ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ГОСТ Р 56920— 2016/ ISO/IEC/IEEE 29119-1:2013**

# СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ

## Тестирование программного обеспечения

### Часть 1

## Понятия и определения

**ISO/IEC/IEEE 29119-1:2013**
**Software and systems engineering — Software testing —**
**Part 1: Concepts and definitions**
**(IDT)**

Издание официальное

Москва
Стандартинформ
2016

gost.e-ecolog.ru

## Предисловие

1 ПОДГОТОВЛЕН Обществом с ограниченной ответственностью «Информационно-аналитический вычислительный центр» (ООО ИАВЦ) на основе собственного аутентичного перевода на русский язык международного стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 22 «Информационные технологии»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 18 мая 2016 г. № 331-ст

4 Настоящий стандарт идентичен международному стандарту ISO/IEC/IEEE 29119-1:2013 «Программная и системная инженерия. Тестирование программного обеспечения. Часть 1. Понятия и определения» (ISO/IEC/IEEE 29119-1:2013 «Software and systems engineering — Software testing — Part 1: Concepts and definitions»).

Наименование настоящего стандарта изменено относительно наименования указанного международного стандарта для приведения в соответствие с ГОСТ Р 1.5 (пункт 3.5)

5 ВВЕДЕН ВПЕРВЫЕ

*Правила применения настоящего стандарта установлены в ГОСТ Р 1.0—2012 (раздел 8). Информация об изменениях к настоящему стандарту публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок — в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (*gost.e-ecolog.ru

© Стандартинформ, 2016

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

## Содержание

1 Область применения . . . 1
2 Соответствие . . . 1
3 Нормативные ссылки . . . 1
4 Термины и определения . . . 1
5 Понятия тестирования программного обеспечения . . . 9
5.1 Введение в тестирование программного обеспечения . . . 9
5.2 Тестирование программного обеспечения в организационном контексте и контексте проекта . . . 11
5.3 Общие процессы тестирования в жизненном цикле программного обеспечения . . . 16
5.4 Тестирование на базе рисков . . . 20
5.5 Подпроцесс тестирования . . . 21
5.6 Методики тестирования . . . 26
5.7 Автоматизация в тестировании . . . 29
5.8 Управление дефектами . . . 29
Приложение А (справочное) Роль тестирования в верификации и валидации . . . 30
Приложение В (справочное) Метрики и показатели . . . 31
Приложение С (справочное) Тестирование в различных моделях жизненного цикла . . . 32
Приложение D (справочное) Примеры подпроцессов тестирования в деталях . . . 38
Приложение Е (справочное) Роли и обязанности в тестировании . . . 46
Библиография . . . 48

## Введение

Международная организация по стандартизации (ИСО) и Международная электротехническая комиссия (МЭК) образуют специализированную систему для всемирной стандартизации. Национальные органы по стандартизации, которые являются членами ИСО или МЭК, участвуют в разработке международных стандартов через технические комитеты, созданные соответствующей организацией для определенных областей технической деятельности. Технические комитеты ИСО и МЭК сотрудничают в сферах, представляющих взаимный интерес. Другие международные правительственные и неправительственные организации, связанные с ИСО и МЭК, также принимают участие в работе по разработке стандартов. В сфере информационной технологии ИСО и МЭК учредили совместный технический комитет ИСО/МЭК СТК 1.

Документы стандартов Института Инженеров по Электротехнике и Радиоэлектронике (ИИЭР) разработаны в сообществах и комитетах по координации стандартов ИИЭР, входящих в состав бюро стандартов ассоциации стандартов ИИЭР. ИИЭР разрабатывает свои стандарты на основе одобренного Американским национальным институтом стандартов консенсусного процесса разработки, который для достижения конечного результата объединяет различные точки зрения и интересы добровольцев. Добровольцы не обязательно являются сотрудниками института и работают на безвозмездной основе. При том, что ИИЭР курирует процесс и устанавливает правила обеспечения справедливости в консенсусном процессе разработки, ИИЭР не проводит независимые оценку, испытание или проверку точности какой-либо информации, входящей в состав своих стандартов.

Международные стандарты разрабатывают в соответствии с правилами, приведенными в Директивах ИСО/МЭК, часть 2.

Основная задача совместного технического комитета состоит в подготовке международных стандартов. Проекты международных стандартов, принятые совместным техническим комитетом, распространяются среди национальных органов по стандартизации для вынесения решения. Для публикации в качестве международного стандарта требуется одобрение по крайней мере 75 % национальных органов по стандартизации, участвующих в голосовании.

Следует обратить внимание на то, что при внедрении настоящего стандарта может потребоваться использование предмета, защищенного патентными правами. При публикации настоящего стандарта не рассматривались вопросы существования или соответствия каких-либо связанных со стандартом патентных прав. ИСО/ИИЭР не несет ответственность ни за идентификацию существенных для стандарта патентов или патентных заявок, для которых может требоваться лицензия, ни за расследование юридической правильности или области применения патентов, или патентных заявок, ни за определение каких-либо условий лицензирования или условий предоставления гарантийных писем, или патентных заявлений и форм лицензирования, если таковые имеются, ни за какие-либо приемлемые недискриминационные лицензионные соглашения. Пользователи настоящего стандарта должны учитывать то, что определение действия каких-либо патентных прав и риска нарушения таких прав полностью лежит на их собственной ответственности. Дополнительная информация может быть получена в ИСО или ассоциации стандартов ИИЭР.

ИСО/МЭК/ИИЭР 29119-1 был подготовлен Подкомитетом 7 «*Системная и программная инженерия*» совместного технического комитета ИСО/МЭК СТК 1 «*Информационные технологии*» в сотрудничестве с комитетом по стандартам системной и программной инженерии компьютерного сообщества ИИЭР в соответствии с организационным соглашением о партнерском сотрудничестве по разработке стандартов между ИСО и ИИЭР.

Серия стандартов ИСО/МЭК/ИИЭР 29119 под общим названием «Системная и программная инженерия. Тестирование программного обеспечения» состоит из следующих частей:

- Часть 1. Понятия и определения;
- Часть 2. Процессы тестирования;
- Часть 3. Документация тестирования;
- Часть 4. Методики тестирования.

Цель создания серии стандартов ИСО/МЭК/ИИЭР 29119 «Тестирование программного обеспечения» состоит в том, чтобы определить на международном уровне согласованную совокупность стандартов, которая может использоваться любой организацией при выполнении различных форм тестирования программного обеспечения.

Существование множества различных типов программного обеспечения, организаций программного обеспечения и методологий общеизвестно. Предметные области программного обеспечения включают в себя информационные технологии (ИТ), персональные (ПК), встроенные, мобильные,

научные компьютеры и многие другие категории. Разнообразие организаций программного обеспечения простирается от малого до большого размера, от локальных до глобальных, от коммерческих до сервис-ориентированных. Методология программного обеспечения включает различные подходы: объектно-ориентированный, традиционный, управляемый данными и динамичный. Все эти и другие факторы оказывают влияние на тестирование программного обеспечения. Настоящая серия стандартов предназначена для поддержки тестирования в разных контекстах.

Настоящий стандарт предоставляет словарь терминов, используемых в серии стандартов ИСО/МЭК/ИИЭР 29119, который упрощает применение других стандартов этой серии, и приводит примеры применения их на практике. Настоящий стандарт предоставляет понятия тестирования программного обеспечения и способы применения тестирования программного обеспечения и является руководством для других частей ИСО/МЭК/ИИЭР 29119.

