и платформенных приложений.

Отметим, что и сейчас актуально утверждение, приведенное в «Архитектуре цифрового мира»: частные случаи и конкретные технологические решения будут упоминаться лишь в качестве примеров, иллюстрирующих комплексное рассмотрение.

Отдельно следует заметить, что автор сознательно избегает рассматривать в настоящем изложении вопросы архитектуры конкретных платформ. Любые совпадения с частными архитектурными, технологическими или организационными решениями существующих платформ случайны.

Проблематика современных цифровых платформ

Читатель, увидев заголовок настоящей главы, может задаться вопросом: «Какая же проблематика? Повсеместно можно услышать про необходимость создания или развития цифровых платформ, любая уважающая себя компания либо создает собственную, либо использует внешнюю/внешние. Нет здесь никаких проблем, за исключением, может быть, финансовых». Но в том-то и дело, что зачастую словесный вал лишь затушевывает суть: вместо создания и развития платформ во многих случаях происходит экстенсивное развитие унаследованного ИТ-ландшафта. Нередко отсутствует даже экстенсивное развитие, а организации уже выходят на тропу мысленной и технологической деградации. И финансовые проблемы играют здесь совсем не основную роль. Попробуем разобраться, в чем же истинная проблематика современных платформ.

В «Архитектуре цифрового мира» (в разделе «Архитектура и цифровые платформы») мы уже говорили о том, что термин «цифровая платформа» является многозначным в современном мире, приводили примеры такого использования данного термина, как «платформа Google», «омниканальная платформа», «платформа Java» и некоторые другие. К сожалению, и на сегодняшний день указанная многозначность не изжита, она присутствует и в специальной литературе, и в дискуссиях самого разного уровня, и в публичных выступлениях. Как результат, создаются все новые и новые «платформы», при этом крайне сложно выявить ценность их внедрения и использования. Продукты у компаний, внедряющих подобные «платформы», зачастую попросту отсутствуют, рассматриваются лишь классические процессные подходы и методологии, сами же внедряемые «платформы» являются потребителем финансовых ресурсов, последние безостановочно направляются на «развитие», реальным же «выхлопом» являются застой и деградация. Разумеется, далеко не все платформы являются таковыми, но автор вынужден констатировать, что попадание в ментальные ловушки, характерные как для цифровизации в целом, так и для работы с платформами, является нередким случаем в современном мире.

Другим полюсом, характеризующим совершенно иные результаты применения платформенных подходов, являются мировые технологические гиганты, которые (благодаря достижению высокого технического уровня) получают возможность реагировать на изменения в конъюнктуре, потребительских предпочтениях, экономических тенденциях на совершенно иной скорости, а также с недоступной еще относительно недавно адресностью. В начале 2024 года один из крупнейших мировых технологических гигантов представил результаты мониторинга, показавшие, что в среднем изменения в используемую им технологическую платформу и продукты, предоставляемые посредством платформенных приложений, вносятся каждые 11,6 секунды. Во-первых, подобная оперативность позволяет реагировать на современные вызовы с адекватной скоростью, сохраняя конкурентоспособность организации. Во-вторых, указанная скорость реакции позволяет задавать ориентир для конкурентов, усугубляя положение тех из них, кто не в состоянии реагировать на вызовы с сопоставимым уровнем оказания услуг. Таким образом, подобный технологический гигант не просто сохраняет адекватность требованиям рынка при предоставлении продуктов и услуг клиентам и партнерам, но и наращивает отрыв от конкурентов, повышая собственные шансы на монополизацию тех или иных сегментов рынка. И основой здесь является именно платформенный подход.

Предположим, что организация, ИТ-ландшафт которой реализован в классической парадигме сервис-ориентированной архитектуры (SOA), столкнется с необходимостью столь же быстрого внесения изменений, связанного, например, с быстро меняющейся рыночной конъюнктурой. Для примера предположим, что ИТ-ландшафт выстроен в формате, представленном на Рисунке 1. Организация стремится оставаться в рыночных трендах и поэтому перешла

(хотя бы на декларативном уровне) к продуктовой методологии. Но «продукт», предоставляемый организацией, является результатом выполнения операций в нескольких (предположим, что в четырех) информационных системах, созданных с использованием различных технологий.

Рисунок 1. ИТ-ландшафт в парадигме SOA

Рассмотрим значимое изменение в «продукте», предоставляемом организацией, как результат работы указанных четырех информационных систем. Под значимым изменением в продукте будем понимать такое изменение, которое влияет на его функциональные (например, перерасчет ставки по кредиту с одновременным изменением рассматриваемого клиентского пула) и нефункциональные (например, требования к новым дистанционным каналам, предполагающие взрывной рост нагрузки) характеристики. Подобное значимое изменение в продукте в общем случае предполагает изменения во всех информационных системах, результатом проводимых операций в совокупности которых является продукт. Таким образом, команда доработки продукта должна включать в себя специалистов, компетентных в соответствующих системах (в рассматриваемом примере – четырех). Используемые в настоящее время информационные системы не являются монотехнологичными, а содержат развитый технологический базис построения. Подобная ситуация предполагает (в случае значимых изменений) привлечение специалистов различного технологического профиля, профессионализм которых позволяет производить масштабные изменения, затрагивающие весь перечень технологий, используемых в информационных системах. Уже привлечение трех-четырех специалистов по каждой информационной системе обеспечит превышение традиционной мощности команд гибких практик (если таковые применяются в организации). Учтем, что значимые изменения, как правило, не выполняются силами выделенных единичных специалистов, поскольку являются трудоемкими, требуют документирования, обмена опытом и т. д. Получается большой коллектив с заметным профессиональным и технологическим разнообразием, к тому же внимание отдельных специалистов резонно будет сфокусировано на доверенных им информационных системах, что дополняет проблематику внесения значимого изменения необходимостью синхронизации деятельности соответствующих специалистов, представляющих подкоманды разработки. Разумеется, ни о каких 11,6 секунды на изменение говорить уже не приходится. Учтем необходимость тестирования, выполнения релизной политики (зачастую также весьма громоздкой) – и получим совершенно неприемлемые с точки зрения современной конъюнктуры значения скорости внесения изменений. Как результат, организацию ожидают застой и деградация, за которыми следует утрата конкурентоспособности.

Приведенный выше пример может показаться надуманным. Но еще в середине 2010-х годов проводилось сравнение крупных российских организаций, осуществлявших цифровую трансформацию, с мировыми технологическими гигантами. Сравнение осуществлялось по показателю количества изменений, вносимых ими в ключевые приложения. Показатели российских организаций за квартал уступали аналогичным показателям технологических гигантов в день. Основными составляющими подобного преимущества назывались следование платформенному подходу (разумеется, каждый гигант формирует собственную платформу) и гибким практикам разработки. Как мы видим, мировые лидеры не сбавляют оборотов. Соответственно, следует не просто не игнорировать составляющие их успеха, но и внимательнейшим образом их изучать и адаптировать. И здесь речь идет не только и не столько о гибких практиках (пусть это и исключительно важный аспект обеспечения конкурентоспособности в современном мире), сколько о комплексном архитектурном mindset, а также о его воплощении в платформенном подходе.

Настало время поговорить о еще одном негативном аспекте, возникающем при обсуждении платформ. Одним из результатов такого обсуждения (собственно негативным аспектом) становится многообразие терминов: «омниканальная платформа», «учетная платформа»

и т. д. Предположим, что организация, рассмотренная выше, рассчитывает встать на путь интенсивного развития, применив платформенный подход. В результате осмысления проблем, возникающих при развитии ИТ-ландшафта, характеризующегося многообразием информационных систем, организация начинает создавать платформы и обеспечивать путем их исполнения предоставление продуктов клиентам и партнерам. Пример «продукта», характерного для указанного изменения подходов, представлен на Рисунке 2.

Рисунок 2. «Продукт», формируемый множеством «платформ»

И предположим, что рассматриваемый «продукт» претерпевает значимое изменение. Для прозрачности рассмотрения укажем, что речь идет об end-to-end продукте (описан в «Архитектуре цифрового мира»). Уточним, что мы рассматриваем в примере финансовый продукт, который можно подразделить на следующие составляющие (для демонстрации применения нескольких «платформ»):

- Учетная составляющая, в рамках которой предполагается ведение синтетического учета данных о продукте.
- Частная учетная составляющая, на уровне которой обеспечивается связь синтетического и аналитического учета, формируется видение учета продукта для организации (в случае нашего примера – финансовой).
- Процессная составляющая, обеспечивающая автоматизацию операций по созданию, заведению, постановке на учет, сопровождению продукта, то есть весь комплекс сложных бизнес-процессов, поддерживающих жизненный цикл продукта.
- Фронтальная составляющая, формирующая пользовательское представление о продукте и операциях, проводимых с ним (включая как внешних, так и внутренних пользователей).

Что же происходит в организации, которая, приняв за руководство к действию следование платформенным подходам, попадает в ментальную ловушку «множества платформ» (как мы увидим ниже)?

В общем случае для каждого уровня продуктовой автоматизации, представленного выше, выбирается собственная платформа, обладающая ключевыми для обеспечения необходимого каждому уровню качества характеристиками.

Ведение синтетического учета предполагает выполнение большого количества транзакционных операций, характеризующихся высокой степенью унификации, минимальными (а зачастую отсутствующими в принципе) требованиями по гибкости настройки (ввиду указанной унификации), серьезными вызовами в части производительности, необходимостью проведения высокоэффективных групповых операций (в случае финансовой организации в качестве примера можно привести необходимость формирования регуляторной отчетности), скромными требованиями к удобству пользовательского интерфейса (в данном случае говорить о вроде бы ставших общепринятыми терминах «клиентский путь», «дизайн-мышление» и им подобных не приходится).

При ведении аналитического учета предъявляются требования гибкой настройки параметров продукта и их связи с показателями синтетического учета. В данном случае имеет смысл говорить об адресности проводимых операций (уже их результаты преобразуются в синтетические проводки, требования к которым становятся базовыми для платформы синтетического учета). То есть производительность в случае аналитического учета рассматривается не с точки зрения эффективности выполнения групповых или пакетных операций, а с точки зрения возможности сегментации работы с продуктами различных типов (например, депозитными и кредитными продуктами в случае финансовой организации) и снижения нагрузки на каждый продукт в отдельности, а также исключения их взаимовлияния. Требования к пользовательскому интерфейсу формируются в соответствии с параметрами аналитического учета, а также ширины спектра продуктов организации: необходимо предоставить развитые пользо-

вательские настройки для формирования параметров продуктов, организации связи кросс-продуктов, адаптации интерфейсов к развитию продуктового ряда, характерному для современного цифрового мира. Поскольку соответствующие настройки проводят сотрудники организации (в более редких случаях – сотрудники компаний-партнеров), то мы говорим скорее об удобстве интерфейсов, нежели о производительности последних.

Автоматизация бизнес-процессов предполагает возможность быстрого создания и изменения последних, а также обеспечение их мониторинга, расчета КПЭ и т. д. Продуктовые бизнес-процессы предполагают исключительно широкие возможности настройки, необходимость создания и исполнения кросс-продуктовых процессов (задействующих различные типы продуктов с генерацией совместной аналитической и синтетической учетной логики), потенциал создания большого количества точек интеграции, использование современных практик low-code и no-code для проектирования и исполнения процессов. Можно констатировать, что требования к удобству и гибкости настроек для данного слоя исключительно актуальны. Долгое время считалось, что вопросы производительности на уровне настройки и исполнения бизнес-процессов уходят на второй план по сравнению с гибкостью, но современный мир диктует необходимость изменения подобного подхода. В условиях повсеместного развития дистанционных каналов обслуживания невысокая производительность автоматизированных бизнес-процессов приведет к утрате конкурентоспособности организации, допустившей указанный недостаток. Например, низкая скорость обработки поступающих заявок на кредит исключает привлекательность для клиентов каналов, не требующих аутентификации (например, сайты компании или ее партнеров), обнуляя возможность лидогенерации, что недопустимо для организации, ставящей во главу угла интенсивное развитие. При этом вопросы выполнения групповых и пакетных операций (по аналогии с автоматизацией аналитического учета) практически неактуальны. Вопросы мониторинга исполняющихся процессов, напротив, крайне важны: специалисты, ответственные за развитие бизнес-процессов организации, должны иметь возможность моделирования и установки КПЭ процессов, клиенты и партнеры нуждаются в развитой статусной модели процессов, позволяющей получать информацию о состоянии интересующих их процессов и обрабатываемых в ходе их исполнения данных. При этом традиционной характеристикой автоматизации бизнес-процессов является высокое потребление вычислительных ресурсов при выполнении данной задачи. Соответственно, необходим мониторинг ресурсов, потребляемых теми или иными бизнес-процессами и их экземплярами, на основании данных которого оказывается возможным обеспечивать «здоровье» ИТ-ландшафта организации. В рассматриваемом случае речь идет о командах DevOps и/или сопровождения, являющихся пользователями мониторинга бизнес-процессов. То есть речь идет как о системном, так и о прикладном характере мониторинга. Пользователи же мониторинга являются как внутренними с точки зрения организации, так и внешними. Проанализировав все указанные аспекты, мы приходим к необходимости высоких требований к возможностям пользовательского интерфейса «платформы» автоматизации бизнес-процессов. При этом автоматизация бизнес-процессов в большинстве случаев не предполагает транзакционного характера исполняемых операций (в части того, что операции на уровне данной составляющей не являются типовыми и/или короткоживущими).

