

Карл Вигерс и Джой Битти

Разработка требований к программному обеспечению

Издание третье, дополненное

 РУССКАЯ РЕДАКЦИЯ



2014

УДК 004.738.5
ББК 32.973.202
В41

Вигерс Карл, Битти Джой

В41 Разработка требований к программному обеспечению. 3-е изд., дополненное / Пер. с англ. — М. : Издательство «Русская редакция» ; СПб. : БХВ-Петербург, 2014. — 736 стр. : ил.

ISBN 978-5-7502-0433-5 («Русская редакция»)

ISBN 978-5-9775-3348-5 («БХВ-Петербург»)

Эта книга — подробное руководство по разработке качественных требований к программному обеспечению. Здесь описаны десятки проверенных на практике приемов выявления, формулирования, разработки, проверки, утверждения и тестирования требований, которые помогут разработчикам, менеджерам и маркетологам создать эффективное ПО. Настоящее издание дополнено новыми приемами, посвященными разработке требований в проектах гибкой разработки (agile).

Основная аудитория — бизнес-аналитики и разработчики, а также дизайнеры, программисты, тестировщики и другие члены команды, задача которых понять и удовлетворить чаяния клиентов, а также маркетологи, менеджеры по продуктам и менеджеры проекта, которые должны проникнуться «духом» и особенностями продукта, чтобы сделать его в полной мере конкурентоспособным.

Книга состоит из 32 глав, 3 приложений и словаря терминов.

УДК 004.738.5

ББК 32.973.202

Microsoft, а также содержание списка, расположенного по адресу: <http://www.microsoft.com/about/legal/en/us/IntellectualProperty/Trademarks/EN-US.aspx> являются товарными знаками или охраняемыми товарными знаками корпорации Microsoft в США и/или других странах. Все другие товарные знаки являются собственностью соответствующих фирм. Все адреса, названия компаний, организаций и продуктов, а также имена лиц, используемые в примерах, вымышлены и не имеют никакого отношения к реальным компаниям, организациям, продуктам и лицам.

© 2013, Translation Russian Edition Publishers.

Authorized Russian translation of the English edition of Software Requirements, Third Edition, ISBN 978-0-7356-7966-5 © Karl Wieggers and Seivel.

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

© 2014, перевод ООО «Издательство «Русская редакция».

Авторизованный перевод с английского на русский язык произведения Software Requirements, Third Edition, ISBN 978-0-7356-7966-5 © Karl Wieggers and Seivel.

Этот перевод оригинального издания публикуется и продается с разрешения O'Reilly Media, Inc., которая владеет или распоряжается всеми правами на его публикацию и продажу.

© 2014, оформление и подготовка к изданию, ООО «Издательство «Русская редакция», издательство «БХВ-Петербург».

Вигерс Карл, Битти Джой

Разработка требований к программному обеспечению

Издание третье, дополненное

Совместный проект издательства «Русская редакция» и издательства «БХВ-Петербург».

 РУССКАЯ РЕДАКЦИЯ

 bhv®

Подписано в печать 21.04.2014 г. Формат 70х100/16.

Усл. физ. л. 46. Тираж 1500 экз. Заказ №

Первая Академическая типография «Наука», 199034, Санкт-Петербург, 9 линия, 12/28

Оглавление

Введение.....	XIV
ЧАСТЬ I Требования к ПО: что, почему и кто.....	1
Глава 1 Основы разработки требований к ПО.....	2
Определение требований к ПО	4
Особенности интерпретации требований	5
Уровни и типы требований.....	6
Три уровня требований.....	12
Требования к продукту и требования к проекту.....	14
Разработка и управление требованиями.....	16
Разработка требований.....	16
Управление требованиями.....	18
Каждый проект имеет требования.....	19
Когда плохие требования появляются у хороших людей.....	20
Недостаточное вовлечение пользователей.....	21
Небрежное планирование.....	22
«Разрастание» требований пользователей.....	22
Двусмысленные требования.....	22
Требования-«бантики».....	23
Пропущенные классы пользователей.....	23
Выгоды от высококачественного процесса разработки требований.....	24
Глава 2 Требования с точки зрения клиента.....	26
Разрыв ожиданий.....	28
Кто же клиент?.....	29
Сотрудничество клиентов и разработчиков.....	31
Билль о правах клиента ПО.....	33
Билль об обязанностях клиента ПО.....	36
Создание культуры уважения к требованиям.....	40
Определение ответственных за принятие решений.....	42
Достижение соглашения о требованиях.....	43
Базовое соглашение о требованиях.....	44
Что если не удастся достичь соглашения?.....	45
Согласование требований в проектах гибкой разработки.....	45
Глава 3 Рекомендуемые приемы формулирования требований.....	48
Каркас процесса создания требований.....	51
Выявление требований.....	54
Анализ требований.....	56
Спецификации требований.....	58
Проверка требований.....	59
Управление требованиями.....	60
Обучение.....	62

Управление проектом.....	64
Начинаем применять новые приемы.....	66
Глава 4 Бизнес-аналитик	69
Роль бизнес-аналитика.....	70
Задачи аналитика	72
Навыки, необходимые аналитику.....	73
Знания, необходимые аналитику.....	77
Становление аналитика.....	78
Бывший пользователь.....	78
Бывший разработчик или тестировщик	79
Бывший (или текущий) менеджер проекта.....	80
Специалист предметной области	80
Молодой специалист	81
Роль аналитика в проектах гибкой разработки	82
Создание дружной команды	83
ЧАСТЬ II Разработка требований	85
Глава 5 Определение бизнес-требований	86
Формулировка бизнес-требований.....	87
Определение требуемых бизнес-преимуществ.....	87
Концепция продукта и границы проекта.....	88
Противоречивые бизнес-требования	89
Документ о концепции и границах.....	91
1. Бизнес-требования.....	93
2. Рамки и ограничения проекта.....	99
3. Бизнес-контекст	101
Способы представления границ проекта	103
Контекстная диаграмма	104
Карта экосистемы	105
Дерево функций	106
Список событий.....	107
Не упускайте границы из вида.....	108
Использование бизнес-целей для принятия решений о границах проекта.....	109
Оценка эффекта от изменения границ проекта.....	110
Концепция и границы в проектах гибкой разработки.....	110
Применение бизнес-целей для определения момента завершения проекта	111
Глава 6 Как отобрать пользователей для работы над проектом	113
Классы пользователей	114
Классификация пользователей.....	115
Определение классов пользователей.....	118
Архетипы пользователей	121
Представители пользователей.....	122
Сторонник продукта.....	124
Сторонники продукта, приглашенные со стороны	125
Чего следует ожидать от сторонника продукта.....	126
На что способны несколько сторонников продукта.....	127
Как «продать» идею о необходимости привлечения сторонника продукта	129

В какие ловушки можно угодить, привлекая сторонников продукта	130
Представительство пользователей в проектах гибкой разработки.....	131
Разрешение противоречивых требований	133
Глава 7 Выявление требований	135
Методы выявления требований	137
Интервью	137
Семинары.....	139
Фокус-группы	142
Наблюдение	143
Опросные листы	145
Анализ системных интерфейсов.....	145
Анализ пользовательского интерфейса	146
Анализ документов.....	146
Планирование выявления требований в проекте	147
Подготовка к выявлению требований.....	149
Выявление требований.....	152
Действия после выявления требований	154
Организация и рассылка протоколов	154
Документирование открытых проблем.....	155
Классификация предоставляемой клиентом информации	155
Как понять, что сбор требований завершен	159
Несколько советов о том, как собирать информацию	160
Подразумеваемые и неявные требования.....	161
Поиск упущенных требований	163
Глава 8 Как понять требования пользователей.....	165
Варианты использования и сценарии использования	167
Подход с применением варианта использования продукта	171
Варианты использования и сценарии использования.....	173
Предварительные и выходные условия	175
Определение вариантов использования	183
Анализ вариантов использования.....	185
Проверка вариантов использования	187
Варианты использования и функциональные требования	188
Каких подводных рифов следует опасаться при способе с применением вариантов использования.....	190
Преимущества требований, основанных на вариантах использования	192
Глава 9 Игра по правилам.....	194
Классификация бизнес-правил	197
Факты.....	198
Ограничения.....	198
Активаторы операций.....	200
Выводы	202
Вычисления.....	202
Атомарные бизнес-правила	203
Документирование бизнес-правил.....	204
Выявление бизнес-правил.....	206
Бизнес-правила и требования	208
Сводим все воедино	210

Глава 10 Документирование требований	212
Спецификация требований к ПО	215
Требования к именованию	218
Когда информации недостаточно	221
Пользовательские интерфейсы и спецификация требований к ПО	221
Шаблон спецификации требований к ПО	223
1. Введение	225
2. Общее описание	226
3. Функции системы	227
4. Требования к данным	228
5. Требования к внешним интерфейсам	229
6. Атрибуты качества	231
7. Требования по интернационализации и локализации	233
8. [Остальные требования]	233
Приложение А. Словарь терминов	233
Приложение Б. Модели анализа	234
Спецификация требований в проектах гибкой разработки	234
Глава 11 Пишем идеальные требования	237
Характеристики превосходных требований	238
Характеристики отдельных положений спецификации требований	238
Характеристики наборов требований	240
Принципы создания требований	242
Системная или пользовательская точка зрения	243
Язык и стиль	244
Уровень детализации	247
Способы представления	248
Предотвращение неопределенности	249
Предотвращение неполноты	253
Примеры требований: до и после	255
Глава 12 Лучше один раз увидеть, чем 1024 раза услышать	260
Моделирование требований	261
От желания клиента к модели анализа	263
Выбор правильного представления	265
Диаграмма потоков данных	267
Диаграммы swimlane	272
Диаграмма переходов состояний и таблица состояний	274
Карта диалоговых окон	277
Таблицы и деревья решений	281
Таблицы событий и реакций	283
Несколько слов об UML-диаграммах	287
Моделирование в проектах гибкой разработки	288
Последнее напоминание	288
Глава 13 Определение требований к данным	290
Моделирование отношений данных	291
Словарь данных	294
Анализ данных	298
Спецификация отчетов	300
Сбор требований к отчетности	300

