Оглавление

Благодарности

Хотел бы поблагодарить многих специалистов, приславших свои замечания по ключевым разделам книги, среди которых Дон Багерт (Don Bagert), Джон Бентли (Jon Bentley), Стивен Блэк (Steven Black), Роберт Бернс (Robert C. Burns) из компании «Boeing», Тревор Берридж (Trevor Burridge), Аугусто Коппола (Augusto Coppola), Алан Корвин (Alan B. Corwin) из «Process Builder», Райан Флеминг (Ryan Fleming), Пэт Форман (Pat Forman), Роберт Гласс (Robert L. Glass) из «Computing Trends», Дэвид Гудман (David Goodman), Оуейн Гриффитс (Owain E. Griffiths), Брейди Хонсингер (Bradey Honsinger), Ларри Хьюз (Larry M. Hughes) из компании «Sprint», Роберт Ли (Robert E. Lee), Эйвонелл Ловхог (Avonelle Lovhaug), Марк Лутц (Mark Lutz), Стив Мэттингли (Steve Mattingly), Грант Мак-Лахлин (Grant McLaughlin), Брайан Мак-Лин (Brian P.McLean), Хэнк Мьюрет (Hank Meuret), Х. Фернандо Наведа (J. Fernando Naveda), Энтон Пэнг (Anthon Pang), Дэвид Парнас (David L. Parnas), Мэтт Пелоквин (Matt Peloquin), Том Рид (Tom Reed), Кейти Роуд (Kathy Rhode), Стив Ринн (Steve Rinn), У. Пол Роджерс (Wm. Paul Rogers), Джей Силвермен (Jay Silverman), Андре Синтцофф (Andrй Sintzoff), Тим Старри (Tim Starry), Стив Токи (Steve Tockey), Леонард Трипп (Leonard L. Tripp), Том Вентцер (Tom Ventser) из группы «DMR Consulting Group», Карл Вигерс (Karl Wiegers) и Грег Уилсон (Greg Wilson).

Мои благодарности также многочисленным рецензентам, высказавшимся по отдельным конкретным вопросам.

Особо хочу поблагодарить великолепный коллектив по подготовке книги издательства Addison-Wesley: Майка Хендриксона (Mike Hendrickson), Ребекку Гринберг (Rebecca Greenberg), Эйми Флейшер (Amy Fleischer), Кэрин Хансен (Karin Hansen) и Дженис Оуенс (Janis Owens). Рабо-

тать с каждым из них одно удовольствие, и книга стала значительно лучше в результате их труда.

Я также высоко ценю опыт сотрудничества при подготовке первого издания этой книги – «After the Gold Rush». Хотел бы напомнить об огромной работе редактора проекта Виктории Тульман (Victoria Thulman) и других сотрудников издательства: Бена Райана (Ben Ryan), Роба Нанса (Rob Nance), Черил Пеннер (Cheryl Penner) и Полы Горелик (Paula Gorelick).

Введение

Кажется, что это просто... пока не попробуешь.

Из журнала IEEE Software[1]

Я сидел в самолете, стоявшем на взлетной полосе, когда прозвучало объявление капитана: «У нас неполадки в системе кондиционирования самолета. Эта система поддерживает уровень кислорода на борту, поэтому она должна заработать раньше, чем мы взлетим. Перезапуск кондиционеров не удался, поэтому мы сейчас выключим и снова включим электропитание. *Знаете, все эти новые самолеты управляются компьютерами, поэтому они не слишком надежны*».

Пилот выключил и снова включил питание – по сути «перезагрузил» самолет, и рейс продолжился без происшествий. Нечего и говорить, что по окончании воздушного путешествия я с большой радостью вышел из самолета.

Лучшие времена и худшие

Лучшие разработчики ПО ведут свои проекты так, чтобы обеспечить достижение целевых показателей качества. Они точно планируют сроки сдачи ПО на месяцы и годы вперед. Проекты разработки ПО укладываются в выделенный бюджет, и производительность таких разработчиков постоянно растет. Моральный дух их персонала высок, и клиенты очень довольны.

