1 Вступление

К современному корпоративному ПО предъявляется много новых требований. Уже недостаточно лишь разработать некий программный продукт и развернуть его на сервере приложений. Впрочем, этого никогда не было достаточно.

Новые требования к корпоративным системам

Сегодня мир меняется быстрее, чем когда-либо прежде. Поэтому способность *быстро изменяться* — одно из важнейших требований к современным ИТ-компаниям. Последние способны оперативно приспособиться к реальному миру и потребностям клиентов. Ожидаемое время выхода новых функций на рынок сократилось от нескольких лет или месяцев до недель и дней. Чтобы уложиться в такие сроки, компаниям необходимо не только вводить новые технологии или *выделять больше денег* на реализацию бизнес-логики, но и переосмысливать и реорганизовывать базовые принципы работы их ИТ.

Что в этом контексте означает *быстро меняться*? В чем это выражается? Какие методы и технологии этому способствуют?

Быстро меняться означает быстро адаптироваться к потребностям рынка и клиентов. Если требуется новая функция и она кажется многообещающей, сколько понадобится времени, чтобы она прошла путь от первоначальной идеи до готового инструмента в руках пользователя? Если нужна новая инфраструктура, сколько времени пройдет от принятия решения до получения работающего оборудования? И не будем забывать о том, что, кроме разработки программного обеспечения в требуемые сроки, нужен еще автоматизированный контроль качества, который гарантирует, что все будет работать как ожидалось и не нарушится существующая функциональность.

При разработке программного обеспечения основная часть этих вопросов касается непрерывного развертывания и автоматизации. Новое программное обеспечение должно быть разработано, протестировано и поставлено автоматическим, быстрым, надежным и воспроизводимым способом. Надежный автоматизирован-

ный процесс обеспечивает не только более быстрые изменения, но и в конечном счете более высокое качество. Автоматический контроль качества, например выполнение программных тестов, — часть технологического процесса. В современной разработке ПО непрерывное развертывание, автоматизация и надлежащее тестирование являются одними из наиболее важных принципов.

Самым узким местом в большинстве компаний всегда была инфраструктура. Небольшие компании часто пытаются построить новую инфраструктуру, имея ограниченный бюджет. Более крупным компаниям, как правило, не удается разработать быстрые и эффективные процессы. Проблема многих больших корпораций заключается не в бюджете, а в технологиях. Часто приходится ждать появления новой инфраструктуры в течение нескольких дней или даже недель из-за согласований и чрезмерно сложных процессов, которые технически могли бы быть выполнены за считаные минуты.

Именно поэтому так важны инфраструктура приложения и способ его разработки. В главе 5 мы рассмотрим современные облачные среды. Вы увидите, что на самом деле не имеет особого значения, используются облачные услуги или нет. Быстрые и продуктивные процессы можно реализовать и на локальном оборудовании. Гораздо важнее, чтобы они правильно работали и применялись соответствующие технологии.

Современная инфраструктура должна создаваться быстро и автоматически, быть воспроизводимой и надежной. Она должна без особых усилий адаптироваться к меняющимся требованиям. Для того чтобы соответствовать этим требованиям, инфраструктура должна быть описана в виде кода — либо как процедурный сценарий, либо как набор декларативных описаний. Позже мы увидим, как инфраструктура в виде кода влияет на процессы разработки программного обеспечения и какие технологии ее поддерживают.

Данные требования относятся и к рабочим группам. Команде разработчиков уже недостаточно *просто создать* программный продукт и передать его операционным командам, которые его запустят, а затем будут решать возникающие в процессе эксплуатации проблемы. Такая практика чревата натянутыми отношениями и перекладыванием на других ответственности за критические ошибки. Но ведь общая цель должна заключаться в том, чтобы поставлять программное обеспечение, полностью удовлетворяющее своему функциональному назначению. Описывая необходимую инфраструктуру и конфигурацию в виде кода, группы разработчиков и операционистов должны составлять единую команду, отвечающую за работу программного обеспечения в целом. Это называется *DevOps* — культура разработки и эксплуатации, нацеленная на ответственность всей команды разработчиков ПО за функционирование продукта. Все участники отвечают за то, что клиенты будут использовать надлежащее программное обеспечение. Это скорее организационная, чем техническая задача.