Особенности определения отчетов	301
Шаблон спецификации отчета	303
Панели мониторинга	305
Глава 14 Обратная сторона функциональности: атрибуты качества ПО ..	309
Атрибуты качества ПО	311
Изучение атрибутов качества	312
Определение требований по качеству	317
Внешние атрибуты качества	317
Внутренние атрибуты качества	333
Определение требований по качеству с помощью языка Planguage	340
Компромиссы для атрибутов	342
Реализация требований к атрибутам качества	344
Ограничения	345
Атрибуты качества в проектах гибкой разработки	347
Глава 15 Прототипы как средство снижения риска	350
Что такое прототип и для чего он нужен	351
Модели и экспериментальные образцы	353
Одноразовые и эволюционные прототипы	354
Бумажные и электронные прототипы	358
Работа с прототипами	360
Оценка прототипа	363
Риски создания прототипов	365
Подталкивание к выпуску прототипа	365
Отвлечение на детали	366
Нереалистичные ожидания производительности	367
Слишком много усилий на создание прототипов	367
Факторы успеха использования прототипов	367
Глава 16 Сначала о главном: определение приоритетов требований	370
Зачем определять приоритеты требований	371
Некоторые практические соображения определения приоритетов	372
Игры, в которые люди играют с приоритетами	374
Некоторые приемы определения приоритетов	376
Включать или не включать	376
Попарное сравнение и ранжирование	377
Трехуровневая шкала приоритетов	377
Схема классификации приоритетов MoSCoW	379
100 долларов	380
Определение приоритетов на основе ценности, стоимости и риска	381
Глава 17 Утверждение требований	389
Утверждение и верификация	391
Рецензирование требований	392
Процесс экспертизы	394
Контрольные списки дефектов	400
Советы по рецензированию требований	401
Сложности рецензирования требований	402
Прототипы требований	404
Тестирование требований	405
Утверждение требований с применением критериев приемки	410

Критерии приемки.....	410
Приемочные тесты.....	411
Глава 18 Повторное использование требований.....	414
Зачем нужно повторное использование требований?.....	415
Виды повторного использования требований.....	416
Объем повторного использования.....	417
Объем изменений.....	418
Механизм повторного использования.....	418
Типы информации требований, поддающихся повторному использованию	420
Популярные сценарии повторного использования.....	421
Семейства продуктов.....	421
Реинжиниринг и замена системы.....	422
Другие возможности повторного использования.....	422
Схемы требований.....	424
Средства, облегчающие повторное использование	425
Как сделать требования повторно используемыми.....	426
Препятствия и факторы успеха повторного использования требований.....	428
Препятствия для повторного использования	429
Факторы успеха повторного использования.....	430
Глава 19 От разработки требований — к следующим этапам	433
Оценка объема работ по реализации требований	434
От требований — к планам проекта	438
Оценка размера проекта и объема усилий на основе требований	438
Требования и график	441
От требований — к дизайну и коду	442
Архитектура и выделение ресурсов.....	442
Дизайн ПО.....	444
Дизайн пользовательского интерфейса	445
От требований — к тестированию.....	447
От требований — к успеху	450
ЧАСТЬ III Требования в проектах определенных классов.....	453
Глава 20 Проекты гибкой разработки (agile).....	454
Ограничения водопадной модели.....	455
Гибкий подход к разработке.....	456
Особенность гибкой разработки в применении к требованиям	457
Вовлечение клиентов.....	457
Подробнее о документации	458
Резерв проекта и расстановка приоритетов.....	458
Планирование времени.....	459
Эпики, пользовательские истории и функции	460
Расчет на неизбежные изменения	461
Адаптация приемов работы с требованиями для проектов гибкой разработки	462
Переход к гибкой разработке: что дальше?.....	463
Глава 21 Проекты по доработке или замене систем	465
Ожидаемые затруднения	466
Работа с требованиями при наличии существующей системы	467

Расстановка приоритетов на основе бизнес-целей.....	469
Осторожно с пробелами	470
Сохранение уровня производительности.....	470
Когда старых версий требований нет	471
Какие требования нужно определять	472
Как собирать требования к существующей системе.....	474
Продвижение новой системы.....	475
Уместны ли итерации.....	477
Глава 22 Проекты с серийным продуктом	478
Требования к выбору тиражируемых решений	479
Разработка пользовательских требований.....	480
Работа с бизнес-правилами	480
Определение потребностей в данных	481
Определение требований по качеству	481
Оценка решений	482
Требования к внедрению серийных решений.....	484
Требования к конфигурированию.....	485
Требования по интеграции.....	486
Требования по расширению	486
Требования к данным.....	486
Изменение бизнес-процессов	487
Обычные сложности с пакетными решениями.....	487
Глава 23 Проекты, выполняемые сторонними организациями.....	489
Необходимый уровень детализации требований.....	490
Взаимодействие заказчика и подрядчика	492
Управление изменениями.....	494
Критерии приемки	495
Глава 24 Проекты автоматизации бизнес-процессов.....	496
Моделирование бизнес-процессов	497
Использование текущих процессов для вывода требований	498
Проектирование в первую очередь будущих процессов	500
Моделирование метрик бизнес-производительности	501
Приемы, рекомендуемые к использованию в проектах автоматизации бизнес-процессов	502
Глава 25 Проекты бизнес-аналитики.....	504
Общие сведения о проектах бизнес-аналитики.....	505
Разработка требований в проектах бизнес-аналитики.....	507
Приоритизация работы на основе решений	508
Определение, как будет использоваться информация	508
Определение потребностей в данных	511
Определение аналитических операций преобразования данных	513
Эволюционная природа аналитики	515
Глава 26 Проекты встроенных и других систем реального времени ..	517
Системные требования, архитектура и назначение	518
Моделирование систем реального времени	520
Контекстная диаграмма	520
Диаграмма переходов состояний.....	521

Таблица событий и реакций	522
Архитектурная диаграмма	523
Создание прототипов.....	525
Интерфейсы.....	525
Требования к временным характеристикам	526
Атрибуты качества для встроенных систем	528
Проблемы встроенных систем	534
ЧАСТЬ IV Управление требованиями	535
Глава 27 Приемы управления требованиями к ПО	536
Процесс управления требованиями.....	537
Базовое соглашение о требованиях.....	538
Управление версиями требований.....	540
Атрибуты требований	542
Отслеживание состояния требований	544
Разрешение проблем с требованиями	546
Определение усилий, необходимых для управления требованиями.....	548
Управление требованиями в проектах гибкой разработки	549
Почему нужно управлять требованиями	551
Глава 28 Изменения случаются	552
Почему нужно управлять требованиями	553
Управление незапланированным ростом объема.....	554
Политика управления изменениями	555
Основные понятия процесса управления изменениями	556
Описание процесса управления изменениями	557
1. Назначение и границы	558
2. Роли и ответственность	558
3. Состояние запроса на изменение	559
4. Входные критерии	560
5. Задачи	560
6. Выходные критерии.....	561
7. Отчет о состоянии контроля изменений	561
Приложение. Атрибуты запросов на изменение.....	561
Совет по управлению изменениями	562
Состав совета по управлению изменениями	563
Устав совета по управлению изменениями	563
Пересмотр обязательств	565
Средства управления изменениями.....	565
Измерение активности изменений.....	566
Анализ влияния изменений	568
Процедура анализа влияния изменения	568
Шаблон отчета об анализе влияния изменения.....	571
Управление изменениями в проектах гибкой разработки	572
Глава 29 Связи в цепи требований.....	575
Отслеживание связей требований.....	575
Мотивация для отслеживаемости требований	578
Матрица отслеживаемости требований.....	580
Средства отслеживания требований.....	583
Процедура отслеживания требований	584
Осуществимость и необходимость отслеживания требований.....	586

Глава 30 Инструментальные средства разработки требований.....	589
Средства разработки требований.....	591
Средства выявления требований.....	591
Средства создания прототипов.....	592
Средства моделирования.....	592
Средства управления требованиями.....	593
Преимущества использования средств управления требованиями.....	593
Возможности средств управления требованиями.....	595
Выбор и реализация средства управления требованиями.....	598
Выбор инструментального средства.....	598
Настройка средств и процессов.....	599
Освоение средств пользователями.....	601
ЧАСТЬ V Реализация процесса построения требований.....	605
Глава 31 Совершенствование процессов работы с требованиями	606
Как требования связаны с другими составляющими проекта.....	607
Требования и различные группы заинтересованных лиц.....	610
Как добиться готовности к изменениям.....	611
Основы совершенствования процессов разработки ПО.....	613
Анализ первопричин.....	615
Цикл совершенствования технологических процессов.....	617
Оценка текущих приемов.....	617
Планирование действий по совершенствованию.....	618
Создание, проба и реализация новых технологических процессов.....	620
Оценка результатов.....	620
Документы процесса разработки и управления требованиями.....	622
Документы процесса разработки требований.....	623
Документы процесса управления требованиями.....	624
Достигнута ли цель?.....	625
Дорожная карта совершенствования работы с требованиями.....	628
Глава 32 Требования к ПО и управление рисками	630
Основы управления рисками при создании ПО.....	631
Составляющие управления рисками.....	632
Документирование рисков проекта.....	633
Планирование управления рисками.....	635
Риск, связанный с требованиями.....	636
Выявление требований.....	637
Анализ требований.....	639
Спецификация требований.....	639
Утверждение требований.....	640
Управление требованиями.....	640
Управление рисками — ваш друг.....	641
Приложение А.....	643
Приложение Б	651
Приложение В	678
Словарь терминов	707
Об авторах	718

Введение

Несмотря на более чем пятидесятилетнее существование компьютерной отрасли, многие компании-разработчики программного обеспечения по-прежнему прикладывают значительные усилия для сбора, документирования и управления требованиями к ПО. Недостаточный объем информации, поступающей от пользователей, требования, сформулированные не полностью, их кардинальное изменение и неправильно понятые бизнес-цели — вот основные причины, из-за которых зачастую командам, работающим в области информационных технологий, не удается успешно завершить проект. Многие разработчики не умеют спокойно и профессионально собирать требования пользователей к ПО. У клиентов зачастую не хватает терпения участвовать в разработке требований к ПО. Иногда участники проекта даже не могут прийти к единому мнению, что же такое «требование». Как заметил один писатель, «программисты скорее предпочтут расшифровать слова классической песни Кингсмена (Kingsmen) «Louie Louie», чем требования клиентов» (Peterson 2002).

Второе издание книги «Разработка требований к программному обеспечению» вышло за 10 лет до выхода этой книги. В мире информационных технологий это большой срок. Многое поменялось за это время, но кое-что осталось неизменным. Вот основные тенденции прошедшего десятилетия:

- признание бизнес-анализа профессиональной дисциплиной и расширение профессиональной сертификации и организаций, таких как International Institute of Business Analysis и International Requirements Engineering Board;
- достигли высокого уровня развития средства для управления требованиями в базах данных и для поддержки разработки требований, в том числе для создания прототипов, моделирования и имитации;
- распространение методов гибкой разработки (agile) и развитие приемов работы с требованиями в проектах гибкой разработки;
- активное использование визуальных моделей для представления знаний о требованиях.

Так что же *не* изменилось? Есть два обстоятельства, которые делают этот раздел важным и уместным. Во-первых, во многих программах обучения разработке ПО и вычислительным системам все еще недостаточно внимания уделяется важности разработки требований (которая подразумевает как

собственно разработку, так и управление требованиями). Во-вторых, многие из тех, кто работает в области ПО, слишком увлечены техническими и процессными решениями наших задач. Мы иногда забываем, что выявление требований — и большая часть работы в проектах разработки ПО и систем вообще — основаны в первую очередь на взаимодействии людей. Не появилось никаких новых магических приемов для автоматизации этой деятельности, хотя на рынке имеются инструменты, позволяющие эффективно взаимодействовать географически распределенным людям.

