Введение

Я испытываю страсть к инфраструктурам и никогда не работал над вычислительной платформой, которая делала бы все, в чем я нуждался для продуктивного построения приложения. Платформа .NET от Microsoft замечательна, но она не всегда делает то, что мне нужно или что я хочу. Для удовлетворения своих потребностей я иногда искал подходящие инструментальные средства и инфраструктуры, и вот я создал их сам.

Инфраструктура — это просто воплощение в коде архитектуры или шаблона проектирования. Прежде чем получить хорошую инфраструктуру, необходимо иметь некую архитектуру. Это означает иметь видение и набор задач, как для архитектуры, так и для приложений, которые она должна поддерживать.

Эта книга об архитектуре приложений, проектировании и разработке на платформе .NET с использованием объектно-ориентированных концепций. Основное внимание здесь уделяется созданию *бизнес-объектов* и их реализации для работы в различных распределенных средах, включая веб и клиент/сервер. В книге описано много технологий .NET, объектно-ориентированное проектирование, концепции программирования и распределенной архитектуры.

В книге изложена большая часть логического процесса, использованного мной при разработке и создании инфраструктуры CSLA .NET, обеспечивающей объектно-ориентированную разработку приложения в среде .NET. Это подразумевает описание множества архитектурных концепций и идей, а также привлечение некоторых передовых методик .NET по созданию инфраструктур.

Книга демонстрирует также использование инфраструктуры для построения примера приложения с несколькими разными интерфейсами. При желании вы можете пропустить главы, относящиеся к проектированию инфраструктуры, и сразу перейти к разделам, где она используется для построения объектно-ориентированных приложений.

Одна из главных задач создания инфраструктуры CSLA .NET заключалась в упрощении разработки .NET. Разработчики, использующие описанную в этой книге инфраструктуру, могут не заботиться о подробностях таких базовых технологий, как дистанционный доступ, сериализация и рефлексия. Все они встроены в инфраструктуру так, чтобы разработчик мог использовать их, практически полностью сосредоточившись на бизнес-логике и проекте приложения, а не на решении многочисленных низкоуровневых проблем.

Эта книга — существенно переработанная версия предыдущих изданий книги *Expert C# 2005 Business Objects*. Нынешняя обновленная версия учитывает преимущества новых возможностей .NET 3.5, а также уроки, извлеченные из использования .NET 2.0 и 3.0 на протяжении нескольких последних лет.

Данная книга — это новое выражение тех концепций, над которыми я работал больше двенадцати лет. Все это время моя задача заключалась в том, чтобы обеспечить продуктивное использование объектно-ориентированного проектирования в распределенных многоуровневых приложениях. За эти годы существенно развились и технологии, и мое понятие этих концепций.

От CSLA .NET 2.0 до CSLA .NET 3.6

За прошедшие восемь лет инфраструктура CSLA .NET стала одной из наиболее широко используемых инфраструктур разработки на платформе Microsoft .NET. Со времени моего представления версии инфраструктуры в 2001 году, она развилась довольно существенно, частично в связи с изменениями самой платформы .NET, а частично благодаря отзывам сообщества, возникшего вокруг инфраструктуры CSLA .NET.

Инфраструктура CSLA .NET является отражением базовой архитектуры, которую я называю *CSLA* (Component-based, Scalable, Logical Architecture — компонентно-ориентированная, масштабируемая, логическая архитектура). За эти годы я получил по электронной почте сотни сообщений от людей, которые использовали CSLA в качестве основы собственных архитектур создаваемых приложений — от малых, однопользовательских, программ до полнофункциональных корпоративных приложений, являющихся основой их бизнеса.

Данная инфраструктура помогает решить два основных вопроса объектно-ориентированной разработки программного обеспечения.

- Как использовать бизнес-объекты для эффективного построения приложений Windows, веб-приложений и приложений, ориентированных на службы.
- Как обеспечить возможность использования объектно-ориентированного проектирования в среде распределенных вычислений.