При автоматизации фронтальной составляющей продукта особое внимание уделяется упомянутым выше «клиентскому пути» и «дизайн-мышлению», а также аналогичным им аспектам. В противном случае конкурентоспособность продукта оказывается чрезвычайно низкой в современном цифровом мире, инвестирование средств в создание, развитие и продвижение такого продукта теряет всяческий смысл. При несомненной важности рассмотренных выше аспектов продуктовой автоматизации фронтальное представление является базовой составляющей формирования эффективного P&L (profit and loss analysis) продукта. Составляющей исключительной важности в случае фронтального представления является произ-

водительность этого самого автоматизируемого фронтального представления: в случае медленной отрисовки графики по продукту, длительного переключения экранов представления потенциальный (да и действующий) клиент потеряет интерес к продукту, обратит внимание на предложения конкурентов, и организация понесет убытки. Особенно актуален вопрос производительности в условиях дистанционных каналов, далеко не все из которых в процессе лидогенерации предъявляют требования к обязательности аутентификации или заполнению пространных анкет. В условиях резкого роста числа каналов продукты должны быть представлены если не в каждом из них, то в некоем значимом множестве. В противном случае можно неоправданно сократить воронку продаж. И организация приходит к необходимости поддержки концепции омниканальности, при которой все каналы коммуникации с пользователями (в их число могут входить не только клиенты организации, но и ее сотрудники и партнеры) объединяются в единую связанную экосистему, при этом коммуникация в рамках продуктового взаимодействия носит сквозной характер. Таким образом достигается целостное взаимодействие пользователей с автоматизируемым продуктом. Но как обеспечить рассматриваемую омниканальность? Зачастую организации добавляют в свой ландшафт очередную «платформу» – омниканальную, в задачи которой входит:

- Отделение логики представления от процессной.
- Предоставление развитых инструментов создания интерфейсов.
- Поддержка различных каналов взаимодействия с пользователем как на уровне восприятия, так и на уровне фронтальных процессов (дополнительной сущности, отделяемой от продуктовых процессов прикладного характера, рассмотренных выше).
- Обеспечение непрерывности коммуникации с пользователями.
- И многие другие...

Также отметим уже ставшее фактически стандартом де-факто требование, предъявляемое к технологиям создания фронтальных интерфейсов, заключающееся в легковесности последних. Фронтальные приложения и их компоненты выполняются на самых разных устройствах, обеспечивают различные каналы коммуникации, вынуждены учитывать проблемы сетевого взаимодействия и т. д. Создание тяжеловесных фронтальных компонентов пользовательского приложения фактически обнуляет конкурентоспособность продукта, а вместе с ним и организации, его предоставляющей. Соответствующее требование легковесности транслируется и в отношении омниканальной «платформы», отличая ее от предшествующих.

Изучив приведенное выше описание, читатель непременно задаст вопрос: «Так в чем же отличие этого платформенного подхода от классической SOA-архитектуры? В такой ситуации надо создавать микросервисы, которые обеспечат распределенность, а то по факту просто системы заменили платформами!» И принципиально внимательный читатель будет прав: фактически значимая доработка продукта, созданного в подобной архитектуре, будет весьма схожа с его доработкой при реализации принципов SOA. Действительно, столь разная организация автоматизации каждого уровня продукта, предоставляемого организацией клиентам и/или партнерам, потребует фактически выделенной продуктовой команды на уровне. Да, в случае простого продукта можно составить продуктовую команду, отвечающую принципам гибких практик, но современные продукты (а особенно если речь идет о кросс-продуктах) крайне сложны, требуют привлечения специалистов разных компетенций. А потому даже на одном уровне автоматизации (при приведенном выше примере принципиально различных технологических и организационных требованиях к базису каждого) может оказаться весьма затруднительным собрать единую команду гибких практик, развивающую соответствующий продукт. После чего возникнет необходимость синхронизации работы команд для обеспечения качества создаваемого продукта, дабы последний не стал подобен лоскутному одеялу из слабо связанных между собой доработок, что в свою очередь может повлечь как рыночную слабость продукта, так и нарушение непрерывности его предоставления. При этом команда развития

каждого уровня автоматизации продукта реализует требуемые доработки с разной скоростью, зависящей от функциональной и нефункциональной составляющей, например: *каждом*

- Доработки могут проводиться на некотором подмножестве уровней автоматизации продукта: доработки фронтального представления, например, не затрагивать учетные составляющие.

- Различный технологический базис уровней автоматизации приводит к разной скорости их развития в соответствии с нефункциональными требованиями.

Добавим сюда еще и возможное попадание в ментальную ловушку, рассмотренную в главе «Архитектуры цифрового мира», посвященной цифровым платформам, а именно создание выделенных команд разработки и развития платформ. В случае масштабирования указанной ошибки на рассматриваемую в примере организацию может оказаться вероятным, что продукт разрабатывается набором команд гибких практик разработки, при этом для каждой платформы в организации существует отдельная команда развития. Получается, что в общем случае (при условии высокой сложности автоматизируемого продукта) мы можем столкнуться с ситуацией, при которой в реализации значимой продуктовой доработки могут быть задействованы до восьми (!) команд разработки. Синхронизация их деятельности становится весьма нетривиальной задачей. И даже совмещение платформенных и продуктовых команд в отдельных случаях (например, это вполне возможно для случая автоматизации функций синтетического учета) не привносит принципиального упрощения ситуации. И стоит учесть, что в рассматриваемом примере мы сознательно учитывали относительно простую структуру как продукта, так и организации. Последняя может внедрять еще и «платформу CRM», и расширяющую «омниканальную платформу» «платформу» ДБО, к примеру, и «платформу MDM», и многие другие аналогичные «платформы».

Мы видим, что подобное следование платформенному подходу, заключающееся фактически в бездумном производстве все новых и новых «платформ», нисколько не упрощает автоматизацию деятельности организации, не приближает ее к перманентному интенсивному развитию. Более того, создание платформ само по себе является трудоемким и финансово недешевым процессом. Результат же, который нами был продемонстрирован на достаточно простом примере, вовсе не приводит к экономии каких-либо ресурсов, доступных организации. В то же время мы регулярно говорим о необходимости перехода к платформенному подходу. В чем же дело? Мы просто следуем моде либо обосновываем бесконечный рост затрат на автоматизацию и цифровизацию? Поставим вопрос по-другому: какой должна быть платформа для достижения результатов автоматизации/цифровизации, заключающихся в переводе организационных процессов и продуктов на новый качественный уровень, какая платформа становится инструментом, обеспечивающим перманентное интенсивное развитие, какая платформа соответствует *mindset* современности, то есть какой должна быть современная платформа? Также впору задать вопрос: что можно, а что нельзя считать современной цифровой платформой?

Ответ на вопрос о платформе и ее видении необходимо начать с краткого обзора современного *mindset*. Мы не будем подробно рассматривать все его аспекты, желающие могут ознакомиться с ними в «Архитектуре цифрового мира». В отношении особенностей современного архитектурного *mindset*, обеспечивающего перманентное интенсивное развитие, можно выделить следующие:

- Ключевым элементом *mindset* является фокус на продукте, представляющем ценность для партнеров и клиентов организации, его создающей. Далеко не всегда определить продукт оказывается простой задачей, во многом корректное формирование продуктового видения конкретной организации требует серьезного изменения мышления и позиционирования себя на рынке, частным случаем которого является уход от продуктологов к владельцам продуктов.

- Как уже отмечено, современный архитектурный mindset должен обеспечивать перманентное интенсивное развитие. Соответственно, его инструментальное обеспечение (куда можно отнести и платформы) должно отвечать указанному требованию.

- Современный архитектурный mindset должен успешно обходить ментальные ловушки, связанные с гибкими практиками, информационной безопасностью, шаблонами проектирования, эффектом Даннинга – Крюгера (подробно рассмотрены в «Архитектуре цифрового мира»). И платформы также должны успешно противостоять той ментальной лени, которая тянет нас в указанные ловушки.

Возвращаясь к представленному выше примеру платформенного подхода, основанного на множестве «платформ», можно отметить, что он полностью игнорирует современный mindset. Фактически в нем отсутствует видение продукта – нет общей продуктовой концепции (а во многих случаях она окончательно «добита» наличием платформенных команд). Создание и развитие продукта, отвечающие требованиям современного цифрового мира, при таком подходе невозможны ввиду необходимости синхронизации рабочего процесса огромного количества специалистов разных направлений деятельности. В особо запущенных случаях деградации организации понятие продукта сводится к его визуальному или процессному представлению. В таком ракурсе невозможно обеспечить перманентное интенсивное развитие, так как векторы «развития» каждой применяемой «платформы» оказываются строго разнонаправленными, результат же получается аналогичным басне И. А. Крылова «Лебедь, щука и рак». Ведь самостоятельное развитие каждой «платформы» может затормозить общее продуктивное развитие, останов же платформенного развития приведет к бесконечному наращиванию продуктового функционала на стремительно устаревающей технологической базе.

При подобном развитии событий крайне сложно избежать и ментальных ловушек, перечисленных выше. При отсутствии общего продуктового видения корректные адаптация и применение гибких практик становятся невозможными, так как отсутствует продукт, на реализацию которого и направлены сами гибкие практики. Наиболее частым случаем, встречающимся на практике при применении такого подхода, является бесконечный «Waterfall со стендами», в который погружаются разрозненные команды, отвечающие за автоматизацию конкретных уровней рассмотрения продукта (в нашем примере их представлено четыре, но может быть и существенно больше при мелкой грануляции технологий автоматизации, применяемых в организации). При этом можно клеить на доску сколько угодно стикеров с описанием решаемых задач, синхронизировать бэклоги команд на соответствующих статусах, использовать современные инструментальные средства поддержки гибких практик, но не будет главного – согласованной работы над продуктом. Для упрощения производственной деятельности, пытаясь ускорить разработку тяжеловесных платформ, будут применяться новые и новые шаблоны проектирования, не ведущие к созданию ценности, но с какого-то момента существующие сами по себе. Ну и выход на «плато стабильности» эффекта Даннинга – Крюгера для подобным образом собранного сообщества также будет затруднен. Некая команда (например, реализующая фронтальное представление продукта при помощи «омниканальной платформы») будет продолжать находиться на «пике тупости», основываясь на относительно невысокой скорости реализации требуемого продуктового функционала другой командой, обладающей априори меньшей емкостью в терминах гибких практик (в качестве примера можно привести команду, отвечающую за ведение частного аналитического продуктового учета), которая, погрязнув в задачах и техническом долге, окажется зажатой в «обрыве сознания».

Разумеется, говорить об интенсивном развитии в данном случае не приходится, а потому и считать приведенный выше пример реальным следованием платформенному подходу также нельзя. Более того, невозможно именовать приведенные в примере «платформы» таковыми, поскольку они затрудняют развитие, обеспечивают попадание в ментальные ловушки, исклю-

чают формирование комплексного архитектурного mindset. И здесь содержится важная часть ответа на вопрос, что можно считать цифровой платформой будущего.

Мы сразу оговоримся, что платформа не предоставляет продукты сама по себе. Автоматизация продуктов, предоставляемых организацией клиентам и партнерам, осуществляется посредством платформенных приложений, которые мы рассмотрим в соответствующей главе. Но платформа должна предоставлять полноценный спектр инструментального обеспечения для автоматизации всех аспектов продуктов организации. То есть концепция продукта организации должна иметь инструментальную платформенную поддержку. Важное свойство отсутствия замкнутости платформы позволяет ей поддерживать развитие продуктов и их представлений, требующих новых технологических решений. При этом свойство открытости позволяет продуктовым командам вносить изменения в платформу, публиковать их на уровне платформы, делая их доступными для смежных команд, повышать тем самым общую производительность труда. Таким образом резко снижается потребность в синхронизации команд, столь критичная в приведенном выше примере.

Синхронизация технологического развития платформы (отсутствие замкнутости) и продуктового развития организации позволяет обеспечить ту самую интенсивность, которая является ключом к конкурентоспособности в современном цифровом мире. Технологии, обеспечивая эффективную сквозную автоматизацию, создают простор для продуктового развития, ускоряя и стимулируя его, а развитие в свою очередь требует новых технологий для автоматизации, которые (опять же вследствие отсутствия замкнутости) добавляются в платформу и платформенные приложения.

Читатель вновь может задать вопрос: «Но разные уровни продуктовой автоматизации требуют различных технологий и подходов, как все это обеспечится платформой?» И ответ даст архитектура платформы: последняя должна обладать составной многомодульной структурой, которая обеспечивает создателей приложений сервисами для решения различных задач продуктовой автоматизации в универсальном формате. Взаимосвязь платформ и платформенных сервисов мы рассмотрим в соответствующей главе. Открытость платформы же позволяет создавать расширения модулей, сервисов и самой платформы командам развития продуктов, расширяя возможности инструментального обеспечения.

