Памяти Дениса Уильяма Фаулера (1922–2000)
посвящается
— Мартин

Предисловие

Весной 1999 года меня пригласили в Чикаго для консультаций по одному из проектов, осуществляемых силами ThoughtWorks — небольшой, но быстро развивавшейся компании, которая занималась разработкой программного обеспечения. Проект был достаточно амбициозен: речь шла о создании корпоративного лизингового приложения уровня сервера, которое должно было охватывать все аспекты проблем имущественного найма, возникающих после заключения договора: рассылку счетов, изменение условий аренды, предъявление санкций нанимателю, не внесшему плату в установленный срок, досрочное возвращение имущества и т.п. Все это могло бы звучать не так уж плохо, если не задумываться над тем, сколь разнообразны и сложны формы соглашений аренды. "Логика" бизнеса редко бывает последовательна и стройна, так как создается деловыми людьми для ситуаций, в которых какой-нибудь мелкий случайный фактор способен обусловить огромные различия в качестве сделки — от полного краха до неоспоримой победы.

Это как раз те вещи, которые интересовали меня прежде и не перестают волновать поныне: как прийти к системе объектов, способной упростить восприятие конкретной сложной проблемы. Я действительно убежден, что основное преимущество объектной парадигмы как раз и состоит в облегчении понимания запутанной логики. Разработка хорошей **модели предметной области (Domain Model, 140)**[1] для изощренной проблемы реального бизнеса весьма трудна, но ее решение приносит громадное удовлетворение.

Модель предметной области — это, однако, еще не все. Нашу модель следовало отобразить в базе данных. Как и во многих других случаях, мы использовали реляционную СУБД. Необходимо было снабдить решение пользовательским интерфейсом, обеспечить поддержку удаленных приложений и интеграцию со сторонними пакетами — и все это с привлечением новой технологии под названием J2EE, с которой никто не умел обращаться.

Несмотря на все трудности, мы обладали большим преимуществом — обширным опытом. Я длительное время проделывал аналогичные вещи с помощью C++, Smalltalk и CORBA. Многие члены команды ThoughtWorks в свое время серьезно поднаторели в Forte. В наших головах уже вертелись основные архитектурные идеи, оставалось только выплеснуть их на холст J2EE. (Теперь, по прошествии трех лет, я могу констатировать, что проект не блистал совершенством, но проверку временем выдержал очень хорошо.)

Для разрешения именно таких ситуаций и была задумана эта книга. На протяжении долгого времени мне доводилось иметь дело с массой корпоративных программных приложений. Эти проекты часто основывались на сходных идеях, эффективность которых

---

[1] Таким образом обозначается ссылка на страницу книги, где описано указанное типовое решение.

была доказана при решении сложных проблем, связанных с управлением на уровне предприятия. В книге делается попытка представить подобные проектные подходы в виде *типовых решений (patterns)*.

Книга состоит из двух частей. Первая содержит несколько ознакомительных глав с описанием ряда важных тем, имеющих отношение к сфере проектирования корпоративных приложений. Здесь бегло формулируются различные проблемы и варианты их решения. Все подробности и нюансы вынесены в главы второй части. Эту информацию уместно трактовать как справочную, и я не настаиваю на том, чтобы вы штудировали ее от начала до конца. На вашем месте я поступил бы так: детально ознакомился с материалом первой части для максимально полного понимания предмета и обратился к тем главам или типовым решениям из второй части, близкое знакомство с которыми действительно необходимо. Поэтому книгу можно воспринимать как краткий учебник (часть I), дополненный более увесистым руководством (часть II).

Итак, наше издание посвящено проектированию корпоративных программных приложений. Подобные приложения предполагают необходимость отображения, обработки и сохранения больших массивов (сложных) данных, а также реализации моделей бизнес-процессов, манипулирующих этими данными. Примерами могут служить системы бронирования билетов, финансовые приложения, пакеты программ торгового учета и т.п. Корпоративные приложения имеют ряд особенностей, связанных с подходами к решению возникающих проблем: они существенным образом отличаются от встроенных систем, систем управления, телекоммуникационных приложений, программных продуктов для персональных компьютеров и т.д. Поэтому, если вы специализируетесь в каких-либо "иных" направлениях, не связанных с корпоративными системами, эта книга, вероятно, не для вас (хотя, может быть, вам просто хочется "вкусить" нового?). За общей информацией об архитектуре программного обеспечения рекомендую обратиться к работе [33].