Хотя платформа .NET поддерживает использование объектов, автор объекта должен проделать довольно много работы, чтобы полностью обеспечить такие важные концепции .NET, как связывание данных. Основное внимание инфраструктуры CSLA .NET и этой книги уделено обеспечению полной поддержки объектами связывания данных, а также других важнейших концепций, таких как проверка и авторизация. Для большинства пользователей инфраструктуры CSLA .NET — это главные предоставляемые ею преимущества.

Распределенные многоуровневые приложения и приложения, ориентированные на службы, создает множество людей. Использование объектно-ориентированного проектирования и бизнес-объектов в распределенной среде имеет собственные проблемы, а инфраструктура CSLA .NET использует различные подходы для решения этих проблем. Для многоуровневого приложения "клиент/сервер", инфраструктура предоставляет концепцию *мобильных объектов* (mobile object), которые фактически перемещаются между компьютерами в многоуровневой среде. Мобильные объекты предоставляют мощнейший способ реализации объектно-ориентированного проектирования в распределенных средах. Для приложений, ориентированных на службы, инфраструктура CSLA .NET применима при построении как приложений, так и служб. Инфраструктура незаменима при создании конечных приложений, а зачастую полезна при создании служб и рабочего цикла.

Поскольку платформа .NET и инфраструктура CSLA .NET существенно развились, я внес множество изменений и добавил новые возможности. В некоторых случаях применение новых концепций и возможностей требовало изменения существующих бизнес-объектов и кода пользовательского интерфейса. Я не берусь утверждать, что совместимость с прежней версией проста, но все же очень важно продвигать новые концепции, чтобы не отставать от изменений в технологии.

По возможности я минимизировал воздействие на существующий код; таким образом, для большинства приложений переход не должен быть слишком сложным. Хоть между версиями 3.0 и 3.6 существуют кардинальные различия, *наиболее* важный код должен обновиться легко. Относительно легко должен обновиться даже код версии 2.1. Бизнес-классы, написанные на CSLA .NET версии 1.x или 2.0, потребуют для обновления не так уж и много усилий.

За эти годы я получил по электронной почте несколько сообщений от людей, для которых инфраструктура CSLA .NET *не была* успешна, но это и не удивительно. Для эффективного использования инфраструктуры CSLA .NET вы должны быть сведущи в объектно-ориентированном проектировании, понимать концепцию мобильных объектов и обладать рядом других навыков. У архитектуры мобильных объектов есть множество преимуществ, но она не является ни самой простой, ни самой понятной.

Однако за тот же период времени я получил бесчисленное количество сообщений от людей, которым использование CSLA .NET обеспечило огромный успех при построении приложений — от Windows до веб, от малых приложений для розничной продажи до производственных и военных систем. Я удивлен и обрадован этими сообщениями и всеми замечательными областями, где использование инфраструктуры SLA .NET оказалось полезным людям.

Разработка с использованием CSLA .NET

Одной из особенностей платформы .NET является то, что она зачастую предоставляет несколько способов решения одной и той же проблемы. Некоторые из доступных способов лучше других, но наилучшие подходы для данной конкретной проблемы, возможно, не сразу очевидны. За прошедшие восемь лет я потратил много времени на исследование большинства этих возможностей и способов. Хотя разносторонность — это свидетельство жизнеспособности, в конце концов, я пришел к выводу, что некоторые из них лучше подходят для моих задач.

У меня был определенный набор задач для архитектуры и книги. Эти задачи важны, поскольку являются ключом к пониманию того, почему я сделал тот или иной выбор технологии .NET и способа ее использования. Задачи таковы.

- Обеспечить полную поддержку объектно-ориентированной модели программирования.
- Позволить разработчику использовать архитектуру, не вдаваясь в подробности ее внутренней реализации.
- Обеспечить высокую масштабируемость.
- Обеспечить высокую эффективность.
- Обеспечить разработчику высокую продуктивность использования бизнес-объектов, включая следующее:
  - поддержка связывания данных в Windows и Web Forms;
  - поддержка для многих типов пользовательских интерфейсов на основании тех же объектов;
  - управление правилами проверки;
  - управление правилами авторизации;
  - многоуровневая отмена на основе объектов (правка, отмена, применение);
  - интеграция с распределенными транзакционными технологиями, такими как Enterprise Services и `System.Transactions`.
- Обеспечить использование объектно-ориентированного проектирования в распределенной среде при помощи мобильных объектов.
- Упростить платформу .NET за счет решения таких сложных проблем, как сериализация, рефлексия и связь по сети.
- Обеспечить использование инструментальных средств, предоставляемых корпорацией Microsoft, включая такие компоненты Visual Studio .NET, как IntelliSense и Autocomplete.