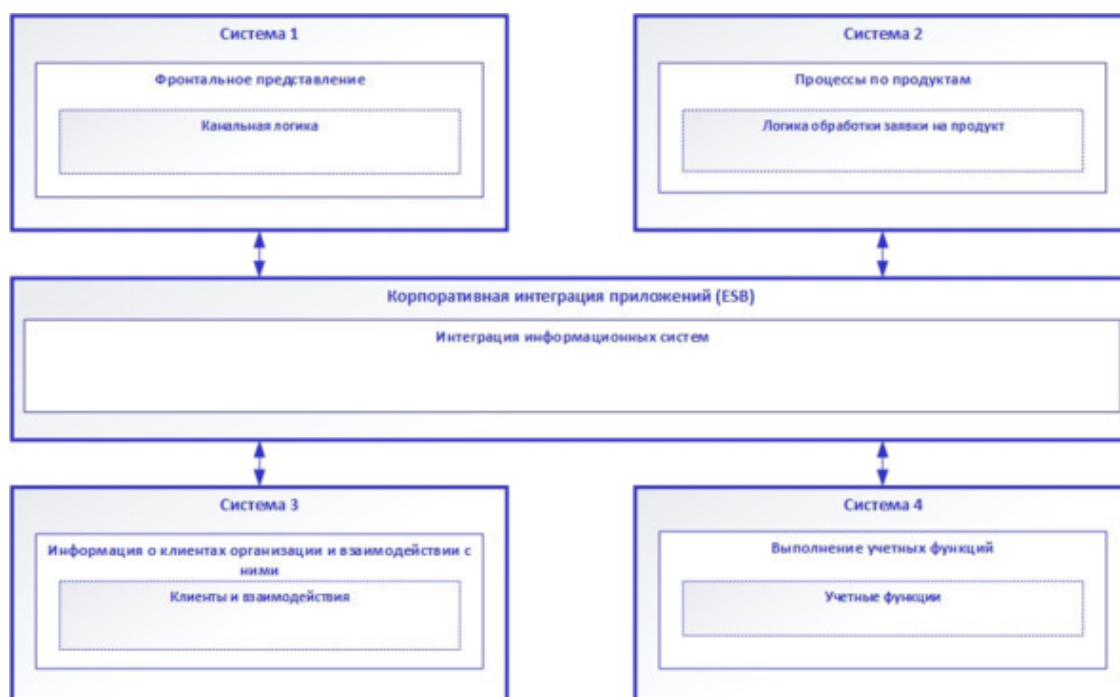
Указанное согласованное развитие продуктовых команд, эффективно адаптирующих и применяющих гибкие практики разработки, их совместная работа с использованием платформенных технологий и сервисов позволяют обеспечить и совместное развитие mindset на пути достижения уровня, адекватного современному цифровому миру. И пройти этапы эффекта Даннинга – Крюгера до уровня «плато стабильности» становится не в пример легче, главное же – достичь его оказываются в состоянии все команды, задействованные в развитии организации.

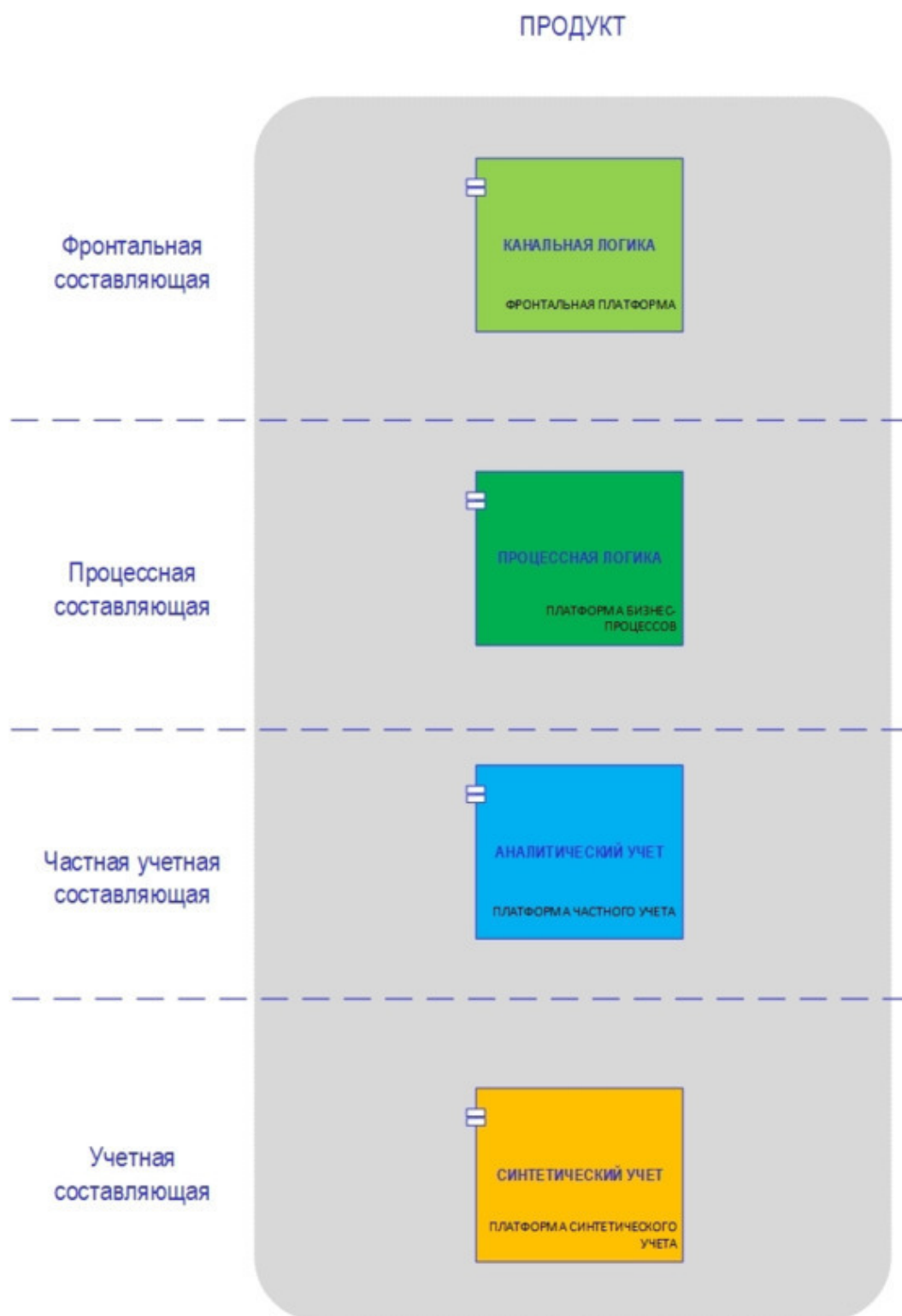
Фактически, если применять закон S-кривой и взять за старт последней классическую SOA-архитектуру, то можно констатировать, что рассмотренный выше пример платформенного подхода, заключающийся в использовании множества «платформ», находится на рабочем подготовительном этапе (значительные инвестиции с невысоким результатом), при этом использование современной платформы относится к этапу интенсивного внедрения технологий. Говорить об этапе стагнации платформенного подхода пока рано, поскольку рынок автоматизации нельзя считать насыщенным современными платформенными технологиями (речь идет именно о полноценных цифровых платформах, а не о частных «платформах», которые скорее осложняют автоматизацию). Схематично развитие платформенного подхода представлено на Рисунке 3.

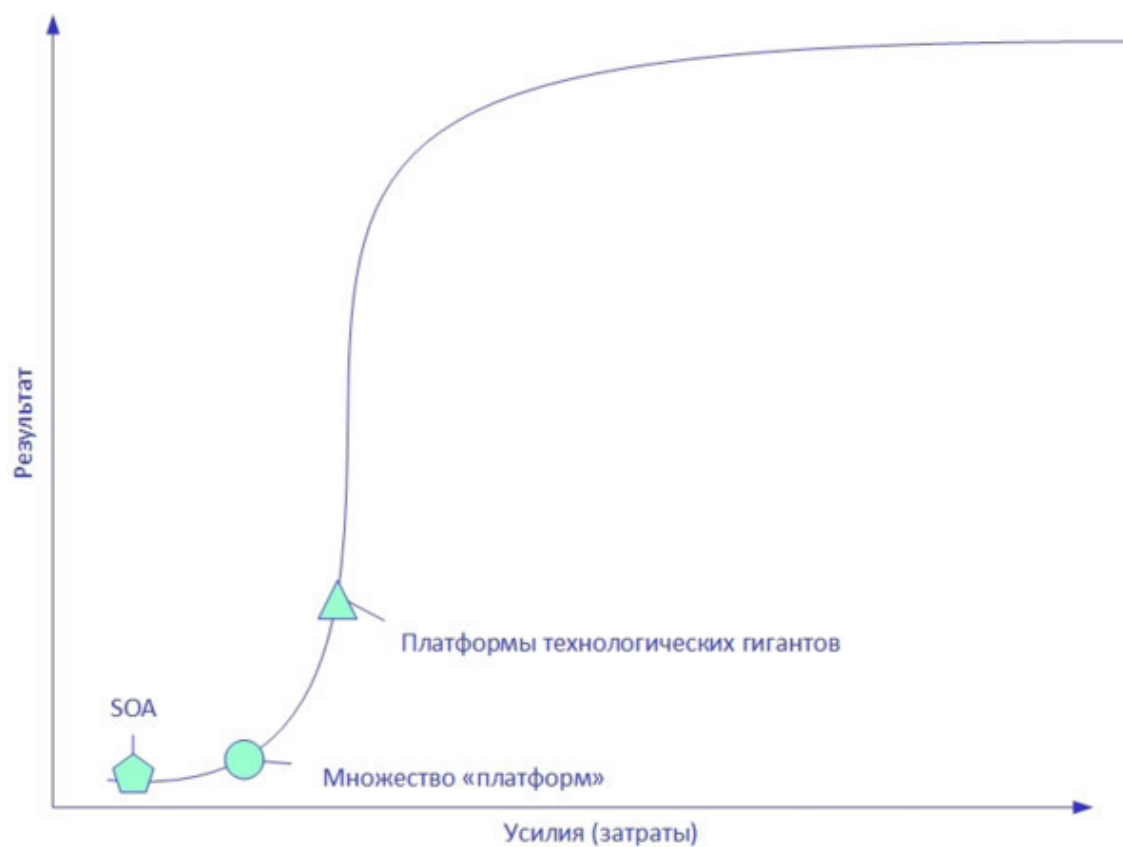
Рисунок 3. S-кривая платформенного подхода

Получается, что создание цифровой платформы, позволяющей обеспечивать достижение современного архитектурного mindset, является ключевым элементом перехода организации

к интенсивному развитию в цифровую эпоху. Отметим, что технологические гиганты, создающие и развивающие собственные платформы, идут именно этим путем. Развитая архитектура платформы, тезисно представленная выше, позволит создавать продукты, несущие ценность клиентам и партнерам организации. При этом архитектура платформы должна следовать ключевым тенденциям развития архитектуры в целом, рассмотрению чего будет посвящена следующая глава.







Цифровые платформы и ключевые тренды развития архитектуры

Еще раз о ключевых трендах развития архитектуры

В предыдущей книге «Архитектура цифрового мира» ключевым трендам развития архитектуры был посвящен специальный раздел ввиду исключительной важности данной темы. При этом мы говорили о тех трендах, которые актуальны для современной архитектуры (а также актуальны для цифрового мира) в целом. С учетом той важности платформ, которая заставила нас вновь «взяться за перо», данные тренды актуальны и для платформенного подхода. Напомним их читателю:

- Открытый код.
- Распределенность.
- Бизнес-процессы.
- Данные.
- Искусственный интеллект.

Применение решений с открытым исходным кодом позволило резко повысить производительность труда при создании ИТ-решений. Основой повышения стало значительное увеличение цепочки разделения труда по сравнению с созданием «закрытых» (vendor-lock) решений. Если последние были ограничены масштабами конкретной организации, выступавшей поставщиком (вендором) ИТ-решения, то в случае успешного свободно распространяемого решения, основанного на открытом исходном коде, становится возможным вовлекать принципиально большее сообщество разработчиков в развитие такого решения. Например, являющуюся одной из самых популярных нереляционных СУБД Apache Cassandra развивают десятки крупнейших технологических компаний по всему миру, используют же и вносят точечные исправления сотни. Подобный масштаб недостижим для любой корпорации, развивающей собственные закрытые решения, пусть даже и исключительно масштабной. Указанный подход позволяет резко наращивать производительность труда вследствие включения в цепочку развития все новых и новых специалистов.

Платформы также являются средством повышения эффективности деятельности организации и роста производительности труда (разумеется, мы говорим именно о тех платформах, к архитектуре которых надо стремиться, что было отмечено в предыдущей главе). Было бы странно предположить, что два фактора, обеспечивающих рост производительности труда в современном цифровом мире, не должны присутствовать и взаимодействовать в современной организации, более того, не учитывать друг друга. К сожалению, именно такой деструктивный подход регулярно встречается и на сегодняшний день в ряде организаций. Тем не менее лучшие практики говорят о том, что решения с открытым исходным кодом стремятся к платформенному подходу, лучшие же платформы активно используют решения с открытым исходным кодом. Поэтому первая ключевая тенденция развития платформ – синергия платформ и решений с открытым исходным кодом.

Распределенность рассматривалась нами в «Архитектуре цифрового мира» в двух смысловых плоскостях: распределенное исполнение современных технологических решений и организационная распределенность команд, создающих и развивающих эти решения. Обе смысловые составляющие более чем применимы к платформам. Платформы должны обеспечивать эффективное создание и исполнение приложений (платформенных приложений), несущих ценность клиентам и партнерам организации. Ценность несут продукты организации, авто-

матизация (и цифровизация) которых осуществляется с помощью платформенных приложений. Современные продукты доступны клиентам во всех часовых поясах и климатических зонах в режиме 24×7. Подобная доступность предполагает непрерывность бизнеса, которая диктует требования к используемой платформе. Платформа должна поддерживать указанный режим функционирования и предоставления продуктов, то есть сама стать распределенной. Длительные периоды недоступности, сервисный останов, ожидание времени отклика – все это признаки технологического застоя и деградации, ведущих к утрате организацией конкурентоспособности на рынке. Поэтому современная платформа (как и платформа будущего) – это распределенная платформа, обеспечивающая эффективное исполнение распределенных же платформенных приложений.