Мы полагаем, что представленные во втором издании приемы разработки и управления требованиями по-прежнему работают и применимы к обширному множеству проектов по разработке ПО. Изобретательный бизнес-аналитик, менеджер или владелец продукта будет разумно адаптировать и масштабировать эти приемы в соответствии с особенностями конкретной ситуации. Новым в этом, третьем издании является глава о работе с требованиями в проектах гибкой разработки и разделы во многих других главах, описывающие, как применять и адаптировать описываемые в этих главах приемы при гибкой разработке.

Разработка ПО включает по крайней мере столько же общения, сколько и обычная работа с компьютером, но зачастую мы делаем акцент на работе с компьютером и не уделяем достаточно внимания общению. В этой книге описаны десятки способов, позволяющих реализовать это общение и помогающих разработчикам ПО, менеджерам, маркетологам и клиентам использовать на практике эффективные методы разработки требований к ПО. В книге описаны основные «отличные способы» формулирования требований, а не экзотичные или тщательно разработанные методологии, обещающие решить все проблемы, возникающие при формулировании требований. В многочисленных врезках я привожу реальные примеры, иллюстрирующие типичные ситуации, возникающие при формулировании требований.

С тех пор как в 1999 году вышло первое издание этой книги, каждый из авторов поработал над многочисленными проектами и провел сотни семинаров, посвященных требованиям к ПО для сотрудников коммерческих и правительственных организаций всех толков. Мы поняли, что эти способы годятся практически для любого проекта: маленьких и крупномасштабных, предусматривающих разработку новых программ и расширение уже имеющихся, силами локальных и распределенных команд и с использованием традиционных и гибких методов разработки. Кроме того, эти приемы не ограничиваются только областью разработки ПО, и вполне годятся для проектирования оборудования и систем. Как и в случае с любыми другими способами разработки ПО, чтобы понять, какие из способов формулирования требований подходят вам более всего, следует руководствоваться здравым смыслом и опираться на собственный опыт. Используйте описанные здесь приемы как инструменты, помогающие гарантировать, что вы эффективно общаетесь с нужными людьми в своих проектах.

Что дает эта книга

Из всех возможных способов совершенствования процесса разработки ПО наибольшее преимущество за формулированием требований. Мы описываем проверенные на практике способы, которые помогут вам:

- повысить качество требований к проекту на ранней стадии цикла разработки, что позволит снизить число доработок и повысить производительность;
- предоставлять высококачественные информационные системы и коммерческие продукты, которые решают поставленные задачи;
- управлять расползанием границ проекта и изменениями требований, обеспечивая контролируемость и отсутствие отклонений от цели;
- повысить удовлетворенность клиентов;
- снизить затраты на обновление, обслуживание и поддержку.

Наша цель — помочь вам усовершенствовать процессы выявления и анализа требований, написания и оценки спецификаций требований и управления требованиями на протяжении всего цикла разработки продукта. Описываемые нами приемы прагматичны и реалистичны. Мы использовали их неоднократно и всегда получали хорошие результаты.

Кому адресована эта книга

Она для тех, кто так или иначе вовлечен в сбор и формулирование требований к программному продукту. Основная аудитория — бизнес-аналитики и разработчики в проекте разработки независимо от того, работают они полный день или привлекаются к проекту время от времени. Более широкий круг читателей — архитекторы, дизайнеры, программисты, тестировщики ПО и другие члены команды, задача которых понять и удовлетворить чаяния клиентов, а также маркетологи и менеджеры по продуктам, которые должны проникнуться «духом» и особенностями продукта, чтобы сделать его в полной мере конкурентоспособным. Менеджеры проектов, отвечающие за своевременный выпуск, также найдут здесь для себя немало интересного: они узнают, как управлять показателями, связанными с требованиями к проекту, а также реагировать на изменение этих требований. Еще один сегмент аудитории — заинтересованные лица, участвующие в определении продукта, который соответствует их функциональным и бизнес-нуждам, а также потребностям в области качества. Настоящая книга поможет конечным пользователям, клиентам, создающим или заказывающим разработку ПО, а также многим другим заинтересованными лицам понять важность процесса формулирования требований и свое место в этом процессе.

Структура книги

Книга состоит из пяти частей. Первая часть «Требования к ПО: что, почему и кто» начинается с ряда определений. Если вы занимаетесь техническими вопросами, надеюсь, вы поделитесь концепциями главы 2, посвященной взаимодействию клиентов и разработчиков, со своими ключевыми клиентами. В главе 3 описано несколько десятков рекомендованных приемов создания и управления требованиями, а также процесс формулирования требований в целом. Глава 4 посвящена роли бизнес-аналитика.

Вторая часть «Разработка требований» начинается с описания способов для определения бизнес-требований в проекте. Другие главы этой части посвящены тому, как правильно выбрать представителей клиента, узнать у них требования и задокументировать пользовательские требования, бизнес-правила, функциональные требования, требования к данным и нефункциональные требования. В главе 12 обсуждается несколько моделей анализа, представляющих требования с разных точек зрения и дополняющих текстовую информацию, в главе 15 — применение прототипов ПО, позволяющих снизить риск выпуска продукта ненадлежащего качества. Прочие главы посвящены расстановке приоритетов, утверждению и оценке требований. Завершается вторая часть описанием, как требования влияют на другие аспекты работы над проектом.

Третья часть содержит совершенно новый материал, содержащий рекомендации по работе с требованиями в различных классах проектов: проектов гибкой разработки, проектов по доработке или замене систем, проектов, в которых используются готовые серийные пакетные решения, проектов, выполняемых сторонними организациями, проектов автоматизации бизнес-процессов, а также проектов встроенных и других систем реального времени.

В четвертой части «Управление требованиями» речь пойдет о принципах и способах управления требованиями, особое внимание уделяется технологиям, позволяющим реагировать на изменение требований. В главе 29 рассказано, как отслеживать изменение требований с самого начала до их реализации в продукте. Четвертая часть завершается описанием серийных утилит, расширяющих возможности разработки и управления требованиями к проекту.

Заключительная часть книги «Реализация процесса построения требований» поможет вам перейти от теории к практике. В главе 31 рассказано, как включить новые способы формулирования требований в имеющийся процесс разработки. Глава 32 посвящена рискам, связанным с требованиями к проекту. Приложение А позволит вам оценить применяемые способы формулирования требований и определить области, требующие совершенствования. В прочих приложениях представлены руководство по устранению проблем, возникающих при формировании требований, а также примеры задокументированных требований к проектам, чтобы вы могли увидеть, как это все работает на деле.

Примеры из практики

Методы, описанные в книге, мы иллюстрируем примерами, за основу которых взяты реальные проекты, это касается и информационной системы среднего размера под названием Chemical Tracking System. Не волнуйтесь — чтобы разобраться в этом проекте, знание химии вам не потребуется. Диалоги участников проектов из примеров разбавляют текст книги. Думаю, вне зависимости от того, что за ПО разрабатывает ваша команда, вы найдете эти диалоги интересными и полезными.

От принципов — к практике

Трудно представить, сколько энергии потребуется на преодоление препятствий, мешающих изменению и практическому применению новых знаний. Чтобы облегчить вам применение новых знаний на практике, каждую главу я заканчиваю разделом «Что дальше?», где перечислены конкретные действия, которые вы можете выполнить в отношении реального проекта. В разных главах есть шаблоны документов для определения требования, контрольные списки проверки, таблица приоритетов требований, описание процесса «изменение-контроль», а также полезные вещи для многих других процессов. Все это можно загрузить с веб-сайта этой книги по адресу: <http://aka.ms/SoftwareReq3E/files>.

Используйте эти материалы, чтобы применять полученные значения на практике. Начните с небольших усовершенствований, но именно сегодня, не откладывая на завтра.

Отдельные участники проекта с большой неохотой возьмутся за применение новых приемов работы с требованиями. Используйте материал книги для обучения коллег, клиентов и менеджеров. Напоминайте им о связанных с требованиями проблемах, возникавших при работе над предыдущими проектами, и обсуждайте потенциальные преимущества использования новых способов.

Не обязательно запускать новый проект, чтобы начать применять усовершенствованные способы формулирования требований. В главе 21 рассказывается, как использовать описываемые в книге способы в проектах по улучшению или замене существующей системы. Постепенное внедрение новых способов — сопряженный с незначительным риском подход к совершенствованию процесса разработки, который подготовит вас к использованию новых способов при работе над следующим крупным проектом.

Цель разработки требований — накопить требования, которые позволят проектировать приложения с приемлемым уровнем риска. Потратив на это достаточно времени, вы можете быть уверены, что свели к минимуму риск переделки продукта, создания негодного ПО или срыва сроков сдачи проекта. Эта книга научит вас, как объединить усилия нужных людей для разработки качественных требований к нужному продукту.

Часть I

Требования к ПО: что, почему и кто

Глава 1	Основы разработки требований к ПО.....	2
Глава 2	Требования с точки зрения клиента.....	26
Глава 3	Рекомендуемые приемы формулирования требований	48
Глава 4	Бизнес-аналитик	69

Глава 1

Основы разработки требований к ПО

«Привет, Фил, это Мария из отдела персонала. У нас проблема с системой управления персоналом, которую вы разрабатывали для нас. Одна из сотрудниц изменила имя на Спаркл Старлайт, а система не принимает это изменение. Ты можешь помочь?»

«Она вышла замуж за парня по имени Старлайт?»

«Нет, она просто взяла другое имя, — ответила Мария. — В этом-то и проблема. Похоже, что мы можем менять имя только при изменении семейного положения».

«Да, я не предусмотрел такой возможности. Я не помню, чтобы и ты говорила мне о ней, когда мы обсуждали систему», — сказал Фил.

«Я полагала, ты знаешь, что люди могут законным образом менять имена, когда захотят, — ответила Мария. — Мы должны исправить это к пятнице, а то Спаркл не сможет обналчить чек. Ты сможешь исправить этот дефект к этому сроку?»

«Это не дефект, — резко парировал Фил. — Я и не знал, что вам нужна эта функциональность. Сейчас я занимаюсь новой системой оценки производительности системы, и скорее всего смогу исправить это в конце месяца, но не к пятнице. Приношу свои извинения. Но в следующий раз с такими просьбами обращайся пораньше и, пожалуйста, излагай их письменно».

«А что же я скажу Спаркл? — возмутилась Мария. — Она сильно расстроится, если не сможет обналчить чек».

«Эй, Мария, это не моя вина, — запротестовал Фил. — Если бы ты сказала мне с самого начала, что вам понадобится изменять имена в любое время, этого не произошло бы. Я не медиум и читать твои мысли не умею».

Сердитая и смирившаяся, Мария отрезала: «Это как раз то, из-за чего я ненавижу компьютерные системы. Позвони мне, как только сможешь исправить недостаток».

Если вы когда-либо бывали в шкуре клиента на переговорах, подобных этому, то знаете, как расстраиваются пользователи продукта, не позволяющего решать элементарные задачи. Не очень приятно находится в милости разработчиков, которые *может быть* удовлетворят ваш запрос. Разработчики тоже не очень довольны, когда узнают о функциональности, которую пользователи ожидали получить, только после развертывания системы.