---

[1] Из книжного обзора, посвященного [137].

- Телекоммуникационной компании понадобилось изменить около 3 тысяч строк в базовом ПО объемом примерно в 1 000 000 строк. Изменения были внесены столь тщательно, что через год работы не обнаружилось ни одной ошибки. Время, которое потребовалось для внесения изменений, включая анализ требований, планирование, реализацию и тестирование, составило *9 часов* [110].
- Группа разработчиков ПО для BBC США взялась реализовать некий проект за год с бюджетом $2 000 000, хотя другие вполне достойные разработчики предлагали срок до 2 лет при бюджете до $100 000 000. Когда же эта группа сдала ПО на месяц раньше срока, менеджер проекта заявил, что успех достигнут за счет методик, известных уже несколько лет, но редко применяемых на практике [49], [131].
- Авиастроительная компания разрабатывает ПО для клиентов по фиксированной цене, при этом только 3% ее проектов превышают сметную стоимость; 97% из 100 укладываются в бюджет.[1]
- Организация, твердо следующая политике достижения исключительного качества ПО, в течение 9 лет добивалась ежегодного снижения на 39% количества дефектов, обнаруживаемых после выпуска версий; итоговое снижение составило 99% [56].

Вместе с огромными успехами, примеры которых приведены выше, отрасль ПО приносит в экономику миллиарды долларов как за счет прямых продаж самого ПО, так и в результате повышения эффективности и производительности, а также создания продуктов и услуг, которые возможны только при использовании соответствующего ПО.

Методики, необходимые для создания качественного программного продукта, известны уже 10, а то и 20 лет. Тем не менее, несмотря на впечатляющие достижения, отрасль ПО не использует весь свой потенциал. Между передовыми разработчиками и общей массой существует огромный разрыв, а многие широко применяемые методики сильно устарели и не обеспечиваются достаточными ресурсами. Эффективность среднего проекта ПО оставляет желать лучшего, о чем свидетельствуют многие хорошо известные провалы.

- Налоговая служба США провалила программу модернизации ПО стоимостью $8 000 000 000, что обошлось в $50 000 000 000 несобранных доходов в год [3].

[1] Из разговора с автором.

- Улучшенная АСУ Федерального управления авиации (FAA) превысила выделенный бюджет примерно на $3 000 000 000 [17], [53], [48].
- Неполадки в системе обработки багажа привели к задержке открытия международного аэропорта в Денвере более чем на год. Потери оцениваются в $1 100 000 в день [48], [53].
- Ракета «Ариан-5» взорвалась при первом пуске из-за ошибки в ПО [99].
- Бомбардировщик «Б-2» также не взлетел с первого раза из-за проблем с ПО [44].
- Управляемые компьютером паромы в г. Сиэттл (штат Вашингтон) около полутора десятка раз врезались в доки, нанеся ущерб на сумму свыше $7 000 000. Власти штата рекомендовали выделить более $3 000 000 на перевод паромов обратно на ручное управление [98].

Подобным ошибкам подвержены и другие проекты. Около четверти из них терпят полную неудачу с самого начала [132], [66]. Очень часто к моменту сворачивания проекта выявляется двукратный перерасход бюджета. Примерно половина всех проектов либо затягивается, либо превышает сметную стоимость, либо обеспечивает меньше функциональных возможностей, чем предусматривалось [132].

Для предприятий такие свернутые проекты означают упущенные возможности: если бы закрытие проекта обходилось в 10% выделенных средств, а не в 200%, нетрудно представить, чего можно было бы добиться, просто перенаправив эти ресурсы в проекты, которые были завершены.

