RSTQB
Russian Software Testing Qualifications Board

Сертифицированный тестировщик

Программа обучения Продвинутого уровня в управлении тестированием

Версия 3.0 RU-3

International Software Testing Qualifications Board

ISTQB®
International Software Testing Qualifications Board

Уведомление об авторских правах

История изменений

Версия	Дата	Содержание
ISEB v1.1	04 сентября 2001	ISEB Practitioner Syllabus
ISTQB 1.2E	Сентябрь 2003	ISTQB Advanced Level Syllabus из EOQ-SG
V2007	12 октября 2007	Certified Tester Advanced Level syllabus version 2007
D100626	26 июня 2010	Внесение изменений, принятых в 2009 году, разделение глав на отдельные модули
D101227	27 декабря 2010	Принятие изменений в формате и правок, не влияющих на суть содержания
D2011	31 октября 2011	Изменения по разделению программы обучения, переработаны цели обучения и текст для согласования с целями. Добавлены бизнес-цели.
Alpha 2012	09 февраля 2012	Внесение изменений по комментариям от национальных коллегий по октябрьскому релизу.
Beta 2012	26 марта 2012	Внесение изменений по комментариям от национальных коллегий по альфа-релизу.
Beta 2012	07 апреля 2012	Бета-версия для GA
Beta 2012	08 июня 2012	Отредактированная версия для национальных коллегий
Beta 2012	27 июня 2012	Внесены комментарии от EWG и по Глоссарию
RC 2012	15 августа 2012	Релиз-кандидат – включены финальные правки от национальных коллегий
GA 2012	19 октября 2012	Финальные правки для релиза GA
RSTQB 2017	30 марта 2017	Перевод на русский язык
2012 RU-2	12 мая 2024	Приведение к шаблону ISTQB v4.0 Обновление логотипа RSTQB
Beta v3.0	31 октября 2023	Внесение изменений по комментариям от национальных коллегий по альфа-рецензированию
POST Beta v3.0	31 января 2024	Внесение изменений по комментариям от национальных коллегий по бета-рецензированию
POST Beta v3.0	29 февраля 2024	Минимальные изменения после корректуры
RC v3.0	28 марта 2024	Релиз-кандидат – внесение формальных изменений последнего шаблона от процессной рабочей группы
V.3.0	3 мая 2024	Переработка после релиза; изменения коснулись только опечаток и несоответствий
V.3.0 RU-1	27 августа 2024	Перевод на русский язык
V.3.0 RU-2	3 октября 2024	Исправлена автонумерация разделов
V.3.0 RU-3	24 января 2025	Терминология приведена в соответствие с последней версией Глоссария ISTQB, добавлен предметный указатель, исправлены ошибки и опечатки

Содержание

Благодарности

Перевод версии документа 2023 выполнен рабочей группой АНО «Коллегия экспертов по качеству программного обеспечения» (Russian Software Testing Qualifications Board, RSTQB): Андрей Конушин (руководитель группы), Александр Александров (редактор), Ксения Кондакова, Илья Кулаков, Юрий Левченко, Александр Торговкин и Павел Шариков.

Документ был официально выпущен Генеральной Ассамблеей ISTQB 3 мая 2024 года.

Этот документ был подготовлен рабочей группой ISTQB: Horst Pohlmann (владелец продукта, заместитель председателя рабочей группы AELWG), Tauhida Parveen, Francis Fenner, Laura Albert, Matthias Hamburg, Maud Schlich, Tanja Tremmel, Ralf Bongard, Erik van Veenendaal, Jan Giessen, Bernd Freimut, Andreas Neumeister, Georg Sehl, Rabi Arabi, Therese Kuhfuß, Ecaterina Irina Manole, Veronica Belcher, Kenji Onishi, Pushparajan Balasubramanian, Meile Postuma и Miroslav Renda.

Команда благодарит Gary Mogyorodi за его техническое рецензирование бета-версии, Julia Sabatine – за ее корректуру, а также команду рецензентов и Национальные коллегии за их вклад и предложения.

Следующие специалисты принимали участие рецензировании, комментариях и обсуждении этой программы обучения:

Альфа-рецензенты: Benjamin Timmermans, Mattijs Kemmink, Rik Marselis, Jean-Francois Riverin, Gary Mogyorodi, Ralf Bongard, Ingvar Nordström, Yaron Tsubery, Imre Mészáros, Mattijs Kemmink, Ádám Bíró, Ramit M Kaul, Chinthaka Indikadahena, Darvay Tamás Béla, Beata Karpinska, Young jae Choi, Stuart Reid, Tal Pe'er, Meile Posthuma, Daniel van der Zwan, Klaudia Dussa-Zieger, Jörn Münzel, Ralf Bongard, Petr Neugebauer, Derk-Jan de Grood, Rik Kochuyt, Andreas Hetz, Laura Albert, Eszter Sebestyeni, Tamás Szőke, Henriett Braunné Bokor, Ágota Horváth, Péter Sótér, Ferenc Hamori, Darvay Tamás Béla, Paul Weymouth, Lloyd Roden, Kevin Chen, Huang qin, Pushparajan Balasubramanian, Szilard Szell, Tamas Stöckert, Lucjan Stapp, Adam Roman, Anna Miazek, Márton Siska, Erhardt Wunderlich, László Kvintovics, Murian Song, Mette Bruhn-Pedersen, Petra Schneider, Michael Stahl, Ramit M Kaul, Imre Mészáros, Dilhan Jayakody, Francisca Cano Ortiz, Johan Klintin, Liang Ren, Ole Chr. Hansen, Zsolt Hargitai, Tamás Rakamazi, Kenji Onishi, Arnika Hryzszko, Rabih Arabi, Veronica Belcher, и Vignesh Balasubramanian.

Бета-рецензенты: Maria-Therese Teichmann, Dominik Weber, Thomas Puffler, Peter Kunit, Martin Klonk, Michaël Pilaeten, Wim Decoutere, Arda Ender Torçuk, Piet de Roo, Rik Marselis, Jakub Platek, Ding Guofu, Zheng Dandan, Liang Ren, Yifan Chen, Hallur Helmsdal, Ole Chr. Hansen, Klaus Skafte, Gitte Ottosen, Tanzeela Gulzar, Arne Becher, Klaudia Dussa-Zieger, Jan Giesen, Florian Fieber, Carsten Weise, Arnd Pehl, Matthias Hamburg, Stephanie Ulrich, Jürgen Beniermann, Márton Siska, Sterbinszky Ádám, Ágnes Srancsik, Marton Matyas, Tamas Stöckert, Csilla Varga, Zsolt Hargitai, Bíró Ádám, Horváth Ágota, Sebestyéni Eszter, Szilárd Széll, Péter Sótér, Giancarlo Tomasig, Nicola de Rosa, Kaiwalya Katyarmal, Pradeep Tiwari, Sreeja Padmakumari, Seunghee Choi, Stuart Reid, Dmitrij Nikolajev, Mantas Aniulis, Monika Stoecklein-Olsen, Adam Roman, Mahmoud Khalaili, Ingvar Nordström, Beata Karpinska, Armin Born, Ferdinand Gramsamer, Mergole Kuaté, Thomas Letzkus, Nishan Portoyan, Ainsley Rood, Lloyd Roden, Sarah Ireton.

Перевод версии документа 2012 года выполнен Рабочей Группой Продвинутого уровня Russian Software Testing Qualifications Board: Маргарита Трофимова (руководитель группы), Александр Александров (редактор).

Благодарим команду переводчиков: Андрей Конушин, Елена Костина, Александр Мешков, Александра Титова

Этот документ был создан рабочей группой Программы обучения Руководитель тестирования Продвинутого уровня ISTQB: Rex Black (председатель), Judy McKay (заместитель председателя), Graham Bath, Debra Friedenberg, Bernard Homès, Paul Jorgensen, Kenji Onishi, Mike Smith, Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

Рабочая группа благодарит команду редакторов и национальные коллегии за их вклад и замечания.

Состав рабочей группы «Руководитель тестирования Продвинутого уровня» на момент завершения программы обучения Продвинутого уровня состоял из (в алфавитном порядке): Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (Заместитель председателя), Paul Jorgensen, Judy McKay, Jamie Mitchell, Thomas Mueller, Klaus Olsen, Kenji Onishi, Meile Posthuma, Eric Riou du Cosquer, Jan Sabak, Hans Schaefer, Mike Smith (Председатель), Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

В редактировании и предоставлении замечаний принимали участие: Chris van Bael, Graham Bath, Kimmo Hakala, Rob Hendriks, Marcel Kwakernaak, Rik Marselis, Don Mills, Gary Mogyorodi, Thomas Mueller, Ingvar Nordstrom, Katja Piroué, Miele Posthuma, Nathalie Rooseboom de Vries, Geoff Thompson, Jamil Wahbeh, Hans Weiberg.

0 Предисловие к программе обучения

0.1 Цель этого документа

Эта программа обучения представляет основу международной сертификации на квалификацию Продвинутого уровня ISTQB® в управлении тестированием. ISTQB® предоставляет эту программу:

1. Национальным коллегиям для перевода на национальный язык и аккредитации организаторов обучения. Национальные коллегии могут адаптировать программу обучения к особенностям местных языков и определить ссылки для адаптации к местным публикациям.
2. Экзаменационным комиссиям для формирования экзаменационных вопросов на местном языке, адаптированные к целям обучения каждого курса.
3. Аккредитованным ISTQB® Организаторам обучения для разработки материалов обучения и определения соответствующих методов обучения.
4. Кандидатам на получение сертификатов для подготовки к экзамену (ISTQB® рекомендует пройти аккредитованный ISTQB® тренинг перед сдачей экзамена ISTQB® Продвинутого уровня).
5. Международному сообществу разработки ПО и систем для продвижения профессии тестировщика ПО и систем, и использования как основы для книг и статей.

0.2 Сертифицированный тестировщик Продвинутого уровня в управлении тестированием

Квалификация Продвинутого уровня предназначена для всех, связанных с управлением тестированием ПО, включая такие роли, как тестировщики, консультанты по тестированию, руководители тестирования, тестировщики пользовательского приемочного тестирования, скрам-мастера, руководители проектов или владельцы продуктов. Данная квалификация Продвинутого уровня в управлении тестированием также подходит для всех, кто хочет получить углубленное понимание тестирования ПО, например, для руководителей проектов, руководителей по качеству, руководителей разработки ПО, бизнес-аналитиков, ИТ-директоров и консультантов по управлению. Обладатели сертификата Продвинутого уровня в управлении тестированием могут готовиться к более высокой квалификации Экспертного уровня ISTQB® в тестировании ПО. Сертификат сертифицированного тестировщика ISTQB® Продвинутого уровня — Руководитель тестирования или Управление тестированием действителен пожизненно и не требует продления. Сертификат признается на международном уровне и демонстрирует профессиональную компетентность и надежность кандидатов в управлении тестированием.

0.3 Карьерный путь тестировщиков

Программа ISTQB® обеспечивает поддержку специалистов по тестированию на всех этапах их карьеры. Специалисты, получившие сертификат ISTQB® Продвинутого уровня в управлении тестированием, также могут быть заинтересованы в остальных квалификациях Продвинутого уровня (Тест-аналитик и Технический тест-аналитик), а затем и Экспертного уровня (Управление тестированием или Улучшение процесса тестирования). Любой, кто стремится развить навыки тестирования в рамках гибкой методологии разработки программного обеспечения (Agile), может

рассмотреть возможность получения сертификата Технический тестировщик в гибких методологиях или Масштабирование тестирования в организации с гибкими методологиями. Направление «Специалист» предлагает глубокое погружение в области, где используются конкретные подходы к тестированию и действия по тестированию (например, автоматизация тестирования, тестирование искусственного интеллекта, тестирование на основе моделей, тестирование мобильных приложений), которые связаны с конкретными областями тестирования (например, тестирование производительности, тестирование удобства использования, приемочное тестирование, тестирование безопасности) или ноу-хау тестирования для определенных областей промышленности (например, автомобилестроения или игр). Посетите сайт www.istqb.org для получения последней информации о программе сертифицированных тестировщиков ISTQB®.

0.4 Бизнес-результаты

В этом разделе перечислены 11 бизнес-результатов, ожидаемые от кандидата, имеющего сертификат Продвинутого уровня в управлении тестированием.

Тестировщик Продвинутого уровня в управлении тестированием может…

TM_01	Руководить тестированием в различных проектах разработки ПО, применяя процессы управления тестированием, определенные для проектной группы или организации, выполняющей тестирование.
TM_02	Определять заинтересованные стороны процесса тестирования и модели жизненного цикла разработки ПО актуальные в заданном контексте.
TM_03	Организовывать встречи по выявлению и оценке рисков в рамках любого жизненного цикла разработки ПО и использовать их результаты для достижения целей тестирования.
TM_04	Определять стратегию тестирования проекта, соответствующую стратегии тестирования, принятой в организации, и контексту проекта.
TM_05	Непрерывно отслеживать и контролировать тестирование для достижения целей проекта.
TM_06	Оценивать и сообщать о ходе тестирования заинтересованным сторонам проекта.
TM_07	Определять необходимые навыки специалистов и развивать их в своей команде.
TM_08	Подготавливать и представлять экономическое обоснование для тестирования в различных контекстах, описывающее затраты и ожидаемые выгоды.
TM_09	Руководить деятельностью по улучшению процессов тестирования в проектах или программах разработки продуктов ПО и вносить вклад в инициативы по улучшению процессов тестирования в организации.
TM_10	Планировать действия по тестированию, включая необходимую тестовую инфраструктуру, и оценивать необходимые для тестирования усилия.
TM_11	Создавать отчеты о дефектах и жизненный цикл дефектов, подходящие для жизненного цикла разработки ПО.

0.5 Проверяемые цели обучения и когнитивные уровни целей обучения

Цели обучения поддерживают достижение бизнес-результатов, а также используются для разработки сертификационных экзаменов Продвинутого уровня в управлении тестированием. В целом все главы данной программы обучения содержат цели обучения для уровня К1, кроме Предисловия, Ссылок, Заключения и Приложений. Другими словами, кандидат на сертификат должен распознать, запомнить и воспроизвести термин или понятие, упоминаемое в любом из трех разделов. Конкретные уровни целей обучения указаны в начале каждой главы и классифицируются следующим образом:

- К2: Понять
- К3: Применить
- К4: Проанализировать

Более подробная информация и примеры целей обучения приведены в Приложении А. Все термины, перечисленные в качестве ключевых слов сразу под заголовками глав, должны быть запомнены (К1), даже если они не упомянуты явно в целях обучения.

0.6 Экзамен Продвинутого уровня в управлении тестированием

Сертификационные экзамены Продвинутого уровня в управлении тестированием основаны на данной программе обучения. Ответы на экзаменационные вопросы могут потребовать использования материала, основанного более чем на одной главе этой программы обучения. Все разделы программы обучения поддаются экзаменационной проверке, кроме Предисловия, Ссылок и Приложений. Стандарты и книги включены в качестве ссылок (Глава 5), но их содержание не проверяется на экзамене, за исключением того, что обобщенно в этой программе обучения из этих стандартов и книг. См. документ «Структура и правила экзамена v1.1», применимый к программам обучения Базового и Продвинутого уровней и модулям Специалист. Англоязычная версия документа «Exam Structures and Rules v1.1» доступна на сайте istqb.org.

- Критерием допуска к сдаче экзамена на получение сертификата Продвинутого уровня в управлении тестированием является:
  - Наличие у кандидата сертификата Базового уровня ISTQB®. Однако, настоятельно рекомендуется наличие у кандидата хотя бы минимального опыта в разработке или тестировании ПО, например, шестимесячного опыта работы тестировщиком или разработчиком ПО

0.7 Аккредитация

Коллегия-член ISTQB® может аккредитовать обучающие организации, учебные материалы которых соответствуют этой программе обучения. Обучающие организации должны получить руководство по аккредитации от коллегии-члена или органа, который проводит аккредитацию. Аккредитованный курс признается соответствующим этой программе, и в рамках курса разрешается сдавать экзамен ISTQB®. Руководство по аккредитации для этой программы обучения соответствует общему Руководству по аккредитации, опубликованному Рабочей группой по управлению процессами и соответствию ISTQB®.

0.8 Поддерживаемые стандарты

В программе обучения Продвинутого уровня в управлении тестированием есть ссылки на стандарты (например, IEEE, ISO и IEC). Цель этих ссылок — предоставить основу (как в ссылках на ISO 25010 в отношении характеристик качества) или предоставить источник дополнительной информации, если читатель пожелает. Стандарты не предназначены для проверки на экзамене. Дополнительную информацию о стандартах см. в главе 5. Ссылки.

0.9 Уровень детализации

Уровень детализации этой программы обучения позволяет проводить согласованные на международном уровне курсы и экзамены. Для достижения этой цели программа обучения состоит из:

- Общих целей обучения, описывающих намерения сертифицированного тестировщика Продвинутого уровня в управлении тестированием.
- Списка терминов, которые кандидаты должны запомнить.
- Целей обучения для каждой области знаний с описанием когнитивных результатов обучения, которые должны быть достигнуты.
- Описание ключевых понятий, включая ссылки на такие источники, как литература или стандарты.

Содержание программы не является описанием всей области знаний по тестированию ПО; оно отражает уровень детализации, который необходимо охватить в учебных курсах Продвинутого уровня в управлении тестированием. Основное внимание уделяется областям тестирования и методам, которые можно применить к большинству проектов разработки ПО независимо от используемого жизненного цикла.

0.10 Как организована эта программа обучения

Программа обучения содержит 3 главы с подлежащим изучению содержанием. В заголовке верхнего уровня каждой главы указано время обучения для этой главы. Ниже уровня главы время не указывается. Для аккредитованных учебных курсов программа требует минимум 22 часа 45 минут обучения, распределенных по трем главам следующим образом:

- Глава 1: Управление активностями тестирования (750 минут)
  - Обучающийся учится объяснять действия по управлению тестированием (т.е. планирование, мониторинг, контроль и завершение тестирования).
  - Обучающийся учится определять стратегию тестирования проекта, включая цели тестирования и выбирать правильный подход к тестированию, соответствующий организационной стратегии тестирования и контексту проекта.
  - Обучающийся изучает управление проектами в различных контекстах.
  - Обучающийся учится применять тестирование на основе рисков, чтобы сфокусировать тестирование на выявленных рисках.
  - Обучающийся изучает управление деятельностью по улучшению процесса тестирования с помощью ретроспективы проекта или итерации.

- Обучающийся учится улучшать инструментальную поддержку тестирования, принимая во внимание риски, затраты и преимущества инструментальной поддержки.

- Глава 2: Управление продуктом (390 минут)
  - Обучающийся изучает, как отслеживать и контролировать тестирование для достижения целей тестирования с использованием показателей тестирования, а также оценивать и сообщать о ходе тестирования.
  - Обучающийся учится выбирать правильные методы оценки результатов тестирования в различных организациях, используя разные модели разработки ПО.
  - Обучающийся учится организовывать жизненный цикл дефектов в рамках управления дефектами, чтобы он соответствовал последовательной, гибкой и гибридной моделям разработки ПО.
- Глава 3: Управление командой (225 минут)
  - Обучающийся учится анализировать контекст конкретного проекта, чтобы определить навыки, необходимые команде тестирования.
  - Обучающийся учится управлять командой по принципу общекомандного подхода.
  - Обучающийся учится экономически обосновывать мероприятия по тестированию в проекте.

Примечание: каждой цели обучения в данной учебной программе соответствует подраздел с содержанием (например, для LO-1.2.3 есть подраздел 1.2.3).

0.11 Фундаментальные предпосылки для данной программы обучения

Эта программа обучения предназначена для всех, кто хочет достичь продвинутого уровня компетенций в управлении тестированием, например, для руководителей тестирования, тест-аналитиков, инженеров по тестированию, консультантов по тестированию, координаторов тестирования и руководителей проектов. Программа обучения соответствует программе ISTQB® Базового уровня V.4, которая обеспечивает базовые знания и понимание тестирования программного обеспечения.

Эта программа охватывает две основные роли в тестировании: руководитель тестирования и тестировщик. Роль руководителя тестирования используется в контексте модели последовательной разработки, где она обычно является отдельной ролью, не совпадающей с ролями руководителя проекта или владельца продукта. Руководитель тестирования отвечает за общий процесс тестирования, команду тестирования и управление тестированием. Эти активности включают определение стратегии тестирования, планирование действий по тестированию, мониторинг и контроль хода тестирования, составление отчетов о результатах тестирования и управление рисками и проблемами тестирования. Руководитель тестирования также обеспечивает соответствие целей тестирования потребностям бизнеса и заинтересованных сторон, а также координацию действий по тестированию с другими заинтересованными сторонами проекта.

Тестировщик также выполняет оценку тестирования, управление дефектами и действия по завершению тестирования. Тестировщик использует различные методы тестирования, чтобы гарантировать качество и надежность артефактов тестирования и тестируемой системы. Тестировщик также использует инструменты тестирования и автоматизацию для поддержки процесса тестирования, а также повышения эффективности, и результативности тестирования.

Действия и задачи, выполняемые этими двумя ролями, могут различаться в зависимости от контекста, например, проекта, продукта, навыков и организации. (см. программу обучения ISTQB® Базового уровня V.4).

Термин «член команды тестирования» используется в данной программе обучения для обозначения любого специалиста, занимающего должность руководителя тестирования или тестировщика, который выполняет тестирование, независимо от организационного контекста и других ролей. Команды тестирования состоят из людей с разными навыками и компетенциями. Члены команды также могут иметь разный уровень опыта и сертификации, например, базовый, продвинутый или экспертный уровень. Члены команды тестирования также могут иметь разные роли и обязанности в зависимости от подхода к тестированию и используемой модели процесса тестирования, таких как тестирование в проектах с гибкими методологиями, тестирование на основе моделей, тестирование на основе рисков и т.д.

Важным является тот факт, что данная программа обучения фокусируется на управлении тестированием на уровне проекта, а не на управлении тестированием на уровне организации. Таким образом, эта программа содержит информацию, которая может быть использована для деятельности по управлению тестированием на уровне проекта, но в меньшей степени для управления тестированием на уровне организации.

Гибридная разработка программного обеспечения используется в этой программе для обозначения подхода к разработке ПО, который сочетает в себе элементы различных моделей жизненного цикла программного обеспечения, таких как V-модель, итеративная, инкрементная и гибкая. Гибридная разработка ПО направлена на использование сильных сторон и смягчение недостатков каждой модели в зависимости от контекста и потребностей проекта. Например, гибридный подход к разработке программного обеспечения может использовать V-модель для этапов первоначального планирования и анализа требований, за которой следует гибкая модель для этапов проектирования, разработки и тестирования. Альтернативно, гибридный подход к разработке программного обеспечения может использовать итеративную модель для общего управления проектом, применяя при этом инкрементальную модель для каждой итерации и гибкую модель для каждого приращения. Гибридная разработка программного обеспечения требует повышенной гибкости, взаимодействия и сотрудничества между заинтересованными сторонами, а также четкого понимания целей, рисков и ограничений каждого этапа и модели.

Согласно данной программе обучения и Глоссарию, стратегия тестирования — это описание того, как будет проводиться тестирование для достижения целей тестирования в заданных обстоятельствах. Стратегия тестирования определяет общий объем, подход и ресурсы для тестирования системы или продукта. Обычно это документируется в плане тестирования или как часть других документов, в зависимости от контекста тестирования. На стратегию тестирования влияет организационная стратегия тестирования, которая представляет собой стратегию тестирования высокого уровня, описывающую, как тестирование проводится в организации. Стратегия тестирования также может существовать для одного уровня или типа тестирования, которые представляют собой конкретные аспекты тестирования, ориентированные на различные цели, задачи и критерии. Общий термин «стратегия тестирования» можно использовать в любом контексте (проект, организация, продукт). Подход к тестированию — это способ реализации задач тестирования, особенно выбор и сочетание уровней тестирования, типов тестирования и методов тестирования для статического и динамического тестирования, а также другие методы тестирования, такие как сценарное тестирование, ручное тестирование, сравнительное тестирование и т.д. Подход к тестированию, выбранный руководителем тестирования, является ключевым решением при формулировании подходящей стратегии тестирования для заданного контекста.