Проектирование корпоративных приложений сопряжено со слишком большим числом проблем архитектурного толка, и книга, боюсь, не сможет дать исчерпывающих ответов на все. Я поклонник итеративного подхода к созданию программного обеспечения. А сердцем концепции итеративной разработки является положение о том, что пользователю следует показывать первые, пусть не полные, результаты, если в них есть хоть толика здравого смысла. Хотя между написанием программ и книг существуют, мягко говоря, заметные различия, мне хотелось бы думать, что эта — далеко не всеобъемлющая — книга все-таки окажется своего рода конспектом полезных и поучительных советов. В ней освещаются следующие темы:

- "расслоение" приложения по уровням;
- структурирование логики предметной области;
- разработка пользовательского Web-интерфейса;
- связывание модулей, размещаемых в памяти (в частности, объектов), с реляционной базой данных;
- принципы распределения программных компонентов и данных.

Список тем, которых мы *не* будем касаться, разумеется, гораздо обширнее. Помимо всего остального, я предполагал обсудить вопросы проверки структуры, обмена сообщениями, асинхронных коммуникаций, безопасности, обработки ошибок, кластеризации, интеграции приложений, структурирования интерфейсов "толстых" клиентов

и прочее. Но ввиду ограничений на объем, отсутствия времени и нехватки оформившихся идей мне это не удалось. Остается надеяться, что в недалеком будущем какие-либо типовые решения в этих областях все-таки появятся. Возможно, когда-нибудь выйдет второй том книги, который вберет в себя все новое, или кто-то другой возьмет на себя труд заполнить эти и другие пробелы.

Среди всего перечисленного наиболее важной и сложной является проблема поддержки системы асинхронных коммуникаций, основанной на сообщениях. Особо острую форму она принимает при необходимости интеграции многих приложений (да и вариант системы коммуникаций для отдельно взятого приложения также "подарком" не назовешь).

В намерения автора *не* входила ориентация на какую бы то ни было конкретную программную платформу. Рассматриваемые типовые решения прошли первое испытание в конце 1980-х и начале 1990-х годов, когда я работал с C++, Smalltalk и CORBA. В конце 1990-х я начал интенсивно использовать Java и обнаружил, что те же подходы оказались приемлемыми при реализации и ранних гибридных систем Java/CORBA, и более поздних проектов на основе стандарта J2EE. Недавно я стал присматриваться к платформе Microsoft .NET и пришел к заключению, что решения вновь вполне применимы. Мои коллеги по ThoughtWorks подтвердили аналогичные выводы в отношении Forte. Я не собираюсь утверждать, что то же справедливо для *всех* платформ, современных и будущих, используемых для развертывания корпоративных приложений, но до сих пор дело обстояло именно так.

Описание большинства типовых решений сопровождается примерами кода. Выбор языка программирования обусловлен только вероятными предпочтениями читателей. Java в этом смысле выглядит наиболее привлекательно. Всякий, кто знаком с C или C++, разберется и в Java; кроме того, Java намного проще, нежели C++. Практически любой программист, использующий C++, способен воспринимать Java-код, но не наоборот. Я стойкий приверженец объектной парадигмы, поэтому речь могла идти только об одном из объектно-ориентированных языков. Итак, примеры написаны преимущественно на языке Java. Период работы над книгой совпал с этапом становления среды .NET и системы программирования C#, которая, по моему мнению, обладает многими свойствами, присущими Java. Поэтому я реализовал некоторые примеры и на C#, впрочем, с определенным риском, поскольку у разработчиков еще нет достаточного опыта взаимодействия с платформой .NET, так что идиомы ее использования, как говорится, не созрели. Оба выбранных мною языка наследуют черты C, поэтому если вы владеете одним, то сможете воспринимать — хотя бы поверхностно — и код, написанный на другом. Моей задачей было найти такой язык, который удобен для большинства разработчиков, даже если он не является их основным инструментом. (Приношу свои извинения всем, кому нравится Smalltalk, Delphi, Visual Basic, Perl, Python, Ruby, COBOL и т.д. Я знаю: вы хотите сказать, что есть языки получше, чем Java или C#. Полностью с вами согласен!)