На национальном уровне отмененные проекты представляют собой чудовищную и бесполезную трату сил и ресурсов. Грубые подсчеты показывают, что свернутые проекты ПО обходятся экономике приблизительно в $40 000 000 000.[1]

---

[1] Этот грубый расчет основан на статистике занятости, представленной в табл. 7.2 «Структура занятости работников сферы ПО» по должностям ученых компьютерно-информационной отрасли, проводящих исследования; программистов-компьютерщиков; инженеров прикладного ПО; инженеров системного ПО и аналитиков компьютерных систем. Другие должности в этом анализе не учитывались. Итоговые расходы экономики США на разработку ПО рассчитывались путем умножения средних совокупных расходов на оплату труда ($95 000) на 1 741 000 работников этих должностей. Четверть итоговой суммы в $160 000 000 000 25% – это доля, затраченная на отмененные проекты. В этом анализе может не учитываться влияние отмененных проектов, поскольку риск отмены увеличивается с ростом объема проекта, поэтому отмененные проекты могут оказаться более дорогостоящими по сравнению со средними расчетами.

Но и успешные проекты могут представлять угрозу безопасности и общественному благополучию. Руководителю проекта в Lotus звонил хирург, который использовал электронную таблицу для анализа состояния пациента во время операции на открытом сердце [142]. В журнале «Ньюсвик» публиковались фотографии военных, планирующих военные операции при помощи Microsoft Excel на своих переносных компьютерах; группа технической поддержки Excel принимала телефонные звонки с поля боя во время активных военных действий.

Цель данной книги

Процесс разработки ПО можно сделать прогнозируемым, контролируемым, экономичным и управляемым. Обычно разработка ПО ведется иначе, однако есть все возможности делать именно так. Эта книга посвящена зарождающейся профессии – инженерии ПО (т. е. технологии разработки ПО) и практической профессиональной методологии, которая обеспечивает экономичное создание высококачественного программного обеспечения.

В книге обсуждаются следующие вопросы:

- Что такое инженерия разработки ПО?
- Как инженерия ПО связана с компьютерной наукой, т. е. с наукой о вычислительных системах?
- Почему обычного программирования недостаточно?
- Зачем нужна *профессия* инженерии разработки ПО?
- Почему *инженерия* является наилучшей моделью профессиональной разработки ПО?
- В чем отличия эффективных методик, используемых в различных проектах (или различных компаниях), и какие принципы практически одинаковы?
- Что могут предпринять организации, чтобы поддержать профессиональный подход к разработке ПО?
- Что нужно сделать индивидуальным разработчикам ПО, чтобы стать профессионалами?
- Что могут предпринять представители отрасли разработки ПО в целом, чтобы создать настоящую профессию «инженерия ПО»?

Структура книги

Материал книги постепенно переходит от рассмотрения программирования как ремесла в его сегодняшнем состоянии к изучению технологии ПО как профессии, которая может сформироваться в будущем.

Часть I «Смоляная яма программного обеспечения» содержит рассказ о том, как отрасль эволюционировала к своему нынешнему состоянию. Этот процесс определялся многими факторами, понимать которые необходимо, для того чтобы ускорять, а не замедлять наступление перемен, призванных сделать успешные проекты ПО повседневной реальностью.

Часть II «Индивидуальный профессионализм» рассматривает шаги, которые специалист может сделать самостоятельно для достижения высокого профессионального уровня в разработке ПО.

Проекты ПО настолько сложны, что многие ключевые факторы невозможно обсудить на уровне индивидуального разработчика. В части III «Организационный профессионализм» представлены организационные методики, необходимые для поддержания более высокого профессионализма в программных проектах. Часть IV «Индустриальный профессионализм» рассматривает меры, которые должны быть предприняты в масштабе отрасли, чтобы обеспечить профессиональный подход на индивидуальном и организационном уровнях.

У этой книги есть партнерский сайт, *www.construx.com/profession*, на котором размещены материалы, связанные с этой книгой, включая списки профессиональной литературы для чтения, планы самообучения, описание существующих инициатив по сертификации и лицензированию, а также ссылки на многие другие полезные сайты.

Что я узнал с 1999 г.