1 Управление активностями тестирования – 750 минут

Ключевые слова

TPI NEXT, анализ рисков, вероятность риска, влияние риска, гибридная модель разработки программного обеспечения, жизненный цикл разработки программного обеспечения, завершение тестирования, инкрементная модель разработки, интегрированная модель зрелости тестирования (TMMi), итеративная модель разработки, контроль тестирования, методология достижения целей S.M.A.R.T., мониторинг рисков, мониторинг тестирования, нефункциональное тестирование, определение рисков, оценка риска, план тестирования, планирование тестирования, последовательная модель разработки, ретроспектива, риск качества, риск продукта, смягчение рисков, совершенствование процесса тестирования, стратегия тестирования, тестирование на основе опыта, тестирование на основе рисков, тип тестирования, управление рисками, уровень риска, уровень тестирования, функциональное тестирование, цель тестирования.

Ключевые термины, специфичные для управления тестированием

IDEAL, измерение, индикатор, метрика, Цель-Вопрос-Метрика (GQM)

Цели обучения главы 1:

1.1 Процесс тестирования

TM-1.1.1 (K2) Обобщить планирование тестирования

TM-1.1.2 (K2) Обобщить мониторинг и контроль тестирования

TM-1.1.3 (K2) Обобщить завершение тестирования

1.2 Контекст тестирования

TM-1.2.1 (K2) Сравнить заинтересованность различных заинтересованных сторон в тестировании

TM-1.2.2 (K2) Объяснить, почему знания заинтересованных сторон важны для управления тестированием

TM-1.2.3 (K2) Объяснить тестирование в гибридной модели разработки программного обеспечения

TM-1.2.4 (K2) Обобщить активности по управлению тестированием для различных жизненных циклов разработки программного обеспечения

TM-1.2.5 (K2) Сравнить активности по управлению тестированием на различных уровнях тестирования

TM-1.2.6 (K2) Сравнить активности по управлению тестированием для различных типов тестирования

TM-1.2.7 (K4) Проанализировать конкретный проект и определить активности по управлению тестированием, в которых особое внимание уделяется планированию, мониторингу и контролю тестирования

1.3 Тестирование на основе рисков

TM-1.3.1 (K2) Объяснить различные измерения, которые принимает тестирование на основе рисков для реагирования на риски

TM-1.3.2 (K2) Привести примеры различных методов, которые руководитель тестирования может использовать для выявления рисков, связанных с качеством продукта

TM-1.3.3 (K2) Обобщить факторы, определяющие уровни риска, связанные с качеством продукта

TM-1.3.4 (K4) Выбрать соответствующие активности тестирования для смягчения рисков в соответствии с их уровнем риска в конкретном контексте

TM-1.3.5 (K2) Различить тяжелые и легкие примеры методов тестирования на основе рисков

TM-1.3.6 (K2) Привести примеры метрик успеха и трудностей, связанных с тестированием на основе рисков

1.4 Стратегия тестирования проекта

TM-1.4.1 (K2) Объяснить типичные варианты подхода к тестированию

TM-1.4.2 (K4) Проанализировать корпоративную стратегию тестирования и контекст проекта, чтобы выбрать подходящий подход к тестированию

TM-1.4.3 (K3) Использовать методологию достижения целей S.M.A.R.T. для определения измеримых целей тестирования и критериев выхода

1.5 Совершенствование процесса тестирования

TM-1.5.1 (K2) Объяснить, как использовать модель IDEAL для совершенствования процесса тестирования в конкретном проекте

TM-1.5.2 (K2) Обобщить подход к совершенствованию процесса тестирования на основе моделей и понять, как его применять в контексте проекта

TM-1.5.3 (K2) Обобщить подход на основе аналитики к совершенствованию процесса тестирования и понять, как его применять в контексте проекта

TM-1.5.4 (K3) Реализовать ретроспективу проекта или итерации, чтобы оценить процессы тестирования и выявить области тестирования, требующие совершенствования

1.6 Инструменты тестирования

TM-1.6.1 (K2) Обобщить лучшие практики внедрения инструментов

TM-1.6.2 (K2) Объяснить влияние различных технических и бизнес-аспектов при выборе типа инструмента

TM-1.6.3 (K4) Проанализировать конкретную ситуацию, чтобы составить план выбора инструмента, охватывающий риски, затраты и выгоды

TM-1.6.4 (K2) Различить этапы жизненного цикла инструмента

TM-1.6.5 (K2) Привести примеры сбора и оценки метрик с помощью инструментов

1.1 Процесс тестирования

Введение

Программа обучения Базового уровня, сертифицированного тестировщика ISTQB® версии 4 описывает процесс тестирования, который включает следующие активности: планирование тестирования, мониторинг и контроль тестирования, анализ тестирования, проектирование тестов, реализация тестов, выполнение тестов и завершение тестирования.

В программе обучения Базового уровня ISTQB® версии 4 указано, что эти активности в процессе тестирования часто реализуются итеративно или параллельно, в зависимости от модели жизненного цикла разработки программного обеспечения и контекста проекта. Обычно требуется адаптация этих активностей в контексте продукта и проекта.

В данной программе обучения основное внимание уделяется следующим ключевым видам деятельности по управлению тестированием:

- **Планирование тестирования:** определение целей тестирования, объема тестирования, ресурсов тестирования, графика тестирования, результатов тестирования и участников тестирования (заинтересованных сторон тестирования).
- **Мониторинг и контроль тестирования:** отслеживание прогресса тестирования, результатов тестирования и отклонений в тестах; принятие корректирующих мер при необходимости; отчет о статусе и результатах тестирования соответствующим заинтересованным сторонам.
- **Завершение тестирования:** доработка и архивирование артефактов тестирования, оценка процесса тестирования и тестового продукта, определение действий по совершенствованию процесса тестирования и сообщение о завершении тестирования соответствующим заинтересованным сторонам.

Стандарт ISO/IEC/IEEE 29119-2 определяет процессы управления тестированием, которые охватывают эти виды активностей по управлению тестированием. Эти процессы управления тестированием могут применяться на разных уровнях тестирования, таких как проект, программа или совокупность программ. Каждый уровень тестирования может иметь свой собственный план тестирования, соответствующий плану тестирования более высокого уровня.

1.1.1. Активности планирования тестирования

В этом разделе основное внимание уделяется активностям по планированию тестирования для различных областей, таких как весь проект, уровень тестирования, тип тестирования или релиз/итерация/спринт в гибкой методологии. В зависимости от объема планирование тестирование может начинаться и заканчиваться на разных этапах процесса разработки. Планирование тестирования — это действия, которые включают определение активностей и ресурсов, необходимых для достижения целей тестирования, определенных в политике тестирования. Планирование тестирования должно быть начато как можно раньше в процессе разработки, желательно до того, как будут определены требования, и должно обновляться по мере выполнения проекта. Планирование тестирования часто представляет собой итеративный процесс, требующий перепланирования в ходе проекта для учета изменений и обратной связи.

Следующие задачи являются частью планирования тестирования (аналогично тем, которые содержатся в ISO/IEC/IEEE 29119-2):

- **Понять контекст и организовать планирование тестирования**

Понимание контекста организации (например, политики тестирования и корпоративной стратегии тестирования), объема тестирования и элемента тестирования (т.е. тестируемого рабочего продукта) имеет решающее значение для планирования тестирования. (см. Раздел 1.2 Контекст тестирования). Это также включает в себя все активности, необходимые для организации разработки плана тестирования и получения одобрения этих активностей и графика заинтересованными сторонами (например, владельцем продукта, руководителем проекта или руководителем группы разработки).

- **Определить и проанализировать риски продукта**
  Анализ рисков включает в себя определение и оценку потенциального воздействия и вероятности возникновения рисков продукта в рамках планирования тестирования. См. Раздел 1.3 Тестирование на основе рисков данной программы обучения для получения более подробной информации о рисках, связанных с продуктом.
- **Определить подходы к обработке рисков**
  На основе анализа рисков выбираются соответствующие подходы к обработке рисков, которые документируются в плане тестирования. Они могут включать предупреждающие, корректирующие или смягчающие активности для определенных ранее рисков.
- **Определить подход к тестированию, оценить и распределить ресурсы тестирования**
  На основе корпоративной стратегии тестирования, стандартов, любых ограничений, заданных проектом, и подходов к обработке рисков определяется подход к тестированию для текущего объема тестирования (см. Раздел 1.4 Стратегия тестирования проекта). Когда определен подход к тестированию, важно оценить необходимые ресурсы тестирования, такие как персонал тестирования, инструменты тестирования, тестовое окружение и тестовые данные, а также распределить эти ресурсы по активностям тестирования.
- **Составить план тестирования**
  План тестирования должен быть одобрен всеми заинтересованными сторонами, и, следовательно, разногласия между ними должны быть урегулированы.

1.1.2. Активности мониторинга и контроля тестирования

Чтобы управление тестированием могло обеспечить эффективный контроль тестирования, необходимо установить график тестирования и систему мониторинга, позволяющую отслеживать статус и прогресс тестирования. Эта система должна включать подробные измерения и цели, которые необходимы для соотнесения статуса рабочих продуктов тестирования и ресурсов с планом и стратегическими целями.

Для небольших и несложных проектов может быть относительно легко связать рабочие продукты и активности тестирования с планом и стратегическими целями, но обычно для достижения этой цели необходимо определить более подробные цели. Это может включать определение измерений и целей тестирования, а также охвата базиса тестирования.

Особое значение имеет необходимость соотносить статус рабочих продуктов тестирования и активностей таким образом, чтобы это было понятно и актуально для заинтересованных сторон проекта и бизнеса.

Мониторинг и контроль тестирования — это непрерывная активность.

Контроль тестирования сравнивает фактический прогресс с планом тестирования и при необходимости осуществляет корректирующие активности. Он направляет тестирование в соответствии со стратегиями и целями тестирования (см. Раздел 1.4 Стратегия тестирования

проекта) и при необходимости пересматривает активности планирования тестирования. Данные контроля тестирования требуют подробной информации о планировании соответствующей реакции тестирования. Эта активность предполагает:

- Внедрение плана тестирования и указаний по контролю
- Управление отклонениями от запланированного тестирования
- Обработка вновь выявленных и изменившихся рисков
- Определение готовности начала тестирования
- Предоставление и получение разрешения на завершение тестирования на основе критериев выхода

Мониторинг тестирования включает сбор и запись результатов тестирования, выявление отклонений от запланированного тестирования, выявление и анализ новых рисков, требующих тестирования, а также мониторинг изменений выявленных рисков.

1.1.3. Активности завершения тестирования

Завершение тестирования обычно происходит на вехах проекта (например, релиз, окончание итерации или достижение критериев завершения уровня тестирования). Для любых не устраненных дефектов создаются запросы на изменения или элементы набора планируемых задач (бэклога) по продукту – см. программу обучения Базового уровня ISTQB® версии 4. Как только критерии выхода будут выполнены, ключевые результаты должны быть зафиксированы, заархивированы и предоставлены соответствующим заинтересованным сторонам. Для завершения тестирования необходимо выполнить следующие задачи:

- **Создать и утвердить отчет о завершении тестирования**
  Эта задача гарантирует, что все тестирование завершено и все цели тестирования достигнуты. Эта задача включает в себя сбор соответствующей информации из различного тестового обеспечения, такой как планы тестирования, результаты тестирования, отчеты о ходе тестирования, отчеты о завершении тестирования и отчеты о дефектах. Собранная информация оценивается и обобщается в отчете о завершении тестирования. Отчет о завершении тестирования утверждается и передается соответствующим заинтересованным сторонам.
- **Заархивировать тестовое обеспечение**
  Эта задача определяет тестовое обеспечение, которое может пригодиться в будущем или которое, как ожидается, будет использовано повторно, например, сценарии тестирования. Это делает их доступными и понятными для будущего повторного использования. Кроме того, результаты тестирования, протоколы тестирования, отчеты о тестировании и другое тестовое обеспечение должны быть временно заархивированы в системе управления конфигурацией.
- **Передать тестовое обеспечение**
  Эта задача предусматривает поставку ценных рабочих продуктов тем, кто в них нуждается. Например, известные дефекты, отложенные или принятые, должны быть доведены до сведения тех, кто будет использовать или поддерживать использование тестового обеспечения.
- **Выполнить все необходимые задачи по очистке тестового окружения и его восстановлению до ранее заданного состояния**
  Эта задача гарантирует, что тестовое окружение готово к следующему циклу тестирования

или проекту. Она включает в себя удаление всех тестовых данных, инструментов тестирования, тестовых драйверов, тестовых заглушек и сценариев тестирования из тестового окружения. Это также включает в себя восстановление тестового окружения до исходного или желаемого состояния.

- **Выполнить/собрать/задокументировать извлеченные уроки**
  Эта задача выполняется в ретроспективах, где обсуждаются и документируются важные уроки, извлеченные в процессе тестирования. Сюда могут входить результаты всего жизненного цикла разработки программного обеспечения (ЖЦ ПО). Извлеченные уроки можно использовать для совершенствования процесса тестирования, как описано в Разделе 1.5 Совершенствование процесса тестирования данной программы обучения.

1.2 Контекст тестирования

Введение

Контекст тестирования включает в себя уникальные условия и ограничения, которые влияют на процесс тестирования, формируя решения и стратегии планирования, проектирования и выполнения тестов. Для руководителей тестирования крайне важно понимать этот контекст, чтобы согласовать тестирование с конкретными потребностями и целями проекта разработки программного обеспечения. Этот контекст может различаться в зависимости от типа продукта, отрасли, регуляторных требований и, что особенно важно, от используемого жизненного цикла разработки программного обеспечения.

Руководителям тестирования поручено применять установленные стратегии тестирования и выбирать методы тестирования, а не разрабатывать их. Руководители тестирования играют ключевую роль в разработке планов тестирования, адаптированных к контексту проекта. Понимая и учитывая эти различные факторы, руководители тестирования могут гарантировать, что тестирование будет надлежащим, эффективным и результативным для достижения целей тестирования.

1.2.1. Заинтересованные стороны тестирования

Участники тестирования — это лица или группы, имеющие прямую или косвенную заинтересованность в качестве продукта. Ниже приведен типичный список потенциальных заинтересованных сторон, отражающий их разнообразные интересы в области тестирования:

- **Разработчики, ведущие разработчики и руководители разработки:** помимо реализации тестируемой системы и действий по результатам тестирования, эти заинтересованные стороны также участвуют в компонентном тестировании и вносят свой вклад в процесс тестирования.
- **Тестировщики, ведущие специалисты по тестированию и руководители тестирования:** эти специалисты подготавливают тестовое обеспечения, которое включает в себя разработку планов тестирования и участвуют в процессе тестирования посредством таких активностей, как анализ требований, проектирование тестов, выполнение тестов, отслеживание дефектов и отчетность по ним, автоматизация тестирования и отчетность о ходе тестирования.
- **Руководители проектов, владельцы продуктов и бизнес-пользователи:** они определяют требования, устанавливают требуемый уровень качества и рекомендуют необходимое покрытие на основе предполагаемых рисков. Они также рецензируют

рабочие продукты, участвуют в пользовательском приемочном тестировании и принимают решения на основе результатов тестирования.

- **Персонал, занимающийся операционной деятельностью:** участвуя в операционном приемочном тестировании, они обеспечивают готовность системы к эксплуатации и вносят вклад в определение нефункциональных требований.
- **Клиенты и пользователи:** клиенты приобретают продукт, а пользователи непосредственно его используют. Обе категории являются ключевыми в определении требований и должны участвовать в пользовательском приемочном тестировании, чтобы подтвердить, что продукт соответствует их потребностям.

Этот список не включает все потенциальные заинтересованные стороны. Руководители тестирования должны провести анализ заинтересованных сторон в рамках создания стратегии тестирования и плана тестирования, учитывая объем тестирования при определении конкретных заинтересованных сторон для своего проекта.

1.2.2. Важность знаний заинтересованных сторон в управлении тестированием

В управлении тестированием крайне важно учитывать точки зрения и влияние различных заинтересованных сторон. Матрица заинтересованных сторон, часто называемая матрицей власти и интересов, помогает руководителям тестирования приоритизировать взаимодействие с заинтересованными сторонами и эффективно управлять ожиданиями. Матрица заинтересованных сторон является стратегическим инструментом в управлении тестированием, который:

- Использует экспертизу заинтересованных сторон, при этом конечные пользователи и технические команды предоставляют обратную связь и мнения относительно производительности и безопасности.
- Способствует управлению рисками, выделяя интересы и влияние заинтересованных сторон, стимулируя проактивные усилия по смягчению рисков.
- Ценит разнообразные точки зрения и полезную обратную связь.

Матрица заинтересованных сторон состоит из четырех квадрантов:

- **Промоутеры (высокое влияние, высокий интерес):** ключевые сотрудники с высоким уровнем влияния и заинтересованности, имеющие решающее значение для формирования стратегии и плана тестирования.
- **Латентные (высокое влияние, низкий интерес):** хотя они могут не проявлять сильного интереса к повседневным задачам, их решения критически важны для распределения ресурсов и определения высокоуровневого направления проекта.
- **Защитники (низкое влияние, высокий интерес):** они часто предоставляют качественную обратную связь и могут поддерживать вовлеченность за счет регулярных обновлений и участия в конкретных обсуждениях.
- **Апатичные (низкое влияние, низкий интерес):** несмотря на то, что они не участвуют в тесном сотрудничестве, информирование их о важных вехах и обращение к ним с просьбой внести свой вклад в решение конкретных вопросов может дать уникальные сведения.

Роль руководителя тестирования включает составление подробного списка заинтересованных сторон и понимание связи каждой из них с активностями по тестированию, используя матрицу заинтересованных сторон для повышения эффективности практик управления тестированием.

Рисунок 1. Различные типы заинтересованных сторон

1.2.3. Управление тестированием в гибридной модели разработки программного обеспечения

Гибридные модели разработки программного обеспечения интегрируют элементы как из традиционных последовательных подходов, так и из практик гибкой методологии, чтобы соответствовать конкретным потребностям проекта или организационным переходам. Следующие причины являются типичными для использования гибридной модели разработки программного обеспечения, хотя в зависимости от организации и проекта могут быть и другие причины:

- **Гибрид как переход к гибкой методологии:** Переход от традиционных методологий к гибким может быть сложным из-за фундаментальных изменений в рабочих процессах, культуре и динамике команды. Гибридные модели предоставляют сбалансированный подход, который облегчает этот переход, сочетая структуру традиционных методов с особенностями гибких практик.
- **Гибрид, соответствующий назначению:** Некоторые организации или проекты, могут быть не в состоянии перейти на гибкие методологии. Проекты с высоким уровнем риска могут требовать последовательного выполнения некоторых задач и гибких практик для других. Они могут использовать гибридную модель, поскольку это соответствует их целям.

В гибридной среде активности по управлению тестированием могут включать:

- Оценку понимания и способности команды плавно переходить от традиционных к гибким методологиям
- Выявление сильных и слабых сторон при адаптации к гибридному подходу
- Обеспечение того, чтобы команда умела сочетать структурированные процессы с гибкостью гибких методологий
- Улучшение сотрудничества между командой тестирования и заинтересованными сторонами для лучшего управления тестированием в рамках спринтов и традиционных этапов тестирования

- Участие в скоординированных усилиях, таких как объединение нескольких команд для поставки сложных решений (Scrum-of-Scrums) для тестировщиков, чтобы сохранить фокус на тестировании, одновременно способствуя достижению общих целей разработки
- Отслеживание и анализ трудозатрат по тестированию и сценариев тестирования в рамках спринтов, чтобы убедиться, что они соответствуют практикам гибких методологий.

Дополнительную информацию можно найти в (Fowler, 2010).

1.2.4. Активности по управлению тестированием для различных моделей жизненного цикла разработки программного обеспечения

Чтобы правильно согласовать тестирование с моделью ЖЦ ПО, руководитель тестирования должен понимать различные модели ЖЦ ПО, используемые в его организации, и использовать эти знания для правильного согласования тестирования с активностями по разработке.

В таблице ниже показано сравнение различных активностей по управлению тестированием, основанных на различных моделях ЖЦ ПО:

Аспект	**Последовательная модель разработки например, V-Model**	**Гибкая методология разработки программного обеспечения например, Scrum**
Оценка трудозатрат на тестирование	Ранняя детализированная оценка для каждого уровня тестирования.	Итеративная оценка, часть планирования пользовательских историй на каждую итерацию.
Тестовое обеспечение	Включает стратегию, план, сценарии, расписание и отчеты.	Сосредоточен на критериях приёмки и определении завершенности; минимальная документация.
Роли	Руководитель тестирования контролирует принятие решений и управление командой.	Роли интегрированы; координатор или наставник (коуч) заменяет традиционного руководителя тестирования.
Инструменты	Преимущественно инструменты управления тестированием, подходящие для тестирования на основе этапов.	Инструменты непрерывной интеграции/непрерывного развертывания (CI/CD) и автоматизации являются основными, поддерживая непрерывное тестирование.
Подход к тестированию	Запланирован заранее, соответствует этапам проекта	Встроен в итерации, с акцентом на приспособляемость и обратную связь.
Автоматизация тестирования	Реализуется стратегически, может происходить на различных этапах	Встроена с самого начала, с акцентом на автоматизированное регрессионное тестирование в CI/CD
Мониторинг и отчетность	Отчетность на основе вех, с возможностью автоматизированных сводных таблиц	Непрерывная отчетность с реальными сводными таблицами и ежедневными обновлениями статуса

Аспект	Последовательная модель разработки например, V-Model	Гибкая методология разработки программного обеспечения например, Scrum
Метрики	Сосредоточены на традиционных метриках тестирования и управлении дефектами (например, выполнение тестов, уровни дефектов)	Включают метрики гибкой методологии для отслеживания итераций в дополнение к традиционным метрикам (например, скорость команды, диаграммы сгорания)

Таблица 1: Активности по управлению тестированием для различных моделей жизненного цикла разработки программного обеспечения

1.2.5. Активности по управлению тестированием на различных уровнях тестирования

Компонентное тестирование:

- Определить объем, цели и критерии завершения компонентного тестирования (модульного тестирования).
- Вовлечь тестировщиков в деятельность, выходящую за рамки традиционных ролей тестирования, такую как рецензирование кода, где их аналитические навыки приносят добавленную стоимость.
- Скоординировать работу с командой разработчиков для решения проблем и участия в компонентном тестировании.

Интеграционное тестирование компонентов:

- Определить последовательности интеграции и комбинации тестов в сотрудничестве с командой разработки, учитывая модель жизненного цикла разработки программного обеспечения, инструменты и процессы.
- Контролировать прогресс, чтобы убедиться, что он соответствует стратегиям системного и приемочного тестирования.
- Управлять этим этапом совместно с разработчиками, учитывая интеграционное тестирование компонентов (модулей).

Системное интеграционное тестирование (интеграция систем):

- Убедиться, что объем и цели системного интеграционного тестирования ясны и соответствуют оценкам рисков и целям качества.
- Поддерживать контроль прогресса, результатов и управление проблемами во время системного интеграционного тестирования.

Системное тестирование (системы систем):

- Адаптировать планирование к модели жизненного цикла разработки программного обеспечения, с тщательным распределением ресурсов, выбором инструментов и составлением расписания.
    - Для проектов, работающих по гибкой методологии разработки программного обеспечения (Agile), интегрировать системное тестирование с итеративным тестированием пользовательских историй, избегая отдельных этапов тестирования

и обеспечивая непрерывное и интегрированное тестирование. В то время как в последовательных моделях тестирование может следовать запланированным этапам.

Приемочное тестирование:

- Сотрудничать с заинтересованными сторонами для рецензирования и подтверждения выполнения критериев приёмки и планирования активностей по тестированию, включая управление пользовательским тестированием в пользовательском приёмочном тестировании.
- Координировать логистику приёмочного тестирования, способствуя проведению тестов на площадке заказчика, чтобы убедиться, что продукт соответствует бизнес-требованиям и стандартам качества вне окружения разработки.
- Способствовать разрешению любых проблем с пользовательским приёмочным тестированием и сопровождению заинтересованных сторон в процессе утверждения продукта при выполнении критериев приёмки.

1.2.6. Активности по управлению тестированием для различных типов тестирования

Эффективное управление тестированием требует интегрированного подхода, который учитывает уникальные требования функционального, нефункционального, тестирования методом черного ящика и тестирования методом белого ящика. Для руководителей, осуществляющих функциональное тестирование, основное внимание уделяется обеспечению того, чтобы все функциональности были тщательно протестированы и соответствовали конкретным требованиям. Управление нефункциональным тестированием сосредоточено вокруг проверки атрибутов системы, таких как производительность и безопасность. Управление тестированием методом черного ящика включает в себя обеспечение того, чтобы тесты были ориентированы на пользователя и охватывали все возможные внешние взаимодействия. Управление тестированием методом белого ящика подчеркивает понимание структуры кода и обеспечение того, чтобы тесты тщательно покрывали внутреннюю логику.