Это избавит разработчика от прыжков сквозь обручи, т.е. позволит ему заниматься "нормальным" программированием, что немалое преимущество. Чтобы выполнить все эти задачи без инфраструктуры, разработчик вынужден будет писать много дополнительного кода, чтобы отслеживать бизнес-правила, реализовать многоуровневую отмену и обеспечивать сериализацию объектных данных. Весь этот код, конечно, важен, но он ничего не делает для решения практической задачи приложения.

К счастью, платформа .NET предоставляет немало мощнейших технологий, позволяющих сократить или устранить большую часть "вспомогательного" кода. Если эти технологии собраны под оболочкой инфраструктуры, то разработчику не придется иметь дело с ними вообще. В некоторых случаях именно задача упрощения управляла моими архитектурными решениями. В итоге разработчик, как правило, может просто написать стандартизированный класс C# и автоматически воспользоваться всеми преимуществами многоуровневой отмены, отслеживания бизнес-правил и т.д.

Это, конечно, заняло много времени и стоило многих усилий, но я объединил архитектуру и книгу, и, надеюсь, вы найдете их полезными во время разработки собственных приложений.

Лицензия на инфраструктуру

5. Бинарное или объектное распространение.

   Вы можете распространять данное программное обеспечение в бинарной или объектной форме без отображения объявления об авторском праве для конечного пользователя. Бинарная или объектная форма должна сохранять объявление об авторском праве, включенное в исходный код программного обеспечения.

6. Ограничения.

   Вы не можете продавать данное программное обеспечение. Если вы создаете инфраструктуру разработки программного обеспечения на основании данного программного обеспечения как производный продукт, не можете продавать полученный производный продукт. Это не ограничивает использование данного программного обеспечения для создания других типов некоммерческих или коммерческих приложений или производных продуктов.

7. Гарантийная оговорка.

   Программное обеспечение предоставляется "как есть", без гарантий. Вообще никаких. Это не означает наличия сокращенной, подразумеваемой, установленной законом или некой другой гарантии, включая неограниченную гарантию, гарантию высокого спроса или здравоохранения для специфических целей, патентную либо гарантию наличия или отсутствия ошибок. Кроме того, вы должны оговаривать это всякий раз, когда распространяете данное программное обеспечение.

8. Ответственность.

   Ни Рокфорд Лотка, ни кто-либо из участников разработки данного программного обеспечения не несут никакой ответственности за убытки любых видов, включая косвенные, специальные, последовательные, непредвиденные, штрафные или назидательные, связанные с этим программным обеспечением или этой лицензией, в максимальной степени юридического допуска, независимо от используемой базовой юридической теории. Кроме того, вы должны передавать это ограничение ответственности всякий раз, когда распространяете данное программное обеспечение.

9. Патенты.

   Если вы предъявляете иск по любому патенту, который, как вы полагаете, может относиться к данному программному обеспечению, за использование людьми данного программного обеспечения, ваша лицензия на данное программное обеспечение автоматически заканчивается. Патентные права, если таковые вообще имеются, лицензируются ниже и относятся только к данному программному обеспечению, но не к любым производным продуктам, которые вы создаете.

10. Прекращение соглашения.

    Ваши права заканчиваются автоматически, если вы каким-либо образом нарушаете лицензию. Рокфорд Лотка оставляет за собой право выпускать данное программное обеспечение с любыми сроками действия лицензии или прекратить распространение данного программного обеспечения в любое время. Такие решения не будут служить поводом для аннулирования этого соглашения. Данное соглашение сохраняет полную силу, если оно не прекращается так, как указано выше.