Книга «Профессиональная разработка программного обеспечения» представляет собой обновленный и значительно расширенный вариант моей книги, вышедшей в 1999 г. [86]. С 1999 г. я усвоил несколько положений, которые нашли отражение в моей новой книге.

- Вопрос лицензирования разработчиков ПО оказался более спорным, чем я ожидал. Я по-прежнему считаю, что лицензирование небольшого количества инженеров-программистов – это важная часть защиты жизнедеятельности людей и их безопасности. Я старался разъяснить, что лицензирование представляет собой всего лишь

одну из множества инициатив, направленных на повышение профессионализма разработчиков ПО, и не самая важная.

- Образовательную подготовку инженеров-программистов не обязательно жестко увязывать с лицензированием. Программы подготовки на младших и выпускных курсах могут быть направлены на формирование инженерного склада мышления у разработчиков ПО, но при этом не обязательно на их подготовку к получению лицензии профессионального инженера. Если лицензию в конечном итоге получат менее 5% разработчиков ПО, что кажется вполне вероятным, то ориентация большинства программ подготовки и обучения на получение обучающимися лицензии представляется неразумной.
- Мир не рухнул первого января 2000 г., когда считалась актуальной угроза масштабных сбоев в работе компьютерных систем (я не думал, что проблемы, связанные с наступлением 2000 г., будут катастрофическими). Мрачные прогнозы не подтвердились. Более того, сама проблема Y2K была в определенном смысле вызвана *успешной* технологией разработки ПО. Она не возникла бы, если бы столь многие системы ПО не просуществовали значительно дольше, чем предполагалось.
- Современные разработки ПО действительно во многом впечатляют, поэтому любые рассуждения о профессионализации отрасли должны учитывать ее успехи. Необходима чрезвычайная осторожность, чтобы в попытках усилить слабые стороны процесса не отказаться от проверенных и успешных методик.

Кому адресована эта книга

Тем, для кого *разработка ПО служит источником средств к существованию*, эта книга даст представление о шагах, которые следует предпринять, чтобы стать настоящим профессионалом в этой области.

Менеджеры проектов разработки ПО найдут здесь свод отличительных особенностей, по которым хорошо управляемые проекты ПО можно отличить от проектов, управляемых плохо, а также обзор методов, которые могут сделать проекты более успешными.

Руководителям организаций-разработчиков ПО эта книга укажет преимущества и выгоды системного подхода к разработке ПО, а также действия, необходимые для их реализации.

Студенты, которые хотели бы работать в области создания ПО, познакомятся здесь с основами технологии ПО и получат представление о карьере в этой отрасли.

На пути к профессиональной разработке ПО

Исследователи, специализирующиеся в этой области, давно указывали, что производительность компаний, конкурирующих в одной отрасли, может отличаться более чем в десять раз [91]. Последние исследования показали, что разница может достигать и 600 раз [145]. Самые эффективные компании действительно процветают.

Выгоды формирования настоящей профессии разработчика ПО весьма убедительны. Согласно традиционным воззрениям, наибольший риск представляют перемены. Если же говорить о программном обеспечении, то наибольший риск представляет как раз их отсутствие и погружение в болото порочных, экстравагантных технологий разработки ПО вместо принятия методов, давно уже доказавших свою практическую эффективность.

Как реализовать такой переход? Это и есть главная тема книги.

– Беллвью, штат Вашингтон.
День Поминовения, 2003 г.

ЧАСТЬ ПЕРВАЯ

Смоляная яма программного обеспечения

ГЛАВА ПЕРВАЯ

Динозавры в смоляной яме

Тот, кто не хочет прибегать к новым средствам, должен ожидать новых бед.

Френсис Бэкон

Фредерик Брукс еще в 1975 г. сравнил разработку крупных систем программного обеспечения с борьбой динозавров, мамонтов и саблезубых тигров с засасывающей липкой смолой в яме [21], предсказав на долгие годы вперед сохранение подобной ситуации в проектировании и разработке ПО, когда одну ногу можно освободить, только завязнув другой ногой.