В настоящем стандарте представлены общие понятия тестирования программного обеспечения. Описывается роль тестирования программного обеспечения в организационном контексте и контексте проекта. Тестирование программного обеспечения рассматривается в контексте общего жизненного цикла программного обеспечения. Представлен способ, который позволяет устанавливать процессы и подпроцессы тестирования программного обеспечения для определенных элементов тестирования или с определенными целями. Рассматривается, как тестирование программного обеспечения вписывается в различные модели жизненного цикла. Демонстрируется использование различных методов планирования тестирования, а также то, как может быть использована автоматизация для поддержки тестирования. Обсуждается роль тестирования в управлении дефектами. Приложение А описывает роль тестирования в более широкой предметной области верификации и валидации. Приложение В представляет краткое введение в метрики, используемые для мониторинга и управления тестированием. Приложение С содержит ряд примеров, показывающих, как применить настоящий стандарт в различных моделях жизненного цикла. Приложение D предоставляет примеры подпроцессов тестирования в деталях. Приложение Е предоставляет дополнительную информацию о ролях и обязанностях, с которыми обычно имеют дело группы тестирования и независимые тестеры. В конце стандарта представлен элемент «Библиография».

Следует обратить внимание на то, что заглавные буквы используются в настоящем стандарте в названиях процессов и документов, которые определены в ИСО/МЭК/ИИЭР 29119-2 и ИСО/МЭК/ИИЭР 29119-3 (например, Процесс Планирования Тестирования, План Тестирования), тогда как строчные буквы используются для документов, являющихся частями других документов (например, стратегия тестирования проекта — элемент Плана Тестирования Проекта).

Модель процесса тестирования, на которой основывается серия стандартов ИСО/МЭК/ИИЭР 29119 «Тестирование программного обеспечения», подробно описана в ИСО/МЭК/ИИЭР 29119-2 «Процессы тестирования». ИСО/МЭК/ИИЭР 29119-2 рассматривает процессы тестирования программного обеспечения на организационном уровне, уровне управления тестированием и уровнях динамического тестирования. Тестирование — это основной подход к обработке рисков в разработке программного обеспечения. Этот стандарт определяет подход к тестированию, базирующийся на рисках. Тестирование на базе рисков — это рекомендуемый подход к разработке стратегии и менеджмента тестирования, который позволяет расставлять приоритеты и акценты в тестировании.

ИСО/МЭК/ИИЭР 29119-3 «Документация тестирования» определяет шаблоны и примеры документации тестирования. ИСО/МЭК/ИИЭР 29119-4 «Методики тестирования» определяет методы тестирования программного обеспечения, которые могут быть использованы в ходе тестирования.

В целом серия стандартов ИСО/МЭК/ИИЭР 29119 дает возможность заинтересованным сторонам контролировать и выполнять тестирование программного обеспечения в любой организации.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

# СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ

## Тестирование программного обеспечения

## Часть 1

## Понятия и определения

Software and systems engineering. Software testing. Part 1. Concepts and definitions

**Дата введения — 2017—06—01**

## 1 Область применения

В настоящем стандарте представлены определения и понятия тестирования программного обеспечения. Это представление обеспечивает идентификацию терминов и ключевых концепций тестирования, необходимых для правильного толкования серии стандартов ИСО/МЭК/ИИЭР 29119.

## 2 Соответствие

Настоящий стандарт носит информативный характер и не требует какого-либо соответствия.

Серия стандартов ИСО/МЭК/ИИЭР 29119 «Тестирование программного обеспечения» содержит три стандарта, которые могут потребовать соответствия:

- Процессы тестирования;
- Документация тестирования;
- Методики тестирования.

Соответствие рассмотрено в ИСО/МЭК/ИИЭР 29119-2, ИСО/МЭК/ИИЭР 29119-3 и ИСО/МЭК/ИИЭР 29119-4.

## 3 Нормативные ссылки

Настоящий стандарт не требует каких-либо нормативных ссылок. Стандарты и документы, полезные для применения и интерпретации настоящего стандарта, перечислены в элементе «Библиография».

## 4 Термины и определения

В настоящем стандарте используются термины и определения, приведенные в ИСО/МЭК/ИИЭР 24765, а также перечисленные ниже термины с соответствующими определениями.

Примечание — Нижеследующие термины и определения представлены для понимания и удобства восприятия частей 1, 2, 3 и 4 серии стандартов ИСО/МЭК/ИИЭР 29119 «Тестирование программного обеспечения». Некоторые термины, определенные в настоящем стандарте, не используются непосредственно в нем, а применяются только в других частях серии ИСО/МЭК/ИИЭР 29119. В этом разделе представлены только термины, критически важные для понимания этих стандартов. Представление полного списка терминов тестирования не является целью данного раздела. Для терминов, не определенных в этом разделе, следует пользоваться словарем системной и программной инженерии ИСО/МЭК/ИИЭР 24765. Он доступен на веб-сайте: gost.e-ecolog.ru

**Издание официальное**

4.1 **тестирование доступности** (accessibility testing): Тип тестирования удобства использования, предназначенный для оценки степени возможности управления элементом тестирования пользователями с самыми разными характеристиками и способностями.

4.2 **фактические результаты** (actual results): Совокупность поведения или условий элемента тестирования, или совокупность условий, связанных данных или тестовой среды, полученные в результате выполнения теста.

***Пример — Вывод на аппаратные средства, изменения в данных, отчеты и отправленные информационные сообщения.***

4.3 **тестирование копирования и восстановления** (backup and recovery testing): Тип тестирования надежности, который измеряет степень состояния системы, до которой в случае отказа может быть произведено восстановление из резервной копии при указанных параметрах времени, стоимости, полноты и точности.

4.4 **тестирование методом «черного ящика»** (black-box testing): См. термин «тестирование на основе спецификации» согласно 4.39.

4.5 **тестирование потенциальных возможностей** (capacity testing): Тип тестирования уровня производительности для оценки уровня, при котором с увеличением нагрузки (числа пользователей, транзакций, элементов данных и т.д.) элемент тестирования подвергается угрозе не обеспечивать требуемую производительность.

4.6 **тестирование совместимости** (compatibility testing): Тип тестирования, который измеряет степень того, насколько удовлетворительно элемент тестирования может функционировать параллельно с другими независимыми продуктами в общей среде (сосуществование) и, по мере необходимости, обменивается информацией с другими системами или компонентами (функциональная совместимость).

4.7 **элемент покрытия** (coverage item): См. термин «элемент тестового покрытия» согласно 4.54.

4.8 **решение** (decision): Тип оператора выбора одного из двух или более возможных результатов для определения выбора конкретного набора действий.

4.9 **динамическое тестирование** (dynamic testing): Тестирование, при котором требуется выполнение элемента тестирования.

4.10 **тестирование износостойкости** (endurance testing): Тип тестирования уровня производительности для определения того, может ли элемент тестирования постоянно выдерживать требуемую загрузку в течение установленного периода времени.

4.11 **раздел эквивалентности** (equivalence partition): Подмножество области значений переменной или совокупности переменных внутри элемента тестирования или на его интерфейсах такое, что можно обоснованно ожидать того, что все значения подмножества будут обработаны элементом тестирования подобным образом (то есть они могут считаться «эквивалентными»).

4.12 **покрытие раздела эквивалентности** (equivalence partition coverage): Доля идентифицированных разделов эквивалентности элемента тестирования, которая покрывается набором тестов.

Примечание — В большинстве случаев идентификация разделов эквивалентности субъективна (особенно в разбиении «недопустимых» разделов); таким образом, окончательный подсчет числа разделов эквивалентности для элемента тестирования может быть невозможен.

4.13 **разбиение эквивалентности** (equivalence partitioning): Метод проектирования тестирования, при котором контрольные примеры разработаны таким образом, чтобы проверить разделы эквивалентности с помощью одного или более представительных элементов каждого раздела.

4.14 **предположение об ошибках** (error guessing): Метод проектирования тестирования, в котором контрольные примеры получены на основе знаний тестера о прошлых отказах или общих знаний о видах отказа.

Примечание — Соответствующие знания могут исходить из личного опыта или могут быть получены, например, из базы данных дефектов или «таксономии ошибок».

4.15 **ожидаемые результаты** (expected results): Характерное предсказанное поведение элемента тестирования при указанных условиях на основе его спецификации или другого источника.

4.16 **исследовательское тестирование** (exploratory testing): Тестирование, основанное на опыте, при котором тестер спонтанно разрабатывает и выполняет тестирования на основе существующих соответствующих знаний тестера, предшествующих исследований элемента тестирования (включая и результаты предыдущих тестирований) и эвристических «эмпирических правил» для общего поведения программного обеспечения и типов отказа.