Управление функциональным тестированием:

- Стратегическое планирование и отслеживание прогресса: создайте детализированную стратегию тестирования, которая соответствует функциональным требованиям и целям проекта, а также мониторинг прогресса.
- Координация ресурсов: эффективно распределите людские и технические ресурсы для покрытия всех функциональных аспектов системы.

Управление нефункциональным тестированием:

- Оценка производительности: установите эталоны производительности и управляйте активностями по тестированию, которые оценивают систему по этим критериям.
- Верификация соответствия: осуществляйте контроль тестов, которые обеспечивают соответствие системы нефункциональным стандартам, таким как безопасность, практичность и надежность.

Управление тестированием методом черного ящика:

- Анализ покрытия тестов: обеспечьте, чтобы тесты методом черного ящика охватывали все пользовательские сценарии и бизнес-требования.

- Учет обратной связи: управляйте процессом сбора обратной связи от заинтересованных сторон для уточнения подходов к тестированию методом черного ящика и исправления дефектов.

Управление тестированием методом белого ящика:

- Оптимизация покрытия кода: контролируйте использование инструментов покрытия кода для выявления пробелов в тестировании методом белого ящика и направление ресурсов на устранение этих областей.
- Интеграция технических знаний: управляйте включением технической информации в процесс планирования тестирования, обеспечивая, чтобы тесты разрабатывались с пониманием внутренней работы приложения.

1.2.7. Активности по управлению тестированием для планирования, мониторинга и контроля

Эффективное управление тестированием является краеугольным камнем любого успешного тестирования, охватывая широкий спектр активностей, требующих тщательного планирования, бдительного мониторинга и стратегического контроля. Руководители тестирования играют ключевую роль в обеспечении того, чтобы процесс тестирования был не только результативным и эффективным, но и адаптированным к уникальным требованиям текущего проекта.

Планирование тестирования:

- **Всестороннее определение объема:** План тестирования должен быть тщательно составлен, включая подробное определение объема тестирования. Это включает в себя идентификацию всех функциональных и нефункциональных требований для обеспечения полного покрытия рабочими продуктами тестирования. Также необходимо учитывать последствия использования методологий тестирования методом черного ящика и методом белого ящика, гарантируя, что разработанные сценарии тестирования были способны валидировать тестируемую систему со всех сторон.
- **Оценка и план смягчения рисков:** Неотъемлемой частью плана тестирования является надежная структура управления рисками. Руководители тестирования должны провести детальный анализ рисков, выявляя потенциальные уязвимости и проблемы, которые могут повлиять как на рабочий процесс проекта, так и на конечный продукт. Разработка стратегий смягчения рисков имеет решающее значение, включая упреждающее планирование для эффективного обхода или минимизации этих рисков.
- **Стратегия распределения ресурсов:** Планирование ресурсов является еще одним критически важным элементом. Это выходит за рамки простого назначения ресурсов и включает в себя определение структуры команды, распределение ролей и установление протоколов коммуникации. В условиях, где команды распределены, как в моделях onsite/offsite, это становится особенно важным для поддержания синхронности и обеспечения бесшовного сотрудничества.

Мониторинг тестирования:

- **Контроль выполнения:** Мониторинг играет центральную роль в процессе управления тестированием. Он включает в себя постоянное рецензирование выполнения тестов в соответствии с установленным планом тестирования, отслеживание прогресса тестирования и управление любыми возникающими дефектами. Корректировка приоритетов тестирования на основе оценки риска и текущих событий обеспечивает, что

тестирование остается сфокусированным и согласованным с наиболее критическими областями.

- **Оптимизация инструментов и окружения:** Разумный выбор и использование инструментов тестирования и тестовых окружений имеют решающее значение для поддержки стратегии тестирования. Непрерывный мониторинг гарантирует их эффективную интеграцию в конвейер CI/CD, способствуя циклам непрерывного тестирования и немедленной обратной связи, которые жизненно важны для гибкой методологии разработки программного обеспечения (Agile).
- **Сотрудничество с разработчиками:** Поддержание тесных рабочих отношений с командой разработчиков имеет важное значение для успешных результатов тестирования. Это сотрудничество должно поддерживать исчерпывающий подход к тестированию, используя информацию как с точки зрения методов белого ящика, так и с точки зрения методов черного ящика, для упреждающего решения потенциальных проблем.

Контроль тестирования:

- **Адаптивное управление процессом:** Контроль тестирования — это динамическая корректировка процесса тестирования в ответ на новые идеи, проблемы и развивающуюся динамику проекта. Руководитель тестирования должен быть отзывчивым и гибким, способным вносить изменения в подход к тестированию, которые отражают текущее состояние проекта.
- **Управление критериями качества:** Структурированный подход к управлению критериями качества (воротами качества) имеет основополагающее значение. Это включает в себя определение того, что представляет собой контроль качества в жизненном цикле тестирования и принятие обоснованных решений о ходе этапа тестирования, что играет важную роль в поддержании целостности продукта.

Сосредотачиваясь на этих конкретных активностях в рамках планирования, мониторинга и контроля тестирования, руководители тестирования могут гарантировать, что процесс тестирования будет четко определен, адаптируем к изменениям в проекте и приведет к созданию продукта, который соответствует как требованиям проекта, так и ожиданиям заинтересованных сторон.

1.3 Тестирование на основе рисков

Введение

Тестирование на основе рисков включает в себя определение, оценку, мониторинг и смягчение рисков управления тестированием. Эти риски определяются различными заинтересованными сторонами и могут использоваться для выбора и приоритизации тестов. Чем выше уровень риска, тем раньше должно начинаться и тем более интенсивным и продолжительным должно быть тестирование.

1.3.1. Тестирование как активность по смягчению рисков

Риск продукта — это потенциальная ситуация, при которой в продукте могут возникнуть проблемы с качеством. Когда тесты выявляют дефекты, тестирование помогает смягчить риск продукта, предоставляя информацию о дефектах и возможности для их устранения до выпуска. Когда тесты перестали находить дефекты это указывает на то, что уровень риска продукта ниже, чем ожидалось.

Среди прочего, руководитель тестирования отвечает за предоставление корректной и надежной оценки качества продукта. Это требует его активного участия в управлении рисками проекта с

акцентом на риски проекта, связанные с обеспечением качества (например, неоднозначные требования, которые приводят к серьезным проблемам на поздних этапах валидации, недостаточные тестовые окружения, препятствующие выполнению тестов).

Тестирование на основе рисков фокусирует тестирование на рисках качества. Оно следует общему процессу управления рисками, который состоит из следующих основных активностей:

- Анализ рисков, включающий определение и оценку рисков
- Контроль рисков, включающий мониторинг и смягчение рисков

Эти основные активности упорядочены, но могут пересекаться.

Чтобы сфокусировать тестирование на рисках качества, необходимо определить и оценить риски качества. Для достижения наибольшей эффективности анализ рисков должен включать различные заинтересованные стороны. Будучи основным заинтересованным лицом в анализе рисков качества, руководитель тестирования должен понимать и осуществлять мониторинг этих активностей и быть в состоянии их координировать.

Мониторинг тестирования должен включать мониторинг рисков. Помимо мониторинга эволюции известных рисков качества, он должен включать анализ любых новых рисков качества и корректировку реестра рисков.

Руководитель тестирования является одним из нескольких специалистов, которые управляют смягчением рисков качества. Смягчение рисков распределено по нескольким активностям тестирования. Например, результаты анализа рисков качества используются при планировании тестирования для сосредоточения тестирования на правильных областях с использованием правильных методик. При анализе тестирования уровни рисков направляют выбор тестовых условий, которые должны быть покрыты. При выполнении тестов приоритизация на основе рисков определяет последовательность выполнения тестов.

1.3.2. Определение рисков качества

Задача руководителя тестирования заключается в сборе рисков от заинтересованных сторон. Заинтересованные стороны могут выявлять риски качества с помощью одного или нескольких из следующих методов:

- Получение информации от экспертов
- Независимые оценки
- Ретроспективы
- Совещания по рискам
- Мозговой штурм
- Чек-листы
- Обращение к прошлому опыту

Привлечение максимально широкого круга заинтересованных сторон часто позволяет выявить большинство значительных рисков продукта. Определить, какие заинтересованные стороны должны участвовать на этом этапе, очень важно. Необходимо убедиться, что список участвующих заинтересованных сторон является исчерпывающим и согласован с руководителем проекта. Пропуск ключевых заинтересованных сторон при определении рисков может быть очень проблематичным. Важно обеспечить, чтобы все соответствующие заинтересованные стороны имели возможность участвовать. Если они не могут участвовать, они должны хотя бы иметь

возможность делегировать задачу. Если ключевые заинтересованные стороны могут быть не представлены, можно провести установочное совещание, чтобы определить, отсутствуют ли они.

В тестировании на основе рисков важно понимать, что риск не распределен равномерно среди тестируемых объектов. Например, компоненты приложения, ориентированные на конечных пользователей, на заказчика, могут иметь очень разные риски удобства использования по сравнению с административными компонентами. Определение индивидуальных рисков различных элементов тестирования является важной задачей при планировании тестирования.

Определение рисков часто приводит к побочным результатам, таким как выявление ситуаций, которые не являются рисками продукта. Примеры включают общие ситуации или вопросы, связанные с продуктом или проектом, или проблемы в документации, таких как требования и спецификации проектирования. Риски проекта также часто выявляются как побочный результат определения рисков качества, но не являются целью тестирования на основе рисков. Руководитель тестирования может играть значительную роль в подчеркивании этих побочных результатов и разъяснении, что качество является заботой всех. Плохие или отсутствующие требования часто указывают на более фундаментальную проблему в планировании и подготовке, так как обеспечение качества вовлечено на протяжении всего жизненного цикла разработки программного обеспечения.

1.3.3. Оценка рисков качества

После выявления рисков их можно оценить. Оценка рисков качества включает категоризацию рисков по типу (риск продукта или проекта) и по затронутым характеристикам качества.

Определение уровня риска обычно включает оценку вероятности возникновения риска и его влияния при возникновении. Факторы, влияющие на вероятность риска для рисков качества, включают:

- Сложность технологии, инструментов или архитектуры системы
- Зрелость организации
- Проблемы с персоналом, связанные с навыками, доступностью, мотивацией или автономной работой, включая знание используемого жизненного цикла разработки программного обеспечения
- Конфликты в команде
- Проблемы с контрактами с поставщиками
- Географически распределенные команды
- Слабое управленческое или техническое руководство
- Нехватку времени, ресурсов, бюджета и управления
- Отсутствие ранних активностей по обеспечению качества
- Высокие темпы изменений базиса тестирования, продукта или персонала

Факторы, воздействующие на влияние риска, включают:

- Частоту использования затронутой функции
- Критичность затронутой функции
- Критичность затронутой бизнес-цели
- Ущерб репутации

- Потерю дохода от бизнеса
- Потенциальные финансовые, экологические или социальные потери, или ответственность
- Гражданские или уголовные правовые санкции
- Проблемы с интерфейсами и интеграцией
- Отсутствие разумных обходных путей
- Потребности в безопасности

Руководитель тестирования комбинирует вероятность риска и его влияние для определения уровня риска.

Если анализ рисков основан на обширных и статистически значимых данных о рисках, уместна количественная оценка. Например, вероятность риска может быть выражена в процентах, а влияние риска — в сумме. В таком случае уровень риска можно рассчитать, как произведение этих двух факторов. Однако, как правило, вероятность риска и его влияние можно определить только качественно на порядковых шкалах, например, как очень высокий, высокий, средний, низкий или очень низкий. Значения вероятности риска и его влияния затем комбинируются с использованием матриц рисков для создания совокупного уровня риска. Этот совокупный уровень риска следует интерпретировать как качественную, относительную оценку на порядковой шкале.

Если анализ рисков не основан на обширных и статистически значимых данных о рисках, анализ рисков будет качественным, основанным на субъективных восприятиях заинтересованных сторон относительно вероятности риска и его влияния.

1.3.4. Смягчение рисков качества путем подходящего тестирования

В разработке программного обеспечения тестирование является наиболее важной активностью по смягчению рисков качества и позволяет снизить вероятность отказов. Другие возможные меры по смягчению рисков включают план действий на случай непредвиденных обстоятельств (например, предоставление обходных решений), передачу риска третьей стороне (например, поставщику компонента) или принятие риска.

При планировании тестирования время и усилия, связанные с разработкой и выполнением тестирования, должны быть пропорциональны уровню риска: тестирование для более высоких уровней риска должно начинаться рано и использовать более строгие методы тестирования, в то время как тестирование для более низких уровней риска может начинаться позже и использовать менее строгие методы тестирования. Чтобы наилучшим образом смягчить общий риск через тестирование, руководитель тестирования должен проанализировать следующие контекстные факторы и выбрать соответствующий подход к тестированию:

- **Элементы тестирования:** разные элементы тестирования в рамках объекта тестирования могут иметь разные уровни одного и того же типа риска, поэтому объект тестирования не нужно тестировать с единой строгостью.
- **Характеристики качества:** риски, влияющие на конкретные характеристики качества, должны быть смягчены соответствующими типами тестирования, которые требуют определенных усилий по тестированию, тестовых окружений и навыков тестирования.
- **Уровни тестирования и типы тестирования:** некоторые риски могут быть протестированы только динамически на определенных уровнях тестирования; другие — статическим тестированием (например, статический анализ и рецензирование кода на предмет сопровождаемости) или комбинацией обоих методов (например, рецензирование

архитектуры и динамическое тестирование интегрированной системы на уязвимости безопасности). Тестирование каждого элемента тестирования как можно раньше смягчает риск обнаружения критических дефектов на поздних стадиях жизненного цикла, который в свою очередь может привести к более высоким внутренним затратам и задержкам.

- **Жизненный цикл разработки программного обеспечения (ЖЦ ПО):** активности тестирования имеют свои собственные критерии входа. Различные ЖЦ ПО выполняют их в разное время.
- **Команда тестирования:** наиболее квалифицированные специалисты должны проводить тестирование элементов тестирования с наивысшими уровнями риска.
- **Регуляторные требования:** некоторые стандарты, связанные с безопасностью (например, стандарт IEC 61508), предписывают методы тестирования и требуемое покрытие на основе уровня целостности. Руководитель тестирования должен обеспечить соблюдение этих стандартов.

Кроме того, уровень риска должен влиять на решения по контролю качества, такие как использование рецензирования рабочих продуктов, таких как сценарии тестирования, уровень независимости тестирования от разработки и объем выполняемого регрессионного тестирования.

В ходе мониторинга и контроля тестирования, тестирование на основе рисков позволяет сообщать о прогрессе тестирования с точки зрения остаточного уровня риска в любой момент времени. Это помогает команде разработчиков и заинтересованным сторонам отслеживать и контролировать разработку программного обеспечения, включая принятие решений о выпуске на основе остаточного уровня риска. Для этого необходимо сообщать о результатах испытаний с точки зрения рисков таким образом, чтобы заинтересованные стороны могли их понять.

Во время реализации тестов приоритизация тестирования основана на уровнях риска. Во время выполнения тестирования это обеспечивает раннее покрытие наиболее критических областей и смягчение рисков наивысшего уровня.

- В некоторых случаях тесты приоритизируются для выполнения в строгом порядке убывания уровней риска, начиная с наивысшего. Этот подход называется тестированием вглубь (depth-first) и подходит, когда важно смягчить риски наивысшего уровня как можно раньше.
- В качестве альтернативы, по крайней мере один тест для каждого риска назначается с наивысшим приоритетом. Все остальные тесты приоритизируются на основе уровней риска, которые они покрывают. Этот подход называется тестированием вширь (breadth-first) и подходит, когда заинтересованные стороны хотят получить общее представление о качестве продукта как можно раньше. На практике тестирование часто начинается с тестирования вглубь, но по мере сокращения времени переключается на тестирование вширь, тестируя все оставшиеся элементы риска хотя бы один раз.

Независимо от того, проводится ли тестирование на основе риска вглубь, вширь или комбинированно, время, выделенное на тестирование, может быть израсходовано без выполнения всех запланированных тестов. В этот момент тестирование на основе рисков облегчает предоставление обоснованной рекомендации руководству о продлении тестирования или принятии оставшегося риска.

1.3.5. Методы тестирования на основе рисков

Существует ряд конкретных методов с различной степенью формальности для реализации тестирования на основе рисков. Пригодность метода зависит от особенностей проекта, процесса и

продукта. Существуют два основных типа методов: тяжеловесные (heavyweight) или облегченные (lightweight). В критически важных для безопасности системах часто используются тяжеловесные методы. В приложениях, не критичных по безопасности, обычно используются облегченные методы.

Тяжелые методы являются формальными, используют определенные процедуры и подробную документацию. Они вовлекают широкие группы заинтересованных сторон. Оценка рисков в рамках тяжелых методов использует детализированные факторы вероятности и влияния риска, а также математические формулы для расчета вероятности и влияния риска на основе этих факторов. Примеры тяжеловесных методов:

- **Анализ опасностей:** расширяет аналитический процесс, пытаясь выявить опасности, лежащие в основе каждого риска
- **Анализ стоимости:** определяет для каждого элемента риска качества вероятность отказа, стоимость потери, связанной с типичным отказом, и стоимость тестирования на такие отказы
- **Анализ видов и последствий отказов и его варианты:** идентифицирует риски качества, их потенциальные причины и вероятные последствия, а затем присваивает уровни критичности, приоритета и обнаружения
- **Анализ дерева отказов:** связывает потенциальные отказы с дефектами, которые могут вызвать отказ, затем с ошибками, которые могут вызвать эти дефекты, продолжая до тех пор, пока не будут выявлены различные первопричины.

Напротив, легковесные методы менее детализированы и требуют меньших усилий от команды тестирования и заинтересованных сторон. Как и тяжеловесные методы, они также основаны на вовлечении заинтересованных сторон, используя результаты анализа рисков в качестве основы для планирования тестирования и тестовых условий. Однако группа заинтересованных сторон может быть не такой широкой, и факторы риска обычно сводятся к влиянию риска и вероятности риска по порядковой шкале. Некоторые из этих методов, такие как Систематическое тестирование программного обеспечения (Craig & Jaskiel, 2002), могут использоваться только при наличии спецификаций требований. Другие методы, включая Прагматический анализ и управление рисками (Black, 2009) и Управление продуктовыми рисками (van Veenendaal, 2012), используют требования и/или другие спецификации в качестве входных данных для анализа рисков, но могут функционировать полностью на основе вклада от заинтересованных сторон.

1.3.6. Метрики успеха и трудности, связанные с тестированием на основе рисков

В ретроспективе команда тестирования должна измерить степень, в которой они достигли преимуществ тестирования на основе рисков. Во многих случаях это включает ответы на некоторые или все из следующих вопросов с использованием метрик и консультаций:

- Были ли вовлечены или представлены соответствующие заинтересованные стороны в анализе рисков?
- Было ли участие заинтересованных сторон в анализе рисков соответствующим?
- Если в промышленном окружении произошли критические инциденты, указывающие на критические дефекты, были ли они решены?
- Были ли большинство дефектов высокого приоритета обнаружены на ранних этапах выполнения тестирования?
- Смогла ли команда тестирования объяснить результаты тестирования заинтересованным сторонам с точки зрения риска?

- Были ли пропущенные тесты связаны с более низким уровнем риска, чем выполненные тесты?

В большинстве случаев успешное тестирование на основе рисков приводит к утвердительному ответу на все эти вопросы. В долгосрочной перспективе должны быть установлены цели по совершенствованию процесса для метрик успеха, а также стремление к повышению эффективности процесса анализа рисков качества.

Управление рисками часто сталкивается с неожиданными трудностями из-за сложностей, которые часто упускаются из виду.

- **Сложность в оценке уровня риска:** оценка влияния риска и вероятности риска может быть очень сложной. Решение: использовать исторические данные и попросить ключевых заинтересованных сторон проекта дать свою оценку.
- **Сложность в управлении рисками:** из-за необходимости добиться быстрого успеха часто пренебрегают разработкой и поддержанием надлежащего подхода к тестированию на основе рисков. Решение: регулярный мониторинг и отчетность о рисках для заинтересованных сторон.
- **Дежавю:** один и тот же набор рисков принимается для каждого проекта, что приводит к самоуспокоенности по отношению к рискам. Решение: вовлекать правильных специалистов в идентификацию рисков и смягчать только те риски, которые считаются важными.
- **Ключевые риски не учитываются:** первопричина этой проблемы обычно связана с участием неопытных или неподходящих специалистов в процессе. Решение: привлекать подходящих специалистов и обучать их.
- **Отток заинтересованных сторон:** заинтересованные стороны могут меняться со временем, и также могут появляться новые риски, поэтому анализ рисков является непрерывной, итеративной активностью и не должен выполняться только один раз в начале.

1.4 Стратегия тестирования проекта

Введение

В рамках данной программы обучения принимается, что корпоративная стратегия тестирования уже известна. Разработка стратегии тестирования, а также поддержание ее в актуальном состоянии должны проводиться в контексте стандарта ISO/IEC/IEEE 29119-3 (где она называется «корпоративной методикой тестирования») и программ обучения Экспертного уровня «ISTQB Управление тестированием»), а также программы обучения для сертифицированных тестировщиков «ISTQB Масштабирование тестирования в организации с гибкими методологиями».

Если корпоративная стратегия тестирования отсутствует либо не охватывает необходимые аспекты, задача процесса управления тестированием состоит в прояснении недостающих элементов с ключевыми заинтересованными сторонами.

Определение стратегии тестирования проекта в рамках данного раздела является подробным примером, который может использоваться для различных типов стратегий тестирования проекта, релиза, продукта или любой другой инициативы по разработке или приобретению системы. Стратегия тестирования проекта (или же «стратегия тестирования» согласно ISO/IEC/IEEE 29119-3) характеризует подход к тестированию в контексте достижения целей организационного уровня, в

частности, в том, что касается качества продукта и собственно активностей тестирования. Стратегия тестирования может быть задана на одном уровне тестирования или для одного типа тестирования.

Стратегия тестирования является основным результатом планирования тестирования в проекте. Обычно она документируется в плане тестирования, но может быть включена и в другие документы. Стратегию тестирования рекомендуется документировать, однако она необязательно должна быть представлена формальным планом тестирования. Необходимость документирования зависит от контекста тестирования (см. Раздел 1.2 Контекст тестирования). Если разработка в проекте строится по последовательной модели, стратегию тестирования принято документировать, причем желательно в плане тестирования (см. ISO/IEC/IEEE 29119-3). Документирование также часто требуется условиями договоров и соглашений, регуляторными органами и законодательством.

1.4.1. Выбор подхода к тестированию

Стратегия тестирования проекта определяет активности всего тестирования внутри проекта, а также конкретизирует задачи, ресурсы, расписание и обязанности. Стратегию тестирования необходимо адаптировать к конкретным требованиям конкретного проекта. Ключевое значение имеет выбор уровней и типов тестирования, методов статического и динамического тестирования и других принятых практик (например, сценарное тестирование, ручное тестирование, сравнительное тестирование).

Теоретически все типы тестирований могут выполняться на любом уровне тестирования, а к любому типу тестирования на любом уровне можно применить любой метод тестирования. На практике же выбор отдельных элементов и их сочетание оказывают большое влияние на эффективность тестирования. Например, сопровождаемость кода часто можно более эффективно и действенно оценить с помощью статического анализа кода или рецензирования кода. С другой стороны, оценку производительности гораздо эффективнее выполнять через сценарное системное тестирование, поскольку учитывается взаимодействие внутренних компонентов, а практическую пользу какой-либо функциональности лучше всего могут валидировать сами пользователи с помощью приемочных тестов. Выбор наилучшего подхода к стратегии тестирования может быть сложным процессом, на который оказывают влияние корпоративная стратегия тестирования, контекст проекта и другие аспекты.

Таким образом, выбор уровней, типов и методов тестирования, а также того, как они комбинируются между собой, имеет решающее значение для эффективности стратегии тестирования проекта, поскольку этот выбор существенно влияет на эффективность и результативность тестирования.