Масса проблем с ПО возникает из-за несовершенства способов, которые люди применяют для сбора, документирования, согласования и модификации требований к ПО. Как в случае с Филом и Марией, проблемы могут возникать из-за неформального сбора информации, предполагаемой функциональности, ошибочных или несогласованных предположений, недостаточно определенных требований и бессистемного изменения процесса. Многие исследования показывают, что на ошибки, внесенные на этапе сбора требований, приходится от 40 до 50% всех дефектов, обнаруженных в программном продукте (Davis, 2005). Некорректные сведения от пользователей и недостатки определения и управления требованиями клиентов — основные причины провалов проектов. Но невзирая на эти сведения многие организации все еще применяют неэффективные методы сбора требований.

Нигде более, как на стадии сбора требований так тесно не связаны интересы всех заинтересованных в проекте лиц с успехом проекта. (Подробнее о заинтересованных лицах см. главу 2.) К заинтересованным в проекте лицам относятся клиенты, пользователи, бизнес-аналитики, разработчики и многие другие. Хорошо справившись с этой стадией процесса, вы получите отличный продукт, восхищенных заказчиков и удовлетворенных разработчиков. В противном случае вам грозит непонимание, разочарование и разногласия, которые подрывают веру в продукт и его ценность. Так как требования представляют собой основу для действий и разработчиков и менеджеров проекта, все заинтересованные в проекте лица должны быть вовлечены в создание этого документа.

Но разработка требований и управление ими — трудный процесс! Здесь нет быстрых или волшебных решений. Однако многие организации борются с одними и теми же трудностями, для преодоления которых мы можем найти общие приемы, годные в различных ситуациях. В этой книге описаны десятки таких приемов. Их рекламируют как средства построения абсолютно новых систем, однако они отлично подходят для улучшения, замены и реинжиниринга проектов (см. главу 21) и выбора коммерческих готовых решений (см. главу 22). Командам, которые выбирают итеративный процесс, следуя методике гибкой разработки (agile), необходимо знать, что же делать на каждом этапе (см. главу 20).

Из этой главы вы узнаете, как:

- разобраться с ключевыми терминами, применяемыми при сборе требований к ПО;
- различать требования к *продукту* и к *проекту*;
- различать требования по *разработке* и *управлению*;
- обнаруживать тревожные симптомы некоторых связанных с требованиями проблем, которые могут иногда возникать.

Внимание! Мы используем термины «система», «продукт», «приложение» и «решение» для определения любого вида ПО или содержащего ПО компонента, который создается для внутрикорпоративного использования, на продажу или по заказу.

Проверка текущего состояния дел с требованиями

Чтобы проверить текущую ситуацию с требованиями в вашей организации, посмотрите сколько из следующих условий выполняются в вашем последнем проекте. Если в вашей ситуации применимы три или четыре элемента, эта книга для вас:

- Бизнес-цели, концепция и границы вашего проекта никогда четко не определялись.
- У клиентов никогда не хватало времени на работу над требованиями с аналитиками или разработчиками.
- Ваша команда не может напрямую взаимодействовать с непосредственными пользователями, чтобы разобраться с их потребностями.
- Клиенты считают все требования критически важными, поэтому они не разбивают их по приоритетам.
- В процессе работы над кодом разработчики сталкиваются с многозначностью и отсутствием информации, поэтому им приходится догадываться о многих вещах.
- Общение между разработчиками и заинтересованными лицами концентрируется на окнах и функциях интерфейса, а не на задачах, которые пользователи будут решать с использованием программного продукта.
- Ваши клиенты никогда не одобряли требования.
- Ваши клиенты одобрили требования к выпуску или итерации, после чего продолжили вносить в них изменения.
- Область действия проекта увеличилась вместе с принятием изменений в требования, но сроки были нарушены из-за отсутствия дополнительных ресурсов и отказа от удаления части функциональности.
- Затребованные изменения требований были утеряны, и никто не знает состояние запроса на указанные изменения.
- Клиенты запросили определенную функциональность и разработчики реализовали ее, но никто ее не использует.
- В конце проекта спецификация была выполнена, но бизнес-задачи не были выполнены, и клиент не удовлетворен.

Определение требований к ПО

Когда группа людей начинает обсуждать требования, они обычно начинают с проблемы терминологии. Разные эксперты, говоря об одном и том же документе, могут называть его пользовательскими требованиями, требованиями к ПО, бизнес-требованиями, функциональными требованиями, системными требованиями, требованиями к продукту или проекту, пользовательской точкой зрения, функцией или ограничением. Имена, которые они применяют к различным требованиям, также различаются. Заказчики зачастую считают, что определенные ими требования — это высокоуровневая концепция про-

дукта, предназначенная для разработчиков. Те, в свою очередь, полагают, что в отношении клиентов это детализированная разработка интерфейса пользователя. Такое многообразие ведет к сумятице и раздражающим проблемам коммуникации.

Особенности интерпретации требований

Десятки лет спустя после появления программирования для компьютеров специалисты по ПО все еще увлеченно спорят о том, что из себя представляет *требование*. Мы не станем участвовать в этих дебатах, а просто представим ряд определений, которые мы считаем полезными.

Консультант Брайан Лоренс предположил, что *требования* — это «нечто такое, что определяет выбор дизайнера» (Lawrence, 1997). Это неплохое неформальное определение, потому что в эту категорию можно отнести много видов информации. И, в конце концов, весь смысл разработки требований заключается в принятии соответствующих проектировочных решений, которые в конечном итоге должны удовлетворить клиента. Другое определение требования заключается в том, что продукт должен обеспечивать выгоду заинтересованному лицу. Тоже неплохо, но не очень точно. Наше любимое определение сформулировано Иеном Соммервиллем и Питом Сойером (Ian Sommerville и Pete Sawyer, 1997):

Требования это спецификация того, что должно быть реализовано. В них описано поведение системы, свойства системы или ее атрибуты. Они могут служить ограничениями в процессе разработки системы.

Это определение учитывает самые разные типы информации, которые в совокупности называются *требованиями*. Требования охватывают как видение пользователя, так и внешнее поведение системы, а также представление разработчика о некоторых внутренних характеристиках. Они включают как поведение системы в определенных условиях, так и свойства, которые делают систему полезной и даже доставляющей удовольствие конечным операторам.

Внимание! Не рассчитывайте, что все заинтересованные лица вашего проекта разделяют общее представление о том, что же такое требования. Задайте определения с самого начала, чтобы вы все говорили на одном языке.

Определение термина «требование» в словаре

Специалисты по ПО используют термин «требование» не совсем в том же смысле, что и его значение в словаре, то есть как что-то требуемое и обязательное, потребность или необходимость. Люди иногда задаются вопросом, стоил ли вообще приоритизировать требования, ведь низкоприоритетные требования никогда не реализуются. Если это не очень нужная вещь, то ее и требованием-то назвать сложно. Возможно, но как тогда называть эту информацию? Если перенести требование из нынешнего проекта в неопределенный будущий выпуск, то можно ли его все равно считать требованием? Конечно да.

Требования к ПО включают измерение времени. Они могут относиться к настоящему времени — в этом случае они описывают текущие функции системы. Или могут относиться к ближайшему (высокоприоритетные), среднему (среднеприоритетные) или гипотетическому (низкоприоритетные) будущему. Они могут даже относиться к прошлому времени, когда описывают запросы, которые были ранее определены, а затем отброшены. Не стоит тратить время на споры является ли что-то требованием или нет, даже если вы знаете, что оно скорее всего не будет реализовано по какой-то серьезной причине. Это требование и точка.

Уровни и типы требований

Из-за такого большого числа разных типов информации требований нам требуется единый набор описаний, изменяющих слишком перегруженный термин *требование*. В этом разделе приводятся определения, которые будут использоваться для терминов, наиболее часто применяемых у такой сфере, как разработка требований (см. табл. 1-1).

Табл. 1-1. Информация о некоторых типах требований

Понятие	Определение
Бизнес-требование	Высокоуровневая бизнес-цель организации или заказчиков системы
Бизнес-правило	Политика, предписание, стандарт или правило, определяющее или ограничивающее некоторые стороны бизнес-процессов. По своей сути это не требование к ПО, но оно служит источником нескольких типов требований к ПО
Ограничение	Ограничение на выбор вариантов, доступных разработчику при проектировании и разработке продукта
Внешнее требование к интерфейсу	Описание взаимодействия между ПО и пользователем, другой программной системой или устройством
Характеристика	Одна или несколько логически связанных возможностей системы, которые представляют ценность для пользователя и описаны рядом функциональных требований
Функциональное требование	Описание требуемого поведения системы в определенных условиях
Нефункциональное требование	Описание свойства или особенности, которым должна обладать система, или ограничение, которое должна соблюдать система
Атрибут качества	Вид нефункционального требования, описывающего характеристику сервиса или производительности продукта
Системное требование	Требование верхнего уровня к продукту, состоящему из многих подсистем, которые могут представлять собой ПО или совокупность ПО и оборудования
Пользовательское требование	Задача, которую определенные классы пользователей должны иметь возможность выполнять в системе, или требуемый атрибут продукта

Требования к ПО состоят из трех уровней — бизнес-требования, пользовательские и функциональные требования. Вдобавок в каждой системе есть свои нефункциональные требования. Модель на рис. 1-1 иллюстрирует способ представления этих типов требований. Как гласит знаменитое высказывание статистика Джорджа Е. П. Бокса (George E. P. Box), «В конечном счете все модели врут, но некоторые оказываются полезными» (Box и Drgar, 1987). Это, конечно же, применимо к рис. 1-1. Как и все модели, она не полная, но схематично показывает организацию требований.

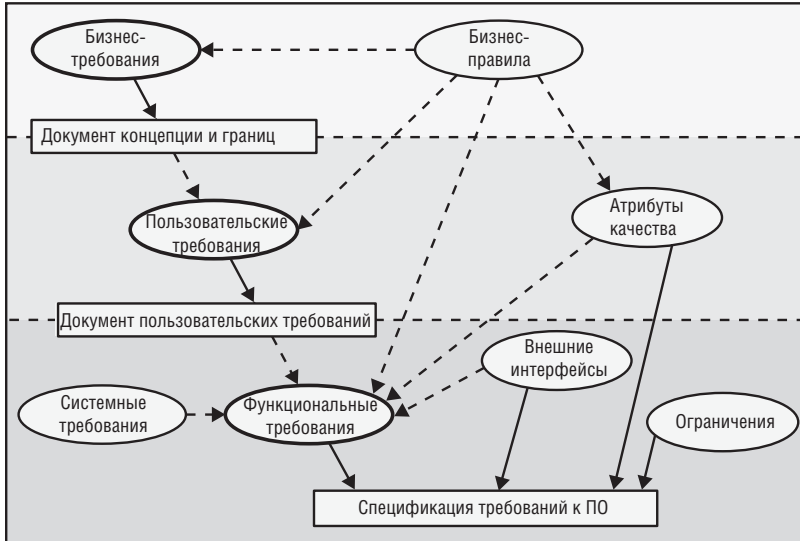


Рис. 1-1. Взаимосвязи нескольких типов информации для требований. Сплошные линии означают «содержатся в», а пунктирные — «являются отправной точкой» или «влиют на»

Внимание! Хотя мы в этой книге и называем требования «документами», как на рис. 1-1, это не обязательно традиционные бумажные или электронные документы. Их можно считать просто контейнерами, в которых хранится знание о требованиях. Такой контейнер может быть традиционным документом или же электронной таблицей, набором диаграмм, базой данных, средством управления требованиями или сочетанием всех этих артефактов. Для удобства мы будем называть документом любой такой контейнер. Мы предоставим шаблоны, которые определяют типы информации, которые стоит хранить в каждой из таких группировок независимо от того, в какой форме они хранятся. Как называть такие артефакты не так важно, как добиться согласия в компании относительно их имен, какие виды информации в них содержатся, а также как эта информация организована.