Примечание — Исследовательское тестирование направлено на выявление скрытых свойств (включая и скрытое поведение), которые сами по себе, с одной стороны, вполне возможно, безобидны, но, с другой стороны, могут повлиять на другие свойства тестируемого программного обеспечения и тем увеличить риск того, что программное обеспечение перестанет работать.

4.17 **набор функций** (feature set): Совокупность, в которую входят тестовые условия проверяемого элемента и могут быть включены риски, требования, функции, модели и т. д.

Примечание — Это может быть набор всех функций элемента (полный набор его функций) или подмножество, определенное для конкретной цели (совокупность функциональных возможностей и т. д.).

4.18 **Отчет об Инциденте** (Incident Report): Документация по инциденту о его проявлении, природе и состоянии.

4.19 **тестирование устанавливаемости** (installability testing): Тип тестирования переносимости для оценки того, могут ли должным образом элемент тестирования или совокупность элементов тестирования быть установлены во всех указанных средах.

4.20 **нагрузочное тестирование** (load testing): Тип тестирования уровня производительности, проводимого для оценки поведения элемента тестирования при ожидаемых условиях переменной загрузки, обычно для ожидаемых условий низкого, типичного и пикового использования.

4.21 **тестирование сопровождаемости** (maintainability testing): Тип тестирования, проводимого для оценки степени эффективности и продуктивности возможных изменений элемента тестирования.

4.22 **Организационная Политика Тестирования** (Organizational Test Policy): Руководящий документ, в котором описаны назначение, цели, полная предметная область применения тестирования в организации и определено, почему выполняется тестирование и что ожидается получить в результате.

Примечание — В общем случае для конкретного контекста предпочтительно иметь Организационную Политику Тестирования максимально короткой. Это может быть набор всех функций элемента (полный набор его функций) или подмножество, определенное для конкретной цели (совокупность функциональных возможностей и т. д.).

4.23 **Организационный Процесс Тестирования** (Organizational Test Process): Процесс тестирования для разработки и управления организационными спецификациями тестирования.

4.24 **Организационная Спецификация Тестирования** (Organizational Test Specification): Документ, в котором представлена информация о тестировании для организации, то есть информация, которая не специфична для проекта.

***Пример — Наиболее общими примерами Организационной Спецификации Тестирования являются Организационная Политика Тестирования и Организационная Стратегия Тестирования.***

4.25 **Организационная Стратегия Тестирования** (Organizational Test Strategy): Документ, в котором изложены универсальные требования к тестированиям, которые будут выполняться для всех проектов организации, а также подробности того, как должно производиться тестирование.

Примечания

1 Организационная Стратегия Тестирования согласована с Организационной Политикой Тестирования.

2 Для покрытия существенно различных контекстов проектов у организации может быть более одной Организационной Стратегии Тестирования.

4.26 **критерий успешного/неуспешного прохождения** (pass/fail criteria): Правила решения, используемые для определения того, прошли ли тестирование элемент тестирования или функция элемента тестирования или перестали работать после тестирования.

4.27 **тестирование производительности** (performance testing): Тип тестирования, проводимого для оценки степени, в которой элемент тестирования выполняет свои определенные функции при заданных ограничениях времени и других ресурсах.

4.28 **тестирование переносимости** (portability testing): Тип тестирования, проводимого для оценки простоты переноса элемента тестирования из одних аппаратных средств или программной среды в другие, включая уровень его изменений, необходимых для выполнения в средах различных типов.

4.29 **тестирование процессов** (procedure testing): Тип тестирования функциональной пригодности, проводимый для определения того, отвечают ли требованиям пользователя и обеспечивают ли цель их применения процедурные инструкции по взаимодействию с элементом тестирования или по использованию его выходных данных.

4.30 **риск продукта** (product risk): Риск того, что продукт может иметь дефект в некотором определенном аспекте его функций, качества или структуры.

4.31 **риск проекта** (project risk): Риск, относящийся к менеджменту проекта.

***Пример — Отсутствие укомплектования персоналом, строгие крайние сроки, изменения требований.***

4.32 **регрессионное тестирование** (regression testing): Тестирование после изменений элемента тестирования или его рабочей среды для определения того, происходят ли регрессивные отказы.

Примечание — Достаточное количество регрессионных тестов зависит от тестируемого элемента и от изменений этого элемента или его рабочей среды.

4.33 **тестирование надежности** (reliability testing): Тип тестирования, проводимый для оценки возможности элемента тестирования выполнять свои требуемые функции, включая оценку частоты, с которой происходят отказы при использовании в установленных условиях в течение заданного периода времени.

4.34 **повторное тестирование** (retesting): Повторное выполнение контрольных примеров, для которых ранее был получен результат «сбоя» для оценки эффективности произведенных корректирующих действий.

Примечание — Используется также термин «тестирование подтверждения».

4.35 **тестирование на базе рисков** (risk-base testing): Тестирование, для которого менеджмент, выбор, расстановка приоритетов и использование действий и ресурсов тестирования преднамеренно основаны на базе проанализированных рисков соответствующих типов и уровней.

4.36 **тестирование на основе сценария** (scenario testing): Класс методик проектирования тестирования, при которых разрабатываются тестирования для выполнения конкретных сценариев.

Примечание — Сценарий может быть историей пользователя, примером использования, операционным понятием или последовательностью событий, с которыми программное обеспечение может встретиться и т. д.

4.37 **тестирование по сценарию** (scripted testing): Динамическое тестирование, в котором действия тестера предписаны записанными в контрольном примере инструкциями.

Примечание — Этот термин обычно применяется для тестирования, выполняемого вручную, а не для выполнения автоматизированного сценария.

4.38 **тестирование защищенности** (security testing): Тип тестирования, проводимый для оценки степени защищенности элемента тестирования и связанных с ним данных и информации от доступа посторонних лиц или систем для использования, чтения или изменения их при том, что доверенным лицам или системам доступ к ним обеспечивается.

4.39 **тестирование на основе спецификации** (specification-based testing): Тестирование, основным базисом которого являются внешние вводы и выводы элемента тестирования, обычно на основе спецификации, а не ее реализация в исходном коде или исполнимом программном обеспечении.

Примечание — Синонимами тестирования на основе являются тестирование методом «черного ящика» и тестирование закрытого ящика.

4.40 **покрытие операторов** (statement coverage): Процент совокупности всех исполнимых операторов элемента тестирования, которые покрываются набором тестов.

4.41 **тестирование операторов** (statement testing): Метод проектирования тестирования, при котором создаются контрольные примеры для выполнения отдельных операторов элемента тестирования.

4.42 **статическое тестирование** (static testing): Тестирование, при котором элемент тестирования анализируется с использованием совокупности критериев качества или других свойств без выполнения кода.

***Пример — Ревизия, статический анализ.***

4.43 **стрессовое тестирование** (stress testing): Тип тестирования уровня производительности, проводимого для оценки поведения элемента тестирования при условиях загрузки, выше ожидаемой или указанной в требованиях к производительности, или при доступности ресурсов, ниже минимальной, указанной в требованиях.

4.44 **структурное тестирование** (structure-based testing): См. термин «тестирование на основе структуры» согласно 4.45.

4.45 **тестирование на основе структуры** (structure-based testing): Динамическое тестирование, для которого тесты являются результатом анализа структуры элемента тестирования.

Примечания

1 Тестирование на основе структуры не ограничено использованием на уровне компонентов, а может использоваться на всех уровнях, например при покрытии пункта меню, как части тестирования системы.

2 Методика включает в себя тестирование ветвей, тестирование альтернатив и тестирование операторов.

3 Синонимами тестирования на основе структуры являются структурное тестирование, тестирование стеклянного ящика и тестирование методом «белого ящика».

4.46 **критерии приостановки** (suspension criteria): Критерии, используемые для того, чтобы (временно) остановить все или часть тестирующих действий.

4.47 **базис тестирования** (test basis): Свод знаний, используемых в качестве базы проекта тестирования и контрольных примеров.