Примеры, приведенные в книге, преследуют цель объяснить и проиллюстрировать основные идеи, лежащие в основе типовых решений. Их не нужно трактовать как окончательные результаты; если вы намерены воспользоваться ими в реальных ситуациях, вам придется проделать определенную работу. Типовые решения — удачная отправная точка, а не пункт назначения.

Для кого предназначена книга

Я писал эту книгу в расчете на программистов, проектировщиков и архитекторов, которые занимаются созданием корпоративных приложений и стремятся улучшить качество принимаемых стратегических решений.

Я подразумеваю, что большинство читателей относятся к одной из двух групп: первые, со скромными потребностями, озабочены созданием собственных программных продуктов, а вторые, более требовательные, намерены пользоваться соответствующими инструментальными средствами. Если говорить о первых, предлагаемые типовые решения помогут им сдвинуться с места, хотя затем потребуются, разумеется, и дополнительные усилия. Вторым, как я надеюсь, книга поможет понять, *что* происходит в "черном ящике" инструментальной системы, и сделать осознанный выбор в пользу того или иного поддерживаемого системой типового решения. Например, наличие средства отображения объектных структур в реляционные отнюдь не означает, что вам не придется делать самостоятельный осознанный выбор в конкретных ситуациях. В этом поможет знакомство с типовыми решениями.

Существует и третья, промежуточная, категория читателей. Им я порекомендовал бы осторожно подходить к выбору инструментальных средств. Я не раз наблюдал, как некоторые буквально погрязают в длительных упражнениях по созданию рабочей среды программирования, не имеющих ничего общего с истинными целями проекта. Если вы убеждены в правильности того, что делаете, дерзайте. Не забывайте, что многие примеры кода, приведенные в книге, намеренно упрощены для облегчения их восприятия, и вам, возможно, придется немало потрудиться, чтобы применить их в особо сложных случаях.

Поскольку типовые решения — это общеупотребительные результаты анализа повторяющихся проблем, вполне вероятно, что с *некоторыми* из них вы уже когда-либо сталкивались. Если программированием корпоративных приложений вы занимаетесь долгое время, не исключено, что вам знакомо *многое*. Я не утверждаю, что книга представляет собой коллекцию свежих знаний. Напротив, я стараюсь убедить вас в обратном: книга трактует (пусть зачастую по-новому) *старые* идеи, проверенные временем. Если вы новичок, книга, я надеюсь, поможет вам изучить предмет. Если вы уже с ним знакомы, книга поспособствует формализации и структурированию ваших знаний. Важная функция типовых решений связана с выработкой общего терминологического словаря: если вы скажете, что разрабатываемый вами класс является, например, разновидностью **интерфейса удаленного доступа (Remote Facade, 405)**, собеседникам должно быть понятно, о чем идет речь.

Благодарности

Книга многим обязана людям, с которыми мне пришлось сотрудничать в течение долгих лет. Проявления помощи были крайне разнообразны, поэтому каюсь, но порой я даже не могу вспомнить, *что* именно — значимое, принципиальное и заслуживающее упоминания на страницах книги — сообщил мне тот или иной человек, и тем не менее я признателен всем.

Начну с непосредственных участников проекта. Дейвид Райс (David Rice), мой коллега по ThoughtWorks, внес громадный вклад: добрый десяток процентов всего содержимого — это его прямая заслуга. На завершающей фазе проекта, стремясь поспеть к сроку (а Дейвид к тому же обеспечивал взаимодействие с заказчиком), мы провели вместе немало вечеров, и в одной из бесед он признался, что наконец-то понял, почему работа над книгой требует полной самоотдачи.