Архитектура распределенных программных комплексов, к которым относятся и сами платформы, и платформенные приложения, является достаточно сложной по сравнению с архитектурными решениями предыдущих поколений. И создавать ее ограниченной командой крайне затруднительно (конечно, такой вариант также возможен, но он приводит к неприемлемым в современном цифровом мире временным затратам). Практика (которая, как известно, является критерием истины) показывает, что современные команды создания и развития платформ и платформенных приложений являются распределенными: как члены одной команды могут находиться в самых разных точках земного шара, так и работа нескольких команд над современными технологическими решениями не только допускает, но и настаивает на их распределенной работе. Здесь есть прямое пересечение с вопросами использования решений с открытым исходным кодом: если успешные решения развиваются десятками (а иногда и сотнями) организаций, то отдельные специалисты и команды развития априори не могут не быть распределенными. Соответственно, и платформы, составляющие вместе с платформенными приложениями полноценные экосистемы, использующие развитые решения с открытым кодом, также развиваются распределенными командами.

Мы уже отмечали ранее, что современные компании, ставящие целью сохранение конкурентоспособности в цифровом мире, стараются уйти от процессной организации деятельности, отказываются от нее в пользу продуктовой. Подобный переход достаточно легко объясним: лишь эффективное взаимодействие с клиентами и партнерами, позволяющее решать их задачи, обеспечивает конкурентоспособность, а решение клиентских и/или партнерских задач осуществляется за счет предоставления им ценности. Ценность же выражается в продукте. В задачи настоящей книги не входит описание всей продуктовой и ценностной концепции, желающие могут изучить разнообразную литературу, посвященную гибким практикам, разработкам клиентского пути и т. д. Мы лишь констатируем, что продукт в организации первичен. Но первичность продукта не означает отсутствия иных сущностей. Продуктовый подход декларирует первичность продукта, но не отрицает бизнес-процессы. Последние в обязательном порядке присутствуют в современной организации, но их задача в реалиях интенсивного развития – описывать жизненный цикл продукта, таким образом, мы приходим к сегментации бизнес-процессов. Вслед за продуктами, группами продуктов или продуктовыми доменами мы имеем все основания определить группы бизнес-процессов, характеризующих конкретные продукты.

Так как платформа поддерживает (и даже продвигает) продуктовый подход, то она обязана поддерживать и бизнес-процессы, связанные с продуктами. Под поддержкой в данном случае понимается широкий спектр активностей: здесь и проектирование бизнес-процессов, и исполнение их экземпляров, и поддержка версионности, и мониторинг (технический и прикладной), и выставление КПЭ процессов, поиск узких мест. Разумеется, непосредственное исполнение экземпляров процессов ассоциируется с конкретными платформенными приложениями, но в задачи платформы входит предоставление инструментальных средств поддержки столь важного функционала, что снимает повышенную нагрузку с каждого выделен-

ного приложения, позволяя не изобретать велосипед, а решать конкретные задачи, связанные с потребностями клиентов и/или партнеров организации. Распределенность же платформ и платформенных приложений диктует необходимость обеспечения распределенности исполнения бизнес-процессов, что также является одним из важнейших аспектов поддержки платформами тренда развития архитектуры, связанного с автоматизацией процессной деятельности (в рамках привязки ее к продуктовому подходу, учитывая указанное выше первостепенное, но не единственное значение продуктов).

Значимость данных и их использования растет с каждым годом. Необходимость следования соответствующему тренду развития архитектуры стала еще более очевидной с момента публикации предыдущей книги автора, где данный тренд также специально подчеркивался. Данным, необходимым организациям, становится все больше, потребности в их эффективном хранении и обработке все выше. Использование для решения подобных задач архитектурных подходов предыдущих поколений попросту недопустимо: представим себе очередную (уже которую по счету) BI-систему, продаваемую лицам, принимающим решения в организации, в качестве «серебряной пули», обеспечивающей качественную аналитику и принятие решений. И раз за разом подобные «спасительные» инструменты терпят крах вследствие низкого качества данных организации, необходимости взаимодействия с несвязанными источниками данных, сложнейшими алгоритмами обработки последних и т. д. Подобный пример приведен с целью подчеркнуть актуальность проблемы с данными, которая не исчезла, а скорее усугубилась за последние годы. И цифровая платформа, предоставляющая средство обеспечения перманентного интенсивного развития, должна помогать в решении проблем с данными, обеспечивать следование указанной тенденции развития архитектуры.

В рамках поддержки управления данными в современной архитектуре платформа должна обеспечивать:

- Возможность следования самым разным концепциям управления данными (за разработку подобных концепций отвечают, как правило, выделенные специалисты), используемым в организации.
- Осуществлять поддержку сбора данных из источников (распределенных, что показывает связь с обозначенной выше распределенностью как тенденцией развития архитектуры), их обработку, подготовку витрин (как аналитических, создающихся за продолжительные промежутки времени, так и создаваемых в режиме реального времени).
- Создание универсальных форматов отображения (в этом случае BI-инструменты могут и не входить в состав платформы, но, работая с предоставляемыми выгрузками, выполнять заявленные функции вполне качественно).

Еще раз необходимо уточнить, что непосредственную работу с данными осуществляет не сама платформа, а платформенное приложение (в некоторых случаях платформенный сервис; разграничение между платформенным сервисом и платформенным приложением будет рассмотрено в соответствующей главе). Но платформа предоставляет все необходимые инструменты для создания и работы приложений.

Искусственный интеллект как тенденция развития архитектуры (а скорее как тенденция развития мира) пережил с момента публикации «Архитектуры цифрового мира» целый взрыв интереса к технологии, при этом спектр эмоций менялся от страха и недоверия до обожания, граничащего с почитанием. Неудивительно, что существенная часть мнений в отношении технологий искусственного интеллекта относится к тому, что принято именовать «хайпом». Как правило, хайп имеет вполне конкретное приложение усилий, в последние годы этим приложением является ChatGPT. Но наличие хайпа не может и не должно заслонять главного: в мире существует огромный и объективный запрос на массовое применение технологий искусственного интеллекта. Этот запрос обусловлен ограниченностью определенных возможностей человеческого интеллекта (быстродействие, скорость принятия решений, объем

и долговременность памяти и т. д.). Следовательно, современная платформа должна поддерживать соответствующее технологическое направление (искусственный интеллект), его развитие и применение в приложениях. Если в предыдущей книге автор писал, что «на сегодня применение искусственного интеллекта ограничивается решением узкоспециализированных задач: выявление мошенничества, диагностика заболеваний, формирование стратегии интеллектуальных игр (шахматы, го), принятие решений в области рисков, промышленная роботизация», то по прошествии времени можно уверенно сказать, что соответствующий рубеж пройден. Пусть пока и на относительно примитивном уровне, но решаются задачи комплексного характера, связанные с систематизацией научных исследований, разработкой программного кода, подготовкой судебных решений, развитой роботизацией и т. д. Нельзя не отметить, что современные решения искусственного интеллекта весьма ресурсоемки, требуют значительных вычислительных мощностей (что приводит даже к перекосам на фондовом рынке). Поэтому к задачам поддержки соответствующей тенденции развития архитектуры можно отнести в том числе и оптимизацию потребления ресурсов при решении актуальных задач, которые ставятся перед технологиями искусственного интеллекта.

Нельзя не отметить, что следованием ключевым тенденциям развития архитектуры вопросы развития цифровых платформ не исчерпываются. Обеспечение интенсификации соответствующего развития требует и следования ряду тенденций, которые можно условно именовать частными. К числу последних можно отнести использование событийно-ориентированного взаимодействия, обеспечение эластичного масштабирования, облачное исполнение и т. д. В настоящей главе мы рассмотрим и вопросы следования частным тенденциям, пусть и тезисно по сравнению с ключевыми (детальное рассмотрение каждой тенденции развития архитектуры в применении к платформенному подходу потребует отдельной книги).

Цифровые платформы и открытый код

В «Архитектуре цифрового мира» уже указывалась дата 17 сентября 1991 года, когда публикация Линусом Торвальдсом исходного кода ядра операционной системы Linux стала отсчетом новой эры развития программных продуктов. Широкому кругу потребителей были представлены и заявили о себе программные продукты с открытым исходным кодом. Тем самым идеи разделения труда в цифровой сфере (пусть тогда и поднимались «лишь» вопросы частичной автоматизации) вышли на сцену. Но мало просто выйти на сцену. На сцену выходят миллионы актеров, но звездами становятся лишь единицы. И открытый код, несомненно, стал такой звездой.

Есть и вторая дата, значимая для истории открытого кода и всего цифрового мира: 28 октября 2018 года корпорация IBM объявила о покупке компании RedHat. Официальная стоимость сделки составила 34 миллиарда долларов. При этом результатом покупки стало не закрытие ранее открытых технологий (компания RedHat является ключевым игроком на рынке открытого кода), а выход на рынок открытого кода IBM в качестве не просто участника, но флагамена, развивающего решения с открытым исходным кодом на новом качественном уровне. Чем важна эта дата? Произошедшее событие стало знаковым и показало глобальные изменения, происходящие в цифровом мире. Компания, создавшая экосистему продуктов на все случаи жизни (аппаратные платформы, операционные системы, системы управления базами данных, серверы приложений, интеграционные решения, системы управления контентом и т. д.), сделала выбор в пользу открытого кода. \$34 млрд были вложены в экспертизу, в открытое сообщество, в перспективы развития, но не в заводы по производству электроники, закрытые лицензии или элитную недвижимость. Чуть больше 27 лет занял путь от публикации исходного кода до столь значимого корпоративного поворота – по меркам истории это произошло в исключительно короткие сроки.

Основной составляющей подобного успеха открытого кода стало кратное повышение эффективности при создании, внедрении и использовании продуктов с открытым исходным кодом, экономическое обоснование которого объясняется теорией Адама Смита. Мы не будем повторять все те доводы, что были изложены в «Архитектуре цифрового мира», а примем их за данность. Цепочка разделения труда при создании и развитии продуктов с открытым исходным кодом в предельном случае может включать все мировое сообщество ИТ-специалистов. Конечно, предельный случай практически недостижим, но и имеющиеся на сегодня примеры превосходят возможности даже крупнейших корпораций, а потому последние и совершили (можно говорить о сверившемся факте) поворот в сторону данной тенденции.

Что же данный поворот означает для платформ? Безусловно, платформа не является информационной системой, платформа не является обособленным программным комплексом. Но программным комплексом она является, причем исключительно сложным – у автора и в мыслях нет пытаться убедить читателя в простоте современных платформ. И повышение производительности труда при создании столь сложных комплексов является одной из важнейших задач современности. Если мы говорим, что платформа является ценностным мультипликатором, то создание и последующее развитие такого мультипликатора не приемлют невынужденных издержек в производительности труда. Особенно при необходимости обеспечения развития интенсивного. А альтернативой интенсивному развитию, как мы помним, являются застой и деградация. Если платформа является ценностным мультипликатором при создании и развитии продуктов, предоставляемых организацией клиентам и партнерам, а открытый код является мультипликатором эффективности при создании программных продуктов, то объединение данных мультипликаторов становится требованием времени. То есть современная платформа должна базироваться на решениях с открытым исходным кодом.

Но мало продекларировать использование решений с открытым исходным кодом, необходимо обеспечить следование ключевым принципам указанного использования. Рассмотрим вопросы данного следования с точки зрения характеристик платформы, отмечавшихся ранее: платформа не является информационной системой, платформа не является выделенным программным комплексом, платформа является открытой и не является замкнутой. Также для наглядности рассмотрим продукт, автоматизация которого была показана в главе, посвященной проблематике современных цифровых платформ, с помощью множества «платформ». Визуализация данного продукта представлена на Рисунке 4.

За автоматизацию соответствующих составляющих продукта отвечают различные платформенные сервисы, а сам продукт представлен платформенным приложением. Вопросы соотношения платформ, сервисов и приложений будут рассмотрены в дальнейшем, в настоящем разделе мы концентрируем внимание на поддержке философии открытого кода, а также на получаемых от этого преимуществах.

Рисунок 4. Продукт, автоматизируемый при помощи современной платформы

Предположим, что платформенные компоненты и предоставляемые при их помощи платформенные сервисы реализуются на основании программного обеспечения с открытым исходным кодом. При этом продукт, как и в примере, приведенном в главе, посвященной проблематике современных цифровых платформ (см. Рисунок 2), содержит учетную, частную учетную, процессную и фронтальную составляющие. Также примем за основу современный архитектурный mindset, детально представленный в «Архитектуре цифрового мира».

Основываясь на рассмотренном ранее примере продукта, отметим следующее:

- На уровне учетной составляющей требуется эффективное выполнение групповых операций над большими объемами данных, надежное долговременное хранение информации, высокая скорость записи вновь поступающих данных.
- На уровне частной учетной составляющей (аналитический учет) требуется гибкая настройка продукта (группы продуктов), адресное формирование настроек, гибкая работа пользовательских интерфейсов, клиентами которых являются в первую очередь сотрудники организации, отвечающие за развитие продуктов (владельцы продуктов и смежные им роли). Также обязательным требованием рассматриваемой составляющей является надежное и долговременное хранение информации.
- Настройка бизнес-процессов нуждается в высокой гибкости, кроме того, специфика инструментов, обеспечивающих гибкую алгоритмизацию проводимых операций, предъявляет требование наличия компонентов, обеспечивающих дополнительную производительность.
- Автоматизация фронтальной составляющей немыслима без применения технологий, обеспечивающих высокую клиентскую привлекательность продукта, адаптивность создаваемых интерфейсов, невысокую нагрузку на устройства, посредством которых осуществляется взаимодействие с продуктом, а также высокую производительность, которая является обязательной составляющей современного клиентского пути.

Как видно из перечисленного списка, требования, предъявляемые при автоматизации различных составляющих продукта, пересекаются по принципу кругов Эйлера. Но мы имеем все преимущества, которые предоставляет нам концепция открытого кода. И здесь необходимо учитывать, что многие из тех задач, которые мы планируем решить при автоматизации соответствующего продукта, уже решались ранее (пусть и частным, а не платформенным образом). И современный архитектор, являясь лидером технологических изменений, обязан это учитывать.