Овалы обозначают типы информации требований, а прямоугольники — документы, в которых хранится эта информация. Сплошные линии указывают, что в указанном документе хранится информация определенного типа. (Бизнес-правила и системные требования хранятся отдельно от требований к ПО, обычно соответственно в каталоге бизнес-правил или в спецификации системных требований.) Пунктирная линия указывает, что информа-

ция одного типа является источником или влияет на информацию другого типа или на требование. На этой схеме не показаны требования к данным. Функции манипулируют данными, поэтому требования к данным могут присутствовать на всех трех уровнях. В главе 7 приводится много примеров различных типов информации требований.

Бизнес-требования (business requirements) описывают, *почему* организации нужна такая система, то есть цели, которые организация намерена достичь с ее помощью. Основное их содержание — бизнес-цели организации или клиента, заказывающих систему. Допустим, что авиакомпания хочет на 25% снизить затраты на сотрудников у стойки в аэропорту. В процессе достижения этой цели может возникнуть идея поставить терминал, который пассажиры будут использовать для самостоятельной регистрации в аэропорту. Как правило, бизнес-требования высказывают те, кто финансируют проект, покупатели системы, управляющий реальными пользователями, отдел маркетинга или ответственный за концепцию продукта. Мне нравится записывать бизнес-требования в форме документа *о концепции и границах* (vision and scope document). К другим руководящим документам, которые еще иногда используют в этом качестве, относят устав проекта (project charter), вариант использования (*business case*) или документ рыночных требований (market requirements document). Определение бизнес-требований описано в главе 5. Для целей этой книги мы предполагаем, что бизнес-потребность (business need) или рыночная возможность (market opportunity) уже определена.

Пользовательские требования (user requirements) описывают цели или задачи, которые пользователи должны иметь возможность выполнять с помощью продукта, который в свою очередь должен приносить пользу кому-то. Область пользовательских требований также включает описания атрибутов или характеристик продукта, которые важны для удовлетворения пользователей. К отличным способам представления этого вида требований относятся варианты использования (Kulak и Guiney, 2004), пользовательские истории (Cohn, 2004) и таблицы «событие — отклик». В идеале эту информацию предоставляют реальные представители пользователей. Пользовательские требования описывают, *что* пользователь должен иметь возможность делать с системой. Примером сценария использования является регистрация на рейс с использованием веб-сайта или терминала в аэропорту. Будучи сформулированным в виде пользовательской истории, то же пользовательское требование может звучать так: «Как пассажир я хочу зарегистрироваться на рейс, чтобы можно было сесть на самолет». Важно помнить, что в большинстве проектов есть несколько классов пользователей, а также других участников, потребности которых тоже надо выявить. Этот уровень модели описывается в главе 8. Некоторые люди используют более широкий термин «требования заинтересованных лиц» для обозначения того факта, что требования будут предоставляться различными заинтересованными лицами, отличными от прямых пользователей системы. Это конечно же верно, но на данном уровне нам нужно понять, что реальные пользователи хотят достичь с помощью продукта.

Функциональные требования (functional requirements) определяют, каким должно быть поведение продукта в тех или иных условиях. Они определяют, что разработчики должны создать, чтобы пользователи смогли выполнить свои задачи (пользовательские требования) в рамках бизнес-требований. Такое соотношение между тремя уровнями требований жизненно важно для успеха проекта. Функциональные требования описываются в форме традиционных утверждений со словами «должен» или «должна»: «У пассажира должна быть возможность распечатать посадочные талоны на все рейсы, на которые он зарегистрировался» или «Если в профиле пассажира не указаны предпочтения по выбору места, система резервирования должна сама назначить ему место».

Бизнес-аналитик¹ документирует функциональные требования в *спецификации требований к ПО* (software requirements specification, SRS), где описывается так полно, как необходимо, ожидаемое поведение системы. Спецификация требований к ПО используется при разработке, тестировании, гарантии качества продукта, управлении проектом и в связанных с проектом функциях. Этот артефакт называют по-разному: документ бизнес-требований, функциональная спецификация, документ требований и т. п. Спецификация требований к ПО может представлять собой отчет, сгенерированный на основе информации, хранимой в средстве управления требованиями. Так как этот термин принят в отрасли, мы будем в этой книге использовать обозначение «SRS» (ISO/IEC/IEEE 2011). Подробнее об SRS см. главу 10.

Термин *системные требования* (system requirements) описывает требования к продукту, который содержит многие компоненты или подсистемы (ISO/IEC/IEEE 2011). В этом смысле «система» это не просто любая информационная система. Говоря о системе, мы подразумеваем программное обеспечение или подсистемы ПО и оборудования. Люди и процессы тоже часть системы, поэтому определенные системные функции могут распространяться и на людей. Некоторые используют термин «системные требования» для обозначения подробных требований к программной системе, но в этой книге мы не используем этот термин в таком смысле.

Хороший пример «системы» — рабочее место кассира в супермаркете. В нем есть сканер штрих-кода, интегрированный с весами, а также ручной сканер штрих-кода. У кассира есть клавиатура, монитор и выдвижной ящик-касса. Есть также устройство чтения кредитных карточек и клавиатура для ввода ПИН-кода и чтения карт постоянного покупателя. На рабочем месте может быть до трех принтеров: для печати чека, квитанции кредитной карты и купонов на скидку. Все эти устройства взаимодействуют под управлением программного обеспечения. На основе требований к системе или продукту в целом бизнес-аналитик формулирует конкретную функциональность, кото-

¹ «Бизнес-аналитик» — это роль в проекте, которая прежде всего отвечает за действия по работе с требованиями в проекте. У роли бизнес-аналитика есть и другие имена. Подробнее о роли бизнес-аналитика см. главу 4.

рую должны поддерживать тот или иной компонент или подсистема, а также интерфейсы взаимодействия между ними.

Бизнес-правила (business rules) включают корпоративные политики, правительственные постановления, отраслевые стандарты и вычислительные алгоритмы. Как вы увидите в главе 9, бизнес-правила не являются сами требованиями к ПО, потому что они находятся за пределами любой системы ПО. Однако они часто налагают ограничения, определяя, какими функциями должна обладать система, подчиняющаяся соответствующим правилам. Иногда, как в случае с корпоративными политиками безопасности, бизнес-правила становятся источником атрибутов качества, которые реализуются в функциональности. Следовательно, вы можете отследить происхождение конкретных функциональных требований вплоть до соответствующих им бизнес-правил.

В дополнение к функциональным требованиям спецификация SRS содержит нефункциональные. *Атрибуты качества* (quality attributes) еще называют параметрам качества, требованиями по уровню обслуживания и т. п. Они представляют собой описание различных измерений характеристик продукта, которые важны для пользователей или для разработчиков и тех, кто будет обслуживать систему, таких как производительность, доступность и переносимость. Другие нефункциональные требования описывают *внешние интерфейсы* между системой и внешним миром. Речь идет о подключениях к другим программным системам, аппаратным устройствам и пользователям, а также коммуникационные интерфейсы. *Ограничения* (constraints) проектирования и реализации накладывают границы на возможности выбора разработчика при проектировании продукта.

Если они нефункциональные, то что они из себя представляют?

Многие годы требования к программным продуктам в целом классифицировались как функциональные и нефункциональные. С функциональными требованиями все ясно: они описывают наблюдаемое поведение системы в различных обстоятельствах. Но многим не нравится термин «нефункциональные». Это прилагательное говорит, чем *не являются* требования, но не говорит, чем они *являются*. Мы понимаем проблему, но не можем предложить идеального решения.

Требования, отличные от функциональных, могут описывать не *что* система делает, а *как хорошо* она это делает. Они могут описывать важные характеристики или свойства системы. К ним относятся доступность, легкость и простота использования, производительность и другие характеристики системы, как описано подробнее в главе 14. Некоторые люди считают нефункциональные требования тем же, что атрибуты качества, но это слишком узкое понимание. Например, ограничения проекта или реализации также являются нефункциональными требованиями, как требования внешних интерфейсов.

Другие нефункциональные требования описывают среду, в которой работает система, например платформу, переносимость, совместимость и ограничения. Многие продукты также должны подчиняться определенным правилам, требованиям регулирующих органов или требовать сертификации. Такими могут быть требования по локализации для продуктов, в которых должны учитываться региональные стандарты, языки, законы, валюты, терминология, орфография и другие характеристики пользователей. Хотя такие требования определяются с использованием нефункциональной терминологии, бизнес-аналитик на их основе может определить много функций, чтобы система обладала всеми необходимыми свойствами и вела себя соответствующим образом в разных ситуациях.

Несмотря на описанные ограничения, в этой книге мы будем придерживаться термина «нефункциональные требования» за неимением подходящего всеобъемлющего альтернативного термина. Не надо волноваться о точности названия всей подобной информации — лучше позаботьтесь, чтобы она вошла в ваши действия по выявлению и анализу требований. Можно создать продукт, обладающий всей требуемой функциональностью, но пользователи могут невзлюбить его за то, что тот не соответствует их (обычно невысказанным) ожиданиям по качеству.

Характеристика (feature) — это набор логически связанных функциональных требований, которые представляют ценность для пользователя и удовлетворяют бизнес-цели. Желательные характеристики продукта, которые перечисляет клиент, не эквивалентны тем, что входят в список необходимых для решения задач пользователей. В качестве примеров характеристик продуктов можно привести избранные страницы или закладки веб-браузера, средства проверки орфографии, запись макрокоманды, автоматическое обновление определений вирусов в антивирусной программе. Характеристики могут охватывать множество пользовательских требований, и для каждого варианта необходимо, чтобы множество функциональных требований было реализовано для выполнения пользователем его задач. Рис. 1-2 иллюстрирует *дерево функций (feature tree)* — модель анализа, которая показывает, как функцию можно разложить на иерархию более мелких функций, которые связаны с конкретными пользовательскими требованиями и ведут к определению наборов функциональных требований (Beatty и Chen, 2012).