11. Государственное законодательство.

    Это соглашение создано и применяется в соответствии с законами штата Миннесоты, США.

12. Никакого присвоения.

Ни само соглашение, ни любой интерес в этом соглашении не подлежат лицензированию, без предварительного письменного одобрения разработчика.

13. Окончательное соглашение.

    Это соглашение прекращает и заменяет все предыдущие понятия и соглашения по данному предмету. Это соглашение может быть изменено только при последующей перезаписи, которая должным образом выполняется обеими сторонами.

14. Раздельность положений соглашения.

    Если какой-нибудь пункт этого соглашения будет признан судом соответствующей инстанции недопустимым или не имеющим законной силы, то все остальные пункты данного Соглашения остаются в полной силе, как будто такого недопустимого или не имеющего законной силы пункта никогда не было.

15. Разделы.

    Разделы, используемые в этом соглашении, предоставлены только для удобства и не должны рассматриваться как мнение или намерение.

Что необходимо для использования этой книги

Работоспособность кода этой книги была проверена на Microsoft Visual Studio 2008 Professional Edition SP1 и на версии 3.5 SP1 платформы .NET Framework. Используемая база данных, SQL Server Express, включена в комплект Visual Studio 2008 Professional. Корпоративная версия Visual Studio 2008 и полная версия SQL Server вполне допустимы, но не обязательны.

Для запуска инструментальных средств и перечисленных выше продуктов вам понадобится, по крайней мере, один компьютер с операционной системой Windows Vista, Windows XP SP2 (или выше), Windows Server 2003 или Windows Server 2008. Для проверки поддержки инфраструктурой CSLA .NET нескольких физических уровней вам, безусловно, понадобится дополнительный компьютер для каждого уровня, который вы желаете добавить (либо вы можете использовать инструмент Virtual PC либо похожий).

Как построена эта книга

В этой книге изложен логический процесс, лежащий в основе архитектуры CSLA .NET для Windows версии 3.6. Здесь описана конструкция инфраструктуры, поддерживающая архитектуру, а также продемонстрировано создание пользовательских интерфейсов WPF, веб-форм и приложений службы WCF на основании бизнес-объектов, написанных с использованием инфраструктуры.

Если вы читаете эту книгу, чтобы понять процесс разработки и построения инфраструктуры для разработки на платформе .NET, то вам стоит прочитать все главы. Если вы читаете эту книгу, чтобы понять, как использовать инфраструктуру CSLA .NET, и меньше интересуетесь тем, как разработана и реализована сама инфраструктура, то вам стоит прочитать главы 1–5 и 17–21.

Глава 1, “Распределенная архитектура”, представляет некоторые концепции, сопутствующие распределенной архитектуре, включая логические и физические архитектуры, бизнес-объекты и распределенные объекты. Но что важнее всего, эта глава готовит почву, демонстрируя логический процесс, результат которого представлен в остальной части книги.

Глава 2, "Проект инфраструктуры", демонстрирует, как архитектура, описанная в конце главы 1, используется в качестве отправной точки для инфраструктуры кода, решающей описанные ранее задачи. К концу главы вы познакомитесь с процессом проектирования объектов, которые будут реализованы в главах 6–16; но перед этим предстоит обратить внимание на ряд других вопросов.

В главе 3, "Объектно-ориентированное проектирование приложения", обсуждаются основы управляемого ответственностью объектно-ориентированного проектирования. В качестве примера эта глава рассматривает требования и проект типичного приложения.

В главах 4, "Объектные стереотипы CSLA .NET", и 5, "Шаблоны объектов CSLA .NET", обсуждается, как использовать каждый первичный базовый класс в инфраструктуре CSLA .NET, чтобы создать собственные бизнес-объекты. Здесь подробно обсуждаются объектно-ориентированные стереотипы, поддерживаемые базовыми классами CSLA .NET, наряду со структурой кода для редактируемых и доступных только для чтения объектов, а также коллекций и списков "имя/значение".