Технически для решения этих задач применяется непрерывное развертывание, а также методика, которую называют *12-факторным* подходом и ориентацией на *выполнение в облаке*. Облачный и 12-факторный подходы описывают, как должны разрабатываться современные корпоративные приложения. Они определяют

требования не только к разработке, но и к эксплуатации приложений. В этой книге мы изучим указанные подходы, современные облачные среды и то, как их поддерживает Java EE.

Современный способ построения корпоративных систем

Посмотрим, как разрабатываются корпоративные программные комплексы.

Стремясь удовлетворить потребности реальных клиентов, мы должны спросить себя: каково назначение приложения, которое мы намерены разработать? Прежде чем углубляться в технические детали, необходимо прояснить мотивы и цели создания корпоративных программных систем, иначе получится разработка ради разработки. К сожалению, так происходит слишком часто. Сосредоточив внимание на бизнес-логике и принципах *проблемно-ориентированного проектирования*, как это прекрасно описано в книге Эрика Эванса (Eric Evans), мы гарантируем, что создаваемое программное обеспечение будет отвечать корпоративным требованиям.

Только после того, как назначение и задачи приложения станут понятны всем заинтересованным сторонам, можно перейти к техническим деталям. Группы разработчиков должны выбирать те методики и технологии, которые позволят не только эффективно реализовывать бизнес-сценарии использования продукта, но и уменьшить объем работы и издержки. У разработчиков должна быть возможность сосредоточиться на бизнесе, а не на операционной среде или технологии. Хорошая операционная среда тихо и незаметно помогает реализовывать бизнес-логику, не требуя внимания разработчика.

Выбранная технология должна обеспечивать и максимально эффективные процессы разработки. Это означает не только автоматизацию и быстрое выполнение всех этапов, но и поддержку современной инфраструктуры, такой как контейнеры Linux. В главах 4 и 5 мы подробнее рассмотрим технические особенности современных сред и поговорим о том, как их поддерживает Java EE.

Значение Java EE для современных систем

Перейдем к платформе Java EE, поскольку она имеет прямое отношение к корпоративным системам и это основная тема книги.

Java EE и J2EE используются очень широко, особенно в крупных компаниях. Одно из их преимуществ заключается в том, что платформа состоит из стандартов, гарантирующих обратную совместимость с ранними версиями. Даже старые приложения J2EE в дальнейшем будут гарантированно функционировать. Это всегда было весомым аргументом для компаний, строящих долгосрочные планы. Приложения, разработанные на базе Java EE API, способны работать на всех серверах приложений Java EE. Независимые от производителя приложения

позволяют компаниям создавать перспективное ПО, не ограниченное рамками какого-либо конкретного продукта. Это оказалось разумным решением и в итоге привело к появлению индустриальной культуры, в которой стандарты улучшают ситуацию в целом.

По сравнению с миром J2EE в Java EE многое изменилось. Здесь совершенно иная модель программирования — более компактная и продуктивная. Радикальные изменения произошли вместе со сменой названия с J2EE на Java EE 5 и особенно на EE 6. В главе 3 мы рассмотрим современный способ разработки приложений на Java. Узнаем, какие архитектурные подходы и программные модели используются, и увидим, что благодаря платформе разработка становится гораздо эффективнее, чем прежде. Тогда вы поймете, почему Java EE является современным решением для разработки корпоративных приложений.

Сейчас донести эту информацию до отрасли — в сущности, скорее маркетинговая и политическая, чем техническая задача. Множество разработчиков и архитекторов все еще считают Java EE громоздким, тяжеловесным корпоративным решением эпохи J2EE, требующим много времени, усилий и XML. За *Enterprise JavaBeans (EJB)* и серверами приложений сохраняется очень плохая репутация. Именно этим вызвано предубеждение многих инженеров против этой технологии. По сравнению с другими корпоративными решениями у Java EE было довольно мало маркетинговых решений для разработчиков.

В главе 4 мы увидим, почему современная Java EE является одним из самых простых корпоративных решений. Вы узнаете, за счет чего она столь удобна и почему сейчас более актуальна, чем когда-либо, особенно в современных облачных и контейнерных средах. Влияние ИТ-индустрии на конкретную технологию очень важно для достижения ею успеха.