1.4.2. Анализ корпоративной стратегии тестирования, контекста проекта и других аспектов

Корпоративная стратегия тестирования, контекст проекта и другие дополнительные факторы и ограничения, имеющие отношение к тестированию, должны быть хорошо осмыслены, без чего невозможна разработка стратегии тестирования проекта

Для выбора оптимального подхода к тестированию, как правило, необходимо проанализировать следующие факторы:

- **Предметная область (Домен)** — сфера, для которой создается или модифицируется продукт. Любые специфические для данной сферы правила, стандарты и принятые методики могут влиять на степень скрупулезности тестирования, требования к документации, а также на уровень детальности тестирования. Например, в фармацевтике и медицине подход к тестированию заключается в акценте на интенсивном приемочном

тестировании (то есть тестировании «приемлемости» для пользователей - пациентов), которое сосредоточено на рисках для здоровья пациентов, и в этом подходе используются сценарии тестирования на базе функциональных требований. Приемочное тестирование веб-приложения для подачи заявок на страхование сосредоточено скорее на практичности и функциональности пользовательского интерфейса и способах повысить вероятность заключения новых страховых договоров, что достигается с помощью A/B тестирования.

- **Корпоративные цели и основные характеристики качества**. Такой целью может быть необходимость продемонстрировать значимость тестирования, либо желание повысить уровень автоматизации тестирования или показателей качества процесса тестирования, такие как уровень зрелости тестирования или эффективность выявления дефектов. Исходя из этих целей и показателей определяются уровни и типы тестирования, которых необходимо придерживаться.
- **Цели проекта и тип проекта**. Цели проекта (например, в отношении бюджета, времени и качества) и тип проекта (например, разработка продукта для конкретного заказчика или же продукта, ориентированного на рынок) обычно влекут за собой ограничения и риски, но также и возможности; все это влияет на тестирование. Например, в случае жестких ограничений бюджета и сроков может быть необходимо строго придерживаться тестирования на основе рисков, с тем чтобы приоритет исполнения отдавался сценариям тестирования; в то же время разработка продукта для конкретного заказчика может требовать тестов, покрывающих заранее известные приемочные критерии, заданные договором.
- **Ресурсы тестирования**. Необходимо учитывать любые ограничения, связанные с доступностью ресурсов тестирования, включая инструменты тестирования, инфраструктуру тестирования, окружения разработки и технологические, а также наличие свободных специалистов по тестированию и их навыки (см. Раздел 3.1 Команда тестирования). Например, тестирование на основе опыта требует от тестировщиков хорошего знания предметной области; тестирование мобильных приложений проводится, как правило, на определенном наборе устройств; использование инструментов тестирования может быть ограничено количеством доступных лицензий.
- **Модель жизненного цикла разработки программного обеспечения в проекте**. Для определения оптимальных уровней тестирования, трудозатрат на тестирование, оптимальных критериев входа и выхода см. программу обучения ISTQB Базового уровня версии 4, Разделы 2.2 и 5.1. Жизненный цикл программного обеспечения с непрерывной интеграцией требует большего количества автоматизированных тестов, чем разовая разработка по водопадной модели, поэтому могут использоваться различные типы и методы тестирования.
- **Взаимодействие с другими системам**. В системе систем крайне важно согласовывать тестирование с другими командами и проектами и выбирать оптимальные уровни тестирования, особенно при системном интеграционном тестировании. Например, тестирование на основе рисков помогает повысить приоритет системного интеграционного тестирования, а также определить его масштаб.
- **Доступность тестовых данных**. Необходимо учитывать ограничения, связанные с доступностью тестовых данных, такие как необходимость получения анонимных тестовых данных из окружения промышленной эксплуатации клиента или воссоздание специфических тестовых данных, в предоставлении которых есть сложности и которые необходимо валидировать, таких как данные для тестирования ИИ. Например, в

тестировании на основе модели предусматривается создание тестовых данных и управление ими.

Руководитель тестирования должен определить, какую комбинацию методов, уровней и типов тестирования следует использовать, чтобы обеспечить наилучший подход для удовлетворения корпоративной стратегии тестирования, учета контекста проекта и дополнительных факторов и ограничений, связанных с тестированием.

1.4.3. Определение целей тестирования

Для каждого проекта тестирования следует определить план тестирования, который должен содержать, среди прочего, объем тестирования, цели тестирования и критерии завершения. План тестирования может быть представлен планом на уровне релиза, планом тестирования проекта (его называют «главным планом тестирования») или же, при необходимости, планами различных уровней тестирования. Кроме того, могут быть заданы планы тестирования по различным показателям качества, например, план тестирования безопасности или производительности. В проектах с гибкой методологией разработки программного обеспечения может быть принят план тестирования отдельной итерации. Объем функциональности и нефункциональные характеристики, которые необходимо поставить заказчику, задаются в таком плане тестирования и согласовываются с заинтересованными сторонами для каждого релиза и каждой итерации.

В зависимости от поставляемой функциональности продукта, которая передается в тестирование в рамках проекта, должны быть определены цели тестирования проекта и критерии выхода. Это можно сделать с помощью методологии достижения целей S.M.A.R.T.:

- S = specific (конкретный). Цель тестирования проекта или критерий завершения должен быть ясным и однозначным.
- M = measurable (измеримый). Цель или критерий должна/должен иметь количественное выражение и точные критерии измерения прогресса, чтобы понимать, была ли достигнута цель / был ли удовлетворен критерий.
- A = achievable (достижимый). Цель или критерий должна/должен быть выполнимой с учетом имеющихся ресурсов, сроков и возможностей.
- R = relevant (релевантный, значимый). Цель или критерий должны соответствовать общим целям проекта.
- T = timely (своевременный, ограниченный во времени). Цель или критерий должны иметь конкретные временные рамки и обозначенный срок завершения.

Цели тестирования проекта должны охватывать все целевые качественные и количественные характеристики, поддающиеся измерению или оценке. Ниже приведены примеры целей тестирования проекта:

- достижение заданных критериев завершения в отведенные сроки;
- выполнение целей организации в области качества (например, качество может измеряться через такой показатель эффективности, как количество обращений, поступающих от клиентов к продукту);
- обеспечение выполнения правил и норм, действующих в конкретной отрасли;
- обеспечение того, чтобы доступ к данным имели только авторизованные пользователи (например, с помощью прав доступа);

- проверка функциональной полноты, корректности работы функциональности, производительности, эффективности, переносимости, безопасной миграции данных;
- повышение уровня автоматизации тестирования (например, повышение процента автоматизированных регрессионных тестов или тестов производительности);
- рефакторинг кода и подтверждение, что работы по нему не привели к появлению новых дефектов. Рефакторинг производится, например, для «подчистки» кода со слабой структурой или работ по уменьшению технического долга. При этом должна сохраниться существующая функциональность, что проверяется посредством регрессионного тестирования;
- проверка интерфейсов на безопасность (например, путем проверки XML-сообщений на их соответствие заданной XML-схеме, чтобы убедиться, что вредоносные данные не будут пропущены);
- проверка практичности и функционирования пользовательского интерфейса и получение определенного субъективного показателя (например, путем измерения времени для выполнения определенной задачи в интернет-магазине).

Помимо подсчета и измерения целей тестирования проекта следует учитывать оценку уровня качества, прибегая к мнению экспертов в предметной области и заинтересованных сторон.

В зависимости от контекста проекта и целей тестирования иногда может потребоваться несколько тестовых окружений выделенными ресурсами и/или инструментами тестирования. Не все тестовые окружения могут быть доступны одновременно. Это необходимо учитывать при формулировании достигаемых целей тестирования и критериев выхода.

Контекст проекта может требовать учета дополнительных факторов при задании целей и объема тестирования проекта, как это описано в Разделе 1.2 Контекст тестирования данной программы обучения.

1.5 Совершенствование процесса тестирования

Введение

Тестирование является важной частью разработки программного обеспечения и часто требует не менее 30-40% от общих затрат на проект. Помимо множества (технических) вызовов, с которыми сталкиваются проекты по разработке программного обеспечения (например, увеличение сложности и размера, новые технологии, широкий спектр устройств и операционных систем, а также уязвимости безопасности), существует необходимость оптимизировать результативность и эффективность тестирования, а также совершенствовать процессы тестирования соответственно. Изучение существующих передовых практик и собственных ошибок позволяет совершенствовать процесс тестирования и делать проекты более успешными.

Совершенствование процесса на корпоративном уровне обычно более полезен, чем совершенствование процесса на уровне проекта или команды. Однако также возможно и полезно применять совершенствование процесса на уровне проекта или команды, но оно должно быть адаптировано к потребностям проекта или команды. Совершенствование тестирования может быть инициировано, например, неудовлетворенностью результатами текущих тестов, неожиданными дефектами, изменяющимися обстоятельствами, результатами анализа производительности или отсутствием коммуникации. Существуют различные техники совершенствования тестирования (Bath & van Veenendaal, 2014). Некоторые из этих техник описаны ниже. Техники, описанные в этой программе обучения, могут быть применены как к последовательным моделям разработки, так и к

гибкой методологии разработки программного обеспечения/инкрементным моделям разработки. Программа обучения уровня эксперта ISTQB® «Совершенствование процесса тестирования» предлагает более глубокое понимание.

1.5.1. Процесс совершенствования тестирования (IDEAL)

Как только согласовано, что процессы тестирования должны быть усовершенствованы, активности по реализации совершенствования процесса, которые будут приняты для этой активности, могут быть определены, как в модели IDEAL, которая основана на аналогичных идеях, как и известный цикл планирование-выполнение-проверка-корректировка (PDCA). IDEAL является акронимом, который расшифровывается как инициирование, диагностика, установление, действие и обучение.

Хотя IDEAL изначально был определен для поддержки активностей по совершенствованию на корпоративном уровне, он также может быть применен на уровне проекта или команды, работающей по гибкой методологии разработки программного обеспечения (Agile). В контексте проекта цели мероприятий (см. ниже) еще предстоит достичь. Основное различие, вероятно, заключается в этапе инициации, которая намного меньше на уровне проекта или команды, чем на корпоративном уровне. Диагностика через ретроспективу и установление плана, скорее всего, будут намного меньше, чем на корпоративном уровне. Действие и обучение также будут актуальны на уровне проекта или команды.

Инициация совершенствования процесса

В начале совершенствования процесса цели и объем совершенствований процесса согласовываются заинтересованными сторонами.

Анализ текущей ситуации

Текущий процесс тестирования оценивается для выявления возможных совершенствований. Оценка обычно проводится по стандартной структуре в случае совершенствования процесса тестирования на основе моделей (см. Раздел 1.5.2 Совершенствование процесса тестирования на основе моделей) или может быть основана на анализе конкретных метрик в случае совершенствования процесса тестирования на основе аналитики (см. Раздел 1.5.3 Подход к совершенствованию процесса тестирования на основе аналитики).

Разработка плана совершенствования процесса тестирования

План совершенствования процесса тестирования может быть формальным документом, который перечисляет все детализированные действия, которые должны быть выполнены для достижения совершенствований. В зависимости от контекста, план может быть весьма неформальным и очень легким. Список возможных совершенствований процесса должен быть приоритизирован. Приоритизация может основываться на возврате инвестиций (ROI), рисках, соответствии стратегиям проекта или команды и/или измеримых количественных или качественных выгодах, которые должны быть достигнуты.

Реализация плана совершенствования процесса тестирования

План совершенствования процесса тестирования для поставки совершенствований реализован. Обычно включает обучение и пилотирование измененных процессов и их полное внедрение в проекте или команде.

Обучение усовершенствованным процессам

После полного внедрения совершенствований процесса важно проверить, какие выгоды, запланированные или неожиданные, были получены. Узнав, что сработало, а что нет, мы должны

действовать на основе этой информации, и только после этого может начаться следующий цикл совершенствований.

1.5.2. Совершенствование процесса тестирования на основе моделей

Одной из предпосылок совершенствования процесса тестирования на основе моделей и аналитического подхода является предположение, что качество продукта сильно зависит от качества используемых и применяемых процессов. При применении совершенствования процесса тестирования на основе моделей используется модель совершенствования тестирования. Модели совершенствования тестирования основаны на лучших практиках в тестировании и организуют совершенствование тестирования поэтапно.

Появилось несколько рекомендованных процессных моделей, которые поддерживают совершенствование процесса тестирования. К ним относятся интегрированная модель зрелости тестирования (TMMi®) и TPI NEXT®.

Совершенствование на основе моделей также может быть применено на уровне проекта. В таких случаях оценка и совершенствования процесса специально сосредоточены на процессах тестирования или ключевых областях, определенных в модели, которые относятся к активностям на уровне проекта (например, планирование тестирования и проектирование тестов) и часто в значительной степени опускают те, которые находятся на уровне организации (например, политика тестирования и организация тестирования). В качестве альтернативы, можно также соответствующим образом адаптировать практики, которые касаются уровня организации, к контексту проекта.

Для получения дополнительной информации о совершенствовании процесса тестирования на основе моделей см. программу обучения ISTQB® по совершенствованию процесса тестирования Экспертного уровня.

Интегрированная модель зрелости тестирования

TMMi® (van Veenendaal & Cannegieter, 2011) (van Veenendaal, 2020) состоит из пяти уровней зрелости. Каждый уровень зрелости, за исключением уровня TMMi® 1, содержит области процесса тестирования и цели совершенствования. Кроме того, для облегчения и поддержки его внедрения TMMi® содержит практики, под-практики и примеры. TMMi® изначально был разработан для дополнения интегрированной модели зрелости процессов (CMMI®), но сегодня широко используется независимо от CMMI®.

Для облегчения и поддержки обновления TMMi® в гибкой методологии разработки программного обеспечения (Agile) была разработана специальная инструкция, объясняющая, как TMMi® может быть полезно использован и применен в гибкой методологии разработки программного обеспечения.

Для получения дополнительной информации о TMMi® см. www.tmmi.org.

TPI NEXT®

Модель TPI NEXT® (van Ewijk, 2013) определяет 16 ключевых областей, каждая из которых охватывает конкретный аспект процесса тестирования (например, стратегия тестирования, метрики тестирования, инструменты тестирования и тестовое окружение). В модели определены четыре уровня зрелости для каждой из 16 ключевых областей.

Для оценки каждой ключевой области на каждом уровне зрелости определены конкретные контрольные точки. Результаты оценки суммируются и визуализируются с помощью матрицы зрелости, которая охватывает все ключевые области.

Для получения дополнительной информации о TPI NEXT® см. www.tmap.net.

1.5.3. Подход к совершенствованию процесса тестирования на основе аналитики

Используя подход к совершенствованию на основе модели, как описано в предыдущем разделе, совершенствования вводятся путем сравнения подхода к тестированию проекта или команды с внешними лучшими практиками. Аналитические подходы выявляют проблемы на основе данных из самого проекта или команды. Соответствующие улучшения могут быть выведены из анализа этих проблем. Аналитические подходы могут использоваться вместе с подходом на основе модели для проверки результатов и обеспечения разнообразия.

Проблемы могут быть выявлены с использованием количественных и качественных данных. Раздел 1.5.3 данной программы обучения, подход к совершенствованию процесса тестирования на основе аналитики, вводит аналитические подходы, которые в основном используют количественные данные из процесса тестирования и данные о дефектах для оценки текущего подхода. Раздел 1.5.4 данной программы обучения, ретроспективы, вводит ретроспективы, в которых собираются качественные данные о том, что работает хорошо и что не работает хорошо, от членов команды разработки и тестирования.

Анализ данных важен для объективного совершенствования процесса тестирования и является ценным дополнением к полностью качественным оценкам, которые в противном случае могут привести к неточным рекомендациям, не подкрепленным данными. Применение аналитического подхода к совершенствованию чаще всего включает количественный анализ процесса тестирования для выявления проблемных областей и установления целей, специфичных для проекта. Определение и измерение ключевых параметров необходимо для оценки процесса тестирования и оценки успешности совершенствований.

Примеры аналитических подходов:

- Анализ первопричин
- Анализ с использованием измерений, метрик и индикаторов
- Подход GQM (Цель-Вопрос-Метрика)

Анализ первопричин — это изучение проблем для выявления их первопричин. Это позволяет выявить решения, которые устраняют причины проблем, а не просто устраняют очевидные симптомы. Возможная процедура анализа может включать выбор соответствующего набора дефектов, выявление кластеров в этих данных и использование диаграмм причинно-следственных связей (также называемых диаграммами Ишикавы или диаграммами «рыбий скелет») для выявления первопричин важных кластеров дефектов. Затем выводятся совершенствования для предотвращения возникновения подобных дефектов.

Измерения, метрики и индикаторы используются для количественной оценки того, насколько хорошо выполняется процесс тестирования в проекте или команде. Ключевые атрибуты процесса тестирования, которые следует учитывать, включают результативность, эффективность и предсказуемость. Для каждого из этих атрибутов можно выбрать одну или несколько метрик. Собирая и анализируя соответствующие данные, можно выявить ключевые области, требующие совершенствований.

Подход GQM (Basili и др., 2014) (van Solingen & Berghout, 1999) предоставляет структуру для определения и анализа метрик, которые адаптированы к информационным потребностям соответствующих заинтересованных сторон проекта. Цели измерения определяют аспект качества объекта, который необходимо измерить для конкретной цели, перспективы и контекста. Эти цели уточняются в вопросах, которые определяют аспект качества с точки зрения заинтересованных

сторон. Затем выбираются метрики, которые предоставляют необходимую информацию для ответа на вопрос. Данные, собранные для метрик, отвечают на вопросы, чтобы оценить цель измерения и удовлетворить информационные потребности заинтересованных сторон.

Более подробную информацию об этих подходах к совершенствованию процесса тестирования на основе аналитики можно найти в программе обучения ISTQB® по совершенствованию процесса тестирования Экспертного уровня.

1.5.4. Ретроспективы

Ретроспективы — это встречи, на которых команда рецензирует свои методы и совместную работу, фиксирует извлеченные уроки (как положительные, так и отрицательные) и принимает решения о изменениях и действиях для достижения совершенствования (как для тестирования, так и для вопросов, не связанных с тестированием). Ретроспективы охватывают такие темы, как процесс, сотрудники, организация, сотрудничество и инструменты.

Ретроспективы используются как в последовательных моделях разработки, так и в гибкой методологии разработки программного обеспечения (Agile). В последовательных моделях разработки они являются частью завершения тестирования. В этом контексте ретроспективы направлены на генерацию извлеченных уроков для лучшего управления будущими проектами. В гибкой методологии разработки программного обеспечения (Agile) ретроспективы обычно проводятся в конце каждой итерации для обсуждения того, что было успешным и что нужно совершенствовать, и как эти совершенствования могут быть учтены в следующей итерации. Ретроспективы проводятся всей командой и, таким образом, поддерживают подход всей команды и способствуют непрерывному совершенствованию. Обратите внимание, что иногда требуются специальные ретроспективы для решения вопросов, связанных с тестированием.

Типичная ретроспектива состоит из следующих шагов:

Введение: цель и повестка дня ретроспективы рецензируются, и создается атмосфера взаимного доверия, чтобы проблемы могли обсуждаться без обвинений или осуждений.

Сбор данных: собираются данные о том, что произошло во время итерации или проекта. Возможно собрать качественные данные, такие как хронология ключевых событий, которая выявляет проблемы и перечисляет, как каждый член команды относится к этим проблемам. Кроме того, могут быть представлены количественные данные из метрик, например, данные о прогрессе тестирования, выявлении дефектов, результативности тестирования, эффективности тестирования и предсказуемости, могут предоставить объективное представление о тестировании проекта или итерации.

Выработка усовершенствований: собранные данные анализируются для понимания текущей ситуации и генерации идей для совершенствования. Например, анализ первопричины может быть применен для выявления первопричин выявленных проблем, и может быть проведена сессия мозгового штурма для генерации идей о том, как решить первопричины.

Принятие решений о действиях по совершенствованию: действия по реализации идей совершенствования вырабатываются и приоритизируются. Определяются план совершенствования и обязанности. Могут быть определены цели и связанные с ними метрики для оценки влияния действий на выявленные проблемы. Реализация слишком большого количества совершенствований одновременно затруднительна для управления с проверяемыми шагами.

Закрытие ретроспективы: на последнем шаге сама ретроспектива рецензируется для выявления сильных сторон и совершенствований в процессе ретроспективы. Ретроспектива проводится

регулярно, особенно в гибкой методологии разработки программного обеспечения (Agile). Непрерывное совершенствование применяется и к самой ретроспективе.

Важно правильно документировать результаты ретроспективы. В последовательной модели разработки выводы, заключения и рекомендации должны быть распределены и доведены до сведения членов организации в понятной форме. В гибкой методологии разработки программного обеспечения проблемы и действия также должны быть задокументированы, чтобы можно было просмотреть действия и их потенциальное влияние на проблемы в следующей итерации.

Тестировщики, являясь частью команды (проекта), привносят свою уникальную точку зрения. Они могут поднимать проблемы, связанные с тестированием (и другие), и стимулировать команду думать о возможных совершенствованиях.

Дополнительную информацию можно найти в (Derby & Larsen, 2006).

1.6 Инструменты тестирования

Введение

Существует три типа бизнес-инструментов:

- Платные инструменты
- Инструменты с открытым исходным кодом
- Заказные инструменты

При выборе бизнес-инструмента необходимо учитывать все организационные требования и нормативные акты, а также требования заинтересованных сторон.

Также существуют технические инструменты, такие как инструменты автоматизации тестирования, инструменты управления тестированием и многие другие.

Примеры использования инструментов тестирования можно найти в программе обучения ISTQB® Базового уровня версии 4.

1.6.1. Хорошие практики для внедрения инструментов

Этот раздел содержит необходимые шаги для оценки и внедрения инструмента тестирования.

Руководитель тестирования может быть вовлечен во внедрение инструмента или может способствовать, или координировать процесс внедрения. Руководители тестирования обычно отвечают за конкретный инструмент тестирования или любой инструмент, связанный с тестированием, такой как инструмент управления требованиями, управления дефектами или инструмент мониторинга.

Существуют общие хорошие практики и соображения при оценке и выборе инструмента тестирования. Эти практики и соображения включают следующее:

- Определить возможности для совершенствования процесса с поддержкой соответствующих инструментов
- Понять технологии, используемые в организации, и выбрать инструмент, совместимый с этими технологиями
- Понять, как инструмент технически и организационно интегрируется в жизненный цикл разработки программного обеспечения

- Оценить инструмент по четким требованиям и объективным критериям
- Оценить поставщика, если вы рассматриваете использование платного инструмента. Оценить поддержку для бесплатных (например, с открытым исходным кодом) инструментов
- Определить внутренние требования для наставничества (коучинга) или обучения использованию инструмента
- Рассмотреть плюсы и минусы различных моделей лицензирования
- На заключительном этапе провести оценку концепции.

Общие хорошие практики при внедрении и развертывании инструмента включают:

- Запустить пилотный проект для проверки критериев и требований выбора, а также для оценки того, как инструмент вписывается в существующие процессы и практики.
- Адаптировать и совершенствовать процессы для соответствия использованию инструмента, а также адаптировать инструмент к существующим процессам, если необходимо.
- Определить руководящие принципы для использования инструмента.
- Обеспечить обучение, наставничество (коучинг) для пользователей инструмента.
- Развернуть поэтапно инструмент в организации.
- Внедрить способ сбора информации от фактического использования инструмента для дальнейших совершенствований.
- Определить владельца инструмента.

1.6.2. Технические и бизнес-аспекты принятия решений по инструментам

Множество факторов влияет на внедрение и использование инструмента. Для руководителя тестирования важно знать и учитывать их.

- **Регулирование и безопасность**: организации, которые разрабатывают программное обеспечение, критически важное для безопасности или миссий, или должны соответствовать нормативам, могут предпочитать платные инструменты, так как они чаще соответствуют требуемым стандартам и часто обладают соответствующей сертификацией.
- **Финансовые аспекты:** инструменты с открытым исходным кодом обычно имеют более низкую начальную стоимость благодаря поддержке и разработке сообществом. Платные инструменты могут иметь единовременную стоимость покупки, а также периодические лицензионные расходы. Начальную стоимость инструмента, разработанного на заказ, трудно определить, так как она зависит от требований и стадии разработки инструмента. Помимо начальных затрат, необходимо рассчитать и учитывать стоимость обучения и сопровождения на протяжении всего жизненного цикла инструмента. Все инструменты могут иметь высокие затраты на сопровождение и поддержку.
- **Требования заинтересованных сторон:** важно собрать требования от всех заинтересованных сторон для оценки и определения наиболее подходящего инструмента. Платные инструменты и инструменты с открытым исходным кодом не обязательно удовлетворяют все требования в деталях. Инструменты, разработанные на заказ, могут быть лучшим выбором для удовлетворения всех индивидуальных требований и в случаях, когда ни один другой инструмент не предоставляет необходимую функциональность.