Примечание — Базис тестирования может иметь форму документов, таких как спецификация требований, спецификация проекта или спецификация модуля, но может также представлять собой недокументированное понимание требуемого поведения.

4.48 **контрольный пример** (test case): Совокупность предварительных условий контрольного примера, входов (включая действия, где это применимо) и ожидаемых результатов, разработанных для управления выполнением элемента тестирования для достижения целей тестирования, включая корректную реализацию, идентификацию ошибок, проверку качества и получение другой значимой информации.

Примечания

1 Для подпроцесса тестирования, для которого он предназначен, контрольный пример — это самый низкий уровень входа тестирования (то есть контрольные примеры не состоят из других контрольных примеров).

2 Исходные условия контрольного примера включают тестовую среду, существующие данные (например, базы данных), программное обеспечение для тестирования, аппаратные средства и т. д.

3 Входы — это информация о данных, используемых для начала выполнения теста.

4 Ожидаемые результаты включают в себя критерии успеха, отказы в проверке и т. д.

4.49 **Спецификация Контрольного Примера** (Test Case Specification): Документация ряда одного или большего количества контрольных примеров.

4.50 **Процесс Выполнения Тестирования** (Test Completion Process): Процесс менеджмента тестирования, необходимый для обеспечения доступности полезных активов тестирования для дальнейшего использования, обеспечения удовлетворительного состояния тестовых сред, гарантии документирования и передачи соответствующим заинтересованным сторонам результатов тестирования.

4.51 **Отчет о Завершении Тестирования** (Test Completion Report): Отчет, в котором представлена сводка выполненного тестирования.

Примечание — Иногда также называют сводным отчетом тестирования.

4.52 **тестовое условие** (test condition): Тестируемый аспект компонента или системы, такой как функция, транзакция, возможность, атрибут качества или структурный элемент, идентифицированные как базис тестирования.

Примечание — Тестовые условия могут быть использованы для получения элементов покрытия или же могут сами по себе образовывать элементы покрытия.

4.53 **тестовое покрытие** (test coverage): Степень, выраженная в процентах, в которой специфицированные элементы тестового покрытия были проверены контрольным примером или контрольными примерами.

4.54 **элемент тестового покрытия** (test coverage item): Атрибут или комбинация атрибутов, которые являются производными одного или более тестовых условий, полученными посредством методики проектирования тестирования, которая позволяет оценить основательность выполнения теста.

4.55 **тестовые данные** (test data): Созданные или отобранные данные, удовлетворяющие входным требованиям для выполнения одного или более контрольных примеров, которые могут быть определены в плане тестирования, контрольном примере или процедуре тестирования.

Примечание — Тестовые данные могут храниться в тестируемом продукте (например, в массивах, плоских файлах или базе данных) или могут быть доступны из внешних источников, или предоставлены такими источниками, как другие системы, другие компоненты системы, устройства или операторский персонал.

4.56 **Отчет о Готовности Тестовых Данных** (Test Data Readiness Report): Документ, описывающий состояние каждого требования к тестовым данным.

4.57 **Процесс Разработки и Реализации Тестирования** (Test Design and Implementation Process): Процесс тестирования для получения и определения контрольных примеров и процедур тестирования.

4.58 **Спецификация Проекта Тестирования** (Test Design Specification): Документ, определяющий функции, которые будут проверены, и соответствующие тестовые условия.

4.59 **методика проектирования тестирования** (test design technique): Действия, понятия, процессы и шаблоны, необходимые для создания модели тестирования, которая используется при определении тестовых условий элемента тестирования, для получения соответствующих элементов тестового покрытия, а далее для разработки или выбора контрольных примеров.

4.60 **тестовая среда** (test environment): Различные средства, аппаратное и программное обеспечение, встроенное микропрограммное обеспечение, процедуры и документация, предназначенные или используемые для выполнения тестирования программного обеспечения.

Примечание — Тестовая среда может содержать в себе другие среды, необходимые для выполнения конкретных подпроцессов тестирования (например, тестовая среда объекта, среда теста производительности и т. д.).

4.61 **Отчет о Готовности Тестовой Среды** (Test Environment Readiness Report): Документ, описывающий выполнение всех требований к тестовой среде.

4.62 **Требования к Тестовой Среде** (Test Environment Requirements): Описание необходимых свойств тестовой среды.

Примечание — Все или часть требований к тестовой среде могут иметь ссылки, необходимые для поиска информации, например, ссылку на соответствующую Организационную Стратегию Тестирования, План Тестирования и/или Спецификацию Тестирования.

4.63 **Процесс Установки Тестовой Среды** (Test Environment Set-up Process): Процесс динамического тестирования для установки и поддержания требуемой тестовой среды.

4.64 **выполнение теста** (test execution): Процесс выполнения теста на элементе тестирования, приводящий к фактическим результатам.

4.65 **Журнал Выполнения Теста** (Test Execution Log): Документ, в который записываются детали выполнения одной или более процедур тестирования.

4.66 **Процесс Выполнения Теста** (Test Execution Process): Процесс динамического тестирования для выполнения процедур тестирования, созданных в процессе разработки и реализации тестирования в подготовленной тестовой среде, и записи результатов.

4.67 **Процесс Отчетности об Инцидентах Тестирования** (TestIncident Reporting Process): Процесс динамического тестирования для создания отчетов для соответствующих заинтересованных сторон о проблемах, требующих дальнейших действий, которые были идентифицированы во время процесса выполнения теста.

4.68 **элемент тестирования** (test item): Рабочий продукт, который является объектом тестирования.

***Пример — Система, элемент программного обеспечения, документ требований, спецификация проекта, руководство пользователя.***

4.69 **уровень тестирования** (test level): Конкретная реализация подпроцесса тестирования.

***Пример — Как подпроцессы тестирования можно рассматривать следующие общие уровни тестирования: уровень/подпроцесс покомпонентного тестирования, уровень/подпроцесс интеграционного теста, уровень/подпроцесс тестирования системы, уровень/подпроцесс приемочного испытания.***

Примечание — Синонимом уровня тестирования является фаза тестирования.

4.70 **менеджмент тестирования** (test management): Планирование, составление графика, оценка, мониторинг, отчетность, управление и выполнение действий по тестированию.

4.71 **Процесс Менеджмента Тестирования** (Test Management Process): Процесс тестирования, содержащий подпроцессы, необходимые для менеджмента проекта тестирования.

Примечание — См. процесс планирования тестирования, процесс мониторинга и управления тестированием, процесс завершения тестирования.

4.72 **Процесс Мониторинга и Управления Тестированием** (Test Monitoring and Control Process): Процесс менеджмента тестирования для обеспечения соответствия выполнения тестирования плану тестирования и организационным спецификациям тестирования.

4.73 **объект тестирования** (test object): См. термин «элемент тестирования» согласно 4.68.

4.74 **фаза тестирования** (test phase): Определенная реализация подпроцесса тестирования.

Примечание — Фазы тестирования означают то же, что и уровни тестирования, поэтому в примерах фазы тестирования совпадают с уровнями тестирования (например, фаза/подпроцесс тестирования системы).

4.75 **План Тестирования** (Test Plan): Подробное описание требуемых целей тестирования, средств и расписания их достижения, предназначенное для координации тестирующих действий для отдельного элемента тестирования или совокупности элементов тестирования.

Примечания

1 В проект может входить более одного плана тестирования, например может быть план тестирования проекта (также именуемый основным планом тестирования), который охватывает все тестирующие действия для проекта, а более подробная информация об определенных действиях тестирования может быть определена в одном или более планах подпроцессов тестирования (то есть план тестирования системы или план теста производительности).

2 Обычно план тестирования представляет собой печатный документ, хотя возможны и другие форматы плана, определяемые локально для организации или проекта.

3 Планы тестирования могут содержать деятельность, выходящую за рамки проекта, например план тестирования обслуживания.

4.76 **Процесс Планирования Тестирования** (Test Planning Process): Процесс менеджмента тестирования, используемый для выполнения планирования тестирования и разработки планов тестирования.