Компании выбирают Java EE в основном из-за надежности и обратной совместимости. Я предпочитаю Java EE из-за ее производительности и простоты использования. Подробнее об этом мы поговорим в главах 4 и 5. В этой книге я хотел бы доказать читателям, что Java EE — решение, хорошо удовлетворяющее потребности современных предприятий. Я также представлю вам технологии и стандарты — не углубляясь в подробности. Скорее расскажу, как они связаны между собой. Считаю, что, сосредоточившись на этой связи, мы лучше поймем, как эффективно строить промышленные приложения.

Обновление и перспектива развития Java EE 8

Сейчас мы кратко рассмотрим, что появилось в Java EE версии 8. Цель создания этой версии — сделать продукт еще удобнее для разработчика, оптимизировать работу API и привести Java EE в соответствие с новыми требованиями, предъявляемыми облачными средами. Платформа располагает двумя полностью новыми JSR — *JSON-B* (Java API для JSON Binding) и *Security*. Кроме того, усовершенствованы существующие стандарты. В частности, внедрение JSON-B упрощает независимую от поставщика интеграцию интерфейсов JSON HTTP API.

Назначение Java EE — улучшить разработку корпоративных приложений в соответствии с современными средами и условиями. Оказывается, современные среды не просто совместимы с Java EE — они поддерживают подходы, которые уже много лет являются частью платформы, например отделение API от реализации или мониторинг сервера приложений.

В долгосрочной перспективе предполагается оптимизировать поддержку современных методов мониторинга, проверки работоспособности и устойчивости. Сегодня для реализации каждой из этих возможностей приходится слегка дополнять код, в чем мы убедимся в следующих главах. Предполагается, что в дальнейшем такая интеграция упростится. Java EE организована так, чтобы разработчик мог сосредоточиться на своих прямых задачах — заняться реализацией бизнес-логики.

Java Community Process

Уникальность платформы Java EE — в процедуре ее определения. Стандарты Java EE разрабатываются как часть *Java Community Process (JCP)*. JCP — яркий пример отрасли, где активно поощряется участие всех, кто интересуется данной технологией. Платформа включает в себя стандарты в виде запросов на спецификацию *Java Specification Requests (JSR)*. Запросы JSR применимы не только в Java и Java EE, но и в отношении основанных на них технологий, таких как среда разработки Spring. В результате практический мировой опыт разработки этих технологий помогает формировать новые JSR.

При разработке приложений, особенно при возникновении потенциальных проблем, письменные спецификации, сформированные на основе JSR, оказываются чрезвычайно полезными. Поставщики, поддерживающие корпоративную платформу, обязаны обеспечить реализацию так, как указано в стандартах. Таким образом, в спецификационных документах содержится информация и для поставщиков, и для разработчиков о том, как будет работать технология. Если какие-либо функции не работают, поставщики должны исправлять эти проблемы в своих реализациях. Это также означает, что разработчикам теоретически остается только изучать и знать сами эти технологии, а не детали, зависящие от поставщика.

Любой разработчик может участвовать в Java Community Process, чтобы помочь сформировать будущие версии Java и Java EE. *Экспертные группы (expert groups)*, работающие над конкретными стандартами, приветствуют конструктивную обратную связь от любого, кто интересуется этой темой, даже если он не состоит в JCP. Более того, можно ознакомиться со следующими версиями стандартов еще до их выпуска. Все это привлекает разработчиков архитектуры и компаний: они не только понимают направление развития, но и имеют возможность повлиять на него и внести свой вклад в процесс.

Именно по этим соображениям я сам специализируюсь на Java EE. У меня есть опыт разработки корпоративных систем в среде Spring. Кроме того, что обе технологии очень похожи с точки зрения модели программирования, я особенно

ценю эффективность стандарта CDI, а также возможность легко использовать все технологии платформы. Я изучал конкретные JSR, которые являются частью корпоративной платформы, участвовал в разработке инструментов, которые были стандартизированы в то время. Кроме того, я являюсь членом двух экспертных групп — по JAX-RS 2.1 и JSON-P 1.1. Участвуя в определении этих стандартов, я значительно пополнил свои знания в области корпоративных систем. Конечно, придется глубоко изучать конкретную технологию, если вы захотите помогать ее стандартизировать. Но, согласитесь, приятно осознавать, что принимаешь участие в разработке стандарта ИТ-индустрии.