- **Существующий ландшафт программного обеспечения и стратегия использования инструментов:** существующий состав инструментов (ландшафт программного обеспечения) и связанная с ним стратегия использования инструментов должны быть оценены, так как могут быть предпочтительные или заблокированные поставщики, интегрированные системы, которые имеют зависимости с другими продуктами, или специальная модель полного обслуживания для всего ландшафта программного обеспечения с конкретными правилами.

1.6.3. Аспекты процесса отбора и оценка возврата инвестиций (ROI)

Инструменты тестирования могут быть долгосрочной инвестицией, возможно, охватывающей многие итерации одного проекта и/или применимой к многим проектам. Потенциальный инструмент должен рассматриваться с разных точек зрения.

- Для высшего руководства требуется высокий возврат инвестиций (ROI).
- Для команды поддержки и эксплуатации предпочтительно ограниченное, но необходимое количество инструментов, используемых в организации. Поддержание большего количества инструментов, отслеживание их лицензий и управление стеком инструментов не должны быть затратными по времени или деньгам.
- Для руководителей проектов инструмент должен добавлять измеримую ценность проекту или организации.
- Для людей, использующих инструмент, важна практичность. Практичность включает, например, поддержку конкретных задач, возможность обучения и работоспособность.
- Для персонала, занимающегося операционной деятельностью (в частности, служба поддержки), важна сопровождаемость.

Функции должны быть проанализированы для каждого бизнес- и технического типа инструмента. На этот анализ влияют различные перспективы и интересы: управление тестированием, (технический) анализ тестирования, автоматизация тестирования или разработка. Сотрудник в организации, ответственное за инструмент (владелец инструмента), должно убедиться, что анализ выполнен, и вышеупомянутые пункты учтены.

Все инструменты, введенные в процесс тестирования, также должны обеспечивать высокий возврат инвестиций (ROI) для организации. Ответственность за расчет и дальнейшую оценку ROI лежит на руководителе тестирования. В гибкой методологии разработки программного обеспечения (Agile) это может быть обязанностью всей команды разработки.

Перед приобретением или созданием инструмента следует провести анализ затрат и выгод, чтобы убедиться, что он приносит пользу организации. Этот анализ должен учитывать, как повторяющиеся, так и неповторяющиеся затраты.

Неповторяющиеся активности и затраты включают:

- Определение и установление требований к инструменту для достижения целей.
- Оценку и выбор правильного инструмента и поставщика, доказательство концепции.
- Приобретение, адаптацию или разработку инструмента для начального использования.
- Определение руководящих принципов использования инструмента.
- Первоначальное обучение по инструменту.
- Интеграцию инструмента в существующий ландшафт инструментов.

- Закупку аппаратного/программного обеспечения, необходимого для поддержки инструмента.

Повторяющиеся активности и затраты включают:

- Повторяющиеся лицензионные и поддерживающие платежи.
- Затраты на сопровождение.
- Постоянные затраты на обучение.
- Перенос инструмента в разные окружения.

Также должны быть учтены альтернативные издержки. Это означает, что время, потраченное на оценку, администрирование, обучение и использование инструмента, могло бы быть потрачено на фактические активности тестирования. Поэтому может потребоваться больше ресурсов для тестирования, прежде чем инструмент можно будет использовать для запланированных активностей.

Следующие риски, связанные с возвратом инвестиций (ROI), должны быть учтены при выборе инструментов:

- Незрелость организации может привести к неэффективному использованию инструмента.
- Изменения в политике сопровождения поставщика могут увеличить рабочую нагрузку.
- Более высокие затраты, чем ожидалось.
- Меньшая выгода, чем ожидалось.

Следующие выгоды могут применяться к инструментам тестирования:

- Сокращение ручной повторяющейся работы (например, регрессионное тестирование).
- Сокращение времени цикла тестирования за счет автоматизации.
- Снижение затрат на выполнение тестов за счет уменьшения ручных активностей.
- Увеличение покрытия для конкретных типов тестирования, поддерживаемых инструментом.
- Снижение ошибок специалистов за счет уменьшения ручных активностей.
- Быстрый доступ к информации о тестах.

Дополнительные выгоды и риски, особенно для инструментов автоматизации тестирования, можно найти в версии 4 программы обучения ISTQB® Базового уровня и программе обучения ISTQB® Инженер по автоматизации тестирования.

Как правило, организация тестирования редко использует один инструмент. Общий возврат инвестиций (ROI) для организации обычно представляет собой смесь ROI всех инструментов, используемых для тестирования. Инструменты должны обмениваться информацией и работать совместно. Долгосрочная, комплексная стратегия для инструментов тестирования, включающая риски, затраты и выгоды, является целесообразной.

1.6.4. Жизненный цикл инструмента

Существует четыре различных этапа в жизненном цикле инструмента. Должен быть назначен администратор инструмента, чтобы гарантировать, что активности этих этапов определены, выполняются и управляются.

- **Приобретение:** прежде всего принято решение выбрать инструмент. На втором этапе необходимо назначить владельца инструмента. Этот сотрудник принимает решения по использованию инструмента (например, соглашения о наименовании рабочих продуктов и место их хранения). Принятие этих решений заранее может существенно повлиять на конечный возврат инвестиций (ROI).
- **Поддержка и сопровождение:** владелец инструмента отвечает за его сопровождение. Ответственность за активности по сопровождению должна лежать на администраторе инструмента или на выделенной группе инструментов. В случае возможности взаимодействия должны быть учтены обмен данными, процессы сотрудничества и коммуникации. Также требуются решения по резервному копированию и восстановлению артефактов, связанных с инструментом.
- **Эволюция:** со временем окружение, потребности бизнеса или решения поставщика могут потребовать изменений в инструменте. Чем сложнее становится операционное окружение для инструмента, тем легче изменение может нарушить его использование.
- **Вывод из эксплуатации:** в конце своего жизненного цикла инструмент должен быть выведен из эксплуатации. В большинстве случаев функциональность, предоставляемая инструментом, будет заменена, и данные должны быть сохранены и/или архивированы. Это может быть основано на решении поставщика или потому, что он достиг точки, где выгоды и возможности перехода на новый инструмент превышают его затраты и риски.

1.6.5. Метрики инструмента

Объективные метрики инструментов разрабатываются и собираются на основе потребностей команды тестирования и других заинтересованных сторон. Инструменты тестирования в основном захватывают ценные данные в реальном времени и уменьшают усилия по сбору данных. Эти данные используются для управления общими усилиями по тестированию и выявления областей для оптимизации.

Различные инструменты сосредоточены на сборе различных типов данных. Примеры включают:

- Инструменты управления тестированием могут предоставлять различные метрики, связанные с доступными элементами тестирования, тестами, запланированными тестами, а также текущим и пройденным статусом выполнения тестов (например, пройден, не пройден, пропущен, заблокирован или запланирован).
- Инструменты управления требованиями обеспечивают трассируемость покрытия требований пройденными и не пройденными сценариями тестирования.
- Инструменты управления дефектами могут предоставлять информацию о дефектах, такую как статус, критичность, приоритет и плотность дефектов элементов тестирования. Другие ценные данные, такие как процент выявления дефектов, уровни тестирования, на которых дефекты были введены и обнаружены, время обнаружения дефектов, помогают в совершенствовании процесса, но могут не предоставляться исключительно инструментом управления дефектами.
- Инструменты статического анализа, среди прочего, предоставляют метрики, связанные со сложностью кода.
- Инструменты тестирования производительности могут предоставлять ценную информацию, такую как время отклика и уровень отказов при пиковых нагрузках.

- Инструменты покрытия кода помогают понять, какие части объекта тестирования были проверены тестированием.
- Хотя инструменты тестирования могут использоваться для сбора метрик, они также должны проводить мониторинг самих себя. В этом контексте может быть измерено качество процесса тестирования (например, количество дефектов, найденных с использованием и без использования инструментов, и покрытие требований).
- Эффективность тестирования (например, продолжительность выполнения тестов и количество выполненных тестов).

Более подробная информация о сборе и использовании метрик может быть найдена в Разделе 2.1 Метрики тестирования данной программы обучения.

2 Управление продуктом – 390 минут

Ключевые слова

Аномалия, дефект, жизненный цикл дефекта, метрика, отказ, отчет о дефекте, оценка затрат на тестирование, прогресс тестирования, цель тестирования.

Ключевые термины, специфичные для управления тестированием

Оценка по трем точкам, покер планирования, широкополосный метод Дельфи.

Цели обучения для главы 2:

2.1 Метрики тестирования

TM-2.1.1 (K2) Привести примеры метрик для достижения целей тестирования

TM-2.1.2 (K2) Объяснить, как контролировать прогресс тестирования с использованием метрик тестирования

TM-2.1.3 (K4) Проанализировать результаты тестирования для создания отчетов о тестировании, которые дадут возможность заинтересованным сторонам принять решения

2.2 Оценка затрат на тестирование

TM-2.2.1 (K2) Объяснить факторы, которые нужно учитывать при оценке затрат на тестирование

TM-2.2.2 (K2) Привести примеры факторов, которые могут повлиять на оценку затрат на тестирование

TM-2.2.3 (K4) Выбрать подходящую технику или подход для оценки затрат на тестирование в заданном контексте

2.3 Управление дефектами

TM-2.3.1 (K3) Реализовать процесс управления дефектами, включая жизненный цикл дефекта, который можно использовать для отслеживания и контроля дефектов

TM-2.3.2 (K2) Объяснить процесс и определить участников, необходимых для эффективного управления дефектами

TM-2.3.3 (K2) Объяснить особенности управления дефектами при гибкой модели разработки программного обеспечения

TM-2.3.4 (K2) Объяснить возможные трудности управления дефектами при гибридной модели разработки программного обеспечения

TM-2.3.5 (K3) Использовать данные и классификационную информацию, которые должны быть собраны во время управления дефектами

TM-2.3.6 (K2) Объяснить, как статистика отчетов о дефектах может использоваться для разработки процессов по улучшению

2.1 Метрики тестирования

Введение – зачем нужны метрики тестирования?

В науке управления существует поговорка: «То, что измеряется, делается». Аналогично, что не измеряется, то, скорее всего, легко игнорируется. Поэтому важно установить правильный набор метрик для любого начинания, включая тестирование.

Цели тестирования – ответ на вопрос, зачем мы тестируем (см. Раздел 1.4 Стратегия тестирования проекта). Чтобы определить, были ли достигнуты цели тестирования, необходимо определить способ их измерения. Метрики тестирования являются индикаторами, которые помогают ответить на этот вопрос.

Метрики тестирования можно разделить на следующие категории:

- **Метрики проекта** измеряют прогресс в соответствии с существующими критериями завершения проекта, такими как процент выполненных, пройденных и не пройденных тестов.
- **Метрики продукта** измеряют характеристики продукта, такие как степень, в которой продукт соответствует ожиданиям качества целевых пользователей.
- **Метрики процесса** измеряют компетентность и эффективность процесса тестирования. Они используются для составления отчетности об эффективности и производительности процесса.

Более подробная информация об управлении метриками продукта и процесса, а также информация об их использовании содержится в программах обучения ISTQB® Экспертного уровня модуля «Руководитель тестирования».

В следующих разделах будут рассмотрены метрики планирования тестирования, мониторинга тестирования, контроля тестирования и завершения тестирования. Это четыре основные управленческие активности, связанные с метриками.

2.1.1. Метрики активностей по управлению тестированием

Программа обучения Продвинутого уровня "Управление тестированием" сосредоточена на следующих основных видах деятельности по управлению тестированием:

- Планирование тестирования
- Мониторинг и контроль тестирования
- Завершение тестирования (см. Раздел 1.1 Процесс тестирования).

Управление тестированием должно быть способным определить набор метрик тестирования для мониторинга, контроля и завершения тестирования в рамках активностей по планированию тестирования. Каждая метрика должна быть определена, измерена, отслежена и отражена в отчетах.

Во время планирования тестирования определяются метрики тестирования, которые соответствуют целям тестирования, указанным в стратегии тестирования проекта.

Метрики, используемые во время мониторинга и контроля тестирования, могут отличаться от тех, которые используются при завершении тестирования. Во время мониторинга и контроля используются метрики, которые касаются прогресса активностей тестирования. При завершении

тестирования должны быть достигнуты цели тестирования. Одна или несколько метрик могут быть объединены для измерения критериев завершения тестирования, определенных для заданных целей тестирования.

Следующая таблица предоставляет примеры некоторых метрик, используемых в активностях по управлению тестированием:

Метрика (планируется/отслеживается на конкретных этапах)	**Мониторинг и контроль тестирования**	**Завершение тестирования**
Покрытие требований	X	X
Покрытие рисков продукта	X	X
Покрытие кода	X	
Фактическое время по сравнению с запланированным временем для активностей тестирования (в часах)	X	
Процент выполненных сценариев тестирования по каждому статусу (например, не пройдены, заблокированы) по сравнению с запланированными к выполнению сценариями тестирования	X	X
Общее количество устраненных дефектов по сравнению с общим количеством дефектов	X	
Фактическое количество автоматизированных сценариев тестирования по сравнению с количеством сценариев тестирования, запланированных для автоматизации		X
Фактическая стоимость тестирования по сравнению с запланированной	X	

Таблица 2: Примеры метрик, используемых в активностях по управлению тестированием

Метрика, используемая в конкретной активности тестирования, показана в таблице. Метрики, отмеченные знаком X в колонке мониторинга и контроля тестирования, в первую очередь используются для измерения прогресса и включаются в отчеты о ходе тестирования (см. программу обучения ISTQB® Базового уровня). Метрики, отмеченные знаком X в колонке завершения тестирования, в первую очередь используются для измерения достижения целей тестирования и включаются в отчет о завершении тестирования (см. программу обучения ISTQB® Базового уровня). Метрики, отмеченные знаком X в обеих колонках, могут использоваться в обоих случаях.

Также существуют метрики мониторинга эффективности тестирования (например, процент выявления дефектов).

Термин «процент выявления дефектов» рассматривается в программе обучения ISTQB® Экспертного уровня «Совершенствование процесса тестирования», в модуле «Реализация совершенствований и изменений».

2.1.2. Мониторинг, контроль и завершение тестирования

Метрики тестирования являются индикаторами, показывающими, насколько далеко продвинулось тестирование и достигнуты ли критерии завершения тестирования или связанные с ними задачи.

Мониторинг тестирования — это деятельность по сбору данных о тестировании и связанной с ним оценки. Она используется, чтобы оценить прогресс тестирования и проверить выполнение критериев выхода или связанных с ними активностей тестирования (см. Раздел 1.1.2 Активности мониторинга и контроля тестирования). Критерии выхода определяются из целей тестирования.

В процессе контроля тестирования используется информация, полученная при мониторинге тестирования, для предоставления рекомендаций и корректирующих действий с целью достижения эффективного и результативного тестирования. Примеры указаний по контролю тестирования включают изменение приоритетов тестов, когда выявленный риск становится проблемой, переоценку соответствия элемента тестирования критериям входа или выхода из-за доработки, корректировку графика тестирования из-за задержки в поставке тестового окружения, и добавление новых ресурсов, когда и где это необходимо.

Процесс завершения тестирования включает сбор данных из завершенных активностей тестирования для консолидации полученных выводов, тестового обеспечения и другой релевантной информации. Завершение тестирования происходит на таких этапах проекта, как завершение уровня тестирования, завершение итерации, завершение (или отмена) проекта тестирования, выпуск продукта или завершение технического обслуживания.

Общие метрики тестирования, используемые в деятельности по управлению тестированием, включают метрики прогресса проекта и метрики, показывающие прогресс по отношению к запланированному графику и бюджету, текущее качество элемента тестирования, а также эффективность тестирования в отношении целей тестирования или целей итерации.

2.1.3. Отчетность тестирования

При управлении тестированием нужно уметь интерпретировать и использовать метрики для понимания и ведения отчетности о статусе тестирования. Для более высоких уровней тестирования, таких как системное тестирование, системное интеграционное тестирование, приемочное тестирование и тестирование безопасности, основным базисом тестирования обычно являются рабочие продукты, такие как спецификации требований, сценарии использования, пользовательские истории и риски продукта. Метрики структурного покрытия более применимы к нижним уровням тестирования, таким как компонентное тестирование (например, покрытие операторов) и интеграционное тестирование компонентов (например, покрытие интерфейсов). Несмотря на то, что управление тестированием может использовать метрики покрытия кода для измерения степени, в которой их тесты охватывают структуру тестируемой системы, отчетность о результатах тестирования на более высоких уровнях должна быть адаптирована к конкретному контексту и потребностям проекта. Например, в условиях частых изменений метрики покрытия кода могут быть полезны для мониторинга влияния изменений кода на набор тестов и выявления потенциальных пробелов или рисков. Кроме того, при управлении тестированием необходимо понимать, что даже если компонентные тесты и компонентные интеграционные тесты достигают 100% покрытия структуры, остаются дефекты и риски качества, которые необходимо решать на более высоких уровнях тестирования.

Целью показателей отчетности является предоставление мгновенного понимания информации для управленческих целей. Метрики могут предоставляться как снимок состояния метрики в определенный момент времени или как эволюция метрики с течением времени для оценки тенденций.

Риски продукта, дефекты, прогресс тестирования, покрытие, а также связанные с ними затраты и усилия на тестирование измеряются и предоставляются конкретными способами в конце проекта.

Ниже приведены примеры метрик, которые могут использоваться для различных целей:

Метрики, связанные с рисками продукта, включают:

- Процент рисков, для которых все тесты прошли успешно
- Процент рисков, для которых некоторые или все тесты не прошли
- Процент рисков, которые еще не полностью протестированы

Эти метрики могут использоваться для оценки качества базиса тестирования и эффективности сценариев тестирования в покрытии рисков продукта.

Метрики, связанные с дефектами, включают:

- Суммарное количество исправленных дефектов по сравнению с суммарным количеством дефектов
- Распределение количества или процентной доли дефектов по категориям:
  - Элементы или компоненты тестирования
  - Источник дефекта (например, технические требования, новый функционал или регресс)
  - Выпуск версий ПО для тестирования
  - Уровень тестирования или итерация, в которых дефект появился, был обнаружен и устранен
  - Приоритет/критичность
  - Первопричина
  - Статус (например, отклонен, дублирован, открыт, закрыт)

Эти метрики могут использоваться для мониторинга процесса обнаружения и устранения дефектов, выявления областей с высокой плотностью дефектов или их критичностью, а также для оценки эффективности и результативности тестирования.

Метрики, связанные с прогрессом тестирования, включают:

- Статус выполнения тестов: общее количество запланированных, реализованных, выполненных, пройденных, не пройденных, заблокированных и пропущенных тестов
- Затраты на тестирование: количество фактически потраченных часов по сравнению с запланированным временем, выделенным на тестирование.

Метрики, связанные с покрытием, включают:

- Покрытие требований: процент требований, которые покрыты сценариями тестирования
- Покрытие рисков продукта: процент идентифицированных рисков продукта, которые смягчаются с помощью сценариев тестирования
- Покрытие кода: процент операторов, ветвлений, путей или условий в коде, которые выполняются в сценариях тестирования

Метрики, связанные с затратами и усилиями на тестирование, включают:

- Остаточные риски для непроверенных компонентов: потенциальное влияние и вероятность дефектов в компонентах, которые не были протестированы

- Затраты на тестирование: фактические затраты на тестирование по сравнению с планируемыми затратами

Кроме того, полезно комбинировать метрики из разных категорий (например, метрика, показывающая корреляцию между открытыми дефектами и выполненными тестами, или метрика, показывающая качество базиса тестирования на основе количества дефектов, обнаруженных в требованиях). Когда выполнение тестов продолжается, и все меньше и меньше дефектов выявляется, можно принять решение о завершении тестирования. Это решение должно основываться на отчетности по метрикам и согласованных критериях завершения тестирования.

2.2 Оценка затрат на тестирование

Введение

Существуют лучшие практики управления проектами для оценки разработки систем и программного обеспечения в отношении всех типов ресурсов. (например, стоимость, специалисты или время). Оценка затрат на тестирование представляет собой применение этих лучших практик к тестированию, связанному с проектом или операцией.

2.2.1. Оценка того, какие активности тестирования будут выполнены

Оценка затрат на тестирование — это деятельность по управлению тестированием, которая оценивает, сколько времени, усилий и расходов потребуется для выполнения задачи. Оценка затрат на тестирование является одной из основных и важных задач в управлении тестированием.

Основными характеристиками оценки в управлении тестированием являются:

- **Усилия** обычно рассчитываются в человеко-часах или в условных единицах (сложности), необходимых для завершения проектных задач по тестированию. Часто усилия, необходимые на тестирование, и продолжительность тестирования (затраченное время) могут различаться, и управление тестированием может потребовать оценить общую продолжительность деятельности. Сколько человеко-часов потребуется?
- **Время, необходимое для завершения проекта.** Время является критическим ресурсом в проекте. Планирование тестирования должно оценить усилия, необходимые на тестирование, в календарных днях и в рабочих днях. У каждого проекта есть вехи и крайний срок для поставки. Сколько времени потребуется для завершения проекта тестирования?
- **Расходы** – это бюджет проекта. Он включает затраты на ресурсы тестирования инструменты и инфраструктуру. Какова стоимость проекта тестирования?

Тестирование часто является подпроектом в рамках (большого) проекта, иногда распределенным по нескольким тестовым площадкам (например, центрам тестирования). На первом шаге составления оценки затрат на тестирование следует определить уровни, активности и задачи тестирования. Далее следует разделить проект тестирования на основные активности тестирования (например, планирование тестирования и выполнение тестов) в рамках процесса тестирования (см. программу подготовки ISTQB® Базового уровня версии 4). В проектах, построенных по гибкой методологии разработки программного обеспечения, активности тестирования часто оцениваются в рамках работы по разработке, а не как отдельные задачи. Следующий шаг – оценить усилия, необходимые для тестирования и завершения задач или рабочих продуктов, а также, вытекающие из этого ожидаемые затраты.

Поскольку тестирование является частью проекта, всегда существуют некоторые естественные ограничения проекта, которые на него влияют и требуют компромиссов, и мы не можем произвольно

манипулировать этими значениями. В управлении качеством это представлено в виде треугольника «время-стоимость-качество». В управлении проектами треугольник «время-стоимость-качество» включает три взаимозависимых значения, что означает, что они тесно связаны и влияют друг на друга. Это взаимоотношение обычно наблюдается в сценариях проекта.

2.2.2. Факторы, которые могут повлиять на усилия, необходимые для тестирования

Оценка усилий, необходимых для тестирования, включает прогнозирование объема работ, связанных с активностями тестирования, которые потребуются для достижения целей тестирования в конкретном проекте, релизе или итерации. Факторы, влияющие на усилия, необходимые для тестирования, могут включать характеристики:

Продукта:

- Качество базиса тестирования
- Размер продукта, подлежащего тестированию (объекта тестирования)
- Сложность области применения продукта (например, окружение, инфраструктура и история)
- Требования к характеристикам качества (например, безопасность и надежность)

Эти факторы, связанные с продуктом, могут влиять на оценку усилий, необходимых для тестирования, поскольку они создают специфический контекст для тестируемой системы.

Процесса разработки:

- Стабильность и зрелость процессов разработки в организации
- Используемая модель разработки (например, гибкая/итеративная или гибридная модель разработки ПО)
- Материальные факторы (например, доступность автоматизации тестирования, инструментов и тестовых окружений)

Эти факторы, связанные с процессом разработки, могут влиять на оценку усилий, необходимых для тестирования, поскольку тестирование напрямую связано с процессом разработки.

Специалистов:

- Удовлетворенность сотрудников (например, возможность отдыха за счет государственных праздников, отпусков, других ожидаемых преимуществ)
- Навыки и опыт привлеченных специалистов, особенно в отношении аналогичных проектов и продуктов (например, знание прикладной области)

Люди являются самым необходимым ресурсом, поэтому необходимо учитывать любую нестабильность. Следовательно, люди являются важным фактором при оценке усилий, необходимых для тестирования. Смотрите также Раздел 3.1 Команда тестирования.