Чтобы вы лучше восприняли некоторые из различных видов требований, проанализируем проект по разработке следующей версии текстового редактора. Бизнес-требование может звучать так: «Увеличить продажи за пределами США на 25% за следующие полгода». В отделе маркетинга выяснили, что в продуктах-конкурентах есть только англоязычные средства проверки орфографии, поэтому было принято решение включить возможность проверки орфографии на других языках. Соответствующие требования пользователей могут содержать задачи вроде такой: «Выберите язык проверки орфографии», «Найдите орфографические ошибки» или «Добавьте слово в словарь».

Проверка грамматики имеет множество индивидуальных функциональных требований, которые имеют дело с такими операциями, как поиск и выделение слов с ошибками, автоисправление, отображение вариантов замены и замена слова с ошибкой корректным вариантом по всему тексту. Требования по *удобству использования* (usability) определяют, как нужно локализовать ПО для использования с различными языками и наборами символов.

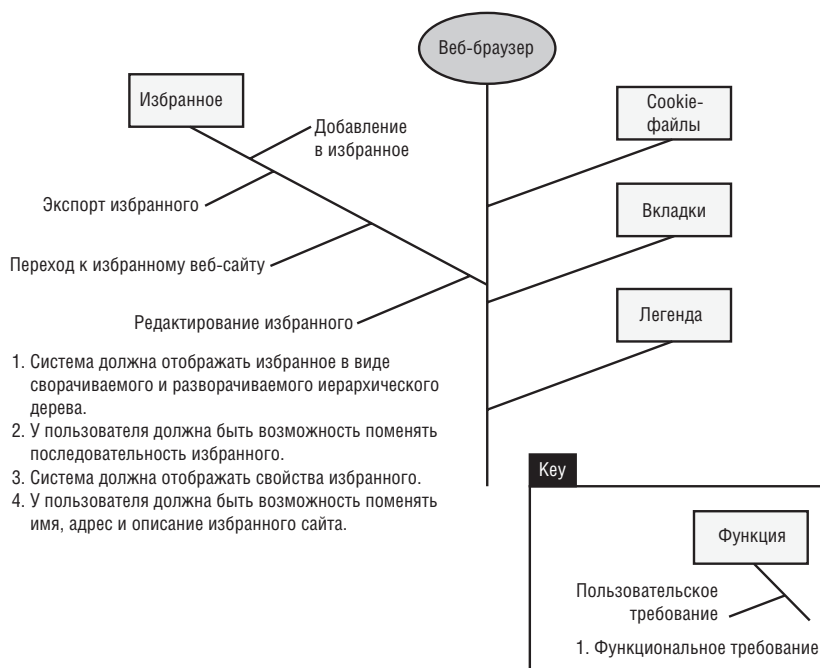


Рис. 1-2. Взаимоотношения между функциями и пользовательскими и функциональными требованиями

Три уровня требований

На рис. 1-3 показано, как различные заинтересованные лица могут участвовать в выявлении трех уровней требований. В разных организациях используют разные имена для ролей, участвующих в этой деятельности; подумайте, кто выполняет эту работу в вашей организации. Имена ролей часто различаются в зависимости от того, является ли подразделение, разрабатывающее продукт, внутренним отделом организации или компанией, создающей ПО для коммерческого использования.

На основе выявленной бизнес-потребности, требования рынка или интересной концепции нового продукта менеджеры и сотрудники отдела маркетинга определяют бизнес-требования для ПО, которые помогут компании работать эффективнее (для информационных систем) или успешно конкурировать на рынке (для коммерческих продуктов). В корпоративной среде

после этого аналитики обычно работают с представителями пользователей для определения пользовательских требований. В компаниях, разрабатывающих коммерческие продукты, часто назначают менеджера продукта, который определяет, какие функции должны включаться в новый продукт. Каждое пользовательское требование должно быть сопоставлено бизнес-требованию. На основе пользовательских требований аналитик или менеджер продукта определяет функции, которые дадут возможность пользователям выполнять их задачи. Разработчикам необходимы функциональные и нефункциональные требования, чтобы создавать решения с желаемой функциональностью, не выходя за рамки налагаемых ограничений. Тестировщики определяют, как проверять правильность реализации требований.

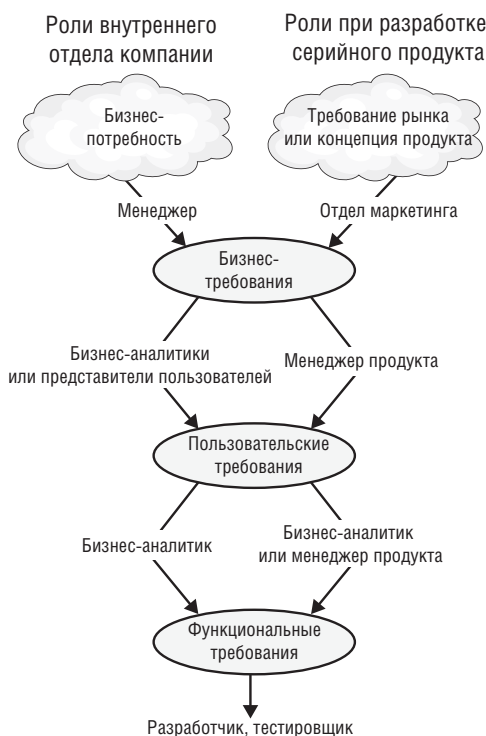


Рис. 1-3. Пример участия различных заинтересованных лиц в разработке требований

Важно понимать ценность письменной фиксации жизненно важных требований в доступной для совместного использования форме, а не сведений, передающихся от человека к человеку изустно. Однажды я работал над проектом, в котором часто менялись команды разработчиков. Основной заказчик был недоволен тем, что из каждой новой команды к нему приходили со словами: «Нам надо поговорить о требованиях». Его реакция на этот запрос звучала так: «Я уже рассказал вашим предшественникам о моих требованиях. Так что просто займитесь созданием системы!» К сожалению, никто не потрудился задокументировать никаких требований, поэтому каждой новой

команде приходилось начинать с нуля. По крайней мере безответственно заявлять, что у вас «есть требование», если на самом деле у вас несколько сообщений электронной почты и голосовой почты, ряд записок-наклеек, протоколы совещаний и туманные воспоминания о разговорах с заказчиком. Аналитик должен выработать осмотрительный подход для определения, насколько полной должна быть документация требований в том или ином проекте.

На рис. 1-1 были показаны три главных документа требований: документ концепции и границ, документ пользовательских требований и спецификация программных требований. Не всегда обязательно создавать эти три отдельных документа в каждом проекте. Часто имеет смысл объединять часть информации, особенно в небольших проектах. Однако надо понимать, что эти три документа содержат разную информацию, разрабатываются на разных этапах проекта, возможно даже разными людьми с разными целями и разной целевой аудиторией.

Модель на рис. 1-1 показывает простой «сверху вниз» поток информации требований. В реальности возможно наличие циклов и итераций пользовательских, функциональных и бизнес-требований. Каждый раз когда кто-то предлагает ввести новую функцию, пользовательское требование или улучшение функциональности, аналитик должен задаться вопросом: «Укладывается ли это в рамки проекта?» Если ответ положительный, требование должно присутствовать в спецификации. В противном случае требования в спецификации быть не должно, по крайней мере в текущем выпуске или итерации. Третий возможный ответ звучит так: «Нет, но это поддерживает бизнес-цели, поэтому должно быть в спецификации». В этом случае, ответственный за область действия проекта — куратор, менеджер или ответственный за проект, должен решить, нужно ли расширять рамки текущего проекта или итерации, чтобы включить новое требование. Это бизнес-решение, которое оказывает влияние на график и бюджет проекта и может потребовать пожертвовать другими возможностями. Эффективный процесс управления изменениями, включающий анализ последствий, гарантирует, что «правильные люди» принимают информированные бизнес-решения о том, какие изменения следует принять, а также что учитываются сопутствующие затраты времени или ресурсов или компромиссы в выборе функциональности.

Требования к продукту и требования к проекту

До этого момента мы обсуждали требования, описывающие свойства программной системы, которую планируется построить. Назовем их требованиями к *продукту*. Для проекта, как правило, создаются требования других типов: документ, где описана среда разработки, ограничения бюджета, руководство пользователя или требования для выпуска продукта и продвижения его в поддерживаемую среду. Это требования к *проекту*, но не к *продукту*. Спецификация требований содержит требования к продукту и не содержит

деталей дизайна или реализации (кроме известных ограничений), данных о планировании проекта, сведений о тестировании и тому подобной информации. Удалите указанные элементы из требований, чтобы из этого документа было абсолютно ясно, что надлежит построить команде разработчиков. К требованиям проекта относятся:

- физические ресурсы, необходимые команде разработки, такие как рабочие станции, специальные аппаратные устройства, тестовые лаборатории, средства и оборудование тестирования, командные комнаты и оборудование для видеоконференций;
- потребности в обучении персонала;
- пользовательская документация, включая обучающие материалы, пособия, справочные руководства и информация о выпусках ПО;
- документация для поддержки, такая как ресурсы службы технической поддержки, а также информация о техническом обеспечении и обслуживании аппаратных устройств;
- инфраструктурные изменения, которые необходимо внести в рабочую среду;
- требования и процедуры для выпуска продукта, установки в рабочей среде, конфигурирования и тестирования;
- требования и процедуры для перехода со старой на новую систему, например требования по переносу и преобразованию данных, по настройке безопасности, переносу производства и обучению для восполнения недостатка квалификации — это требования иногда называют *требованиями по переходу* (transition requirements) (ИВА 2009);
- требования по сертификации продукта и его соответствия требованиям регулирующих органов;
- скорректированные политики, процессы, организационные структуры и аналогичные документы;
- сорсинг, приобретение и лицензирование ПО сторонних производителей и компонентов оборудования;
- требования по бета-тестированию, производству, упаковке, маркетингу и дистрибуции;
- соглашения об уровне обслуживания с клиентами;
- требования по правовой защите (патенты, товарные знаки или авторское право) интеллектуальной собственности, связанной с разрабатываемым ПО.

Эти требования к проекту не будут рассмотрены в этой книге. Это не значит, что они не важны, — просто они находятся за рамками нашей тематики, посвященной разработке и управлению требованиями к программному продукту. Определение этих требований к проекту является совместной ответственностью бизнес-аналитика и менеджера проекта. Они часто возникают при сборе требований к продукту. Информацию требований к проекту лучше всего хранить в плане управления проектом, который должен описывать все

ожидаемые операции и результаты проекта.

В частности бизнес-приложения люди иногда называют «решением», которое охватывает как требования к продукту (за которые отвечает бизнес-аналитик), так и требования к проекту (за которые отвечает менеджер проекта). Они могут употреблять термин «рамки решения» в смысле «все, что нужно сделать для успешного завершения проекта». Однако в этой книге мы сосредоточимся на требованиях к продукту, что бы это ни было — коммерческий программный продукт, аппаратное устройство с встроенным ПО, корпоративная информационная система, ПО для государственных учреждений и т. п.