Итак, требование к надежному и долговременному хранению информации, предъявляемое составляющими синтетического и аналитического учета, выполняется путем использования современных технологий хранения, обеспечивающих эластичное масштабирование. Более

того, в случае если соответствующие технологии предоставляют возможность сегментации (раздельного и независимого хранения и управления массивами данных), то мы можем использовать общую технологическую базу при хранении соответствующей информации как для учетной, так и для частной учетной составляющих. Примерами подобных технологий, которые могут использоваться для решения указанной задачи, могут служить Apache Cassandra и Apache Hadoop, являющиеся продуктами с открытым исходным кодом. Указанные продукты используются технологическими гигантами и их клиентами и партнерами и доказали свою состоятельность. Для групповых операций может использоваться связанная технология – например, Apache Spark, также представляющая собой продукт с открытым исходным кодом. Более того, Apache Spark предоставляет возможности бесшовной интеграции с тем же Apache Hadoop, существенно снижая трудозатраты команд разработки.

Обеспечение высокой скорости записи возможно при использовании технологий как распределенного хранения, так и потоковой интеграции; примерами последних могут выступать Apache Kafka или Apache Pulsar. Указанные технологии обеспечивают высокую скорость записи и множество доступных смежных технологий, отлаженных сообществом разработчиков и готовых к применению. Отметим, что интеграция может быть вертикальной (с точки зрения Рисунка 4) и пронизывать все составляющие автоматизации продукта сквозным образом. Также Apache Kafka и Apache Pulsar (в меньшей степени) поддерживают бесшовную интеграцию (с использованием открытых компонентов) с современными технологиями долговременного хранения информации. Нельзя не отметить, что приведенные технологии потоковой интеграции представляют собой продукты с открытым исходным кодом.

Требования по созданию гибких и эффективных пользовательских интерфейсов являются общими для всех составляющих продукта (в современном мире даже синтетический учет нуждается в развитом пользовательском интерфейсе). Поэтому резонным становится общий подход к созданию как фронтального представления продукта в целом, так и интерфейсных элементов каждой составляющей продукта, предоставление готовых компонентов в виде библиотек, их переиспользование, а также общие подходы к дизайну. И основные фреймворки создания фронтальных компонентов, применяемые на сегодняшний день, – React, Angular, Vue.js – также отвечают принципам открытого кода.

Высокая производительность работы с данными для фронтального слоя может обеспечиваться как кэширующими элементами, применяемыми для обеспечения эффективности индексирования данных, использования быстрого поиска при работе, например, со статическими массивами информации, так и сложными программными комплексами, применяемыми для кэширования, предполагающего частое обновление данных, так называемый «прогрев кэша». Также во втором случае (сложные программные комплексы и связанные с ними задачи) актуальным будет не просто кэширование информации (формат In-Memory Data Grid, IMDG), а полноценная работа с ней в оперативной памяти (формат In-Memory Data Base, IMDB), предполагающая в том числе поддержку языков работы с данными, привычных многим современным разработчикам. Примерами таких инструментов могут служить Apache Ignite или Infinispan; в качестве примера инструмента, поддерживающего первый случай кэширования (индексирование данных и быстрый поиск), можно привести OpenSearch. Указанные продукты являются решениями с открытым исходным кодом. И применяться соответствующие инструменты могут не только для повышения производительности фронтальной составляющей, но и, например, в процессной составляющей автоматизации продукта.

Дело в том, что традиционно инструменты, обеспечивающие гибкую настройку и исполнение бизнес-процессов, их мониторинг, выставление и расчет КПЭ, имеют не самую высокую производительность, а потому нуждаются в дополнительных компонентах, упомянутую производительность повышающих. И это могут быть уже приведенные выше компоненты, применяемые одновременно и для обеспечения корректной автоматизации фронтальной

составляющей продукта. А их (компонентов) распределенное исполнение, эластичное масштабирование и опции сегментации позволяют проводить тонкую настройку, адресно применимую для каждой составляющей автоматизации продукта. В свою очередь, для непосредственной автоматизации процессной составляющей продукта могут использоваться самые разные решения с открытым исходным кодом: Activiti, Camunda, Kogito и другие. Указанные инструменты активно применяются огромным количеством самых разных мировых корпораций, для них разработаны различные топологии развертывания, адекватные тем или иным вариантам использования.

Читатель может задать вопрос в лоб: «Неужели достаточно просто развернуть набор технологий, например приведенный автором, чтобы достичь современного платформенного подхода?» Ответ на подобный вопрос может быть только отрицательным: развертывание технологий является промежуточным и не определяющим, хоть и важным, этапом работы с цифровыми платформами. Более того, этот этап может быть применен и в случае следования парадигме SOA, и в случае применения «платформенного» подхода, разобранный в предыдущей главе, посвященной проблематике современных цифровых платформ. Что же отличает современный платформенный подход?

Во-первых, мы должны исходить из того, что платформа не существует сама по себе, она является средой создания и исполнения приложений. Платформа не является ни информационной системой, ни выделенным программным комплексом, по факту платформа является основой прикладной экосистемы организации. Отсюда может следовать первый вывод: многообразие решений с открытым исходным кодом не означает, что каждая составляющая продукта, рассматриваемого в примере, автоматизируется независимым технологическим стекком. В «Архитектуре цифрового мира» мы говорили о том, что единственным критерием выбора технологических решений является целесообразность. И когда мы говорим об экосистеме платформенных приложений, о продуктах, предоставляемых указанными приложениями, то крайне неразумно будет рассуждать о том, что каждый продукт, каждая составляющая продукта автоматизируется независимо от других. Да, жизненный цикл продуктов различен, каждый продукт обладает собственным P&L, каждый продукт может получать совершенно разные импульсы к развитию, основанные на множестве факторов. Но платформа помогает им развиваться, она, как мы уже отмечали, является ценностным мультипликатором. И архитектор, занимающийся проектированием платформы, приходит к выводу, что технологический стек, использующийся при проектировании и реализации платформы и платформенных приложений должен быть согласованным. Нет никакого смысла использовать собственные системы управления базами данных для каждого продукта и каждой составляющей такого продукта, если это не диктуется объективными потребностями. Если мы говорим о решениях по обработке данных в оперативной памяти, позволяющих обеспечить требуемые производительность и надежность для процессной и фронтальной составляющих, то имеет смысл рассматривать технологическую унификацию данных решений, а также проектировать и реализовывать унифицированные платформенные сервисы инфраструктурного и инфраструктурно-прикладного характера (подробнее о платформенных сервисах – в соответствующей главе) для обеспечения работы указанных продуктовых составляющих. Разумеется, отсюда вовсе не следует необходимость использования единственно правильной технологии, мы утверждаем необходимость решения схожих технологических задач схожими технологиями. Представим себе, что обработка данных в оперативной памяти производится на уровне процессной составляющей продукта средством решения Infinispan, фронтальной – Apache Ignite, при этом для выделенного подмножества дистанционных каналов – Redis. Поддерживать и развивать подобную конфигурацию будет непомерно затратно. Таким образом, мы приходим к логике технологической унификации платформы и ее сервисов. При этом унификация проводится в логике решений с открытым исходным кодом: выбор используемых в рамках унификации технологий может

производиться как выделенно, то есть с точки зрения максимально эффективного технологического решения, так и совместно, когда оценивается эффективность экосистемы технологий (например, с точки зрения возможностей упрощения технологической интеграции).

Во-вторых, структура платформы определяется теми прикладными вариантами использования, исполнение которых планируется для продуктов, создаваемых с применением платформы. А варианты использования диктуют те характеристики платформы, которые обеспечат требуемый P&L продуктов. То есть в рамках технологической унификации, представленной выше, топологии развертывания технологических решений, применяемых в платформе, могут существенно различаться. Например, упомянутый выше Apache Ignite может иметь различные топологии развертывания и сервисное окружение для вариантов автоматизации процессной и фронтальной составляющих. И указанные топологии диктуются вариантами использования. Здесь мы видим очередную синергию платформенного подхода и философии открытого кода: успешные решения с открытым исходным кодом используются огромным числом организаций по всему миру, изменения в данные решения вносятся аналогично огромным числом пользователей, поэтому количество доступных вариантов топологий оказывается априори большим, нежели у закрытых (vendor-lock) решений, даже если последние являются собственностью крупнейших корпораций. И для каждого частного платформенного решения может выбираться (а в случае необходимости и создаваться) собственная топология развертывания технологического продукта. То есть мультипликативный ценностный эффект платформы накладывается на мультипликатор эффективности открытого кода. Мы можем создавать множество топологий в рамках технологической унификации применяемых решений в зависимости от требований к той или иной составляющей продуктовой автоматизации.

Резюмируя вышеизложенное, мы можем отметить, что при использовании современного платформенного подхода обеспечиваются:

- Технологическая унификация, в рамках которой могут оформляться те или иные подсистемы платформы – например, отвечающие за обработку данных в оперативной памяти и соответствующее повышение производительности и надежности платформенных приложений.
- Подсистемы платформы предоставляют сервисы (и сами могут предоставляться в виде сервиса) разного уровня платформенным приложениям, в ходе реализации которых создаются продукты организации, предоставляющие ценность клиентам и партнерам.
- Подсистемы могут предоставлять как общие сервисы приложениям и их компонентам, отвечающим за те или иные продуктовые составляющие, так и адресные сервисы, определяемые вариантами использования.
- Многообразие подсистем и их топологий основывается на развитии решений с открытым исходным кодом, так как не может ограничивать себя привязкой к некоему выделенному поставщику программного обеспечения.

Резонным будет рассмотреть вопрос возможности применения аналогичных принципов к рассмотренному ранее примеру множества «платформ». На самом деле, активным образом применяя указанные принципы, мы приходим к тому, что подобные «платформы» сливаются в единую, при этом остается организационное разграничение команд развития соответствующих «платформ». Ставя целью обеспечение интенсивного развития, современная организация логически приходит к необходимости устранения подобного разграничения, в результате чего мы сталкиваемся уже с примером, рассмотренным в настоящем разделе. То есть интенсивное развитие требует ухода от множества «платформ», и исключительно важную роль в этом уходе играет философия открытого кода.

Безусловно, рассмотренный выше пример не является детальной инструкцией по проектированию и разработке ИТ-решений автоматизации продуктов с использованием платформенного подхода. Задачей примера является показать соответствие современной платформы

открытому коду как ключевой тенденции развития архитектуры, проявления современного архитектурного mindset, актуальные задачи архитектора, являющегося лидером технологических изменений. Приведенные выше технологические решения с открытым исходным кодом являются примером обеспечения необходимой платформенной поддержки решения задач по цифровизации, но никак не панацеей.

Попробуем ответить на вопрос, обязательно ли использование технологий с открытым исходным кодом для приведенного выше решения задач продуктовой автоматизации с использованием платформенного подхода. Да, можно обойтись без решений с открытым исходным кодом. Можно применять закрытые технологические решения, подключить соответствующих поставщиков. Можно разработать как платформу, так и платформенные приложения, обеспечивающие продуктовую автоматизацию, с нуля, не подключать базис открытого кода. Оба приведенных варианта представляют собой альтернативы использованию решений с открытым исходным кодом. Но что же произойдет, если следовать подобным альтернативам? Фактически что в одном, что в другом случае мы отвергнем возможность повышения эффективности за счет увеличения длины цепочки разделения труда. В одном случае мы доверимся некому поставщику программного обеспечения, который априори не может обеспечивать эффективность, сопоставимую с сообществом разработчиков открытого кода (при использовании ведущих решений), ограничим число как применяемых технологий, так и топологий их развертывания. Во втором случае взвалим на себя тяжкую ношу разработки компонентов собственными силами, игнорируя достижения ведущих мировых технологических игроков. В таком случае логично и компиляторы не использовать внешние, а разработать самостоятельно. А там и до реализации собственной электронной компонентной базы недалеко. Мы сознательно доводим примеры до абсурда, показывая, что именно следование открытому коду как ментально (при проектировании и реализации платформенных решений), так и технологически является той золотой серединой, позволяющей организациям добиться максимальной производительности труда и эффективности на пути цифровой трансформации. **всех**

Рассмотрим те аспекты экосистем продуктов с открытым кодом и их развития, использование которых (аспектов) позволяет фундаментально изменить мировоззрение компаний при реализации новых решений в части применения современного платформенного подхода. В «Архитектуре цифрового мира» мы взяли за основу тезисы Джима Уайтхерста (James M. «Jim» Whitehurst), изложенные им в книге «Открытая организация: Страсть, приносящая плоды» (2015, ISBN 978-5-9693-0405-5): мотивация, меритократия, прозрачность принятых решений, развитие новых направлений. Рассмотрим данные тезисы в контексте платформенного подхода.

- Мотивация. Указанный тезис, столь важный для современного цифрового мира, подразумевает, что эффективность работы коллектива резко возрастает при обеспечении слаженной совместной работы всех его участников; при этом каждый член коллектива работает более эффективно, воочию наблюдая значимость собственного вклада в общее дело. В предыдущем своем труде («Архитектура цифрового мира») мы отмечали, что развитие решений с открытым исходным кодом является наглядным примером эффективной совместной работы. Современный платформенный подход, основанный на философии открытого кода, идет дальше: стираются границы между командами, обеспечивающими цифровую трансформацию, исчезают команды развития отдельных информационных систем, команды развития отдельных «платформ». В рамках общей платформы, использующей открытый код, все команды создают и публикуют изменения, доступные коллегам. При этом, развивая платформу, формируя новые топологии и дополнения в решения с открытым исходным кодом, команды должны публиковать их в сообществах, дополнительно повышая производительность. Мы уже отмечали, что платформа становится ценностным мультипликатором, открытый код является мультипликатором эффективности, а их совместное и согласованное использование является и мультипли-

котором мотивационным, позволяющим резко повышать эффективность командного труда, создавать современные решения. И важнейшее слово здесь принадлежит архитектору, который должен возглавлять технологические изменения, определять структуру платформ, состав и компоненты используемых технологий, способы их применения, необходимые дополнения и т. д. Но все это возможно лишь при достижении уровня *mindset*, адекватного современному цифровому миру.