4.77 **методика тестирования** (test practice): Концептуальная основа, применимая к организационным процессам тестирования, процессам менеджмента тестирования и/или процессам динамического тестирования, чтобы упростить тестирование.

Примечание — Методики тестирования иногда называются подходом к тестированию.

4.78 **процедура тестирования** (test procedure): Последовательность контрольных примеров в порядке выполнения и любые связанные действия, которые могут потребоваться, чтобы установить начальные предпосылки и успешно выполнить завершающие действия после окончания тестирования.

Примечание — Процедуры тестирования включают в себя подробные инструкции для выполнения одного или более набора контрольных примеров, выбранных для последовательного выполнения, а также для установки общих исходных условий, обеспечения входа и оценки фактического результата для каждого выбранного контрольного примера.

4.79 **Спецификация Процедуры Тестирования** (Test Procedure Specification): Документ, определяющий одну или более процедур тестирования, представляющих собой наборы контрольных примеров, которые будут выполняться с определенной целью.

Примечания

1 Контрольные примеры в наборе тестов перечислены в порядке, требуемом в процедуре тестирования.

2 Ее также называют сценарием ручного тестирования. Спецификацию процедуры тестирования для автоматизированного тестового прогона обычно называют сценарием тестирования.

4.80 **процесс тестирования** (test process): Обеспечивает информацию о качестве программного продукта, зачастую состоит из множества действий, сгруппированных в один или несколько подпроцессов тестирования.

***Пример — Процесс тестирования для определенного проекта может состоять из множества подпроцессов, например подпроцесса тестирования системы, подпроцесса планирования тестирования (часть большего процесса менеджмента тестирования) или подпроцесса статического тестирования.***

4.81 **тестовое требование** (test requirement): См. термин «тестовое условие» согласно 4.52.

4.82 **результат тестирования** (test result): Индикатор того, прошел ли определенный контрольный пример успешно или нет, то есть соответствует ли фактический результат элемента тестирования ожидаемому результату или наблюдались отклонения.

4.83 **сценарий тестирования** (test script): Спецификация процедуры тестирования для ручного или автоматизированного тестирования.

4.84 **набор тестов** (test set): Один или совокупность нескольких контрольных примеров с общими ограничениями на их выполнение.

***Пример — Определенная тестовая среда, специализированные знания проблемной области или определенная цель.***

4.85 **спецификация тестирования** (test specification): Подробная документация проекта тестирования, контрольных примеров и процедур тестирования для конкретного элемента тестирования.

Примечание — Спецификация тестирования может быть представлена одним документом, набором документов или другими способами, например записями базы данных и документами.

4.86 **Отчет о Ходе Тестирования** (Test Status Report): Отчет, предоставляющий информацию о состоянии тестирования, которое выполняется в указанный отчетный период.

4.87 **стратегия тестирования** (test strategy): Часть Плана Тестирования, в которой описан подход к тестированию определенного проекта тестирования или процессам и подпроцессам тестирования.

Примечания

1 Стратегия тестирования — это производная от Организационной Стратегии Тестирования.

2 Стратегия тестирования обычно определяет некоторые или все из следующих аспектов: используемые методики тестирования, реализуемые тестовые подпроцессы, повторное тестирование и регрессионное тестирование, которые будут использоваться, методы проектирования тестирования, соответствующие критерии завершения тестирования, тестовые данные, тестовую среду, требования к инструментам тестирования и ожидаемые результаты тестирования.

4.88 **подпроцесс тестирования** (test sub-process): Процессы менеджмента тестирования и процессы динамического (и статического) тестирования, используемые для выполнения определенного уровня тестирования (например, тестирование системы, приемочные испытания) или определенного типа тестирования (например, тестирование удобства использования, тестирование производительности) обычно в контексте полного процесса тестирования для проекта тестирования.

Примечание — В подпроцесс тестирования могут быть включены один или несколько типов тестирования. Обычно, в зависимости от используемой модели жизненного цикла, подпроцессы тестирования также называют фазами тестирования, уровнями тестирования, этапами тестирования или задачами тестирования.

4.89 **методика тестирования** (test technique): См. термин «метод проектирования тестирования» согласно 4.59.

4.90 **матрица прослеживаемости тестирования** (test traceability matrix): Документ, электронная таблица или другой автоматизированный инструмент, используемые для идентификации в документации и программном обеспечении связанных элементов, таких как требования соответствующего тестирования.

Примечания

1 Также известна как матрица перекрестных ссылок верификации, матрица проверки требований, таблица верификации требований и др.

2 Различные матрицы прослеживаемости тестирований могут отличаться содержащейся информацией, форматами и уровнями детализации.

4.91 **тип тестирования** (test type): Совокупность тестирующих действий, которая фокусируется на определенных показателях качества.

Примечание — Тип тестирования может быть выполнен одиночным подпроцессом тестирования или несколькими подпроцессами тестирования (например, тестирование производительности, выполненное в ходе подпроцесса покомпонентного тестирования и также выполненное в ходе подпроцесса тестирования системы).

***Примеры — Тестирование защищенности, функциональное тестирование, тестирование удобства использования и тестирование производительности.***

4.92 **тестирование** (testing): Набор операций, проводимых для обеспечения выявления и/или оценки свойств одного или более элементов тестирования.

Примечание — Действия тестирования могут включать в себя планирование, подготовку, выполнение, создание отчетов и менеджмент, поскольку все они направлены на тестирование.

4.93 **средства тестирования** (testware): Артефакты, произведенные во время процесса тестирования, требуемые для планирования, разработки и выполнения тестирования.

Примечание — В средства тестирования могут входить документация, сценарии, входные данные, ожидаемые результаты, файлы, базы данных, среда и любое дополнительное программное обеспечение или утилиты, используемые в ходе тестирования.

4.94 **тестирование без сценария** (unscripted testing): Динамическое тестирование, в котором действия тестера определены инструкциями, записанными в контрольном примере.

4.95 **объемное тестирование** (volume testing): Тип тестирования уровня производительности, проводимого для оценки способности элемента тестирования обработать определенные объемы данных (обычно равных или близких к максимальным указанным потенциальным возможностям) с точки зрения потенциальных возможностей пропускной способности, емкости памяти или того и другого.

4.96 **тестирование методом «белого ящика»** (white box testing): См. термин «тестирование на основе структуры» согласно 4.45.

## 5 Понятия тестирования программного обеспечения

### 5.1 Введение в тестирование программного обеспечения

Необходимость тестирования программного обеспечения может быть продиктована следующими условиями:

- лица, принимающие решения, запрашивают информацию о показателях качества элемента(ов) тестирования;
- проверяемый(ые) элемент(ы) тестирования не всегда делает то, что от него (них) ожидается;
- необходимо произвести верификацию проверяемого(ых) элемента(ов) тестирования;
- необходимо произвести валидацию проверяемого(ых) элемента(ов) тестирования и/или
- необходимо провести оценку элемента(ов) тестирования по всему жизненному циклу разработки программного обеспечения и систем.

Общеизвестно, что создать совершенное программное обеспечение невозможно. Поэтому прежде чем программное обеспечение будет передано пользователям, его необходимо протестировать, чтобы в производстве программного обеспечения снизить риск ошибок, оказывающих негативное влияние на его функционирование. В равной степени необходимо обеспечить качественное выполнение тестирования программного обеспечения.

Ошибки или допущенные дефекты обычно имеют место и неизбежны. Опечатка или ошибка, сделанная человеком, приводит к возникновению дефекта в продукте, над которым человек работает (например, спецификация требований или компонент программного обеспечения). Дефект не оказывает влияния на функционирование программного обеспечения до тех пор, пока он не будет обнаружен при эксплуатации программного обеспечения. Однако если дефект обнаружен в реальных условиях, когда продукт уже сдан в эксплуатацию, то это может привести к тому, что продукт не будет удовлетворять законным потребностям пользователя. Последствия программной ошибки для пользователя могут быть серьезны. Например, дефект может поставить под угрозу бизнес-репутацию, государственную безопасность, бизнес-экономическую жизнеспособность, бизнес или безопасность пользователей и/или окружающую среду.