Разработка и управление требованиями

Путаница в терминологии возникает, как только речь заходит о требованиях, и затрагивает даже то, что именно называть этим словом. Некоторые авторы называют всю эту предметную область *разработкой технических условий* (нам такой подход ближе всего), вторые употребляют термин *управление требованиями* (requirements management), а третьи считают эту деятельность подмножеством более общей предметной области бизнес-анализа.

Мы считаем полезным разделить область разработки технических условий на *разработку требований* (requirements development) — подробнее об этом в части 2 этой книги — и *управление требованиями* (requirements management) — см. часть IV, как показано на рис. 1-4. Независимо от методологии проекта — водопадная, поэтапная, итеративная, гибкая или смешанная — есть вещи, которые надо выполнить в отношении всех требований. В зависимости от методологии эти операции могут выполняться в разное время жизненного цикла проекта и с разным уровнем глубины и детализации.

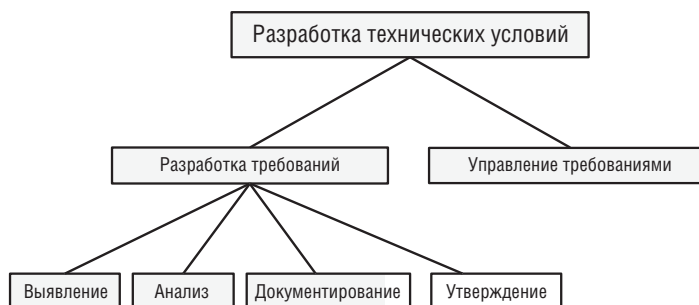


Рис. 1-4. Составные части предмета разработки технических условий

Разработка требований

Как показано на рис. 1-4, мы подразделяем разработку технических условий на *выявление* (elicitation), *анализ* (analysis), *документирование* (specification) и *утверждение* (validation) (Abgran и другие, 2004). В эти составные части входят все действия, включающие сбор, оценку, документирование и утверждение требований для ПО. Далее описаны основные действия в каждой из

составных частей.

Выявление и сбор требований

Выявление и сбор требований (elicitation) охватывает все действия, связанные с выявлением требований, такие как интервью, совещания, анализ документов, создание прототипов и другие. К ключевым действиям относятся:

- Определение классов ожидаемых пользователей продукта и других заинтересованных лиц.
- Понимание задач и целей, а также бизнес-целей, которым соответствуют эти задачи.
- Изучение среды, в которой будет использоваться новый продукт.
- Работа с отдельными людьми, которые представляют каждый класс пользователей, чтобы понять их потребности и ожидания в отношении качества.

На что ориентироваться: на использование или на продукт?

При сборе требований обычно используется подход, ориентированный на использование, или подход, ориентированный на продукт, хотя возможны и другие стратегии. В ориентированном на использование подходе упор делается на понимание и исследование задач пользователей, и на основе этой информации выводится необходимая функциональность системы. Ориентированный на продукт подход сфокусирован на определение функций, которые, как ожидается, приведут к успеху на рынке или успеху бизнеса компании. В стратегиях, ориентированных на продукт, есть риск реализовать функции, которые не будут активно использоваться, несмотря на то, что во время сбора требований они казались очень нужными. Мы рекомендуем сначала изучить бизнес-цели и цели пользователей, а затем на основе этой информации определить нужные функции и характеристики продукта.

Анализ

Анализ требований (analyzing requirements) подразумевает получение более обширного и точного понимания всех требований и представление наборов требований в различном виде. Далее перечислены основные действия:

- анализ информации, полученной от пользователей, чтобы отделить их задачи от функциональных и нефункциональных требований, бизнес-правил, предполагаемых решений и другой информации;
- разложение высокоуровневых требований до нужного уровня детализации;
- выведение функциональных требований из информации других требований;
- понимание относительной важности атрибутов качества;
- распределение требований по компонентам ПО, определенным в систем-

ной архитектуре;

- согласование приоритетов реализации;
- выявление пробелов в требованиях или излишних требований, не соответствующих заданным рамкам.

Документирование

Документирование требований предусматривает представление и хранение совокупного знания о требованиях постоянным и хорошо организованным способом. К ключевому действию относится:

- преобразование собранных потребностей пользователей в письменные требования и диаграммы, пригодные для понимания, анализа и использования целевой аудиторией.

Утверждение

Утверждение требований (requirements validation) должна подтвердить правильность имеющегося набора требований, которые позволят разработчикам создать решение, удовлетворяющее бизнес-целям. Далее перечислены основные действия:

- проверка задокументированных требований для устранения всех недостатков до принятия требований группой разработки;
- разработка приемочных тестов и критериев, которые должны подтвердить, что созданный на основе требований продукт будет отвечать потребностям заказчика и удовлетворять поставленным бизнес-целям.

Итерация — ключевое условия успеха разработки. При планировании нужно предусмотреть не один цикл проверки требований для поступательного уточнения деталей высокоуровневых требований и подтверждения правильности пользователями. На это может уходить много времени, и подобная деятельность может быть не очень захватывающей, но это неизбежная процедура разрешения всех неясностей при определении новой программной системы.

Внимание! Вы никогда не сможете создать идеальные требования. С практической точки зрения цель разработки требований — накопить общее понимание требований, достаточно хорошее для создания очередной порции продукта — будь то один или сто процентов, чтобы продолжить работу при разумном уровне риска. Основной риск заключается в наличии большого объема незапланированных недоделок, потому что команда недостаточно глубоко разобралась с требованиями к очередному этапу работы перед началом проектирования и разработки.

Управление требованиями

К действиям по управлению требованиями относятся:

- определение основной версии требований, моментальный снимок, который представляет согласованный, проверенный и одобренный набор функциональных и нефункциональных требований, обычно для конкрет-

- ного выпуска продукта или итерации разработки;
- оценка влияния предлагаемых требований и внедрение одобренных изменений в проект управляемым образом;
- обновление планов проекта в соответствии с изменениями в требованиях;
- обсуждение новых обязательств, основанных на оцененном влиянии изменения требований;
- определение отношений и зависимостей, существующих между требованиями;
- отслеживание отдельных требований до их проектирования, исходного кода и тестов;
- отслеживание состояния требований и действий по изменению на протяжении всего проекта.

Предмет управления требованиями заключается не в предотвращении изменении или усложнении их внесения — задача состоит в предугадывании и приспособливании к ожидаемым реальным изменениям, чтобы снизить их разрушительное влияние на проект.

На рис. 1-5 показан другой способ разделения областей разработки требований и управления ими. В этой книге описаны десятки специальных приемов выявления требований, их анализа, документации, проверки и управления.

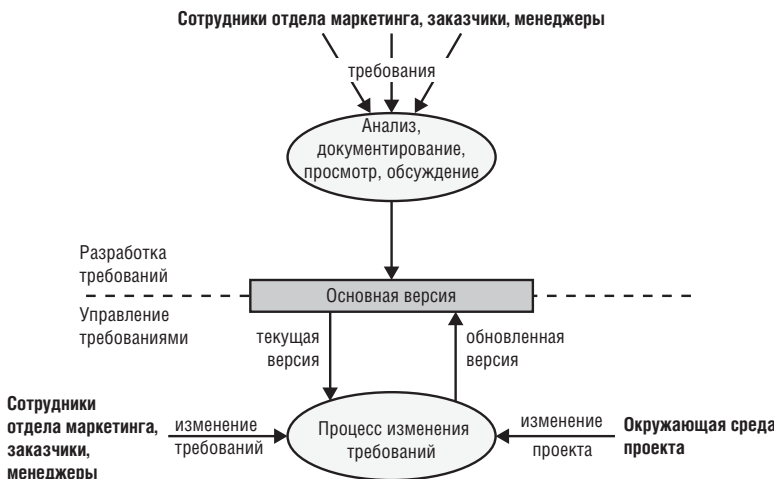


Рис. 1-5. Разделение областей разработки требований и управления ими

Каждый проект имеет требования

Фредерик Брукс выразительно определил критическую роль требований в проекте разработки ПО в классическом эссе 1987 года «No Silver Bullet: Essence and Accidents of Software Engineering»:

«Самая сложная часть построения систем ПО — решить точно, что же создавать. Никакая другая часть концептуальной работы не явля-

ется такой трудной, как выяснение деталей технических требований, в том числе и взаимодействие с людьми, механизмами и иными системами ПО. Никакая другая часть работы так не портит результат, если она выполнена плохо. Никакая другая часть не дает более трудные для исправления ошибки».

Каждая содержащая ПО система имеет пользователей, которые полагаются на нее. Время, которое тратится на выяснение потребностей пользователей, представляет собой высокоэффективную инвестицию в успех проекта. Если в команде проекта не регистрируют в письменном виде требования, с которыми заказчики уже согласились, как могут они удовлетворить этих заказчиков?

Зачастую невозможно — или не нужно — полностью определить требования до начала проектирования и реализации. В этом случае действуйте итеративно и постепенно: разрабатывайте одну порцию требований за раз и обязательно дожидайтесь ответной реакции заказчика, прежде чем приступать к следующему циклу. Это суть разработки по гибкой методологии — выяснение ровно такого объема требований, чтобы выполнить разумную приоритизацию и планирование выпуска, чтобы команда могла максимально быстро начинать создавать ценное ПО. Это не может служить оправданием написания кода до изучения требований перед следующим шагом. Итерации при кодировании стоят гораздо дороже, чем при разработке концепций.

Люди иногда предпочитают не тратить время на написание требований, но этот этап не самый сложный. Самое трудное — *выявить* эти требования. Первоначально написание требований представляет собой процесс выяснения, разработки и расшифровки данных. Четкое понимание требований к продукту дает вам уверенность, что ваша команда работает над теми проблемами, над которыми нужно, и создает лучшее их решение. Не зная, что собой представляют требования, вы не сможете определить момент окончания проекта, установить, достигнуты ли цели, или выбрать компромиссное решение, когда придется корректировать границы проекта. Вместо того, чтобы отказываться от затрат времени на создание требований, откажитесь от потерь денег из-за недостаточного внимания к требованиям в проекте.

Когда плохие требования появляются у хороших людей

Основное следствие проблем с требованиями — переделка того, что, как вы думаете, уже готово. На это расходуется от 30 до 50% общего бюджета разработки (Shull, et al., 2002; GAO, 2004), а ошибки в требованиях стоят от 70 до 85% стоимости переделки (Leffingwell, 1997). Небольшие переделки повышают ценность и улучшают продукт, но очень большой объем переделок не несет ничего кроме растраты ресурсов и разочарования. Вообразите, насколько изменилась бы ваша жизнь, если бы вы сократили объем переделок

наполовину! Члены вашей команды могли бы создавать ПО быстрее и даже приходить домой вовремя. Создание более качественных требований это инвестиции, а не затраты.