Результатов тестирования:

- Количество и критичность дефектов, обнаруженных во время выполнения тестов
- Объем необходимых доработок

Исторические статистические данные поддерживают оценку тестирования. Таким образом, знание этих факторов поможет сделать оценку более точной.

Контекста тестирования:

- Распределение тестирования между несколькими дочерними структурами, состав и расположение команд, сложность проекта (например, множество подсистем)
- Тип работы (например, в офисе или удаленная)

Факторы, связанные с контекстом, влияют на общую оценку тестирования. Смотрите также Раздел 1.2 Контекст тестирования.

2.2.3. Выбор методов оценки тестирования

Оценка тестирования должна покрывать все активности, включенные в процесс тестирования. Оцениваемые затраты, усилия и особенно продолжительность выполнения тестов часто являются наиболее важными для управления тестированием, так как эти значения будут влиять на проект. Однако, оценку выполнения тестов обычно трудно предсказать, когда общее качество программного обеспечения низкое или неизвестное. Кроме того, осведомленность и опыт работы с продуктом, вероятно, также будут влиять на качество оценок. Общей практикой является оценивание количества сценариев тестирования, полученных из базиса тестирования (например, требований или пользовательских историй). Предположения, сделанные во время оценки тестирования, всегда должны быть задокументированы как часть оценки.

Методы или подходы к оценке тестирования можно разделить на методы, основанные на метриках, и методы, основанные на экспертизе. Более подробная информация о техниках оценки тестирования объясняется в программе обучения ISTQB® Базового уровня версии 4.

В большинстве случаев, оценка, после её подготовки, должна быть представлена руководству проекта вместе с обоснованием. Часто некоторые входные параметры изменяются (например, объем тестирования), что приводит к корректировке оценки. В идеале окончательная оценка тестирования представляет собой наилучший возможный баланс между организационными и проектными целями в областях качества, графика работ, бюджета и функциональных возможностей.

Важно помнить, что любая оценка основывается на информации, доступной на момент её подготовки. На ранних стадиях проекта информация может быть довольно ограниченной. Кроме того, эта информация может меняться со временем. Чтобы сохранять точность, оценка должна обновляться с учетом новой и изменяющейся информации.

Выбор метода оценки зависит от различных факторов, таких как:

- **Ошибка оценки:** некоторые методы позволяют рассчитать стандартное отклонение, которое является мерой неопределенности или изменчивости оценки. Например, техника оценки по трем точкам использует оптимистичную, пессимистичную и наиболее вероятную оценки для расчета ожидаемого значения и стандартного отклонения оценки (подробнее см. в Разделе 5.1.4 программы обучения ISTQB® Базового уровня версии 4).
- **Доступность данных:** некоторые методы требуют исторических данных из предыдущих или аналогичных проектов, которые могут быть недоступны или ненадежны. Например, оценка на основе коэффициентов и экстраполяции опирается на исторические данные для получения коэффициентов или тенденций для текущего проекта.
- **Доступность экспертов:** некоторые методы требуют вовлечения экспертов, обладающих знаниями и опытом для предоставления точных и реалистичных оценок. Например,

широкополосный метод Дельфи и покер планирования основываются на мнениях и суждениях экспертов или членов команды.

- **Знания в области моделирования:** некоторые методы требуют использования математических моделей или формул для расчета оценок, что может потребовать определенных навыков и знаний в области моделирования. Например, экстраполяция и оценка по трем точкам используют формулы для вычисления ожидаемого значения и стандартного отклонения оценки.
- **Временные ограничения:** некоторые методы требуют больше времени и усилий для выполнения, чем другие, что может повлиять на их осуществимость и пригодность. Например, покер планирования легко выполним, в то время как экстраполяция может быть более сложной.

Это показывает, что критерии выбора подходящих методов оценки тестирования сильно зависят от контекста тестирования (например, жизненного цикла разработки ПО, заинтересованных сторон, уровней тестирования и типов тестов, используемых в проекте) (см. Раздел 1.2 Контекст тестирования). Руководитель тестирования должен уметь координировать и применять методы оценки тестирования (например, с различными моделями жизненного цикла разработки ПО в одном проекте в разных подразделениях).

Например, чтобы выбрать подходящий метод оценки, сначала нужно определить сложность контекста. Если сложность низкая, можно использовать методы, основанные на метриках. Если сложность высокая, можно использовать методы, основанные на экспертизе. Если используется последовательная модель разработки ПО, для оценки можно выбрать широкополосный метод Дельфи. Если используется гибкая модель разработки ПО, можно применить покер планирования.

2.3. Управление дефектами

Введение

Программа обучения ISTQB® Базового уровня версии 4 описывает активности, которые начинаются после наблюдения фактических результатов, отличающихся от ожидаемых. Программа называет эти действия управлением дефектами. Другие стандарты используют термин «управление инцидентами» (стандарт ISO/IEC/IEEE 29119-3) или «управление аномалиями» (TMAP), чтобы подчеркнуть тот факт, что в начале процесса мы можем не знать, вызвано ли расхождение дефектом в рабочем продукте или чем-то другим (например, отказом автоматизации тестирования или недопониманием требований тестировщиком). Управление дефектами и инструмент управления дефектами, имеют критическое значение для тестировщиков и других членов команды, участвующих в разработке программного обеспечения. Информация, полученная с помощью эффективного процесса управления дефектами, позволяет команде тестирования и другим заинтересованным сторонам проекта иметь представление о состоянии проекта на протяжении всего его жизненного цикла. Управление дефектами также важно для принятия решения о том, какие дефекты будут исправлены. Это гарантирует, что усилия будут потрачены на работу с правильными дефектами. Сбор и анализ данных, связанных с дефектами, с течением времени могут помочь выявить области потенциального совершенствования как для тестирования, так и для других процессов в рамках жизненного цикла разработки программного обеспечения (например, лучшая профилактика дефектов за счет усовершенствованной архитектуры и технического проектирования).

Помимо понимания общего жизненного цикла дефектов и того, как он используется для мониторинга и контроля процессов как разработки, так и тестирования программного обеспечения, руководитель тестирования и тестировщики (или вся команда, работающая по гибким методологиям в случае использования в проекте гибкой методологии разработки ПО) также должны быть осведомлены о том, какие данные имеют критическое значение для сбора. Руководитель тестирования должен быть сторонником правильного использования как процесса управления дефектами, так и выбранного инструмента управления дефектами.

2.3.1. Жизненный цикл дефекта

Каждый этап жизненного цикла разработки программного обеспечения должна включать действия по обнаружению и устранению потенциальных дефектов. Например, методы статического тестирования (т.е. рецензирование и статический анализ) могут быть использованы для проверки проектных спецификаций, технических требований ПО и кода до передачи этих рабочих продуктов на следующие этапы разработки. Чем раньше обнаружен и устранен каждый дефект, тем ниже общая стоимость качества продукта. Стоимость качества минимизируется, когда каждый дефект устраняется на том же этапе, где он был обнаружен (т.е. когда процесс разработки программного обеспечения достигает идеальной фазовой локализации).

Во время статического тестирования мы ищем дефекты. Во время динамического тестирования наличие дефекта выявляется, когда он вызывает отказ, что приводит к расхождению между фактическим и ожидаемым результатами теста (т.е. аномалии). В некоторых случаях возникает ложный негативный результат, когда тестировщик не наблюдает аномалию. Когда аномалия обнаружена, следует провести дальнейшее расследование. Обычно это расследование начинается с заполнения отчета о дефекте в соответствии с конкретным процессом управления тестированием и дефектами. Не пройденный тест не всегда приводит к созданию отчета о дефекте. Например, в разработке на основе тестов, где компонентные тесты, обычно автоматизированные, используются как способ описания исполняемой проектной спецификации. Пока разработка компонента не завершена, некоторые или все тесты должны вначале завершаться неудачно. Следовательно, результат такого теста не обязательно вызван дефектом и обычно не отслеживается через отчет о дефекте.

Отчет о дефекте развивается согласно рабочему процессу (для простоты и согласованности с большинством инструментов управления дефектами мы будем дальше использовать термин «жизненный цикл дефекта») и перемещается через последовательность состояний дефекта. В большинстве этих состояний один человек владеет отчетом о дефекте и несет ответственность за выполнение задачи (например, анализ, устранение дефекта или подтверждающее тестирование). Следующая диаграмма представляет собой простой жизненный цикл дефекта:

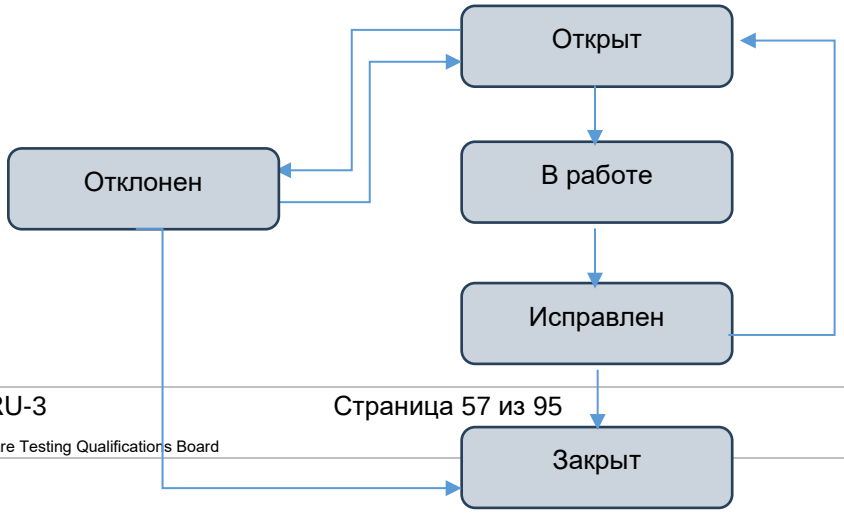

Рисунок 2: простой жизненный цикл дефекта

Простой жизненный цикл дефекта может охватывать следующие состояния дефекта:

- ОТКРЫТ (может называться НОВЫЙ): Исходное состояние при создании отчета о дефекте.
- В РАБОТЕ: Команда работает над анализом отчета о дефекте и/или его исправлением.
- ОТКЛОНЕН: Отчет о дефекте отклонен специалистом, обработавшим его (обычно разработчиком или аналитиком). Причин для отклонения может быть много (например, недействительная информация, неправильный тест, дублирование отчета о дефекте), и эта информация добавляется в отчет о дефекте.
- ИСПРАВЛЕН (может называться ИСПРАВЛЕН И ПРОВЕДЕН, ГОТОВ ДЛЯ ПОВТОРНОЙ ПРОВЕРКИ): Тестировщик проводит подтверждающее тестирование, часто следуя шагам воспроизведения отказа из самого отчета о дефекте, чтобы определить, действительно ли исправление устранило дефект.
- ЗАКРЫТ: Отчет о дефекте достиг своего конечного состояния, и дальнейшая работа с ним не планируется. Тестировщик переводит отчет о дефекте в это состояние либо после успешного подтверждающего тестирования, либо для подтверждения отклонения отчета о дефекте.

Простой жизненный цикл дефекта используется во многих организациях и расширяется за счет использования других состояний дефектов, актуальных для данного контекста (например, ПЕРЕОТКРЫТ, ПРИНЯТ, НА УТОЧНЕНИИ или ОТЛОЖЕН).

Жизненный цикл дефекта может различаться в разных организациях по названиям состояний дефектов, правилам переходов между состояниями дефектов и ролям, ответственным за задачи в данных состояниях дефектов. Часто жизненный цикл дефекта более простой в гибкой модели разработки программного обеспечения, чем в последовательных моделях. Жизненный цикл дефекта должен быть адаптирован к конкретному контексту. При разработке жизненного цикла дефекта рекомендуется учитывать несколько лучших практик:

- Если возможно, жизненный цикл дефекта должен быть определен на уровне всей организации, чтобы обеспечить единое управление дефектами во всех проектах.
- Дублирующие и ложные позитивные дефекты должны быть представлены отдельным состоянием или сочетанием статуса ОТКЛОНЕН с указанием причины отклонения. Они могут быть полезны при дальнейшем анализе дефектов с целью улучшения процесса тестирования.
- Рекомендуется использовать только одно конечное состояние (например, ЗАКРЫТ). Переход в это состояние часто требует указания причины закрытия, что полезно для оценки процесса и активностей по его улучшению.

- Названия состояний в жизненном цикле дефекта должны совпадать с аналогичными состояниями, используемыми для других сущностей (например, пользовательских историй и задач тестирования), чтобы упростить работу с ними.
- Последовательные состояния дефекта должны принадлежать разным ролям. Если два или более последовательных состояния принадлежат одной и той же роли, то для этого должна быть веская причина (например, для измерения времени нахождения дефекта в конкретном состоянии).
- Каждое состояние дефекта, за исключением конечного состояния, должно иметь более одного исходящего перехода, чтобы роль могла принять решение о следующем шаге. Исключения из этого правила должны быть обоснованы (например, для мониторинга времени, затраченного на конкретную активность).
- Набор атрибутов, которые необходимо заполнить при выполнении перехода в другое состояние, должен быть ограничен теми, которые приносят значительную пользу для управления дефектами.

2.3.2. Межфункциональное управление дефектами

Хотя организация, ответственная за тестирование, и руководитель тестирования часто владеют полным представлением о процессе управления дефектами и инструментах, необходимых для этого, обычно за управление дефектами в конкретном проекте отвечает межфункциональная команда. Иногда называемая группой управления дефектами, она может включать руководителя тестирования, представителей команды разработки, поставщиков, руководителей проекта, руководителей или владельцев продукта и других лиц, заинтересованных в тестируемом программном обеспечении.

Как только аномалии обнаружены и зафиксированы с помощью инструмента управления дефектами, группа управления дефектами должна определить, представляет ли каждый отчет о дефектах действительный дефект и должен ли он быть исправлен (и какой стороной, если в поставке участвуют несколько команд разработки), отклонен или отложен. Это решение требует, чтобы группа управления дефектами рассмотрела преимущества, риски и затраты, связанные с устранением дефекта. Полезно обсудить этот вопрос на встрече (часто называемой встречей по сортировке дефектов). Если дефект необходимо исправить, команда должна установить приоритет исправления дефекта относительно других задач. Руководитель и команда тестирования могут проконсультировать относительно важности дефекта, и предоставить имеющуюся объективную информацию.

В очень крупных проектах назначение штатного руководителя дефектов может быть оправдано усилиями, необходимыми для подготовки и выполнения решений, принятых на встречах группы управления дефектами, по крайней мере, на тех этапах жизненного цикла разработки программного обеспечения, когда тестирование наиболее интенсивно. В других ситуациях несколько крупных проектов могут находиться в ведении одного руководителя дефектов.

Инструмент управления дефектами не должен использоваться в качестве замены хорошей коммуникации, так же, как и группа управления дефектами не должна заменять эффективное использование хорошего инструмента управления дефектами. Коммуникация, адекватная инструментальная поддержка, четко определенный жизненный цикл дефектов (включая свойства отчета о дефектах) и вовлеченная команда по управлению дефектами — все это необходимо для эффективного и действенного управления дефектами.

2.3.3. Особенности управления дефектами в командах, работающей по гибкой методологии разработки программного обеспечения

Управление дефектами в организациях, использующих гибкую разработку программного обеспечения, часто является упрощенным и/или менее формальным, чем в моделях последовательной разработки. Если команды, работающие по гибкой методологии, расположены в одном и том же месте или имеют хорошо налаженные средства связи, информация о дефекте или отказе часто распространяется между тестировщиками, представителями клиентов и разработчиками без официального отчета о дефекте. Однако отчеты о дефектах следует создавать для:

- дефектов, которые блокируют другие активности текущей итерации (например, разработку, тестирование и т. д.) и не могут быть исправлены немедленно внутри команды;
- дефектов, которые невозможно устранить в течение текущей итерации. В некоторых командах, работающих по гибкой методологии, есть правило создания отчета о дефекте, если дефект не может быть устранен в течение дня, когда он был обнаружен;
- дефектов, которые должны быть устранены другими командами или в сотрудничестве с ними в организациях с множеством команд;
- дефектов, которые должен устранить поставщик;
- дефектов, отчет о которых явно запрашивается (например, когда разработчик не может немедленно приступить к исправлению).

Обычной практикой является добавление дефектов, которые не могут быть устранены в течение текущей итерации, в набор планируемых задач (бэклог), чтобы их можно было приоритизировать наряду с другими дефектами и пользовательскими историями для более поздней итерации.

Хотя основы управления дефектами должны быть заложены в корпоративной стратегии тестирования, многие аспекты, включая уровень формальности, триггеры создания отчета о дефектах и атрибуты дефектов, которые необходимо фиксировать, могут быть оставлены на усмотрение членов команды. В целом уровень формальности в управлении дефектами и подход к созданию отчетов о дефектах должны отражать следующее:

- совместное размещение членов команды
- распределение членов команды по часовым поясам
- количество команд, которые сотрудничают при разработке продукта
- зрелость команд(ы)
- размер команд(ы)
- риски, связанные с продуктом
- нормативные, договорные или другие требования (если и где это применимо)

Окончательное решение команды, работающей по гибкой методологии разработки программного обеспечения (Agile), относительно деталей управления дефектами всегда должно быть задокументировано (например, вместе с методическими рекомендациями в инструменте управления знаниями).

2.3.4. Проблемы управления дефектами при использовании гибридной методологии разработки программного обеспечения

На практике над созданием системы или комплекса систем часто сотрудничают несколько команд. Примеры включают разработку программного обеспечения по гибридной методологии, когда заказчик использует гибкую методологию разработки программного обеспечения, а один из его поставщиков использует модель последовательной разработки, или когда организация, использующая модель последовательной разработки, требует поставки подсистемы от команды, использующей гибкую методологию разработки программного обеспечения. Такая среда с множеством команд создает различные проблемы:

- **Согласование атрибутов дефектов и инструментов управления дефектами.** В идеальном сценарии все команды используют один инструмент управления дефектами. На практике каждая команда обычно использует разные инструменты управления дефектами, особенно когда в реализации проекта участвуют несколько команд поставщиков. В таких случаях полезно установить синхронизацию между инструментами управления дефектами (желательно автоматическую).
- **Приоритизация дефектов.** Владельцев продукта следует привлекать к участию во встречах по управлению дефектами и активно искать информацию о последствиях и рисках, связанных с дефектами. Встречи по управлению дефектами следует проводить чаще при гибкой разработке программного обеспечения, чем при последовательной, чтобы идти в ногу с быстрыми темпами разработки продуктов Agile-командой. Однако в гибких командах эти встречи могут быть короче. Иногда полезно, чтобы последнее слово в определении приоритетности дефектов оставалось за небольшой группой лиц, заинтересованных в управлении дефектами.
- **Согласованность и прозрачность плана тестирования для новых разработок и исправления дефектов.** Работа всех команд должна строиться по одному и тому же плану проекта, независимо от того, используют ли они гибкую разработку программного обеспечения или модели последовательной разработки. Все практические результаты, включая исправления дефектов, должны быть приведены в соответствие с этим планом проекта. Лучшего соответствия можно достичь за счет активного участия членов всех команд в процессе планирования (например, участие команд последовательной модели разработки во встречах команд гибкой разработки, где обсуждаются дефекты и расставляются приоритеты). Прозрачность планов разработки можно повысить, разделив их между командами (например, через информационные панели или через набор планируемых задач).

2.3.5. Информация отчета о дефектах

Информации в отчете о дефекте должно быть достаточно для следующих целей:

- управление отчетом о дефектах на протяжении всего жизненного цикла дефекта
- оценка общего статуса проекта, особенно с точки зрения качества продукта и хода тестирования
- оценка состояния роста продукта с точки зрения его качества
- оценка возможностей процесса

Информация, необходимая для управления дефектами и определения статуса проекта, может варьироваться в зависимости от того, в какой момент жизненного цикла разработки программного

обеспечения обнаружен дефект. Кроме того, для отчетов о дефектах, связанных с нефункциональными характеристиками качества, может потребоваться дополнительная информация (например, модель нагрузки для проблем с производительностью). Однако собранная основная информация должна быть единообразной во всем жизненном цикле разработки ПО и, в идеале, во всех проектах в организации, чтобы обеспечить значимое сравнение данных о дефектах в рамках проекта и между всеми проектами.

В отчет о дефектах можно собрать множество данных. Руководитель тестирования должен решить, какая информация подходит для эффективного управления дефектами в данном контексте проекта. В связи с тем, что каждый дополнительный атрибут увеличивает время, затрачиваемое на создание отчета о дефектах, и может привести к путанице у специалиста, его заполняющего, рекомендуется собирать только те данные, которые необходимы для управления дефектами в данном контексте и/или будут использоваться для улучшения процессов.

Для управления отчетом о дефектах в большинстве случаев обязательным является:

- название дефекта с кратким описанием аномалии
- подробное описание аномалии, желательно включая шаги по воспроизведению отказа
- критичность воздействия на тестируемую систему и/или лиц, заинтересованных в продукте
- приоритет исправления аномалии

Инструмент управления дефектами часто создает дополнительные важные элементы данных:

- уникальный идентификатор отчета о дефекте
- дата/время создания отчета о дефекте
- имя специалиста, обнаружившего и/или сообщившего об аномалии
- этап проекта и жизненного цикла разработки ПО, на котором была обнаружена аномалия
- текущее состояние отчета о дефекте
- текущий владелец (т. е. лицо, которому в настоящее время поручено работать над дефектом)
- история изменений, такая как последовательность действий, включая информацию о дате и времени, принятая членами проектной группы для изоляции, устранения и подтверждения дефекта как исправленного.
- ссылки (например, на сценарий тестирования, на связанные дефекты).

В зависимости от контекста дополнительная информация (например, трассируемость) также может быть собрана в отчете о дефекте (дополнительную информацию см. в ISO/IEC/IEEE 29119–3). Следующие пункты группируют информацию в соответствии с предполагаемым назначением:

- **Помощь в устранении дефекта:** подсистема или компонент, в котором обнаружен дефект, конкретный элемент тестирования и номер его версии, в которой наблюдалась аномалия, или тестовое окружение, в котором наблюдался дефект.
- **Оценка общего состояния проекта:** информация для отслеживания прогресса (например, риски, затраты, возможности и выгоды, связанные с исправлением или не исправлением дефекта, описание любого доступного обходного пути или требований, на которые влияют дефекты).

- **Оценка статуса роста продукта с точки зрения его качества:** тип дефекта (обычно соответствующий классификации дефектов), рабочий продукт, в котором был обнаружен дефект, или характеристика/подхарактеристика качества, на которую влияет дефект.
- **Оценка возможностей процесса:** информация для отслеживания эффективности и результативности процессов разработки (например, этап внедрения, обнаружения и устранения дефекта или его первопричины).

2.3.6. Определение активностей по совершенствованию процесса с использованием информации отчета о дефектах

Как обсуждалось в Разделе 2.3.5 Информация отчета о дефектах, отчеты о дефектах могут быть полезны для отслеживания статуса проекта и составления отчетов. Хотя влияние метрик на процесс тестирования в первую очередь рассматривается в программе обучения Экспертного уровня управления тестированием, на продвинутом уровне управления тестированием руководители тестирования должны знать, какое значение отчеты о дефектах имеют для оценки возможностей процессов разработки и тестирования программного обеспечения.

В дополнение к информации об отслеживании хода тестирования, упомянутой в Разделе 2.1.2 Мониторинг, контроль и завершение тестирования и в Разделе 2.1.3 Отчетность тестирования, информация о дефектах должна поддерживать инициативы по совершенствованию процессов, как обсуждалось во время ретроспектив. Примеры включают:

- Использование информации об этапах появления, обнаружения и устранения дефектов для оценки фаз сдерживания и/или выполнения анализа стоимости качества с целью предложения способов повышения эффективности обнаружения дефектов на каждом этапе и минимизации затрат, связанных с дефектами.
- Использование информации об этапе появления дефекта для анализа фаз, где появляется наибольшее количество дефектов, чтобы обеспечить целевые улучшения для предотвращения дефектов.
- Использование информации о первопричине дефекта для определения основных причин появления дефекта, чтобы обеспечить возможность совершенствования процесса, позволяющего уменьшить общее количество дефектов.
- Использование информации о местоположении дефектов для выполнения кластерного анализа дефектов, для лучшего понимания технических рисков (для тестирования на основе рисков) и улучшения кода проблемных компонентов.
- Использование информации о повторно открытых дефектах для оценки качества отладки реализаций.
- Использование информации о повторяющихся и отклоненных дефектах для оценки качества создания отчета о дефектах.
- Включение процессов совершенствования, которые уменьшают общее количество дефектов, вводя предупредительные меры, чтобы избежать ошибок заранее.