Проблемы, которые Брукс выявил почти тридцать лет назад, уже тогда не были новы, так что у разработчиков ПО было более четверти века, чтобы поработать над ними. Так далеко ли удалось продвинуться с тех пор?

Многие проблемы, буквально преследующие почти любой проект разработки ПО, практически не изменились. К примеру, вопрос сроков выполнения остается типичным и для проектов сегодняшних дней. По некоторым оценкам примерно 75% средних и свыше 90% крупных проектов ПО сталкиваются с проблемами жестких сроков.[1] При этом сверхурочные переработки программистов являются скорее правилом, чем исключением.[2] Хорошо известно, что современные зарождающиеся предприятия, работающие в области ПО, ожидают от своих сотрудников значительной сверхурочной работы, а рассказы о программистах, засыпающих ночью

---

[1] Данные о статистике напряженности в связи со сроками выполнения взяты из работы [64]. Данные о типичном соблюдении сроков сдачи проектов ПО имеются в работах [64], [66], [132].

[2] Этот вопрос подробно обсуждается в книге автора [83].

за своими дисплеями, можно услышать почти всюду [18]. Однако еще в середине 60-х годов XX века в одном из исследований утверждалось, что «многие программисты, не укладываясь в сжатые сроки, просиживают ночи на рабочих местах» [22]. В 1975 г. Фредерик Брукс указывал, что «из-за нарушения сроков исполнения срывается куда больше проектов, чем в силу всех остальных причин вместе взятых» [19]. Так что превышение сроков исполнения проектов наблюдается уже более тридцати лет, а если учесть, что люди нетерпеливы по природе, то и значительно дольше.

Размах сегодняшних проектов ПО кажется беспрецедентным, что, разумеется, наталкивает их создателей на мысль, что никто и никогда не сталкивался с проблемами подобного масштаба. Тем не менее даже у гигантских проектов, вроде Windows NT, были предшественники. Объемы современных крупных проектов разработки ПО действительно впечатляют, причем каждый разработчик естественно полагает, что он первый берется за столь масштабный проект. Однако, как я уже сказал, прецеденты существовали и ранее. Например, первоначальный проект Windows NT потребовал около полутора тысяч человеко-лет трудозатрат,[1] однако разработка OS/360 фирмы IBM, завершенная в 1966 г., оказалась в три раза более трудоемкой.[2]

Данные последних исследований показывают, что наиболее часто встречающиеся причины неудач проектов по разработке ПО связаны с проблемами их соответствия заданным техническим условиям, которые выстраивают неверную систему. Эти требования слишком размыты, чтобы их можно было точно реализовать, а некоторые из них меняются настолько часто, что вносят полнейшую неразбериху в системный проект [24], [132]. Но все же проблемы с соответствием требованиям также далеко не новы. Еще в 1969 г. Роберт Фрош (Robert Frosch) заметил, что система «может удовлетворять букве технических условий, но тем не менее не являться целиком и полностью удовлетворительной» [47].

Большинство разработчиков ломают голову, пытаясь поспеть за стремительными изменениями, которые происходят благодаря развитию Интернета. Как угнаться за новыми языками, меняющимися стандартами и предложениями новых программных продуктов? Для тех, кто работает

---

[1] Это примерная сумма. Оценка основана на опубликованных данных о стоимости разработки Windows NT ($150 000 000) и совокупных расходах на оплату труда в $100 000 на человека в год [146].

[2] Трудозатраты составили примерно 5000 человеко-месяцев [19].

в сфере программирования уже пару десятков лет, нынешняя ситуация кажется очень похожей на ту, что была в середине 80-х годов прошлого века, когда пришествие персональных компьютеров *IBM* полностью изменило использование вычислительных машин в компаниях.