Еще один "мыслитель"[2], Мэттью Фоммел (Matthew Foemmel), оказался непревзойденным поставщиком примеров кода и немногословным, но острым критиком, хотя легче было бы растопить льды Арктики, нежели заставить его написать забавы ради пару абзацев текста. Я рад поблагодарить Рэнди Стаффорда (Randy Stafford) за его лепту в виде **слоя служб (Service Layer, 156)**, а также вспомнить Эдварда Хайета (Edward Hieatt) и Роберта Ми (Robert Mee). Сотрудничество с Робертом началось с того, что, просматривая материал, он обнаружил какое-то логическое несоответствие. Со временем он стал самым пристрастным и полезным критиком: его заслуги состоят не только в обнаружении каких-то пробелов, но и в их заполнении!

Я в большом долгу перед всеми, кто вошел в число официальных рецензентов книги, — Джоном Бруэром (John Brewer), Кайлом Брауном (Kyle Brown), Дженс Колдуэй (Jens Coldewey), Джоном Крапи (John Crupi), Леонардом Фенстером (Leonard Fenster), Аланом Найтом (Alan Knight), Робертом Ми, Жераром Месарашем (Gerard Meszaros), Дерк Райел (Dirk Riehle), Рэнди Стаффордом, Дейвидом Сигелом (David Siegel) и Каем Ю (Kai Yu). Я мог бы привести здесь полный список служащих ThoughtWorks — так много коллег помогали мне, сообщая о своих проектах и результатах. Многие типовые решения оформились благодаря счастливой возможности общения с талантливыми сотрудниками компании, поэтому у меня нет другого выбора, как поблагодарить коллектив в целом.

Кайл Браун, Рейчел Рейниц (Rachel Reinitz) и Бобби Вулф (Bobby Woolf) отложили все свои дела, чтобы встретиться со мной в Северной Каролине и подробнейшим образом обсудить материалы книги. Я с удовольствием вспоминаю и наши с Кайлом телефонные беседы.

В начале 2000 года я вместе с Аланом Найтом и Каем Ю подготовил доклад для конференции Java One, который послужил первым прототипом книги. Я признателен моим соавторам, а также Джошуа Маккензи (Josh Mackenzie), Ребекке Парсонз (Rebecca Parsons) и Дейвиду Райсу за помощь, оказанную ими и тогда и позже. Джим Ньюкирк (Jim Newkirk) сделал все необходимое, чтобы познакомить меня с новым миром Microsoft .NET.

Я многому научился у специалистов, с которыми мне приходилось общаться и сотрудничать. В частности, хотелось бы поблагодарить Коллин Роу (Colleen Roe), Дейвида Мьюрэхеда (David Muirhead) и Рэнди Стаффорда за то, что они поделились со мной результатами работы над проектом системы Foodsmart в Джемстоуне. Не могу не упомянуть добрым словом всех, кто участвовал в плодотворных дискуссиях на конференциях Crested Butte, проводимых Брюсом Эклем (Bruce Eckel) в течение нескольких последних лет. Джошуа Керивски (Joshua Kerievsky) не смог выкроить время для полномасштабного рецензирования, но он прекрасно справился с функциями технического консультанта.

---

[2] Название компании ThoughtWorks можно перевести, скажем, как "работа мысли". — *Прим. пер.*

Я получил заметную помощь со стороны участников одного из читательских кружков, организованных при университете штата Иллинойс. Вот эти люди: Ариель Герценстайн (Ariel Gertzenstein), Боско Живальевич (Bosko Zivaljevic), Брэд Джонс (Brad Jones), Брайан Футей (Brian Foote), Брайан Марик (Brian Marick), Федерико Бэлэгур (Federico Balaguer), Джозеф Йодер (Joseph Yoder), Джон Брант (John Brant), Майк Хьюнер (Mike Hewner), Ральф Джонсон (Ralph Johnson) и Вирасак Витхаваскул (Weerasak Witthawaskul). Всем им большое спасибо.

Драгос Манолеску (Dragos Manolescu), экс-глава кружка, собрал собственную группу рецензентов, в которую вошли Мухаммад Анан (Muhammad Anan), Брайан Дойл (Brian Doyle), Эмад Гоше (Emad Ghosheh), Гленн Грэйссл (Glenn Graessle), Дэниел Хайн (Daniel Hein), Прабхахаран Кумаракуласингам (Prabhaharan Kumarakulasingam), Джо Куинт (Joe Quint), Джон Рейнк (John Reinke), Кевин Рейнолдз (Kevin Reynolds), Шриприя Шринивасан (Sripriya Srinivasan) и Тирумала Ваддираджу (Tirumala Vaddiraju).