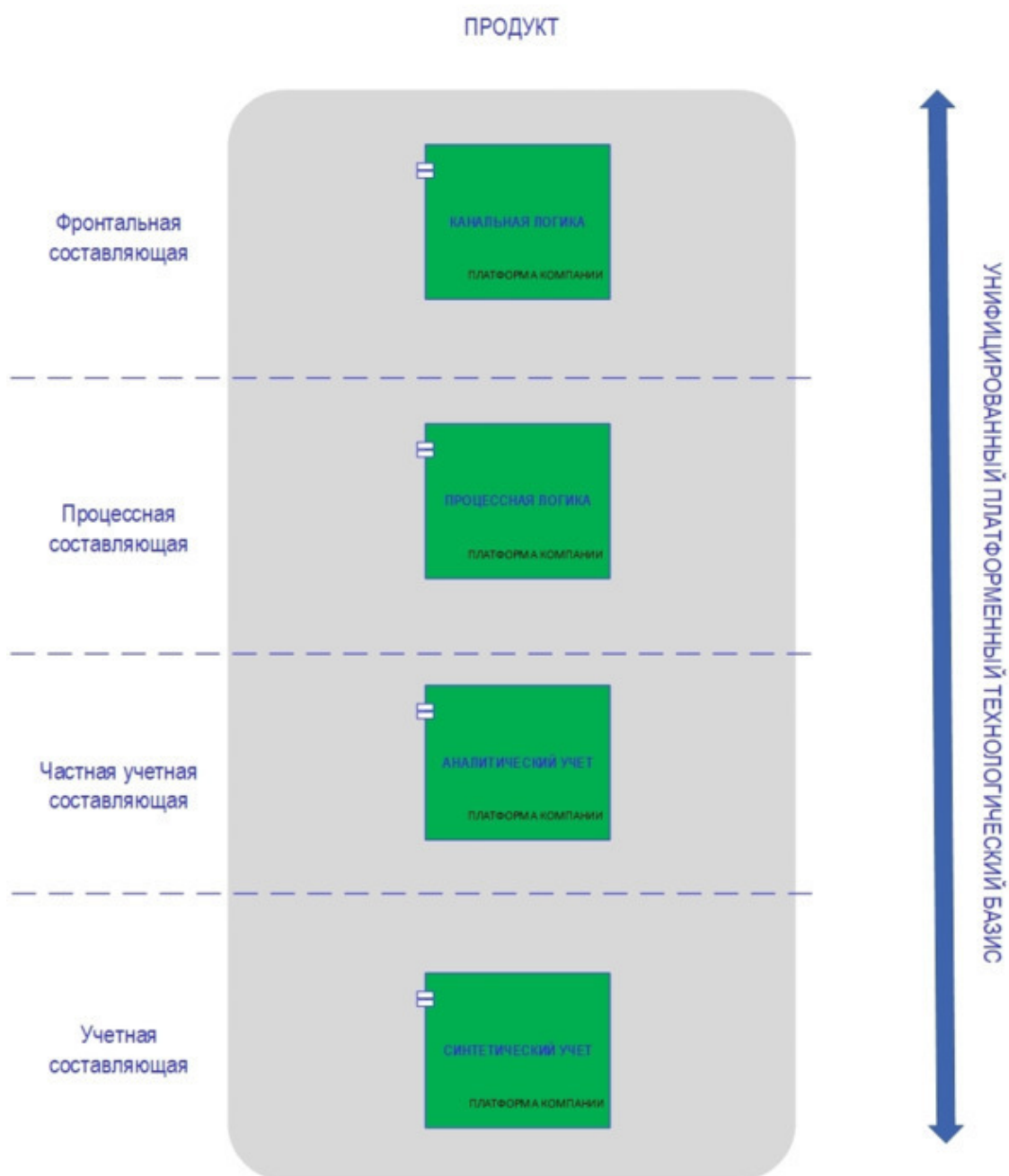
- Меритократия. Суть меритократии в контексте нашего рассмотрения заключается в том, что полномочия при принятии решений соразмерны вкладу участников в создание и/или развитие ИТ-решения. Данный принцип обеспечивает мотивацию ответственных членов команды принимать участие в выработке технологических решений. В случае применения платформенного подхода имеет смысл говорить не только о меритократии отдельных участников рабочего процесса, но о меритократии команд, ведь в рамках платформы и создаваемых на ее основе платформенных приложений в процессе цифровой трансформации участвует множество команд развития. И все команды развития обязаны предлагать решения в том числе уровня платформы, публиковать их, делать общедоступными хотя бы в масштабе организации. И полномочия при принятии решений командами, вес мнения тех или иных команд будут тем выше, чем больший вклад они вносят в развитие платформ. Именно таким образом команды продвигают свое видение, обеспечивают лучшие технологические решения, мотивируют лучших участников рабочего процесса становиться членами именно самых успешных команд; по факту таким образом формируется добросовестная конкуренция на уровне развития не отдельных технологий (пусть и успешных), но целых цифровых платформ. То есть меритократия дополнительно усиливает фактор мотивации, а драйвером усиления становятся платформы (при условии использования философии открытого кода). При этом исключительно важной является роль архитектора, ведь именно представитель данной роли отвечает в том числе за учет предложений, оценку их веса и последующие полномочия команд, основанные на публикуемых обновлениях платформы и ее составляющих.

- Прозрачность принятых решений. Мотивированные команды разработки, их участники, выступающие с принципов меритократии, обеспечивают прозрачность решений, принимаемых в процессе развития платформ и создаваемой с их помощью ценности для клиентов и партнеров организации. Прозрачность решений позволяет обеспечить наглядность тренда развития, задаваемого как для платформы в целом, так и для платформенных сервисов и приложений. А наглядность тренда развития в свою очередь обеспечивает более полное вовлечение всех участников рабочего процесса в дальнейшее развитие, интенсифицируя его. Таким образом, использование платформенного подхода, отвечающего принципам открытого кода, повышает прозрачность принятых решений, мотивацию всех членов команд развития и обеспечивает кардинальное повышение производительности труда. При этом в задачи архитектора входит наглядная выработка решений, помощь в их оценке, формирование архитектурных фреймворков, обеспечивающих прозрачность принятия решений, утверждение их на уровне платформы и постоянное развитие.

- Развитие новых направлений. Рассматриваемый тезис предполагает максимально быструю проверку возникающих идей, их шлифовку уже в ходе практического применения. Платформенный подход полностью укладывается в следование данному тезису. Команда прорабатывает новые идеи, при их успешности публикует их на уровне платформы, делая тем самым доступными для смежных команд развития, которые также обкатывают соответствующие наработки, выявляя ошибки и неточности, повышая качество и добавляя новые применимые варианты использования. Тем самым резко повышается производительность труда, а каждая ошибка (в полном соответствии с фундаментальными постулатами гибких практик) становится не барьером, а вызовом на пути развития.

На основании рассмотрения указанных тезисов мы можем утверждать, что платформы являются фундаментом построения открытых организаций, то есть становятся краеугольным камнем повышения эффективности цепочек разделения труда. В «Архитектуре цифрового мира» мы отмечали, что потенциал расширения цепочек разделения труда в ИТ далеко не исчерпан, но по мере его исчерпания встанет вопрос нового революционного перехода в сфере ИТ (эволюционное и революционное развитие архитектуры описаны в соответствующей главе там же). На сегодняшний день уже можно утверждать, что платформы стали основой революционного перехода, резко повышающего производительность труда в цифровой сфере, позволяющего выстраивать открытые организации во все новых областях человеческой деятельности. И этот переход находится в самом разгаре.

В завершение данного раздела автор выражает глубокую благодарность сообществу разработчиков и пользователей открытого кода за ту страсть, которую они добавляют в ИТ, революционное развитие данной сферы человеческой деятельности, создание прекрасных программных продуктов, используемых автором и его коллегами в своей работе.



Цифровые платформы и распределенность

В «Архитектуре цифрового мира» мы отмечали, что распределенность является одной из ключевых тенденций развития архитектуры в цифровую эпоху. При этом распределенность рассматривалась в двух смысловых плоскостях: организационной, предполагающей реализацию ИТ-решений силами распределенных команд, состоящих из специалистов самого разного профиля, и технологической, подразумевающей распределенное исполнение современных решений. Соответствующая тенденция актуальна и для платформ. Именно в контексте платформенного подхода распределенность будет рассмотрена в настоящем разделе.

Начнем с организационной распределенности. В современном цифровом мире использование гибких практик стало стандартом де-факто (напомним, что подобное использование может привести к попаданию в ментальные ловушки, разобранные в соответствующей главе «Архитектуры цифрового мира»). Гибкие практики (и их адаптации для применения в крупных компаниях) предполагают, что в организации определяются продукты, при этом продукты создаются и развиваются выделенными командами специалистов, минимальным образом зависящими друг от друга. Подобные команды могут находиться в самых разных точках земного шара, при этом все они должны работать согласованно (в архитектурном, а не управленческом смысле, поскольку последнее граничит с «микроменеджментом») для обеспечения создания ценности и сохранения конкурентоспособности организации.

Мы уже неоднократно отмечали (как в текущем изложении, так и в предыдущем труде автора), что открытый код и гибкие практики позволяют увеличивать разделение труда и тем самым кардинально повышать его производительность по сравнению с использованием «закрытых» решений (как технологически, так и при «закрытой» философии). В цепочку создания ценности вовлекаются все новые участники (как отдельные специалисты, так и команды), которые отвечают за свои участки деятельности, обеспечивая максимальную эффективность. При этом возрастает общая производительность труда в цепочке разделения, но обратной стороной медали являются резко возрастающие риски отдельных участников технологической цепочки. Ведь чем больше участников цепочки разделения труда (при создании общей ценности), тем больше они зависят от результатов работы друг друга (увеличивается число как поставщиков составляющих, используемых командами в своей работе, так и потребителей результатов этих работ). И в данном случае платформа, выступающая, как говорилось ранее, ценностным мультипликатором, позволяет обеспечить управление соответствующими рисками в цепочке разделения труда. Рассмотрим это на примере, который визуальным образом представлен на Рисунке 5.

Рисунок 5. Продукты, автоматизируемые с помощью платформы

На Рисунке 5 обезличенно показаны два продукта, автоматизируемые при помощи платформы. Для наглядности изложения мы предположим, что продукты, как и в ранее рассматриваемом примере, содержат учетную (отвечающую за синтетический учет), частную учетную (отвечающую за аналитический учет), процессную (отвечающую за исполнение бизнес-процессов, связанных с соответствующим продуктом) и фронтальную (отвечающую за клиентское представление продукта) составляющие. При этом реализация каждой составляющей определяется собственным платформенным приложением, но может использовать подобный (а в отдельных случаях и идентичный) набор платформенных сервисов. Мы видим, что уже на текущем этапе платформа позволяет схожим образом структурировать продукты (выделить общие составляющие) и использовать схожие платформенные сервисы и компоненты для их автоматизации. Таким образом, архитектурная организация продуктов уже упрощается, снижая избыточные трудозатраты, неизбежные при полностью независимой автоматизации.

Внимательный читатель может сразу задать вопрос: «Почему предполагается общая архитектурная структуризация продуктов? Неужели все продукты должны автоматизироваться по одному и тому же шаблону?» Подобный вопрос является закономерным, ведь ранее мы регулярно говорили о том, что основой выбора технологий, топологий, структур автоматизации является исключительно целесообразность. Поэтому данный момент необходимо разобрать более подробно. В «Архитектуре цифрового мира» (в разделе «Продукты в архитектуре») мы рассматривали проблематику автоматизации end-to-end продукта. Под end-to-end продуктом мы понимаем продукт, предоставляющий ценность и являющийся независимым от других продуктов. Мы не будем здесь расписывать все архитектурные свойства end-to-end продукта, желающие могут ознакомиться с ними в предыдущем труде автора. Ограничимся ключевыми тезисами (опять же в архитектурной плоскости):

- Автоматизация end-to-end продукта должна осуществляться максимально гранулированным образом, дабы иметь возможность как вписать таковой продукт в унаследованные архитектурные ландшафты, так и подготовить его к переводу на новые архитектурные поколения.

- Элементы end-to-end продукта должны быть отчуждаемыми (следствие грануляции) и реализовываться различным образом в соответствии с теми или иными архитектурными практиками (и архитектурными поколениями).

- End-to-end продукты должны иметь независимые релизные циклы.

Поскольку платформа представляет собой среду создания и исполнения цифровых решений, а результатом исполнения цифровых решений является предоставление ценности клиентам и партнерам организации (то есть продуктов), то платформенные средства должны поддерживать все составляющие end-to-end продукта (все уровни максимально мелкой грануляции). Разумеется, частные продукты могут не обладать той или иной составляющей, но в общем случае продукты организации должны соответствовать ее специфике и быть достаточно комплексными, чтобы сохранять конкурентоспособность на рынке, а потому для них актуален максимум составляющих. Платформа же обязана поддерживать все составляющие, для продуктов организации. *возможные*

На основании предыдущего тезиса мы можем сделать важный вывод: платформа и продукты организации оказывают взаимовлияние друг на друга. При проектировании платформы архитектор, являясь лидером технологических изменений, обязан провести предварительное технологическое проектирование продуктов (действующих и потенциальных) организации, чтобы на их основе запланировать необходимый набор составляющих, поддерживаемых платформой (как мы помним, каждая составляющая характеризуется своим технологическим набором и вариантами его использования, диктующими, например, топологии развертывания). Аналогичным образом для продуктов должна производиться декомпозиция на составляющие, для каждой из которых определяется платформенный инструментальный, осуществляющий ее поддержку. Таким образом, формируется унифицированный подход к автоматизации продуктов.