Динамическое тестирование является необходимым, но не достаточным условием, чтобы обеспечить приемлемую уверенность в том, что программное обеспечение будет функционировать, как задумано. В сочетании с эффективными действиями динамического тестирования необходимо произвести дополнительные действия статического тестирования, такие как экспертная оценка и статический анализ.

Основными целями тестирования являются: предоставление информации о качестве элемента тестирования и любых остаточных рисках относительно того, до какой степени элемент тестирования был проверен; обнаружение дефектов в элементе тестирования до его передачи в эксплуатацию; смягчение рисков получения продукта низкого качества заинтересованными сторонами.

Вышеупомянутая информация может использоваться в нескольких целях, включая:

- улучшение элемента тестирования путем устранения дефектов;
- улучшение управленческих решений, предоставляя как основание для решений информацию о качестве и рисках;
- улучшение процессов в организации, особо выделяя процессы, которые позволяют дефектам возникать и/или оставаться скрытыми там, где они могут быть обнаружены.

Качество продукта имеет много аспектов, включая соответствие спецификациям, отсутствие дефектов и удовлетворение продукта требованиям пользователей. В ИСО/МЭК 25010 «Модели качества систем и программного обеспечения» определено восемь показателей качества, которые могут быть измерены или оценены путем тестирования (см. 5.5.3).

Тестирование программного обеспечения должно быть направлено на предоставление информации о программном продукте и нахождение максимально возможного числа дефектов на возможно ранних этапах процесса разработки при заданных ограничениях стоимости и графика разработки.

Основные положения тестирования заключаются в следующем:

- тестирование — это процесс, представляющий собой совокупность взаимосвязанных или взаимодействующих видов деятельности, преобразующих входы в выходы. Цель настоящего стандарта состоит в том, чтобы представить и описать общие процессы тестирования (дополнительная информация доступна в ИСО/МЭК/ИИЭР 29119-2 «Процессы тестирования»);

- Организационный Процесс Тестирования устанавливает и поддерживает политики тестирования и стратегии тестирования, которые повсеместно применяются в проектах и функциях организации;

- тестирование необходимо планировать, контролировать и управлять им. Процессы тестирования, описанные в ИСО/МЭК/ИИЭР 29119-2, включают в себя процесс менеджмента тестирования и могут быть применены к тестированию во всех жизненных циклах разработки и менеджменте исследовательского тестирования;

- процессы и подпроцессы тестирования применимы для любой фазы или уровня тестирования (например, тестирование системы) и для любого типа тестирования (например, тестирование производительности);

- тестирование предполагает исследование элемента тестирования;

- возможно тестирование продукта без выполнения его на компьютере. В настоящем стандарте и во многих областях промышленности такое тестирование называют статическим тестированием, хотя в других стандартах (например, в ИИЭР 1028) оно может называться анализом, пошаговым разбором или проверкой. Для статического тестирования настоящий стандарт подтверждает и определяет роль тестера в этих действиях, даже если они могут быть выполнены другими группами в рамках проекта или определены другими стандартами, не относящимися к тестированию. Это связано с тем, что действия статического тестирования считаются крайне важными для полного тестирования жизненного цикла и, как показала практика, выполнение тестирования критически важно для раннего обнаружения дефектов, снижения полной стоимости проекта и обеспечения лучшего удовлетворения требования графика;

- статическое тестирование может также включать в себя использование инструментов статического анализа, которые находят дефекты в коде или документах без выполнения кода (например, компилятор, цикломатический анализатор сложности или анализатор защищенности кода);

- динамическое тестирование представляет собой нечто большее, чем «просто» выполнение исполнимых элементов тестирования, сюда входят также как действия подготовки, так и последующие действия. Процессы динамического тестирования, описанные в ИСО/МЭК/ИИЭР 29119-2, охватывают каждое из действий, которые будут выполняться в ходе динамического тестирования;

- верификация — это подтверждение путем представления объективных доказательств выполнения данным рабочим элементом установленных требований;

- валидация демонстрирует, что рабочий элемент может использоваться пользователями для решения определенных ими задач;

- тестирование, как статическое так и динамическое, должно быть направлено на получение обоих типов подтверждения, хотя и должно допускать, что подтверждение не будет получено немедленно из-за обнаружения дефектов.

#### 5.1.1 Роль тестирования в верификации и валидации

В настоящем стандарте рассматриваются только некоторые действия верификации и валидации. В частности, рассматривается тестирование программного обеспечения, которое является основным действием при верификации и валидации. Такие стандарты, как ИСО/МЭК 12207 и ИИЭР 1012, рассматривают и другие действия верификации или валидации. Настоящий стандарт ориентирован только на тестирование и в нем не рассматриваются другие действия валидации и верификации (например, анализ валидации и верификации, формальные методы). Для обеспечения полной валидации и верификации продукта организация в составе своей комплексной технической программы должна использовать настоящий стандарт совместно с другими стандартами. Иерархия действий верификации и валидации приводится в приложении А.

#### 5.1.2 Исчерпывающее тестирование

Из-за сложности систем и программного обеспечения не представляется возможным исчерпывающе проверить каждый аспект любого конкретного элемента тестирования. Тестеры должны осознать, что исчерпывающее тестирование невозможно и что тестирующие действия должны быть направлены на возможно лучшее выполнение задач тестирования для элемента тестирования. Тестирование на базе рисков — это подход, при котором риск используется для координации усилий тестирования. Тестирование на базе рисков рассматривается в 5.4.

### 5.1.3 Тестирование как эвристика

Эвристика в технике (и в программной инженерии) — это основанный на опыте метод (метод проб и ошибок), который можно использовать в качестве вспомогательного средства в разрешении проблем и разработках. Хотя эвристика и может использоваться для разрешения проблем, в отдельных случаях она ненадежна в том смысле, что может не решить задачу или решить ее только частично. На эвристике основывается значительная часть тестирований систем и программного обеспечения. Например, эвристика полезна при создании модели тестируемой системы, однако она не может моделировать систему в полной мере, и поэтому дефекты в системе не могут быть выявлены даже при том, что тестирование кажется полным. Признание того, что метод тестирования может быть ненадежен, позволяет снизить риск неэффективной стратегии тестирования, используя несколько стратегий тестирования.

## 5.2 Тестирование программного обеспечения в организационном контексте и контексте проекта

Компании, вовлеченные в разработку или приобретение программных продуктов, заинтересованы в разработке и использовании эффективных, действенных и повторимых процессов. Для осуществления этого компании разрабатывают полноценный набор процессов жизненного цикла программного обеспечения, используемых в проектах выполняемых ими разработок. Настоящий стандарт предназначен как для использования в организации в целом, так и для применения в отдельных проектах. Организация может принять настоящий стандарт и дополнить его по мере необходимости дополнительными процедурами, методами, инструментами и политиками. Конкретный проект разработки программного обеспечения или системы, выполняемый организацией, как правило, соответствует процессам организации, а не соответствует напрямую настоящему стандарту. В отдельных случаях проект может выполняться организацией, которая не располагает надлежащей совокупностью процессов организации. Такой проект может непосредственно применить положения настоящего стандарта.

Для любой производящей программное обеспечение организации, будь ею многонациональная организация с тысячью тестеров или компания из одного человека, обязательства по тестированию программного обеспечения должен нести высший уровень организационного менеджмента, будь то генеральный директор, открытый руководящий комитет или начальник отдела. Желательно, чтобы эти обязательства были заложены в Организационную Политику Тестирования и в одну или более Организационные Стратегии Тестирования, служащие основой для всех выполняемых в организации видов тестирования программного обеспечения. Политики Тестирования и Организационные Стратегии Тестирования обычно присутствуют в наиболее развитых организациях. В организациях менее зрелых тестирование может выполняться и выполняется без формальных политик тестирования и организационных стратегий тестирования, но такая практика не обеспечивает обоснованности тестирования в организации и, как правило, показывает меньшую эффективность и результативность тестирования, выполняемого в проектах.