Что вы найдете в этой книге

Я решил написать книгу о том, что узнал сам, работая со всевозможными корпоративными системами Java. Моя цель — описать вам современную технологию Java EE. Она предназначена прежде всего для разработки корпоративных приложений и современных моделей программирования. Я хочу, чтобы вы хорошо понимали, как Java EE используется в эпоху EE 8 и где проявляются лучшие свойства платформы. На базе современных сред разработки появились новые шаблоны и парадигмы проектирования. Если вы знакомы с миром J2EE, то, надеюсь, оцените и преимущества современной Java EE. Я постарался показать, какие из старых парадигм, ограничений и соображений, из-за которых многие разработчики не любили J2EE, больше неактуальны. Кроме этого, попытаюсь заразить вас своим энтузиазмом и объяснить, почему я убежден в том, что Java Enterprise хорошо подходит для построения корпоративных приложений.

Вам не обязательно разбираться в шаблонах и рекомендованных методах разработки J2EE. В частности, новая модель программирования настолько отличается от старой, что я убежден: имеет смысл продемонстрировать современный подход с чистого листа.

Если вам приходилось разрабатывать J2EE-приложения, это отлично. Вы увидите, как трудно было решать многие задачи с помощью старых шаблонов проектирования J2EE, особенно в современном мире, когда можно сосредоточиться на бизнес-требованиях, а не на технологии, используемой для их реализации. Это тем более актуально, если следовать принципам проблемно-ориентированного проектирования. Вы заметите, скольких громоздких и утомительных методов, в прошлом свойственных системам J2EE, можно избежать в современной Java EE. Простота и мощь платформы Java EE могут вдохновить вас на переосмысление подходов, которые вы применяли до сих пор.

Эта книга предназначена для инженеров-программистов, разработчиков и архитекторов, которые создают корпоративные приложения. В издании я в основном буду использовать термины «разработчики» или «инженеры». Тем не менее я убежден, что архитекторам также время от времени стоит заглядывать в исходный код. Это нужно не только для поддержки других разработчиков в команде — это важно для них самих, ведь так можно получить больше практического опыта.

Точно так же любой разработчик должен иметь по крайней мере базовое представление о системной архитектуре и понимать причины, по которым принимаются определенные решения. Чем лучше специалисты станут понимать друг друга, тем проще им будет общаться и тем успешнее пройдет работа над проектом.

Разработка современных корпоративных приложений — это гораздо больше, чем просто разработка. Это и новые требования, предъявляемые к корпоративным приложениям, и инженеры, которые вникают в процессы, и облачные среды, контейнеры и инструменты управления ими. Мы подробно изучим проблемы непрерывной поставки ПО и автоматизированного тестирования, поговорим о том, почему они имеют такое большое значение и как интегрируются с Java EE. Мы также рассмотрим контейнерные технологии, такие как Docker, и среды управления, например Kubernetes. В современном мире корпоративных систем важно показать, как технология наподобие Java EE поддерживает эти возможности.

Архитектура Microservice — обширная тема, еще одна из сегодняшних проблем. Мы разберем, что такое микросервисы и как они могут быть реализованы с помощью Java EE. Кроме того, обязательно поговорим о безопасности, протоколировании, производительности и мониторинге. Я покажу, что именно должен знать и принимать во внимание архитектор, работая над современным ПО. В частности, ему может потребоваться определить набор нужных технологий, особенно когда речь идет о современных решениях, например, в области 12-факторных или облачных приложений. Однако гораздо важнее понимать, какие концепции лежат в основе этих технологий и каково их назначение. Используемая технология меняется с каждым днем, принципы и концепции информатики живут гораздо дольше.

Мой подход ко всем темам, затронутым в этой книге, заключается в том, чтобы сначала объяснить причины, лежащие в основе принятых решений, а затем показать, как эти решения применяются и внедряются в Java EE. Я считаю, что простое изучение определенной технологии, безусловно, может помочь разработчикам в их повседневной работе, но они не усвоят материал полностью до тех пор, пока не поймут до конца, в чем назначение того или иного подхода. Вот почему я начну с описания корпоративных приложений в целом.

В Java EE много функций, особенно если сравнивать с прошлыми версиями. Но эта книга не является полным справочником по Java EE. При ее написании я стремился рассказать вам о своем практическом опыте, а также дать *практические рекомендации* с подробным разбором решений типичных сценариев. Итак, приготовьтесь насладиться путешествием в мир современного корпоративного программного обеспечения.