Можно ли автоматизировать каждый end-to-end продукт независимо от других с точки зрения не только конкретной разработки, но и архитектуры (в том числе платформенной)? Конечно, можно. Но тогда пострадает эффективность работы организации в целом. В своей книге «Бесконечный матч» великий футбольный тренер В. В. Лобановский писал: «Не кажется ли вам странным, что в наше время унификации элементарных – не говоря уже о сложнейших – производственных и мыслительных процессов почему-то только к футболистам все еще предъявляются требования изобретать на ходу? Им вполне серьезно рекомендуют на ощупь отыскивать позиции, с которых удобно воспользоваться прострельной передачей мяча с фланга или по наитию доходить до того, какой маневр и в какой фазе игры ведет к созданию численного превосходства в контратаке... Я не против того, чтобы так „творили“

на поле наши соперники. Увы, они этим не занимаются. На международной арене мы имеем дело с командами, которые не тратят ни мгновения на выбор коллективных решений, когда возникает знакомая тактическая ситуация». И в случае независимого продуктового проектирования мы оказываемся в аналогичной ситуации – начинаем «творить» продукты и их автоматизацию на ходу, каждый раз заново формируя грануляцию, выбирая средства автоматизации для каждого уровня, теряя эффективность. Подобное является недопустимой роскошью в современном цифровом мире. Думается, что организация, вынужденная работать на высококонкурентном рынке, не против, чтобы ее конкуренты «творили» подобным образом, но итог для рынка в целом будет критичным: компании, «творящие» автоматизацию своих продуктов, будут покидать его, а оставшиеся – останавливаться в своем развитии и деградировать в отсутствие конкуренции.

По итогам приведенных тезисов можно отметить, что платформа определяет набор инструментов создания и исполнения продуктов организации, при этом предлагает схожие наборы указанных инструментов для общих продуктовых составляющих (в нашем примере их четыре). То есть уже на уровне концепции платформенного подхода происходит снижение рисков для отдельных участников рабочего процесса (как конкретных специалистов, так и команд гибких практик) вследствие структуризации (составляющие продуктов) и унифицированной поддержки работы (общие наборы инструментов для продуктовых составляющих). При этом кроме собственно инструментов платформы предлагают и общие направляющие реализации продуктов, учитывающие опыт организации, риски использования тех или иных инструментов, проработанные варианты использования и т. д. И роль архитектора в данном случае крайне важна. Ведь именно на нем лежит задача технологического проектирования будущих продуктов организации, структуризация платформы, определение продуктовых составляющих, технологий их автоматизации (и применяемых топологий), выработка общих направляющих. Разумеется, архитектор не может быть надсмотрщиком, диктующим командам разработки правила их работы – указанные правила формируются в рамках рабочего диалога, отвечающего принципам открытого кода и открытой организации.

Важно отметить, что в разговоре об общих инструментальных наборах, предоставляемых платформами для автоматизации продуктовых составляющих, речь не идет о повторном использовании сервисов и компонентов в вульгарном смысле этого слова. В «Архитектуре цифрового мира» мы уже касались опасности бездумного повторного использования (или переиспользования). К сожалению, нельзя не отметить, что концепция переиспользования (именно в вульгарном смысле) продолжает оставаться популярной, что резко снижает эффективность цифровизации многих рыночных игроков. Итогом следования подобной абсурдной концепции становится позиционирование платформы в качестве овещенного компонента ИТ-ландшафта, фактически информационной системы, что ранее уже отмечалось нами как одна из ключевых ментальных ошибок, допускаемых на пути реализации платформенного подхода. При вульгаризации принципа повторного использования, как правило, происходит следующее:

– Максимум компонентов, содержащих близкий функционал, выделяются в общие и «переиспользуются». Зачастую все такие компоненты становятся «платформенными». Так и появляется, например, «платформенный компонент Клиент», который зачастую либо не реализует требуемые функции платформенных приложений, либо является избыточным. В результате команды развития либо приходят к антипаттерну «Распределенный монолит», либо начинают создавать дублирующий функционал, не руководствуясь никакими общими направляющими и технологическими средами продуктов. Любой из приведенных вариантовкратно увеличивает требуемые для реализации ценности трудозатраты и риски участников рабочего процесса, что недопустимо в современном цифровом мире, так как приводит к застою и деградации организаций, допустивших подобное.

- Продолжением предыдущего пункта, формулирующего чрезмерную перегрузку платформ несвойственными им компонентами, становится формирование платформенных команд, не создающих ценность и являющихся высококонкурентным ресурсом, что уже рассматривалось нами ранее в качестве крайне негативного фактора и барьера на пути интенсивного развития.

- Вульгарное толкование термина «повторное использование» приводит к тому, что возникает огромное количество распределенных компонентов, предоставляющих собственные API (различных версий), а также их потребителей, использующих различные версии указанных API. В подобных условиях формируются зависимости релизных циклов компонентов и платформенных приложений, что крайне негативно влияет на значение такого определяющего показателя эффективности, как время вывода продукта на рынок (time-to-market).

Перечисление негативных факторов, следующих из вульгаризации концепции повторного использования, можно продолжать, но смысл уже понятен: резкое усложнение релизных циклов, увеличение объемов регрессионного тестирования, сложность цифровых платформ, для которых формируются подобные факторы, сводит на нет концепцию платформы как ценностного мультипликатора. Поэтому при рассмотрении вопросов повторного использования, позволяющего повышать производительность цепочки разделения труда, необходимо придерживаться следующих ключевых положений:

- Платформа представляет собой среду создания и исполнения приложений, автоматизирующих продукты организации и предоставляющих ценность клиентам и партнерам. Следовательно, платформа не содержит прикладных компонентов (отвечающих за непосредственное создание ценности), она предоставляет опосредованную ценность.

- Ключевым свойством предоставления API платформенными компонентами является их встраиваемость: API предоставляется в формате встраиваемых библиотек, используемых платформенными приложениями. Указанный подход позволяет избегать необходимости пересборки приложений при обновлениях платформенных компонентов либо максимально снизить данную необходимость.

- Подсистемы, сервисы, компоненты и другие структурные составляющие платформы не предоставляются в виде единой сущности всем потребителям – они могут представлять собой наборы сущностей, при этом настройки каждой сущности (в том числе топология) могут адаптироваться под потребности потребителей (платформенных приложений).

- Кроме собственно технологических компонентов платформа предоставляет общие направляющие для потребителей, включающие (но не ограничивающиеся ими) варианты использования компонентов, рекомендации и другие материалы, позволяющие обеспечить максимально эффективное предоставление платформы платформенным приложениям.

Следуя вышеперечисленным положениям, мы можем разделить технологический и прикладной функционал, обеспечить независимость последнего, разделить релизные циклы компонентов, снизить потенциальные объемы регрессионного тестирования, а также адаптировать платформу под требования потребителей. Тем самым повышается производительность труда технологической цепочки создания ценности и снижаются риски ее участников. Важно отметить, что платформенный подход и принципы открытого кода дополняют друг друга, обеспечивая тот самый ценностный мультипликатор, о котором говорилось по ходу настоящего изложения. В результате организация, применяющая платформенный подход, достигает более высокого уровня конкурентоспособности по сравнению с теми своими конкурентами, кто либо игнорирует соответствующий подход, либо попадает в ментальные ловушки в ходе его применения.

Таким образом, следование платформенному подходу, реализация, внедрение и использование платформ позволяют не просто повышать эффективность цифровизации продуктов, предоставляемых организациями своим клиентам и партнерам. Платформы поддерживают

распределенную структуру команд, обеспечивают вовлечение все большего числа участников в общие технологические цепочки, стирают между ними барьеры. Мы приходим к выводу: чем более полно внедряется современный платформенный подход, чем более современная (и устремленная в будущее) платформа используется в организации, тем в большей степени цифровая архитектура организации соответствует такой ключевой тенденции развития, как распределенность.

При этом для участников технологической цепочки платформа является открытой. Подробно открытость платформ мы рассмотрим в соответствующей главе. В контексте же распределенности отметим, что распределенные команды могут и должны становиться участниками развития платформы, дополняя ее необходимым им функционалом, публикуя его в доступном коллегам виде. Подчиняясь принципам открытого кода, платформа развивается ее же потребителями, в результате чего максимально полно отвечает их нуждам. Синергия публикуемых обновлений позволяет всем распределенным командам повышать свою эффективность, как следствие, растет производительность всей цепочки разделения труда.

Рассмотрим теперь поддержку платформами смысловой составляющей технологической распределенности. Последняя предполагает возможность реализации при помощи платформы таких платформенных приложений, которые могут исполняться в распределенном режиме, предоставлять сервисы клиентам и партнерам в любой точке земного шара. При этом мы должны учитывать, что посредством платформенных приложений переводятся в цифровой вид продукты организации, несущие непосредственную ценность клиентам/партнерам. Перечень продуктов не является законченным – организация постоянно создает новые продукты или вносит изменения в уже существующие, следя за рыночной конъюнктурой. Характеристики продуктов являются совершенно различными. Продукты, отвечающие структурной декомпозиции, показанной на Рисунке 5, могут кардинально отличаться друг от друга функциональным наполнением в рамках общих составляющих, автоматизация которых осуществляется с применением платформенного подхода. Примерами таких отличий могут стать сложность и специфика исполнения бизнес-процессов (например, количество шаблонов процессов и сложность схем продуктовых процессов, подлежащих автоматизации), доступность продуктов в различных множествах дистанционных каналов (подобная характеристика влияет на нефункциональные требования, но принимается на уровне владельца продукта, а потому исходно может считаться функциональной), характеристики аналитического учета и т. д. Таким образом, и требования продуктов к платформе также существенно отличаются.

Приведем следующий пример. Если в продуктах отличается набор дистанционных каналов предоставления, то при проектировании как фронтальной, так и иных составляющих необходимо учитывать детализацию соответствующего набора. Например, может оказаться, что один продукт предполагает предоставление ценности посредством каналов, не предполагающих аутентификации и авторизации клиентов, в то время как другой продукт доступен только авторизованным клиентам. Результатом данного отличия становится необходимость поддерживать принципиально разный уровень производительности при исполнении продуктовых решений (платформенных приложений). Речь идет не только о фронтальной составляющей – все составляющие продукта должны учитывать необходимость собственного корректного исполнения при соответствующих требованиях к производительности. Фронтальная составляющая должна обеспечивать время отклика, гарантирующее удовлетворенность пользователей, при этом наполнение фронтальной составляющей данными невозможно без корректной отработки процессной и учетной составляющих, также сталкивающихся с повышенными требованиями к производительности. При этом продукт, потребителями которого являются только авторизованные пользователи, не предъявляет аналогичных требований. Платформа же, обеспечивающая автоматизацию продуктовых решений, является общей.

Каким же образом достичь эффективности и в то же время учесть все требования, предъявляемые продуктами?

Как мы уже отмечали, платформа предоставляет общую среду создания и исполнения приложений. Таким образом, мы исходим из того, что и набор технологических инструментов для автоматизации продуктовых составляющих является общим (с точки зрения доступности командам разработки). Но здесь вновь вступает в дело архитектор, являющийся лидером технологических изменений. Доступные платформенным приложениям технологические решения, предоставляемые платформой, должны проектироваться и создаваться таким образом, чтобы:

- Охватывать множество продуктов и поддерживать распределенное исполнение предоставляющих их платформенных приложений.
- Допускать множество вариантов использования, чтобы иметь возможность избегать предоставления платформенным приложениям избыточно сложных или тяжеловесных платформенных сервисов.
- Быть открытыми, чтобы иметь возможность развиваться при расширении набора продуктов и появлении новых/дополнительных требований.

Для рассматриваемого примера можно выделить следующие варианты совместного исполнения приложений, автоматизирующих предоставляемые продукты (с учетом вышеприведенных характеристик технологических решений):

- Для обеспечения высокой производительности фронтальной составляющей может использоваться решение по обработке и хранению данных в оперативной памяти (IMDG/IMDB). При этом данное решение должно быть технологически общим, дабы при поддержке и развитии продуктов не приходилось учитывать «зоопарк технологий». Примером неудачного проектирования может служить вариант, при котором в роли IMDG/IMDB-решения для высокопроизводительного продукта выступает Apache Ignite, а для продукта, требования к производительности которого ниже, Redis. В дальнейшем при появлении еще одного продукта с высокими требованиями по производительности команда развития внедрит Hazelcast, после чего драматически возрастет стоимость обеспечения непрерывности, поддержки и развития подобного «унифицированного» решения. Но выбором общей технологии работы (например, Apache Ignite ввиду наличия требований по высокой нагрузочной способности) дело не ограничивается. На первый план выступает вопрос использования топологий. И платформа должна предоставлять возможность развернуть топологию, поддерживающую исполнение продукта в соответствии с требованиями. Например, соответствующее число узлов, правила репликации, даже сложность и развитость встраиваемого платформенного API могут различаться в зависимости от требований, предъявляемых к продукту. Таким образом, и варианты использования платформенного сервиса также могут различаться. В конце концов, и обеспечение групп продуктов может осуществляться множеством платформенных сервисов, каждый из которых отвечает своему подмножеству требований.