Тестирование программного обеспечения выполняется как управляемый контекстом процесс. Это означает, что процесс нужно планировать, контролировать и им нужно управлять. Контекстом тестирования может быть как проект разработки (в пределах от многочисленного, многолетнего формального проекта разработки до неофициальной разработки, требующей нескольких часов работы одного человека), так и текущее сопровождение функционирующей системы. Необходимо отметить некоторые положения контекста тестирования: полный бюджет; требования расписания; риск; организационная культура; ожидания потребителя/пользователя; готовность сред инфраструктуры для тестирования; область применения проекта; критичность проекта и т. д. Накопленный опыт отрасли показывает, что ни одна стратегия тестирования, ни один план, метод или процесс не будут работать во всех ситуациях. Следовательно, организации и проекты должны адаптироваться и совершенствоваться в деталях тестирования, опираясь на стандарты, такие как настоящий стандарт.

Полный план проекта должен включать в себя анализ тестирующих действий, которые будут выполняться как часть проекта. В Плане Тестирования Проекта должны быть отражены как Организационная Политика Тестирования и Организационная Стратегия Тестирования, так и отклонения от этих организационных руководств. Кроме того, в плане должны быть учтены ограничения, заданные в полном плане проекта. План Тестирования Проекта включает в себя стратегию тестирования проекта и специфичные, используемые для выработки стратегии решения проекта (включая предположения). Основной элемент планирования тестирования — это оценка различных нужд тестирования и балансировка ресурсов между различными тестированиями. План тестирования фиксирует результат этого анализа. В соответствии с настоящим стандартом в основу метода определения потребностей тести-

рования закладывается риск (см. 5.4), однако в стратегию также могут быть включены и другие методики тестирования, описанные в 5.6.

Для всего проекта тестирование осуществляется часто через некоторое множество подпроцессов тестирования, каждый из которых может иметь соответствующий План Тестирования (План Подпроцесса Тестирования, например План Тестирования Системы или План Теста Производительности), включающий в себя как стратегию подпроцесса тестирования, согласованную со стратегией тестирования проекта, так и специфические детали подпроцесса тестирования.

На рисунке 1 показано, как тестирование вписывается в иерархический контекст. Тестирование базируется на статусе регулирования, присущем организации (если регулирование имеет место). Упомянутый контекст состоит из законов, инструкций и промышленных стандартов. В соответствии с нормативной ситуацией организация разрабатывает необходимые политики и процедуры, требуемые для успешной работы. Организационная Политика Тестирования работает на этом уровне. Каждый проект организации реализуется для соответствия некоторому требованию или возможности, которые идентифицированы организацией. Контекст проекта помогает определить, какую модель жизненного цикла выбрать. На этом уровне на основе контекста проекта и модели жизненного цикла определяется стратегия тестирования. План проекта и стратегия тестирования формируют базу для Плана Тестирования Проекта.

План Тестирования Проекта описывает общую стратегию тестирования и процессы тестирования, которые будут использоваться. Он устанавливает контекст тестирования проекта, определяя цели, методы, ресурсы и расписание. Кроме того, он идентифицирует применимые подпроцессы тестирования (например, тестирование системы, тестирование производительности). Идентифицированные подпроцессы далее расписываются в плане тестирования подпроцесса (например, План

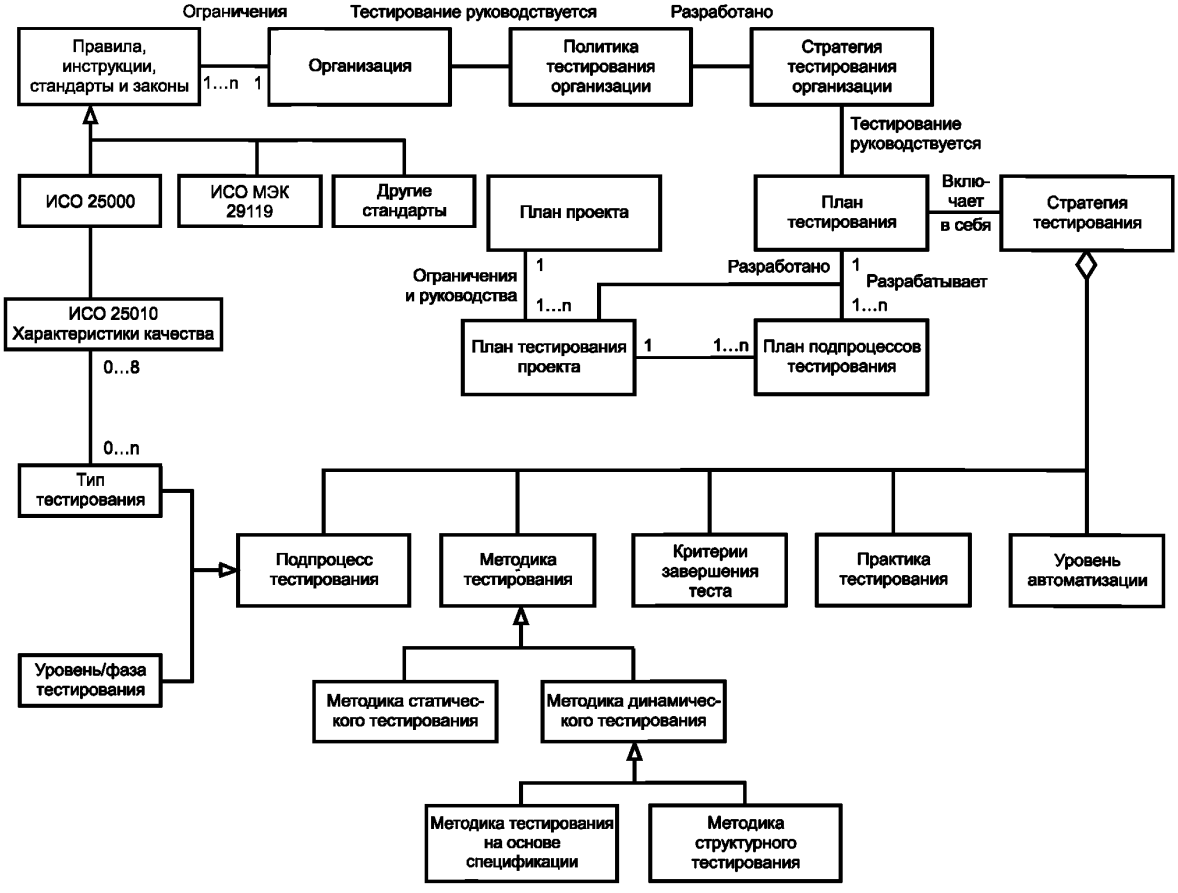


Рисунок 1 — Иерархическая схема тестирования

Тестирования Системы, План Теста Производительности). План тестирования также определяет надлежащий метод проектирования тестирования (статическое или динамическое), необходимый для выполнения подпроцесса тестирования, требуемого конкретным планом. Более подробно методы проектирования тестирования описаны в 5.5.6 и ИСО/МЭК/ИИЭР 29119-4.

Каждый план тестирования подпроцесса может относиться более чем к одному уровню тестирования (например, план тестирования защищенности может относиться к нескольким уровням тестирования), а кроме того, может относиться к более чем одному типу тестирования (например, План Тестирования Системы, который относится к тестированию функциональности и тестированию производительности на уровне тестирования системы). План тестирования подпроцесса также определяет стратегию выполнения теста (например, по сценарию, без сценария или смесь того и другого).

План тестирования включает в себя стратегию тестирования (см. раздел 6). Стратегии тестирования настраиваются на конкретный контекст проекта тестирования. Стратегии тестирования должны соотноситься конкретными элементами, показанными на рисунке 1 и определенными в других частях серии стандартов ИСО/МЭК/ИИЭР 29119. Каждый план тестирования (и стратегия) будет уникален, отличаясь от других: подпроцессами тестирования, уровнями автоматизации, совокупностью методик проектирования тестирования, критериями завершения теста, их планированием и выделением ресурсов. Планирование и выбор этих аспектов начинаются на ранней стадии проекта и будут продолжаться в течение жизненного цикла тестирования, влияя как фактор на изменение риска. Следует ожидать, что многие части плана и стратегии тестирования изменятся, хотя изменения будут, очевидно, в пределах ограничений проекта, организации или регулирующих норм.