Во времена разработки языка программирования FORTRAN в 1954–1958 гг. предполагалось, что он устранит сложность программирования компьютеров: ученые и инженеры просто вводили бы свои формулы в компьютер, а тот транслировал бы их в машинный код; отсюда и название FORTRAN – FORmula TRANslation (ТРАНслятор ФОРмул). Но, разумеется, FORTRAN не исключил программирование, а лишь сократил его объем на машинном языке. Время от времени появляются многообещающие заявления о возможности автоматизации программирования [118]. Компьютеры станут настолько «умными», что нужда в программистах и вовсе отпадет. Однако эта пластинка уже была заезжена более 35 лет назад, когда Джин Билински подметил, что «описание бизнесмена, бодро общающегося со своим всемогущим компьютером на обычном языке, регулярно появляется в прессе» [22]. Реальность же заключается в том, что подробное до мелочей описание проблемы – задача весьма трудная, и эта сторона программирования никуда не уйдет. Новые инструменты полезны, но они не заменяют ясность мышления. Я написал об этом в 1996 г. в своей книге «Rapid Development» [83], однако Роберт Фрош уже утверждал то же самое в издании *IEEE Spectrum* 30 лет назад.

Если говорить о разработке ПО в эпоху Интернета., то он дает возможность его создателям легко распространять обновленные версии своих программ. Пользователи могут загружать обновления программ в электронном виде без необходимости копирования их на компакт-диски CD или DVD – доставка быстро и недорого. В свою очередь, это увеличивает давление на разработчиков, чтобы те чаще выпускали обновления программ в ответ на требования пользователей. Интернет-разработчики утверждают, что пользователи просто хотят скорее получить ПО, нежели обрести его безукоризненную версию. У интернет-разработчиков в ходу поговорка: «Лучше быть первым, чем правым».

Насколько действительно беспрецедентна эта ситуация? Некоторые интернет-разработчики считают, что такая динамичность уникальна именно для веб-проектов, но ветеранам отрасли программирования известно: низкая стоимость распространения ПО, легкость внесения исправлений, невысокие потери в случае неудач – все это сильно смахивает на произ-

водственную среду разработки ПО своими силами для компьютеров коллективного пользования.

Эти тенденции, сложившиеся за 25 лет истории разработки ПО, являются источником и радости, и понимания того факта, что некоторые проблемы существуют уже четверть века и достаточно распространены. Мы действительно увязли в этой смоляной яме слишком давно, но повод для оптимизма все же есть: достаточно долго имея дело с одними и теми же проблемами, мы можем понять их сущность, и, как я намерен показать в этой книге, похоже, что мы находимся на пути их решения.

ГЛАВА ВТОРАЯ

Ложное золото

Надежда хороша на завтрак, но не годится на ужин.
Френсис Бэкон

Проблемы с ПО сохраняются частично в силу завораживающей притягательности нескольких неэффективных практических подходов. Во времена Калифорнийской «золотой лихорадки» в конце 40-х – начале 50-х годов XIX века некоторых золотоискателей обманывало «ложное золото» – пирит железа, который блестит и сияет, как настоящее золото. Однако, в отличие от настоящего, «ложное» золото – вещество хрупкое, слоистое, ломкое и почти ничего не стоит. Опытным золотодобытчикам хорошо известно, что настоящее золото мягкое, пластичное и не крошится под давлением. Уже 50 лет разработчики ПО поддаются соблазну своего «ложного золота». Негодные практические методы имеют соблазнительную привлекательность, но оказываются «ложным золотом», и, как и пирит железа, они хрупкие, ломкие и практически ничего не стоят.

Перемещение каменных глыб

Заглянем еще дальше в историю, на много веков, задолго до Калифорнийской «золотой лихорадки», и предположим, что вы строите одну из древних пирамид. Перед вами стоит задача: переместить огромную каменную глыбу на 10 км от реки на место строительства, как показано на рис. 2.1. У вас есть 20 человек и 100 дней, чтобы переместить этот камень.

Вам разрешено пользоваться любым методом, чтобы камень оказался на нужном месте. Нужно каждый день передвигать камень на 100 метров