Кент Бек (Kent Beck) внес столько предложений, что я затрудняюсь вспомнить их все. Он, например, придумал название для типового решения **частный случай (Special Case, 511)**. Джим Оделл (Jim Odell) был первым, кто познакомил меня с миром консалтинга, преподавания и писательства; мне не хватает слов, чтобы выразить всю глубину моей благодарности.

По мере работы над книгой я размещал ее черновики в Web. Среди тех, кто откликнулся и прислал свои вопросы, варианты возможных проблем и их решения, были Майкл Бэнкс (Michael Banks), Марк Бернстайн (Mark Bernstein), Грэхем Беррисфорд (Graham Berrisford), Бьерн Бесков (Bjorn Beskow), Брайан Борхэм (Brian Boreham), Шен Бродли (Sean Broadley), Перис Бродски (Peris Brodsky), Пол Кемпбелл (Paul Campbell), Честер Чен (Chester Chen), Джон Коукли (John Coakley), Боб Коррик (Bob Corrick), Паскаль Костэнза (Pascal Costanza), Энди Червонка (Andy Czerwonka), Мартин Дайл (Martin Diehl), Дэниел Дрейзин (Daniel Drasin), Хуан Гомес Дуазо (Juan Gomez Duaso), Дон Дуиггинз (Don Dwiggins), Питер Форман (Peter Foreman), Рассел Фриман (Russell Freeman), Питер Гэзмен (Peter Gassmann), Джейсон Горман (Jason Gorman), Дэн Грин (Dan Green), Ларс Грегори (Lars Gregori), Рик Хансен (Rick Hansen), Тобин Харрис (Tobin Harris), Рассел Хили (Russel Healey), Кристиан Геллер (Christian Heller), Ричард Хендерсон (Richard Henderson), Кайл Херменин (Kyle Hermenean), Карстен Хейл (Carsten Heyl), Акира Хирасава (Akira Hirasawa), Эрик Кон (Eric Kaun), Кирк Кнорнсчайлд (Kirk Knoernschild), Джеспер Лейдегаард (Jesper Ladegaard), Крис Лопес (Chris Lopez), Паоло Марино (Paolo Marino), Джереми Миллер (Jeremy Miller), Иван Митрович (Ivan Mitrovic), Томас Нейманн (Thomas Neumann), Джуди Оби (Judy Obee), Паоло Паровел (Paolo Parovel), Тревор Пинкни (Trevor Pinkney), Томас Рестрепо (Tomas Restrepo), Джоуэл Ридер (Joel Rieder), Мэттью Робертс (Matthew Roberts), Стефан Рук (Stefan Roock), Кен Роша (Ken Rosha), Энди Шнайдер (Andy Schneider), Александр Семенов, Стен Силверт (Stan Silvert), Джефф Суттер (Jeoff Soutter), Уолкер Термат (Volker Termath), Кристофер Тейм (Christopher Thames), Уолкер Тюро (Volker Turau), Кнут Ванхеден (Knut Wannheden), Марк Уоллес (Marc Wallace), Стефан Уэниг (Stefan Wenig), Брэд Уаймерслэдж (Brad Wiemerslage), Марк Уиндхолц (Mark Windholtz) и Майкл Юн (Michael Yoon).

Тех, кто здесь не упомянут, я тоже благодарю с не меньшей сердечностью.

Как всегда, моя самая горячая признательность любимой жене Синди (Cindy), чье общество я ценю намного больше, чем кто-либо сумеет оценить эту книгу.

И еще несколько слов

Это моя первая книга, оформленная с помощью языка XML и связанных с ним технологий и инструментов. Текст был набран в виде серии XML-документов с помощью старого доброго TextPad. Я пользовался и доморощенными шаблонами определения типа документа (Document Type Definition — DTD). Для генерации HTML-страниц Web-сайта применялся язык XSLT, а для построения диаграмм — надежный Visio, пополненный замечательными UML-шаблонами от Павла Храби (Pavel Hruby) (намного превосходящими по качеству аналоги из комплекта поставки редактора; на моем сайте приведен адрес, где их можно получить). Я написал небольшую программу, которая позволила автоматически включать примеры кода в выходной файл, избавив меня от кошмара многократного повторения операций копирования и вставки. При подготовке первого варианта рукописи я использовал XSL-FO в сочетании с Apache FOP, а позже для импорта текста в среду FrameMaker создал ряд сценариев XSLT и Ruby.