Гораздо дороже исправить дефекты, которые найдены позднее в проекте, чем сразу после создания. Допустим, что стоимость нахождения и устранения дефекта требования в процессе работы над ним составляет один доллар (по относительной шкале). Если же ошибка обнаружится во время проектирования, придется потратить доллар на исправление ошибки в требованиях и еще 2-3 доллара на переделку проекта, основанного на неправильном требовании. Но представьте, что никто не обнаружил ошибку, пока не позвонил пользователь и не пожаловался на нее. Стоимость устранения дефекта зависит от типа системы и может достигать на нашей относительной шкале 100 или больше долларов (Boehm, 1981; Grady, 1999; Haskins, 2004). Один из клиентов, которому я оказывал консалтинговые услуги, определил, что в среднем трудозатраты на обнаружение и исправление дефекта в их информационной системе составляли 200 долларов, при этом использовалась специальная методика проверки ПО, один из видов дружественной проверки (Wieggers, 2002). С другой стороны, исправление ошибки, обнаруженной пользователем, обходилось в 4200 долларов — в 21 раз дороже. Предотвращение ошибок в требованиях и обнаружение их на самых ранних этапах сильно снижает объем переделок.

Недостатки в требованиях представляют собой угрозу успеху проекта, где *успех* означает выпуск продукта, который удовлетворяет ожиданиям пользователей по качеству и функциональности при соблюдении бюджета и графика проекта. В главе 23 рассказано, как управлять такими рисками, чтобы они не привели к крушению проекта. Некоторые из наиболее общих рисков описаны далее в этой главе.

Недостаточное вовлечение пользователей

Заказчики зачастую не понимают, почему так важно тщательно собрать требования и обеспечить их качество. Разработчики не всегда придают значение вовлечению пользователей в процесс скорее всего из-за того, что они считают, что все уже знают о потребностях пользователей. В некоторых случаях трудно добраться до людей, которые непосредственно будут иметь дело с продуктом, а те, кто заменяет пользователей, не всегда понимают, что тем нужно в реальности. Недостаточное вовлечение пользователей ведет к обнаружению ошибок в требованиях на поздних стадиях проекта, а значит, к задержке завершения проекта.

Еще один риск, связанный с недостаточным вовлечением пользователей, особенно при анализе и проверке требований, заключается в том, что бизнес-аналитик может не понять и неправильно задокументировать реальные бизнес-требования или потребности клиента. Иногда бизнес-аналитик идет по пути определения того, что кажется «идеальными» требованиями, а разработчики реализуют их, но никто не использует решение, потому что

бизнес-задача была понята неправильно. Продолжающиеся совещания с пользователями могут помочь снизить риск, но если пользователи проверят требования недостаточно тщательно, у вас все равно могут быть проблемы.

Небрежное планирование

«Я кое-что придумал для нового продукта. Когда вы сможете это сделать?» Не отвечайте на подобный вопрос, пока больше не узнаете о проблеме. Неопределенные, плохо понятые требования порождают слишком оптимистические оценки, которые возвращаются и не дают нам покоя, когда возникает перерасход. Неподготовленная оценка звучит как обязательство для слушателя. Наибольшие вклады в проект при плохо просчитанной смете составляют затраты на частые изменения требований, пропущенные требования, недостаточное взаимодействие с пользователями, недетализированную спецификацию требований и плохой анализ (Davis, 1995). Оценка трудоемкости и продолжительности проекта на основе требований означает, что вы должны что-то знать об объеме своих требований и продуктивности команды разработчиков. Подробнее о выполнении оценки на основе требований см. раздел «Подробнее о требованиях к ПО» (Wieggers, 2006) в главе 5.

«Разрастание» требований пользователей

В процессе разработки требования могут меняться из-за чего проект часто выходит за установленные рамки как по срокам, так и по бюджету. Чтобы управлять пределами разрастания требований, для начала уточните бизнес-цели проекта, стратегическое видение, ограничения и критерии успеха. Оцените, как все предполагаемые новые характеристики или требования, отразятся на этих параметрах. Требования *будут* изменяться и расти. Менеджер проекта должен предусмотреть «буферы планирования» на случай непредвиденных обстоятельств, чтобы первое же новое требование не привело к срыву графика (Wieggers, 2007). В проектах гибкой методологии объем итерации корректируется так, чтобы вписаться в заданный бюджет и длительность итерации. При появлении новых требований они размещаются в резерве (backlog) и назначаются в будущие итерации на основе приоритета. Изменения зачастую критически важны для успеха, однако они всегда имеют цену.

Двусмысленные требования

Один из симптомов двусмысленности заключается в том, что пользователь может интерпретировать одно и то же положение по-разному. (Lawtence, 1996). Другой симптом состоит в том, что у нескольких читателей требования возникает разное понимание, что оно означает. В главе 11 указано множество слов и фраз, которые порождают двусмысленность, возлагая ответственность интерпретации на читателя.

Двусмысленность ведет и к формированию различных ожиданий у заин-

тересованных лиц. Впоследствии некоторые из них удивляются результату. Разработчики же впустую тратят время, занимаясь не теми задачами. А тестировщики готовятся к проверке не тех особенностей поведения системы.

Один из способов избавиться от двусмысленности — пригласить различных представителей пользователей для официальной экспертизы требований. (Wiegers, 2002). Как говорится в главе 17, неформальная дружественная оценка, в которой участники просто самостоятельно читают требования, часто не обнаруживает двусмысленности. Если они интерпретируют требования различными способами, но это имеет смысл для каждого из них, то неясность не проявится. Совместное выявление и проверка требований стимулирует обсуждение и уточнение требований в группе в рамках совещания. Другой способ обнаружить двусмысленность — написать вариант тестирования для требования и построить прототип.

Требования-«бантики»

Под «бантиками» (gold plating) понимают отсутствующие в спецификации требований функции, добавленные разработчиками, потому что им кажется, что это понравится пользователям. Если избыточные возможности оказываются ненужными для клиентов, получается, что время, отведенное на реализацию, тратится впустую. Прежде чем просто вставлять новые функции, разработчики и бизнес-аналитики должны представить свои творческие идеи на суд заказчиков. Задача команды — четко соблюдать требования спецификации, а не действовать за спиной клиентов, без их одобрения.

Пользователи иногда требуют функции или элементы интерфейса, которые выглядят красиво, но не представляют особой ценности для продукта. Все, что вы захотите добавить, стоит времени и денег, поэтому постарайтесь максимизировать пользу от будущего продукта. Чтобы снизить количество «бантиков», отслеживайте каждый элемент функциональности до его источника, чтобы четко понимать, почему именно он включен в продукт. Убедитесь, что все специфицируемое и разрабатываемое находится в рамках проекта.

Пропущенные классы пользователей

Большинство продуктов предназначены для нескольких групп пользователей, которые могут применять различные наборы функций с разной частотой и иметь самый разный опыт работы с ПО. Если вы не определили важные классы пользователей для вашего продукта заранее, некоторые потребности клиентов не будут учтены. После идентификации всех классов удостоверьтесь, что голос каждого услышан, как описано в главе 6. Помимо очевидных пользователей не забудьте о сотрудниках поддержки, у которых есть собственные требования, функциональные и нефункциональные. У сотрудников, которые конвертируют данные из унаследованных систем, есть требования по переходу, которые не влияют на конечный продукт, но определенно влияют на успех решения. Могут быть заинтересованные лица, которые даже

не знают о существовании проекта, например государственные учреждения, определяющие стандарты, которые могут влиять на вашу систему, и вы должны знать о таких заинтересованных лицах и их влиянии на проект.

Выгоды от высококачественного процесса разработки требований

Многие люди ошибочно считают, что время, которое тратится на обсуждение требований, просто отсрочивает выпуск продукта. Они предполагают, что работа с требованиями не повышает рентабельность проекта. На самом деле, затраты на получение хороших требований практически всегда возвращаются с излишком.

В удачных процессах создания требований к совместной партнерской работе над проектом привлекаются множество заинтересованных лиц, причем на протяжении всего проекта. Сбор требований позволяет команде разработчиков лучше понять пользователей или реалии рынка, что критически важно для любого проекта. Делая упор на задачи пользователей, а не на внешне привлекательные функции, команда избежит необходимости переписывать код, который даже не понадобится. Вовлечение клиентов в процесс снижает вероятность появления ложных ожиданий, возникающих из-за различия того, в чем пользователи нуждаются, и того, что разработчики выпускают. Рано или поздно вы начнете получать отзывы заказчиков. Причем намного дешевле добиться взаимопонимания до начала разработки продукта, чем после финальной поставки. О природе партнерства между заказчиком и разработчиками см. главу 2.

Ясное разделение требований на те, что относятся к ПО, оборудованию или подсистемам, взаимодействующим с людьми, позволяет применять системный подход к разработке продукта. Эффективные процессы управления изменениями минимизируют неблагоприятные последствия от изменения требований. Недвусмысленно составленные документы облегчают тестирование продукта. В совокупности все это повышает шансы создать высококачественный продукт, который удовлетворит всех пользователей.

Никто не станет обещать конкретный возврат инвестиций в процесс улучшения требований. Вы можете мысленно проанализировать, как более качественные требования могут помочь вашим командам (Wiegiers, 2006). Стоимость более качественных требований складывается из стоимости разработки новых процедур и шаблонов документов, тренинга команды и покупки инструментов. Наибольшая инвестиция — это время, которое ваша команда тратит на сбор, документирование, просмотр и управление требованиями. Возможные выгоды таковы:

- меньше дефектов в требованиях и в готовом продукте;
- меньше переделок;
- быстрее разработка и поставка готового продукта;

- меньше ненужных и неиспользуемых функций;
- ниже стоимость модификации;
- меньше недопонимания;
- меньше расползание границ проекта;
- меньше беспорядок в проекте;
- выше удовлетворение заказчиков и членов команды;
- продукты, которые делают то, что от них ожидается.

Даже если вы не можете количественно оценить все преимущества, они все равно реальны.

Что дальше?

- Опишите связанные с требованиями проблемы, с которыми вы сталкивались в текущем или предыдущих проектах. Разбейте их на проблемы разработки или проблемы управления. Опишите основные причины каждой проблемы и ее влияние на проект.
- Организуйте обсуждение проблем с требованиями с членами вашей команды и всеми заинтересованными лицами, записывая проблемы, с которыми вы сталкивались в текущем или предыдущих проектах, их влияние и их основные причины. Поделитесь своими идеями по изменению текущих процессов работы с требованиями, которые позволят противостоять этим трудностям. Возможно, вам окажется полезным руководство по устранению неполадок в Приложении Б.
- Сравните используемую в вашей организации терминологию в области требований и продуктов и ту, что употребляется в этой главе, чтобы определить, охватываете ли вы все рекомендуемые нами категории.
- Выполните простую оценку на нескольких страницах одного из документов требований, чтобы определить области, в которых ваша команда может улучшить свою работу. Особенно полезно было бы, если бы эту оценку выполнил внешний специалист.
- Организуйте тренинг, посвященный требованиям, для команды, которая работает над вашим проектом. Пригласите основных заказчиков, маркетологов, менеджеров, разработчиков, тестировщиков и других заинтересованных лиц. Тренинг позволяет выработать единую лексику и общее понимание эффективных приемов и поведения, чтобы члены команды могли более эффективно решать стоящие перед ними общие задачи.