Использование метрик оценки эффективности и результативности процесса тестирования обсуждается в программе обучения Экспертного уровня управления тестированием.

В некоторых случаях команды решают не отслеживать дефекты, обнаруженные на некоторых или всех этапах жизненного цикла разработки программного обеспечения. Хотя это часто делается ради эффективности и сокращения расходов процесса, однако значительно снижает прозрачность возможностей процесса разработки и тестирования программного обеспечения. Что в свою очередь

затрудняет реализацию предложенных выше улучшений из-за отсутствия надежных вспомогательных данных.

3 Управление командой тестирования

Ключевые слова

Внешние отказы, внутренние отказы, дефект, отказ, оценивание, предотвращение дефектов, стоимость качества.

3.1 Команда тестирования

TM-3.1.1 (K2) Привести примеры типичных навыков в четырех областях компетенции, которыми необходимо обладать членам команды тестирования.
TM-3.1.2 (K4) Проанализировать описанные условия реализации проекта, чтобы определить необходимые навыки членов команды тестирования.
TM-3.1.3 (K2) Объяснить типичные методы оценки навыков членов команды тестирования.
TM-3.1.4 (K2) Объяснить разницу между типичными подходами к развитию навыков членов команды тестирования.
TM-3.1.5 (K2) Объяснить навыки, необходимые для управления командой тестирования.
TM-3.1.6 (K2) Привести примеры факторов мотивации и гигиенических факторов для членов команды тестирования.

3.2 Взаимоотношения с заинтересованными сторонами

TM-3.2.1 (K2) Привести примеры для каждой из четырех категорий, определяющих стоимость качества.
TM-3.2.2 (K3) Применить расчет соотношения затрат и выгод для оценки добавленной стоимости тестирования для заинтересованных сторон.

3.1. Команда тестирования

Введение

Команда тестирования состоит из людей с различными компетенциями. В то время как в некоторых организациях команды самоорганизуются, в других руководители тестирования набирают и развивают эти команды. Правильное сочетание навыков[1] является критическим фактором успешного выполнения задач по тестированию для любых команд.

Требования к навыкам членов команды тестирования могут со временем меняться. Важно выбирать правильных специалистов, предоставлять адекватное обучение и возможности роста. Кроме того, специалисты в других сферах могут предоставлять дополнительные специфические знания членам команды.

В этом разделе рассматривается фундаментальный процесс анализа и развития необходимых навыков членов команды тестирования, а также навыки, необходимые для наставничества или руководства командой тестирования, что включает знание факторов, которые мотивируют или демотивируют членов команды тестирования, а также других факторов, обеспечивающих успешную командную работу.

Каждый специалист обладает определенными навыками и может развивать эти навыки различными способами, включающими приобретение профессионального опыта, образование и обучение. Идеальная команда тестирования обладает всеми необходимыми навыками для выполнения конкретных задач по тестированию или отвечает только за те задачи, для которых у нее есть необходимые навыки. Для успешной работы команда тестирования должна обладать различными навыками на разных уровнях. В зависимости от условий проекта одни навыки будут более важными или необходимыми, чем другие. В некоторых случаях имеет смысл привлекать внешних экспертов для выполнения конкретных задач по тестированию, которые выходят за рамки возможностей команды.

3.1.1. Типичные навыки в четырех областях компетенции

Навыки человека можно распределить по четырем областям компетенции (Sonntag & Schmidt-Rathjens, 2005) (Erpenbeck & von Rosenstiel, 2017)[2]:

Профессиональная компетенция: состоит из навыков выполнения специализированных задач. Примеры включают навыки применения методов тестирования, технологическую и бизнес-экспертизу в прикладных областях, а также навыки управления проектами.

[1] Термин «навык» используется как обобщающий термин для обозначения собственно навыков, наличия знаний о чем-то и способности что-то делать.

[2] Четыре упоминаемые области компетенции являются частью широко известной модели, описанной в указанных источниках. Существуют и другие модели, описанные в литературе, которые группируют навыки иными способами. Эти модели и их изучение не являются частью данной программы.

Методологическая компетенция: включает общие навыки, которые человек может использовать в какой-то прикладной области и которые позволяют самостоятельно выполнять сложные или новые задачи. В качестве примера можно привести аналитические, концептуальные и оценочные навыки.

Социальная компетенция: включает навыки, связанные с коммуникацией, взаимодействием и управлением конфликтами во внутрикультурных и межкультурных средах. Они позволяют сотруднику взаимодействовать с коллегами, действовать адекватно в конкретной ситуации, а также достигать индивидуальных и общих целей. Примерами в данном случае являются коммуникативные навыки, навыки разрешения конфликтов, умение работать в команде, адаптивность и настойчивость.

Личная компетенция: включает способность и готовность специалиста к саморазвитию, развитию своих талантов, мотивацию и готовность к выполнению задач, а также развитие специфических установок и индивидуальности. Примерами таких компетенций являются: самоуправление, личная ответственность, способность воспринимать критику, надежность, стойкость, способность действовать уверенно, дисциплину, открытость к изменениям, готовность помогать и учиться, а также способность делегировать.

Все области компетенции важны для успеха любой команды тестирования. Поскольку методологическая, социальная и личная компетенции не являются специфичными для тестирования, ISTQB® сосредотачивается на развитии профессиональной компетенции. Она включает навыки управления задачами по тестированию, анализа базиса тестирования, проектирования тестов, выявления и анализа рисков, а также разработки, настройки и поддержки тестовых данных, тестовых окружений и сценариев тестирования.

3.1.2. Анализ необходимых навыков членов команды тестирования

Подбор персонала является частью планирования тестирования. Этот процесс включает в себя задачу определения ролей и навыков сотрудников, необходимых для реализации тестирования в рамках стратегии тестирования. Для определения необходимых навыков для проекта требуется детальный анализ условий его реализации.

Профессиональная и методологическая компетенции

В тестировании основной акцент делается на навыках, необходимых для выполнения задач тестирования. Ниже приведены некоторые примеры:

- Планирование тестирования требует концептуальных знаний для разработки стратегии тестирования.
- Мониторинг и контроль тестирования требуют навыков управления проектами для управления всеми задачами тестирования.
- Анализ тестирования требует аналитических навыков для анализа базиса тестирования и рисков продукта.
- Проектирование тестов требует навыков применения методов тестирования для разработки сценариев тестирования и концептуальных знаний для проектирования тестового окружения.

- Реализация тестирования требует навыков оценки объекта тестирования для выбора тестов и технической экспертизы для автоматизации сценариев тестирования, и настройки тестового окружения.
- Выполнение тестирования требует технической экспертизы для выполнения автоматизированных тестов, проведения исследовательского тестирования и оценки результатов тестирования.
- Завершение тестирования требует коммуникативных навыков для представления результатов проекта и личной ответственности за принятые решения.

Различные типы тестирования и уровни тестирования требуют различных навыков (например, бизнес-экспертиза в прикладной области для оценки функциональной пригодности системы или техническая экспертиза для оценки трудозатрат на сопровождаемости кода).

Кроме того, условия реализации проекта предоставляют ценную информацию о необходимой профессиональной компетенции:

Предметная область проекта требует обладания бизнес-экспертизой в таких прикладных областях как информационные технологии, автомобилестроение или индустрия азартных игр.

Программная и системная архитектура, а также технологии, используемые в проекте, требуют, например, технической экспертизы в языках программирования, интерфейсных технологиях или уязвимостях безопасности.

Жизненный цикл разработки программного обеспечения (SDLC) требует, например, знаний об уровнях тестирования, ролях в тестировании и конкретных методиках тестирования.

Социальная компетенция

В контексте тестирования социальная компетенция позволяет членам команды тестирования наладить рабочие отношения другими членами команды и достигать целей тестирования. В частности, это включает коммуникативные навыки, навыки сотрудничества и разрешения конфликтов (например, конструктивное отношение к неблагоприятным условиям работы или сообщению об ошибках разработчикам).

Разработка и тестирование программного обеспечения обычно выполняются разными членами (разных) групп, которые координируют свои задачи посредством коммуникации. Для успешной реализации проекта требуются навыки общения, умение работать в команде и способность разрешать конфликты. Однако требуемый уровень социальных навыков может различаться в зависимости от контекста проекта. Например, разработка программного обеспечения с применением гибких методологий может предъявлять более высокие требования к социальным навыкам по сравнению с документ-ориентированными последовательными моделями разработки, а также в условиях удаленной работы.

Личная компетенция

Эффективность и продуктивность членов команды тестирования также зависят от их способности и готовности развиваться, совершенствовать свои навыки и установки. Например, работа в самоорганизующейся команде, использующей гибкую методологию разработки, может требовать

более высокого уровня самоуправления и дисциплины от всех членов команды, в то время как руководителю тестирования иерархической команды, необходимо уметь делегировать работу. Высокая степень надежности и устойчивости особенно часто требуется в проектах с критическими сроками. Кроме того, готовность помогать, учиться и открытость к изменениям важны в процессе изменений во всех моделях жизненного цикла разработки программного обеспечения (SDLC).

3.1.3. Оценивание навыков членов команды тестирования

В большинстве случаев команды тестирования формируются из сотрудников организации. Чтобы понять возможности членов команды и необходимость их личного развития, руководителю тестирования необходимо оценить существующие навыки команды тестирования и сравнить их с требуемыми навыками, которые могут быть задокументированы в матрице навыков.

Существуют модели, помогающие командам работать более эффективно (например, модель командных ролей Мередита Белбина, DISG® или PCM®). Согласно Белбину (Belbin, 2010), команды работают более эффективно, когда состоят из сотрудников с разными типами личности и ее члены выполняют в ней разные роли. Эти модели помогают командам определить, какие навыки у них имеются, и каких навыков может не хватать.

Профессиональная и методологическая компетенция членов команды тестирования может быть оценена посредством использования типичных задач тестирования:

- Определение стратегии тестирования и обсуждение обратной связи с коллегами
- Анализ базиса тестирования и обсуждение результатов, что также может выявить навыки коммуникации
- Определение техник проектирования тестов для достижения конкретных целей тестирования в условиях реализации конкретного проекта
- Правильное применение различных техник проектирования тестов
- Написание отчета о завершении тестирования, включающего оценку результатов тестирования

Кроме того, навыки могут оцениваться при помощи полученных сотрудниками удостоверений, сертификатов, дипломов и рабочего опыта.

Специфика применения гибких моделей в разработке программного обеспечения заключается в том, что члены команды самостоятельно определяют необходимые им навыки, регулярно участвуя в ретроспективах и получая обратную связь. Опытные наставники помогают им развивать эти навыки, а также выявлять и устранять пробелы в знаниях.

3.1.4. Развитие навыков членов команды тестирования

Команда тестирования может не обладать всеми необходимыми навыками в начале проекта. И хотя идеальный набор специалистов может быть недоступен, сильная команда способна компенсировать слабые стороны отдельных ее членов.

Руководитель тестирования или команда тестирования могут определить необходимые потребности в развитии, сравнив требуемые и доступные навыки в матрице навыков. На этой основе они могут определить подходы к развитию компетенций:

- При помощи обучения могут быть переданы заранее определенные знания и навыки обычно в (виртуальном) классе (например, отправка сотрудников на курсы, проведение обучающих сессий в организации, разработка индивидуального обучения или использование онлайн-курсов).
- Самообучение — это способ изучения предмета, который предполагает самостоятельное обучение, а не в (виртуальном) классе (например, чтение книг, просмотр записанных видео или поиск информации в Интернете).
- Взаимное обучение, при котором коллеги делятся знаниями, идеями и опытом и учатся друг у друга.
- Наставничество, или менторство — это подходы, при которых член команды, выполняя новую для себя роль, получает индивидуальное руководство от тренера или знания, навыки и/или опыт от наставника. Опытный ментор выступает в качестве источника знаний, давая советы и оказывая необходимую помощь.
- Обучение на рабочем месте также хорошо известно и является смесью самообучения, взаимного обучения и наставничества.

Не все подходы к развитию компетенций одинаково эффективны и результативны. Самообучение и обучение, например, хорошо подходят для развития профессиональной и методологической компетенций. В связи с этим базовые знания в области тестирования можно развить, участвуя в учебных сессиях ISTQB® или путем самообучения на основе программ обучения ISTQB®. Однако для развития социальной и личной компетенций рекомендуется использовать такие подходы, как обучение и менторство, которые часто более перспективны, чем самообучение. Социальный обмен, обратная связь и самоанализ являются ключевыми факторами успеха в развитии социальной и личной компетенций.

3.1.5. Навыки управления, необходимые для руководства командой тестирования

Любому, кто хочет успешно руководить командой тестирования, необходимо обладать управленческими навыками. Они предполагают использование профессиональной и методологической компетенций при решении основных управленческих задач (например, планирование, мониторинг прогресса, контроль и отчетность). Для тестирования требуются специфические знания и навыки управления (например, знание различных подходов к тестированию, разработка стратегий тестирования, использование техник проектирования тестов или модели жизненного цикла разработки программного обеспечения).

Руководство командой тестирования означает соответствующее поведение в отношениях с другими членами команды тестирования, а также обладание способностью и готовностью развиваться в изменяющихся условиях. По этой причине обладание социальной и личной компетенциями является важным фактором успеха для руководства командой тестирования. Это включает настойчивость, способность делегировать задачи и умение ладить с членами команды тестирования. Кроме того, это включает в себя способность отстаивать интересы команды тестирования в проекте, быть активным сторонником лучших практик тестирования, а также общаться и разрешать конфликты со всеми заинтересованными сторонами.

Для отбора сотрудников в команду тестирования необходимы навыки анализа социальных, командных и рабочих условий. Эти навыки помогают обеспечить соответствие команды тестирования рабочим условиям или, если возможно, адаптировать рабочие условия к требованиям

команды тестирования. Кроме того, в командах тестирования регулярно происходят изменения, связанные с их развитием, в связи с чем руководителю тестирования необходимо уметь оперативно реагировать на такие изменения (например, в соответствии с фазами модели развития команд Такмана) (Tuckman, 1965) (Bonebright, 2010):

- Готовность помочь членам команды тестирования влиться в коллектив (Формирование)
- Способность разрешать конфликты внутри команды тестирования (Конфликт)
- Дисциплина и целенаправленное руководство для обеспечения соблюдения согласованных ценностей и правил (Нормирование)
- Способность делегировать задачи для формирования у членов команды тестирования чувства личной ответственности (Функционирование)
- Умение проявлять признательность и доверие к уходящим членам команды тестирования (Расставание)

3.1.6. Факторы, мотивирующие и демотивирующие команду тестирования в определенных ситуациях

Удовлетворенные и мотивированные члены команды тестирования гораздо производительнее, что оказывает значительное влияние на успех проектов. В такой ситуации взаимное обучение происходит неформально, члены команды тестирования могут самостоятельно управлять своей рабочей нагрузкой, а у руководителя тестирования появляется больше времени для решения других вопросов. Двухфакторная теория мотивации (Herzberg и др., 1993) различает мотиваторы и гигиенические факторы:

Мотиваторы осознаются индивидуумом и могут приводить к развитию и удовлетворению.

К ним можно отнести:

- Оценку и признательность за выполненную работу (например, поощрения и любые другие индивидуальные подходы в оценке выполненной работы, которые члены команды тестирования считают ценными)
- Повышенную ответственность и самостоятельность (например, возможность определять процессы тестирования в команде тестирования)
- Интересные, значимые и сложные задачи, которые члены команды тестирования считают достижимыми и одновременно стоящими затрачиваемых усилий (например, выбор и внедрение нового инструмента для автоматизации тестирования)
- Профессиональный рост и развитие (например, развитие опытного тестировщика до руководителя тестирования или владельца процесса тестирования)

Гигиенические факторы обычно воспринимаются как само собой разумеющиеся. Их наличие или выполнение не приводит автоматически к большему удовлетворению индивидуума. Однако их отсутствие может оказывать демотивирующее воздействие на членов команды тестирования:

- Соответствующее вознаграждение (например, рыночная зарплата, оплачиваемые переработки, хорошие социальные льготы)

- Кадровая политика и стиль управления, в основе которых лежит ценность персонала (например, бережливое управление, реалистичные цели, защита от внешнего вмешательства и перегрузки)
- Адекватные условия труда (например, однозначные технические требования, зрелые объекты тестирования, адекватный подход к устранению дефектов, соответствующая рабочая обстановка, стабильная тестовая среда)
- Безопасность как экзистенциальная потребность (например, безопасное рабочее место и соблюдение договоренностей)
- Хорошие межличностные отношения (например, с коллегами, руководителями)

Таким образом, задачей руководителя тестирования является устранение демотивирующих факторов и одновременно с этим создание и укрепление мотивирующих факторов.

Дополнительную информацию можно найти в (Belbin, 2010) (Marston, 1999) (Kahler, 2008).

3.2. Взаимоотношения с заинтересованными сторонами

Введение

В управлении тестированием для того, чтобы оказывать услуги, обладающие большой бизнес-ценностью важно оптимизировать процессы тестирования. Избыточное тестирование может привести к неоправданным задержкам и затратам, которые превышают полученные выгоды, в то время как недостаточное тестирование может привести к выпуску продукта низкого качества. Оптимальный подход лежит между этими двумя крайностями. Задача руководителя тестирования — помочь заинтересованным сторонам понять этот баланс и добавленную стоимость тестирования в его достижении, учитывая, например, типичные временные ограничения проекта.

3.2.1. Стоимость качества

Преимущества тестирования уравновешиваются затратами на качество. Количественная оценка, используемая для определения общих затрат на обеспечение качества и устранение дефектов, называется стоимостью качества. В стоимость качества включаются проектные и операционные затраты, разделенные на четыре категории в соответствии со структурой затрат на устранение дефектов продукта:

Затраты на предотвращение дефектов: затраты на все запланированные и реализуемые мероприятия по предотвращению низкого качества (например, обеспечение квалификации разработчиков для выполнения таких задач как обучение созданию поддерживаемого или безопасного кода, как можно более ранний обзор базиса тестирования и соответствующая коммуникация внутри команды).

Затраты на оценку: затраты на все мероприятия, направленные на обнаружение дефектов (например, проведение статического и динамического тестирования и обзор рабочих продуктов).

Затраты на внутренние отказы: затраты на все реактивные мероприятия (например, исправление дефектов, найденных в процессе тестирования, предоставление обходных решений).

Затраты на внешние отказы: затраты на все не добавляющие ценности и являющиеся реактивными мероприятия (например, потеря доходов, активов, здоровья или жизни индивидуума, ущерб окружающей среде, юридические издержки, связанные с исправлением дефектов, тестированием, развертыванием и поддержкой из-за выпуска дефектного продукта ("пост-релизного"), исправление дефектов, выявленных пользователями).

Общие затраты на оценку и внутренние отказы обычно значительно ниже, чем затраты на внешние отказы. Поэтому тестирование является чрезвычайно ценным. Определив затраты в этих четырех категориях, руководитель тестирования может создать убедительное обоснование для проведения тестирования.

Существуют и другие подходы, которые можно рассматривать для определения стоимости качества. В данной программе обучения ISTQB® описываются два из них. В основе этой программе обучения – подход Фейгенбаума, а в программе обучения ISTQB® Базового уровня версии 4 представлен подход Боэма (см. программу обучения ISTQB® Базового уровня версии 4, Раздел 1.3, Принципы тестирования). Эти два подхода были выбраны для более широкого понимания стоимости качества. Подход Фейгенбаума (Feigenbaum, ноябрь/декабрь 1956) рассматривает качество как ориентированный на клиента и общекорпоративный процесс, в то время как подход Боэма (Boehm, 1979) сосредотачивается на компромиссе между стоимостью предотвращения дефектов и стоимостью отказов в разработке программного обеспечения (Hadjicostas, 2004).

3.2.2. Соотношение затрат и выгод тестирования

Хотя большинство организаций считают тестирование ценным в некотором смысле, немногие руководители, включая руководителей тестирования, могут количественно оценить, описать или выразить эту ценность. Кроме того, многие руководители тестирования, лидеры команд тестирования и тестировщики сосредотачиваются на операционных деталях тестирования (т.е. аспектах, специфичных для задач тестирования или уровня тестирования), игнорируя более крупные тактические и стратегические (более высокоуровневые) вопросы, связанные с тестированием, которые волнуют других заинтересованных сторон, особенно руководителей высшего звена.

Тестирование приносит выгоды организации, проекту и/или операциям как в количественном, так и в качественном отношении:

Качественные выгоды включают повышение репутации за обеспечение качества продукта, более плавные и предсказуемые выпуски продукта, повышенную уверенность, защиту от юридической ответственности и снижение риска потери целых миссий или даже жизней.

Количественные выгоды включают найденные, предотвращенные или исправленные до выпуска дефекты, дефекты, которые были выявлены до выпуска (не исправленные, но задокументированные, возможно, с обходными решениями), экономические выгоды (Bohm 1981, Böhler 2008), снижение уровня риска путем проведения тестов и предоставление информации о статусе проекта, процесса и продукта.

Дополнительной выгодой тестирования является то, что все заинтересованные стороны получают адекватную информацию для принятия обоснованных решений о том, является ли качество продукта достаточным для запуска в эксплуатацию, с дефектами или без них. Иногда выпуск продукта с известными дефектами гораздо лучше, чем ожидание выпуска до устранения дефектов.

В случаях, когда дефект может быть допустим, решение сильно зависит от вероятности его возникновения и серьезности дефекта.

Стоимость качества в расчете на один дефект при тестировании рассчитывается следующим образом:

Средняя экономия в расчете на один дефект = Средние затраты на внешние отказы на один дефект - (Средние затраты на предотвращение одного дефекта + Средние затраты на оценку на один дефект + Средние затраты на внутренние отказы на один дефект)

Общая стоимость качества = (Затраты на предотвращение дефектов + (Средние затраты на оценку на один дефект * Количество дефектов, найденных до выпуска)) + ((Средние затраты на внутренние отказы на один дефект * Количество дефектов, найденных до выпуска) + (Средние затраты на внешние отказы на один дефект * Количество дефектов, найденных после выпуска))

В качестве примера предположим, что вы рассчитали следующие затраты на качество на один дефект для продукта:

- Затраты на предотвращение дефектов: $180
- Средние затраты на оценку на один дефект: $500
- Средние затраты на внутренние отказы на один дефект: $200
- Средние затраты на внешние отказы на один дефект: $4,000

Средние затраты на предотвращение дефектов, затраты на оценку и затраты на внутренние отказы рассчитываются с использованием количества дефектов, найденных до выпуска, в то время как средние затраты на внешние отказы рассчитываются с использованием количества дефектов, найденных после выпуска. С этими значениями мы можем рассчитать среднюю экономию на один дефект следующим образом:

Средняя экономия в расчете на один дефект = $4,000 – ($500 + $200) = $3,300.

Кривая Боэма является графическим представлением затрат на исправление дефектов со временем в жизненном цикле разработки программного обеспечения (SDLC). Она позволяет сделать вывод о том, что тестирование должно проводиться на более ранних этапах SDLC, чтобы снизить стоимость исправления дефектов. Кривая Боэма показывает, что затраты на внутренние отказы, или стоимость исправления дефекта, увеличиваются по мере того, как дефект обнаруживается на более поздних этапах SDLC. Используя эту информацию, руководитель тестирования должен стремиться найти оптимальное соотношение между затратами на предотвращение дефектов и внутренними и внешними затратами.

Объемы тестирования должны основываться на конкретном риске проекта и продукта, а также на риске, который бизнес готов принять. Избыточное тестирование может привести к более высоким затратам, чем выгоды от снижения уровня риска. Если тестирование недостаточное, пропущенные дефекты могут представлять высокий риск, приводящий к более высоким затратам, чем стоимость не проведенных тестов. Тестирование на основе рисков (см. Раздел 1.3 Тестирование на основе рисков), обеспечивает соотношение затрат и выгод тестирования, распределяя усилия на тестирование пропорционально уровням риска и приоритизируя тесты на основе их уровней риска.