В процессе работы над книгой мне приходилось применять несколько инструментов из категории программ с открытым исходным кодом — JUnit, NUnit, ant, Xerces, Xalan, Tomcat, Jboss, Ruby и Hsql. Я искренне признателен всем авторам. Не обошлось, разумеется, и без коммерческих продуктов. В частности, я активно пользовался пакетом Visual Studio .NET и прекрасной интегрированной Java-средой разработки Idea от IntelliJ — первой, которая меня по-настоящему вдохновила со времен работы с языком Smalltalk.

Книга была приобретена для издательства Addison Wesley Майком Хендриксоном (Mike Hendrickson) и Россом Венаблзом (Ross Venables), которые руководили проектом. Я начал трудиться над рукописью в ноябре 2000 и закончил работу в июне 2002 года.

Сара Уивер (Sarah Weaver) выполняла обязанности главного редактора и координировала все стороны проекта, связанные с редактированием, верстской, корректурой, составлением предметного указателя и получением окончательной электронной версии книги. Диана Вуд (Dianne Wood), литературный редактор, предприняла все возможное, чтобы "подчистить" словесную форму, не испортив идейного содержания. Ким Арни Мулки (Kim Arney Mulcahy) окончательно скомпоновал материал, "отшлифовал" рисунки и подготовил для передачи в типографию итоговые файлы FrameMaker. При верстке мы придерживались того же формата, который был выбран ранее для книги [18]. Шерил Фергюсон (Cheryl Ferguson), корректор, позаботилась, чтобы в тексте осталось как можно меньше ошибок, а Ирв Хершман (Irv Hershman) составил предметный указатель.

Поддержка

Эта книга отнюдь не совершенна. Несомненно, в ней есть какие-то неточности; может быть, я упустил что-то важное. Если вы найдете то, что считаете ошибочным, или сочтете, что в книгу следует включить дополнительный материал, пожалуйста, пошлите свое сообщение по адресу: `fowler@acm.org`. Обновления и исправления будут представлены на странице `http://www.martinfowler.com/eaaErrata.html`[3].

О фотографии, размещенной на обложке книги

Когда писалась эта книга, в Бостоне происходило нечто более знаменательное. Я имею в виду строительство моста Bunker Hill Bridge (попробуйте-ка втиснуть это название в небольшой дорожный указатель) через Чарлз-ривер по проекту Леонарда П. Закима (Leonard P. Zakim). Мост Закима, изображенный на обложке книги, относится к разряду подвесных, которые до сих пор не приобрели в США такого распространения, как, скажем, в Европе. Мост не особенно длинен, но зато самый широкий в мире (в своей категории) и первый в Штатах, имеющий асимметричный дизайн. Кроме того (может быть, это самое главное), он очень красив.

Мартин Фаулер,
Мелроуз, Массачусетс,
август 2002 года
`http://martinfowler.com`

[3] В переводном издании этой книги учтены исправления, опубликованные на этой Web-странице, по состоянию на 21 декабря 2003 года. — *Прим. ред.*

Ждем ваших отзывов!

Именно вас, уважаемые читатели, мы считаем главными критиками и комментаторами этой книги. Мы ценим ваше мнение и хотим знать, что было сделано нами правильно, что можно было сделать лучше и что еще вы хотели бы увидеть изданным нами. Нам интересно услышать и любые другие замечания, которые вам хотелось бы высказать в наш адрес.

Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо либо просто посетить наш Web-сервер и оставить там свои замечания. Одним словом, любым удобным для вас способом дайте нам знать, нравится или нет вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас.

Посылая письмо или сообщение, не забудьте указать название книги и ее авторов, а также ваш обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию последующих книг.

Наши координаты:

E-mail:	info@williamspublishing.com
WWW:	http://www.williamspublishing.com

Информация для писем:

из России:	115419, Москва, а/я 783
из Украины:	03150, Киев, а/я 152