Главы с 6, "Реализация бизнес-инфраструктуры", по 16, "Другие средства инфраструктуры", посвящены конструкции самой инфраструктуры CSLA .NET непосредственно. Если вы интересуетесь кодом объявлений свойств, правил проверки, правил авторизации, многоуровневой отмены, поддержки мобильных объектов и сохранения состояния объектов, то эти главы для вас. Кроме того, здесь использованы некоторые наиболее передовые и интересные элементы платформы .NET Framework, включая связывание данных, сериализацию, рефлексию, динамический вызов метода, WCF, защиту .NET, Enterprise Services, `System.Transactions`, строгое именование сборок, динамическую загрузку сборок, файлы конфигурации приложения и многое другое.

В главах 17, "Реализация бизнес-объекта", и 18, "Пример доступа к данным", создаются бизнес-объекты для приложения. Здесь иллюстрируется, как вы можете использовать инфраструктуру, чтобы быстро и легко создать мощный набор бизнес-объектов для приложения. Конечный результат — набор объектов, которые не только моделируют бизнес-обязанности, но и поддерживают связывание данных, проверку правильности, авторизацию, многоуровневую отмену и различные физические конфигурации, способные оптимизировать производительность, масштабируемость, защиту и отказоустойчивость, обсуждаемые в главе 1, "Распределенная архитектура".

Глава 19, "Пользовательский интерфейс Windows Presentation Foundation", демонстрирует создание интерфейса WPF для бизнес-объектов.

В главе 20, "Пользовательский интерфейс Web Forms", рассматривается создание интерфейса ASP.NET Web Forms со сравнимыми функциональными возможностями.

Глава 21, "Интерфейс WCF Service", демонстрирует построение служб WCF с использованием бизнес-объектов. Этот подход обеспечивает ориентированную на службы разработку, предоставляя программный интерфейс к бизнес-объектам, к которым может обратиться любая веб-служба или любой клиент WCF.

Инфраструктура реализует логическую модель, которую вы можете развернуть в различных физических конфигурациях для оптимальной поддержки Windows, веб, а также клиентов и служб XML.

Загрузка кода

Код, отражающий содержимое этой книги, доступен в области Source Code/Download веб-сайта Apress (`www.apress.com`). Последняя версия инфраструктуры и примеры приложений доступны по адресу: `www.lhotka.net/cslanet/download.aspx`.

Связь с автором

Вы можете связаться с Рокфордом Лоткой на его веб-сайте, `www.lhotka.net`, на котором содержится его блог, информация об инфраструктуре и книге, а также приведена его контактная информация.

Соглашения, принятые в этой книге

При оформлении книги использованы соглашения, общепринятые в компьютерной литературе.

- Новые термины в тексте выделяются *курсивом*. Чтобы обратить внимание читателя на отдельные фрагменты текста, также применяется *курсив*.
- Текст программ, функций, переменных, URL, веб-страниц и другой код представлены моноширинным шрифтом.
- Все, что придется вводить с клавиатуры, выделено **полужирным моноширинным** шрифтом.
- Знакоместо в описаниях синтаксиса выделено *курсивом*. Это указывает на необходимость заменить его фактическим именем переменной, параметром или другим элементом, который должен находиться на этом месте `BINDSIZE=(`*`максимальная ширина колонки`*`)*(`*`номер колонки`*`)`.
- Пункты меню и названия диалоговых окон представлены следующим образом: Menu Option (Пункт меню).
- Разрыв слишком длинных строк кода, не помещающихся на странице, обозначен специальным символом ↳.

От издательства

Вы, читатель этой книги, и есть главный ее критик. 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 авторам.

Мы ждем ваших комментариев. Вы можете прислать письмо по электронной почте или просто посетить наш веб-сервер, оставив на нем свои замечания, — одним словом, любым удобным для вас способом дайте нам знать, нравится ли вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более подходящими для вас.

Посылая письмо или сообщение, не забудьте указать название книги и ее авторов, а также ваш e-mail. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию следующих книг. Наши координаты:

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

Адреса для писем из:

России: 127055, г. Москва, ул. Лесная, д. 43, стр. 1
Украины: 03150, Киев, а/я 152