Руководители тестирования должны понимать, какие из этих выгод и затрат применимы для их организации, проекта и/или операций, а также уметь донести до заинтересованных сторон добавленную ценность тестирования в терминах выгод и стоимости качества на один дефект.

4 Ссылки

Стандарты

- **IEC 61508** (2010) Functional safety of electrical/electronic/programmable electronic safety-related systems - Parts 1 to 7
- **ISO/IEC/IEEE 29119-2** (2021) ISO/IEC/IEEE 29119-2 Software and systems Engineering-Software testing-Part 2 Test processes
- **ISO/IEC/IEEE 29119-3** (2021) ISO/IEC/IEEE 29119-3 Software and systems Engineering-Software testing-Part 3 Test documentation

Документы ISTQB®

ISTQB® Certified Tester Agile Test Leadership at Scale Syllabus v2.0 (2023)

ISTQB® Certified Tester Foundation Level Syllabus v4.0 (2023) (на русском языке - Программа подготовки ISTQB® Базового уровня v4.0 (2023))

ISTQB® Certified Tester Expert Level Test Management Syllabus v1.0 (2011)

ISTQB® Certified Tester Expert Level Improving the Test Process Syllabus v1.0.1 (2011)

Книги

- Basili, V., Trendowicz, A., Kowalczyk, M., Heidrich, J., Seaman, C., Münch, J., & Rombach, D. (2014). *Aligning Organizations Through Measurement – The GQM+ Strategies Approach.* Springer International.
- Bath, G., & van Veenendaal, E. (2014). *Improving the Test Process - chapter 6: Process for Improvement.* Rocky Nook.
- Belbin, R. M. (2010). *Management Teams: Why They Succeed or Fail.* London: Routledge.
- Black, R. (2009). *Managing the Testing Process, 3rd Edition.* John Wiley & Sons.
- Boehm, B. (1979). *Software Engineering Economics.* Prentice-Hall.
- Bonebright, D. A. (2010). *40 years of storming: a historical review of Tuckman's model of small group development* (1 Ausg., Bd. 13). Human Resource Development International, 1, 2010, Vol. 13.
- Craig, R., & Jaskiel, S. P. (2002). *Systematic Software Testing.* Artech House.
- Derby, E., & Larsen, D. (2006). *Agile Retrospectives – Making Good Teams Great,.* The Pragmatic Bookshelf.
- Erpenbeck, J., & von Rosenstiel, L. (2017). *Handbuch Kompetenzmessung.* Stuttgart: Schäffer-Poeschel.
- Fowler, M. (2010). *Hybrid development processes.* IEEE Software, 27(2), 57-63.
- Herzberg, F., Mausner, B., & Bloch Snyderman, B. (1993). *Motivation to Work.* London: Routledge.

- Kahler, T. (2008). *The Process Therapy Model: The Six Personality Types with Adaptations.* Taibi Kahler Associates, Inc.
- Marston, W. M. (1999). *Emotions Of Normal People.* London: Routledge.
- Sonntag, K., & Schmidt-Rathjens, C. (2005). *Anforderungsanalyse und Kompetenzmodelle.* Wiesbaden: VS Verlag für Sozialwissenschaften.
- Tuckman, B. W. (1965). *Developmental sequence in small groups* (Bd. 63(6)). Psychological Bulletin.
- van Ewijk, A. (2013). *TPI NEXT – Business Driven Test Process Improvement,.* Sogeti Nederland B.V.
- van Solingen, R., & Berghout, E. (1999). *The Goal Question Metric Method – A Practical Guide for Quality Improvement of Software Development.* McGraw-Hill.
- van Veenendaal, E. (2012). *The PRISMA Approach: Practical Risk-Based Testing.* UTN Publishers.
- van Veenendaal, E. (2020). *TMMi in the Agile world, version 1.4.* TMMi Foundation.
- van Veenendaal, E., & Cannegieter, J. J. (2011). *The Little TMMi – Objective-Driven Test Process Improvement.* UTN Publishers.

Статьи

- Feigenbaum, Armand V. (Nov/Dec 1956) Harvard Business Review, Vol. 34 Issue 6, p93-101
- Hadjicostas, Evsevios (2004) Total Quality Management and Cost of Quality, Springer https://link.springer.com/chapter/10.1007/978-3-662-09621-5_7

Веб-ресурсы

- www.tmmi.org Test Maturity Model integration (TMMi®); последняя проверка 31 января 2024
- www.tmap.net Test Process Improvement (TPI); последняя проверка 31 января 2024
- www.wikipedia.org/wiki/PDCA Plan-Do-Check-Act; последняя проверка 31 января 2024 (на русском языке - https://ru.wikipedia.org/wiki/Цикл_Деминга Цикл Деминга; последняя проверка 07 июля 2024)

Ссылки выше указывают на информацию, доступную в Интернете и других источниках. Несмотря на то, что эти ссылки были проверены на момент публикации данной программы обучения, ISTQB® не может нести ответственность, если ссылки больше не доступны.

5 Приложение A – Цели обучения / Уровни знаний

Конкретные цели обучения (LO), применимые к этой программе обучения, указаны в начале каждой главы. Каждая тема в программе обучения будет проверяться в соответствии с целями обучения, определенными для нее. Формулировки целей обучения начинаются с глагола, соответствующего уровню знаний, как перечислено ниже.

Уровень 1: запомнить (K1)

Кандидат запоминает, различает и использует термин или понятие.

Глаголы: запомнить, узнать.

Примеры
Запомнить концепцию пирамиды тестирования.
Узнать типичные цели тестирования.

Уровень 2: понять (K2)

Кандидат указывает причины или поясняет понятия, относящиеся к теме, а также резюмирует, сравнивает, классифицирует, разделяет по категориям и приводит примеры понятий, используемых в тестировании.

Глаголы: классифицировать, сравнить, различить, сопоставить, объяснить, приводить примеры, интерпретировать, обобщить.

Примеры	**Примечания**
Классифицировать инструменты тестирования в соответствии с их назначением и поддерживаемыми ими активностями тестирования.	
Сравнить различные уровни тестирования.	Может использоваться для поиска сходств, различий или того и другого.
Различать тестирование и отладку.	Позволяет различать понятия.
Сопоставлять риски проекта и риски продукта.	Позволяет раздельно классифицировать два (или более) понятия.
Объяснить влияние контекста на процесс тестирования.	
Привести примеры того, почему необходимо тестирование.	
Интерпретировать основную причину дефектов из заданного профиля отказов.	
Обобщить мероприятия процесса рецензирования.	

Уровень 3: применить (K3)

Кандидат может выполнить процедуру, встретившись со знакомой задачей, или выбрать правильную процедуру и использовать ее в заданном контексте.

Глаголы: применить, реализовать, подготовить, использовать.

Примеры	**Примечания**
Применить анализ граничных значения для получения сценариев тестирования из заданных требований.	Должно относиться к процедуре, методу, процессу и т.д.
Реализовать методы сбора метрик для поддержки технических и управленческих требований.	
Подготовить тесты установки мобильных приложений.	
Использовать трассируемость для мониторинга хода тестирования на предмет полноты и соответствия целям, стратегии и плану тестирования.	Может использоваться в LO, которая нацелена на то, чтобы кандидат мог использовать методику или процедуру. Похоже на «Применить».

Уровень 4: проанализировать (K4)

Кандидат может разделить информацию, связанную с процедурой или методикой, на составные части для лучшего понимания и отличить факты от умозаключений. Типичным применением является анализ документа, программного обеспечения или ситуации в проекте и предложение соответствующих действий для решения проблемы или выполнения задачи.

Глаголы: проанализировать, разобрать, выделить, расставить приоритеты, выбрать.

Примеры	**Примечания**
Проанализировать конкретную ситуацию в проекте, чтобы определить, какие методы тестирования «черного ящика» или основанные на опыте следует применить для достижения конкретных целей.	Поддается проверке только в сочетании с измеримой целью анализа. Должна иметь вид «Анализ от xxxx до xxxx» (или аналогичный).
Расставить приоритеты сценариев тестирования в данном наборе тестов для их выполнения с учетом рисков продукта.	
Выбрать соответствующие уровни и типы тестов для проверки определенного набора требований.	Нужно там, где выбор требует анализа.

Ссылки на литературу, описывающую цели обучения и их уровни

Anderson, L. W. and Krathwohl, D. R. (eds) (2001) A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Allyn & Bacon

Конкретные цели обучения, применимые к этой программе, показаны в начале каждой главы.

6 Приложение B – Матрица, показывающая связь между бизнес-результатами и целями обучения

В этом приложении для программы обучения Продвинутого уровня в управлении тестированием перечислены цели обучения, связанные с бизнес-результатами, а также представлена матрица, показывающая связь между бизнес-результатами и целями обучения.

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			**TM_01**	**TM_02**	**TM_03**	**TM_04**	**TM_05**	**TM_06**	**TM_07**	**TM_08**	**TM_09**	**TM_10**	**TM_11**
TM_01	Руководить тестированием в различных проектах разработки ПО, применяя процессы управления тестированием, определенные для проектной группы или организации, выполняющей тестирование		12										
TM_02	Определять заинтересованные стороны процесса тестирования и модели жизненного цикла разработки ПО актуальные в заданном контексте			4									
TM_03	Организовывать встречи по выявлению и оценке рисков в рамках любого жизненного цикла разработки ПО и использовать их результаты для достижения целей тестирования				6								
TM_04	Определять стратегию тестирования проекта, соответствующую стратегии тестирования, принятой в организации, и контексту проекта					11							

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			**TM_01**	**TM_02**	**TM_03**	**TM_04**	**TM_05**	**TM_06**	**TM_07**	**TM_08**	**TM_09**	**TM_10**	**TM_11**
TM_05	Постоянно отслеживать и контролировать тестирование для достижения целей проекта						4						
TM_06	Оценивать и сообщать о ходе тестирования заинтересованным сторонам проекта							3					
TM_07	Определять необходимые навыки специалистов и развивать их в своей команде								6				
TM_08	Подготавливать и представлять экономическое обоснование для тестирования в различных контекстах, описывающее затраты и ожидаемые выгоды									5			
TM_09	Руководить деятельностью по улучшению процессов тестирования в проектах или программах разработки продуктов ПО и вносить вклад в инициативы по улучшению процессов тестирования в организации										5		
TM_10	Планировать действия по тестированию, включая необходимую тестовую инфраструктуру, и оценивать необходимые для тестирования усилия											9	
TM_11	Создавать отчеты о дефектах и жизненный цикл дефектов, подходящие для жизненного цикла разработки ПО												6

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			**TM_01**	**TM_02**	**TM_03**	**TM_04**	**TM_05**	**TM_06**	**TM_07**	**TM_08**	**TM_09**	**TM_10**	**TM_11**
Идентификатор ЦО	Цель обучения	Уровень знаний											
1	Управление активностями тестирования												
1.1	Процесс тестирования												
TM-1.1.1	Обобщить планирование тестирования	K2	X			X							
TM-1.1.2	Обобщить мониторинг и контроль тестирования	K2	X				X						
TM-1.1.3	Обобщить завершение тестирования	K2	X					X					
1.2	Контекст тестирования												
TM-1.2.1	Сравнить заинтересованность различных заинтересованных сторон в тестировании	K2		X		X							
TM-1.2.2	Объяснить, почему знания заинтересованных сторон важны для управления тестированием	K2		X		X							
TM-1.2.3	Объяснить тестирование в гибридной модели разработки программного обеспечения	K2		X		X							
TM-1.2.4	Обобщить активности по управлению тестированием для различных	K2	X	X		X							

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			TM_01	TM_02	TM_03	TM_04	TM_05	TM_06	TM_07	TM_08	TM_09	TM_10	TM_11
	жизненных циклов разработки программного обеспечения												
TM-1.2.5	Сравнить активности по управлению тестированием на различных уровнях тестирования	K2	X			X							
TM-1.2.6	Сравнить активности по управлению тестированием для различных типов тестирования	K2	X			X							
TM-1.2.7	Проанализировать конкретный проект и определить активности по управлению тестированием, в которых особое внимание уделяется планированию, мониторингу и контролю тестирования	K4	X			X							
1.3	Тестирование на основе рисков												
TM-1.3.1	Объяснить различные измерения, которые принимает тестирование на основе рисков для реагирования на риски	K2			X								
TM-1.3.2	Привести примеры различных методов, которые руководитель тестирования может использовать для выявления рисков, связанных с качеством продукта	K2			X								
TM-1.3.3	Обобщить факторы, определяющие уровни риска, связанные с качеством продукта	K2			X								
TM-1.3.4	Выбрать соответствующие активности тестирования для смягчения рисков в	K4			X								

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			TM_01	TM_02	TM_03	TM_04	TM_05	TM_06	TM_07	TM_08	TM_09	TM_10	TM_11
	соответствии с их уровнем риска в конкретном контексте												
TM-1.3.5	Различить тяжелые и легкие примеры методов тестирования на основе рисков	K2			X								
TM-1.3.6	Привести примеры метрик успеха и трудностей, связанных с тестированием на основе рисков	K2			X								
1.4	Стратегия тестирования проекта												
TM-1.4.1	Объяснить типичные варианты подхода к тестированию	K2				X							
TM-1.4.2	Проанализировать корпоративную стратегию тестирования и контекст проекта, чтобы выбрать подходящий подход к тестированию	K4				X							
TM-1.4.3	Использовать методологию достижения целей S.M.A.R.T. для определения измеримых целей тестирования и критериев выхода	K3				X							
1.5	Совершенствование процесса тестирования												
TM-1.5.1	Объяснить, как использовать модель IDEAL для совершенствования процесса тестирования в конкретном проекте	K2									X		
TM-1.5.2	Обобщить подход к совершенствованию процесса тестирования на основе моделей и понять, как его применять в контексте проекта	K2									X		

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			TM_01	TM_02	TM_03	TM_04	TM_05	TM_06	TM_07	TM_08	TM_09	TM_10	TM_11
TM-1.5.3	Обобщить подход на основе аналитики к совершенствованию процесса тестирования и понять, как его применять в контексте проекта	K2									X		
TM-1.5.4	Реализовать ретроспективу проекта или итерации, чтобы оценить процессы тестирования и выявить области тестирования, требующие совершенствования	K3									X		
1.6	Инструменты тестирования												
TM-1.6.1	Обобщить лучшие практики внедрения инструментов	K2										X	
TM-1.6.2	Объяснить влияние различных технических и бизнес-аспектов при выборе типа инструмента	K2										X	
TM-1.6.3	Проанализировать конкретную ситуацию, чтобы составить план выбора инструмента, охватывающий риски, затраты и выгоды	K4										X	
TM-1.6.4	Различить этапы жизненного цикла инструмента	K2										X	
TM-1.6.5	Привести примеры сбора и оценки метрик с помощью инструментов	K2									X	X	
2	Управление продуктом												
2.1	Метрики тестирования												

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			TM_01	TM_02	TM_03	TM_04	TM_05	TM_06	TM_07	TM_08	TM_09	TM_10	TM_11
TM-2.1.1	Привести примеры метрик для достижения целей тестирования	K2					X						
TM-2.1.2	Объяснить, как контролировать прогресс тестирования с использованием метрик тестирования	K2					X						
TM-2.1.3	Проанализировать результаты тестирования для создания отчетов о тестировании, которые дадут возможность заинтересованным сторонам принять решения	K4					X	X					
2.2	Оценка затрат на тестирование												
TM-2.2.1	Объяснить факторы, которые нужно учитывать при оценке затрат на тестирование	K2	X							X		X	
TM-2.2.2	Привести примеры факторов, которые могут повлиять на оценку затрат на тестирование	K2	X							X		X	
TM-2.2.3	Выбрать подходящую технику или подход для оценки затрат на тестирование в заданном контексте	K4	X							X		X	
2.3	Управление дефектами												
TM-2.3.1	Реализовать процесс управления дефектами, включая жизненный цикл дефекта, который можно использовать для отслеживания и контроля дефектов	K3											X

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			TM_01	TM_02	TM_03	TM_04	TM_05	TM_06	TM_07	TM_08	TM_09	TM_10	TM_11
TM-2.3.2	Объяснить процесс и определить участников, необходимых для эффективного управления дефектами	K2											X
TM-2.3.3	Объяснить особенности управления дефектами при гибкой модели разработки программного обеспечения	K2	X										X
TM-2.3.4	Объяснить возможные трудности управления дефектами при гибридной модели разработки программного обеспечения	K2	X										X
TM-2.3.5	Использовать данные и классификационную информацию, которые должны быть собраны во время управления дефектами	K3											X
TM-2.3.6	Объяснить, как статистика отчетов о дефектах может использоваться для разработки процессов по улучшению	K2										X	X
3	Управление командой тестирования												
3.1	Команда тестирования												
TM-3.1.1	Привести примеры типичных навыков в четырех областях компетенции, которыми необходимо обладать членам команды тестирования	K2							X				
TM-3.1.2	Проанализировать описанные условия реализации проекта, чтобы определить необходимые навыки членов команды тестирования	K4							X				

Бизнес-результаты: Программа обучения Продвинутого уровня в управлении тестированием			TM_01	TM_02	TM_03	TM_04	TM_05	TM_06	TM_07	TM_08	TM_09	TM_10	TM_11
TM-3.1.3	Объяснить типичные методы оценки навыков членов команды тестирования	K2							X				
TM-3.1.4	Объяснить разницу между типичными подходами к развитию навыков членов команды тестирования	K2							X				
TM-3.1.5	Объяснить навыки, необходимые для управления командой тестирования	K2							X				
TM-3.1.6	Привести примеры факторов мотивации и гигиенических факторов для членов команды тестирования	K2							X				
3.2	Взаимоотношения с заинтересованными сторонами												
TM-3.2.1	Привести примеры для каждой из четырех категорий, определяющих стоимость качества	K2								X			
TM-3.2.2	Применить расчет соотношения затрат и выгод для оценки добавленной стоимости тестирования для заинтересованных сторон	K3						X		X			

7 Приложение C – Описание изменений

Программа обучения Продвинутого уровня ISTQB® в управлении тестированием версии 3.0 — это крупное обновление программы обучения Продвинутого уровня Руководитель тестирования 2012. Именно по этой причине здесь нет подробного описания изменений по каждой главе и разделу. Ниже приводится краткое содержание основных изменений.

Цели обучения в этой версии были изменены таким образом, чтобы сделать их атомарными и создать прослеживаемость между целями обучения и разделами программы обучения, в которой не осталось содержания без цели обучения. Цель изменений в том, чтобы сделать эту версию более легкой для чтения, понимания, изучения и перевода, уделяя особое внимание повышению практической полезности и балансу между знаниями и навыками.

В этом выпуске были реализованы следующие крупные изменения:

- Уменьшен общий размер учебной программы. Программа обучения — это не учебник, а документ, в котором излагаются основные элементы курса Продвинутого уровня по тестированию ПО, включая то, какие темы следует охватить и на каком уровне. Поэтому, в частности:
  - В большинстве случаев примеры исключены из текста. Предоставить примеры и упражнения во время обучения – задача обучающей организации.
  - Соблюдался «Чек-лист написания программы обучения», который предлагает максимальный размер текста для цели обучения на каждом K-уровне (K2 = 1500 символов, исключая пробелы, K3 = 2500 символов, исключая пробелы, K4 = 3000 символов, исключая пробелы, +/- 20%).
- Сокращено количество целей обучения по сравнению с программой CTAL TM 2012.
  - 36 целей обучения K2 по сравнению с 39 в CTAL TM 2012.
  - 5 целей обучения K3 по сравнению 12 в CTAL TM 2012.
  - 7 целей обучения K4 по сравнению 10 в CTAL TM 2012.
- Пересмотрена структура программы обучения в целом.
- Полная согласованность с программой обучения Базового уровня ISTQB® V.4.
- Основные изменения в бывшей Главе 1 (Процесс тестирования) версии CTAL TM 2012.
  - Глава ограничена до управления активностями тестирования (планирование тестирования, мониторинг тестирования, контроль тестирования и завершение тестирования).
  - Глава интегрирована как раздел в новую главу «Управление активностями тестирования».
- Новая глава **Управление активностями тестирования**
  - Раздел 1.1 – Процесс тестирования: см. выше.
  - Раздел 1.2 – Контекст тестирования: расширен, чтобы охватить непоследовательные модели разработки программного обеспечения.

- Раздел 1.3 – Тестирование, основанное на рисках: полностью переписан, чтобы сделать его более применимым на уровне проекта.
- Раздел 1.4 – Стратегия тестирования проекта: поскольку план тестирования уже определен в программе ISTQB® Базового уровня V.4, основное внимание уделяется выбору соответствующего подхода к тестированию и тому, как определить измеримые цели тестирования.
- Раздел 1.5 – Улучшение процесса тестирования: интегрирован в Управление активностями тестирования с целью продемонстрировать, как применять в контексте проекта и внедрять в рамках итерации или проекта с использованием ретроспективы.
- Раздел 1.6 – Инструменты тестирования: знакомство с инструментами было перенесено из программы ISTQB® Базового уровня V.3.1 (в программе ISTQB® Базового уровня V.4 отсутствует).

- Новая глава **Управление продуктом**
  - Раздел 2.1 – Метрики тестирования: бывшие разделы программы, описывающие метрики и их использование.
  - Раздел 2.2 – Оценка тестирования: программа ISTQB® Базового уровня V.4 уже охватывает расчет оценки тестирования. Расширен для выбора и применения подходящих методов оценки тестирования в моделях ЖЦПО на уровне K4.
  - Раздел 2.3 — Управление дефектами: приведен в соответствие с последними редакциями стандартов и расширен для использования в гибкой и гибридной разработке программного обеспечения.
- Новая глава **Управление командой**
  - Раздел 3.1 – Команда тестирования: основные темы такие же, как в программе TM 2012. Определение индивидуальных навыков и организация команд тестирования.
  - Раздел 3.2 – Взаимоотношения с заинтересованными сторонами: это бывший раздел программы TM 2012 «2.7 Бизнес-ценность тестирования».
- Основные изменения и удаленные разделы/главы по сравнению с CTAL TM Syllabus 2012
  - Удален раздел «Распределенное, аутсорсинговое и внутреннее тестирование».
  - Удален раздел «Управление на основе промышленных стандартов».
  - Удалена глава «Рецензирование».
  - Удалены разделы «Улучшение процесса тестирования с CTP и STEP».
  - Удалены разделы «Анализ тестирования», «Проектирование тестов», «Реализация тестов» и «Выполнение тестов».

8 Приложение D – Термины, специфичные для управления тестированием

Русскоязычный термин	**Англоязычный термин**	**Определение**
IDEAL	IDEAL	Организационная модель совершенствования, которая служит дорожной картой для инициирования, планирования и реализации активностей по совершенствованию.
Измерение	measure	Процесс присвоения числа или категории сущности для описания атрибута этой сущности.
Индикатор	indicator	Измерение, дающее оценку выбранных атрибутов, полученных из некоторой модели с учетом определенных информационных потребностей.
Метрика	metric	Шкала измерений и метод, используемый для измерений.
Оценка по трем точкам	Three-point estimation	Техника на основе экспертной оценки времени или усилий, при которой эксперты рассматривают три сценария: наиболее оптимистичная оценка (о), наиболее вероятная оценка (в) и наиболее пессимистичная оценка (п). Итоговая оценка (О) является их среднеарифметическим значением.
Покер планирования	planning poker	Метод оценки трудозатрат, основанный на всеобщем одобрении. Обычно используется для оценки затрат или относительного объема пользовательских историй в гибких методологиях разработки программного обеспечения. Является вариантом широкополосного предсказателя с использованием колоды карт со значениями, представляющими число квантов, которыми оценивается работа.
Цель - Вопрос - Метрика	goal question metric (GQM)	Подход к измерению программного обеспечения с использованием трехуровневой модели: концептуальный уровень (цель), оперативный уровень (вопрос), количественный уровень (метрика).
Широкополосный метод Дельфи	Wideband Delphi	Методика оценки затрат на тестирование на базе экспертной оценки, ставящая целью точную оценку с помощью коллективного опыта членов команды.

9 Приложение E – Торговые марки

CMMI® является зарегистрированной торговой маркой в Ведомстве по патентам и товарным знакам США Университета Карнеги-Меллон.

ISTQB® является зарегистрированной торговой маркой International Software Testing Qualifications Board.

TMMi® является зарегистрированной торговой маркой TMMi Foundation.

TPI-Next® является зарегистрированной торговой маркой Sogeti, Нидерланды.

10 Приложение F – Предметный указатель