- Подобное технологическое решение по обеспечению высокой производительности может (и должно) применяться при автоматизации процессной составляющей рассматриваемых продуктов. Как мы уже отмечали ранее, стандартные средства автоматизации бизнес-процессов, как правило, не отличаются высокой производительностью. В них закладываются возможности гибкого моделирования процессов, их быстрой сборки и развертывания, возможности мониторинга, но производительность отказывается не самым передовым показателем данного класса инструментов. Одновременно (и мы также обращали внимание читателя на данный факт) в современной архитектуре бизнес-процессы не могут характеризоваться низкой производительностью, в противном случае организация потеряет конкурентоспособность на рынке. Поэтому (так как базовый инструмент управления процессами не может сам по себе обеспечить адекватную требованиям дистанционного обслуживания произво-

дительность) на уровне реализации процессной составляющей продуктовой автоматизации могут применяться упомянутые выше IMDG/IMDB-инструменты. С их помощью, например, может осуществляться высокоэффективная работа с контекстом экземпляра процесса, его бизнес-данными, обеспечиваться надежность и т. д. Конкретные топологии (и сущности) развертывания могут различаться как от продукта к продукту, так и в зависимости от той продуктовой составляющей, к которой они относятся. Но технологическая унификация, а также специфика платформенных сервисов должны быть общими на уровне платформенных приложений и предоставляемых ими продуктов организации клиентам и партнерам последней. Фактически множество платформенных сервисов, обеспечивающее исполнение фронтальной составляющей продукта, пересекается с множеством платформенных сервисов, обеспечивающих исполнение процессной составляющей продукта, по принципу кругов Эйлера. Аналогичным образом пересекаются множества платформенных сервисов, обеспечивающих исполнение различных продуктов (даже при аналогичной структуре последних). А требования к платформенным сервисам, режимам их функционирования, топологиям развертывания технологической составляющей предъявляют продукты. Отметим, что указанные требования не являются статическими. Например, при добавлении новых продуктовых бизнес-процессов или расширении использования существующих требования к инструментальной поддержке высокой производительности процессов могут кардинально измениться – как вариант, возрасти. Подобное происходит при добавлении новых каналов лидогенерации, привлекающих большие объемы клиентов. Соответственно, используемые в платформах технологии, а также предоставляемые платформами сервисы должны быть адаптируемыми к подобным изменениям требований, что, в свою очередь, напрямую следует из свойства отсутствия замкнутости платформ.

- Каждый продукт, реализуемый платформенными приложениями, может функционировать в собственном режиме, а требования к нему могут изменяться независимо от остальных продуктов, предоставляемых организацией клиентам и партнерам. А потому платформа должна обеспечивать достаточную гибкость для независимой поддержки развития каждого продукта. То есть платформа должна быть распределенной, каждый ее модуль, каждый платформенный сервис должен быть распределенным, в противном случае он окажется узким местом развития независимых продуктов.

Мы не будем подробно рассматривать варианты распределенного исполнения остальных составляющих продуктовой автоматизации, представленных на Рисунке 5. Их сопоставление (как между собой, так и в контексте нескольких автоматизируемых продуктов) будет аналогично приведенному выше для фронтальной и процессной составляющих. Здесь важно отметить другое. При технологической унификации решений, предлагаемых платформами для продуктовой автоматизации, необходимо учитывать раздельное и распределенное исполнение как продуктов, так и отдельных уровней автоматизации, то есть платформы должны предлагать (с точки зрения визуализации Рисунка 5) как горизонтальную, так и вертикальную распределенность. И добиться такого результата позволит эластичное масштабирование.

Поддержка той или иной технологией эластичного масштабирования подразумевает, что соответствующая технология обеспечивает:

- Сегментацию собственных сущностей. В приведенном выше примере мы можем, в частности, сформировать отдельные сегменты IMDG/IMDB-решения для поддержки различных составляющих автоматизации продуктов, разделить по сегментам те или иные продуктовые процессы в зависимости от требований.

- Каждый сегмент может масштабироваться независимо. Например, может быть выделен сегмент IMDG/IMDB под каждый тип продуктового процесса, при этом сегменты, обеспечивающие производительность высоконагруженных процессов, масштабируются независимо от других. Таким образом обеспечивается использование ресурсов в соответствии с реальными потребностями приложений, причем гранулированное, исключая избыточные затраты.

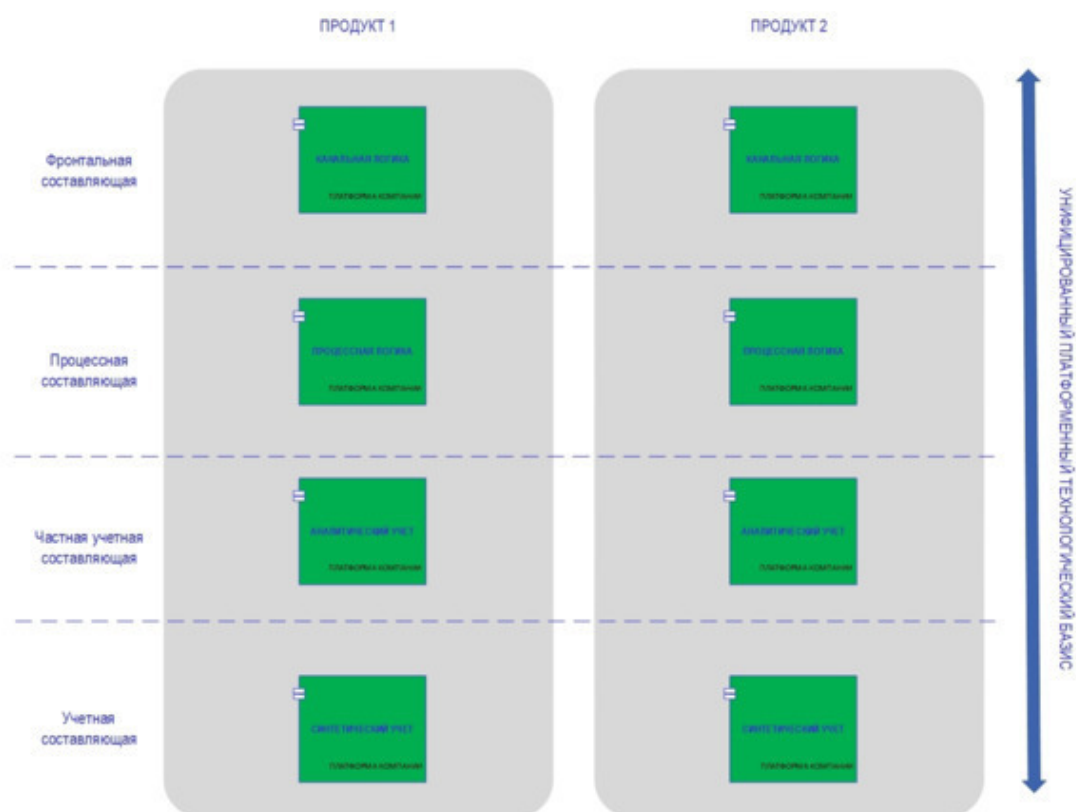
- Технологическое решение, применяемое для автоматизации соответствующего сегмента, может использовать динамически расширяемое или сокращаемое количество вычислительных узлов для своего функционирования.

- Технологические решения сохраняют корректность функционирования при увеличении или уменьшении количества используемых ими вычислительных узлов (разумеется, уменьшение в данном случае не подразумевает сокращения количества узлов ниже требуемого для обеспечения минимальной нагрузочной способности).

Отметим, что приводимые нами в качестве примеров в течение настоящей книги технологии Apache Ignite, Apache Cassandra, Apache Kafka и многие другие ведущие решения с открытым исходным кодом обеспечивают вышеприведенные характеристики.

Но мало применять технологии, поддерживающие эластичное масштабирование. Платформенные сервисы и платформенные приложения, использующие подобные технологии, также должны поддерживать эластичное масштабирование. И здесь задача архитектора совместно с командами развития – спроектировать и реализовать сервисы и приложения таким образом, чтобы они не стали узким местом платформы, чтобы сама платформа не стала барьером между развитыми технологиями с открытым исходным кодом и командами развития, которые в таком случае окажутся вынуждены «бороться» с предоставляемыми им сервисами, чтобы обеспечить распределенность.

Считаем нелишним еще раз подчеркнуть, что мы не рассматриваем конкретные платформенные решения. Реализации конкретных платформ как требуют детальной проработки технологий и их топологий, так и содержат гораздо большее количество поддерживаемых составляющих автоматизации продуктов организации, наполняя их существенно более богатым набором технологических решений. Наша же задача в рамках данного изложения заключается в том, чтобы показать читателю связь цифровых платформ и такой ключевой тенденции развития архитектуры, как распределенность, проиллюстрировать способы поддержки соответствующей тенденции путем платформенного подхода.



Цифровые платформы и бизнес-процессы

Перечисляя ключевые тенденции развития архитектуры, мы говорили о том, что современные организации, ставящие себе целью перманентное интенсивное развитие, стараются уйти от процессной методологии работы к продуктовой. При этом смещение фокуса внимания на продукты не отменяет важности бизнес-процессов, более того, мы относим последние к трендам, определяющим развитие архитектуры. Внимательный читатель и тут не оставит нас без каверзного вопроса: «Нет ли противоречия в указанной логике?» Мы отвечаем, что противоречие отсутствует, а ниже объясним, почему так происходит.

Для обеспечения полноты рассмотрения мы возьмем за основу организацию с относительно длительной историей, прошедшую уже не одно поколение автоматизации (применявшую различные архитектурные парадигмы), поскольку вариант рассмотрения стартапа чреват применением к нему фразы Исаака Ньютона: «Если я видел дальше других, то потому, что стоял на плечах гигантов». Случай стартапа может характеризоваться учетом опыта предшественников, при котором удастся избежать тех или иных ошибок, кроме того, влияние унаследованного ИТ-ландшафта и устаревшего mindset может оказаться пренебрежимо малым.

Как правило, такая организация уже проводила описание выполняемых бизнес-процессов (в ряде случаев они могут именоваться производственными ввиду специфики деятельности), зачастую с приглашением внешних консультантов проводились попытки провести и некую оптимизацию последних. Результаты подобных действий могут различаться:

- Возможно, дело ограничилось просто описанием бизнес-процессов с созданием (и то не всегда) визуализирующих диаграмм в общепринятых форматах – например, BPMN.
- Исполнение бизнес-процессов в автоматизированном режиме может осуществляться в каждой информационной системе отдельно (специфичные процессы выполняются в специфичных информационных системах).
- В организации мог быть внедрен BPM-инструмент для централизованного управления бизнес-процессами (с различным охватом внедрения соответствующего инструментария).

Отметим, что вышеперечисленные пункты не являются фиксированными шагами на пути компании к успеху – возможны самые разные варианты реализации каждого из них, включая различные подводные камни:

- Бизнес-процессы могут быть описаны халатно или в формате «вроде как сейчас так происходит», то есть без необходимой верификации описания.
- Бизнес-процессы, реализованные в информационных системах либо посредством выделенного BPM-инструмента, могут содержать большое количество элементов, автоматизированных с использованием «ручного» программирования, и требовать масштабного привлечения разработчиков при изменениях.
- Элементы бизнес-процессов могут быть реализованы (или управляться) в различных информационных системах, что приводит к избыточным трудозатратам при внесении изменений в автоматизируемые процессы.
- И т. д.

При переходе к современным архитектурным и технологическим принципам необходимо учитывать первичность продуктового подхода, когда деятельность организации определяется тем, какую ценность она приносит клиентам и/или партнерам. При этом ценность в современном мире реализуется автоматизируемыми продуктами – цифровыми продуктами. Бизнес-процессы сегментируются в соответствии с продуктовой логикой. Но далеко не все процессы могут быть ограничены лишь одним продуктом:

- Безусловно, бизнес-процессы, ассоциированные с тем или иным продуктом, автоматизируются в соответствии с ним и размещаются (с точки зрения архитектуры) в соответствующей функциональной области.

- Сложные кросс-продукты (или связанные продукты) могут требовать и сложных бизнес-процессов, которые будут включать в себя инструкции по исполнению продуктовой логики, реализуемой в составе множества продуктов.

- Общесистемные сквозные процессы могут выполняться на уровне организации и вовлекать значимое подмножество продуктов, в пределе – все продукты компании. Данный вариант расширяет и усложняет вариант процессов, автоматизирующих работу с кросс-продуктами.

Наглядным представлением такого разграничения процессов является функционально-информационная архитектура, пример которой представлен на Рисунке 6.

Рисунок 6. Пример представления бизнес-процессов на функционально-информационной архитектуре

Таким образом, переход к современным архитектурным практикам предполагает и наличие важной методологической составляющей: процессы организации должны быть описаны, сегментированы по продуктам (с классификацией в части ограниченности продуктовой областью), а также должны своевременно актуализироваться. Непосредственно при автоматизации необходимо использовать современные средства управления процессами (BPM-движки), которые позволяют наглядным образом создавать шаблоны процессов, моделировать исполнение процессов, собственно осуществлять исполнение, поиск узких мест, мониторинг (системный и прикладной), выставлять и рассчитывать КПЭ процессов и т. д. При этом BPM-движок, используемый при автоматизации, должен отвечать современным тенденциям развития архитектуры – например, поддерживать распределенность. Также отметим, что в предыдущем труде автора («Архитектура цифрового мира») указывались три исключительно важных аспекта автоматизации бизнес-процессов в современном цифровом мире: технический рефакторинг, работа с контекстом, корректное использование шаблонов оркестровки и хореографии. Для полноты восприятия вкратце напомним их читателю.

Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, [купив полную легальную версию](#) на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.