Взаимосвязи между общим процессом тестирования, общими подпроцессами тестирования, уровнями/фазами тестирования и типами тестирования более подробно показаны на рисунке 2. Рисунок показывает реализацию общих подпроцессов тестирования на определенных уровнях тестирования и в соответствии с типом тестирования. Общий подпроцесс тестирования может быть реализован:

- на уровне или фазе тестирования. То есть каждый уровень тестирования представляет собой определенное приложение общего подпроцесса тестирования (например, фаза покомпонентного тестирования, уровень приемочного испытания);

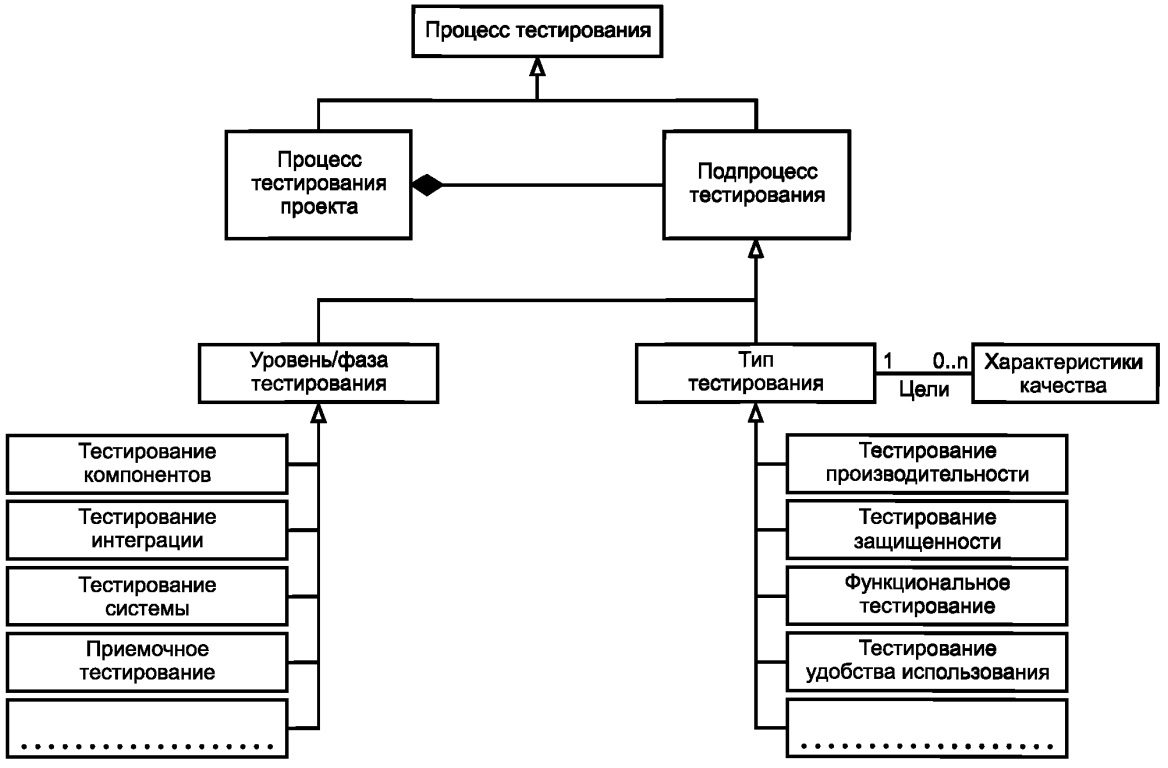


Рисунок 2 — Взаимосвязи между общим подпроцессом тестирования, уровнями тестирования и типами тестирования

- как тестирование определенного типа. То есть каждый тип тестирования представляет собой определенное приложение общего подпроцесса тестирования (например, тестирование производительности, тестирование удобства использования);
- подпроцесс тестирования, соответствующий уровню тестирования, может включать в себя больше одного подпроцесса разного типа тестирования (например, функциональный и тестирование производительности являются частями тестирования системы);
- процесс тестирования проекта может состоять из последовательности подпроцессов тестирования (например, тестирование компонента, интеграционное тестирование, тестирование системы и подпроцессы тестирования приемочных испытаний).

На рисунке 2 в деталях показаны связи между типами тестирования и показателями качества (как определено в ИСО/МЭК 25010 «Модели качества систем и программного обеспечения»). Каждый тип тестирования ориентирован на один определенный показатель качества. Более подробно связи между типами тестирования и показателями качества рассматриваются в 5.5.3.

**5.2.1 Процесс тестирования**

В настоящем стандарте используется трехуровневая модель процесса, подробно описанная в ИСО/МЭК/ИИЭР 29119-2 и показанная в общем виде на рисунке 3. Модель процесса начинается с организационного управления спецификациями тестирования высокого уровня (организационными спецификациями), такими как Организационная Политика Тестирования и Организационная Стратегия Тестирования. Средний уровень показывает менеджмент тестирования (менеджмент тестирования проекта, менеджмент фазы тестирования, менеджмент типа тестирования), в то время как нижний уровень определяет множество процессов тестирования, используемых в динамическом тестировании.

Трехуровневая модель процесса показана на рисунке 3.




Рисунок 3 — Многоуровневая связь процессов тестирования

Политика Тестирования выражает в бизнес-терминах ожидания менеджмента организации и подход к тестированию программного обеспечения. Она ориентирована на руководителей и старших менеджеров, хотя и может быть полезна для каждого, кто связан с тестированием. Политика Тестирования также руководит предпочтительным направлением и динамикой формирования и выполнения Организационной Стратегии Тестирования и процессов тестирования организации. Создание, реализация и поддержка Организационной Политики Тестирования определяется Организационным Процессом Тестирования.

Организационная Стратегия Тестирования выражает требования и ограничения для процессов менеджмента тестирования и процессов динамического тестирования, которые должны удовлетворяться для всех выполненных в организации проектов, если они не слишком отличаются по характеру и когда может быть сформулировано несколько Организационных Стратегий Тестирования. Стратегия согласована с Организационной Политикой Тестирования и определяет, как должно быть выполнено тестирование. Создание, реализация и поддержка Организационной Стратегии Тестирования также определяется Организационным Процессом Тестирования.

Менеджмент выполняемого тестирования определяется Процессами Менеджмента Тестирования. На основе анализа идентифицированных рисков и ограничений проекта и с учетом Организационной Стратегии Тестирования разрабатывается связанная с проектом стратегия тестирования. Эта стратегия разрабатывается с точки зрения определения необходимого статического и динамического тестирования, укомплектованности персоналом, баланса заданных ограничений (ресурсы и время) с

предопределенными охватом и качеством результатов намеченного для выполнения тестирования. Все это записывается в План Тестирования Проекта. Для того чтобы обеспечивать запланированное продвижение тестирования и гарантию соответствующей обработки рисков во время выполнения тестирования, необходимо осуществлять мониторинг. Если в ходе тестирования потребуется внести какие-либо изменения в тестирующие действия, то соответствующему процессу или подпроцессу тестирования должны быть переданы управляющие директивы. Отчеты о Ходе Тестирования могут создаваться регулярно для информирования заинтересованных сторон о прогрессе тестирования в течение всего периода мониторинга и управления. Полный результат тестирования проекта оформляется в виде Отчета Завершения Тестирования Проекта.

Процессы Менеджмента Тестирования показаны на рисунке 4.




Рисунок 4 — Процессы Менеджмента Тестирования

Полное тестирование проекта обычно разбивается на меньшие подпроцессы тестирования (например, тестирование компонентов, тестирование системы, тестирование удобства использования, тестирование производительности), и ими нужно управлять, их нужно выполнять и о них нужно информировать так же, как в случае тестирования полного проекта. Процессы Менеджмента Тестирования также применимы к подпроцессам тестирования. Примерами планов подпроцессов тестирования являются План Тестирования Системы, План Приемочного Испытания или План Теста Производительности.

В подпроцессы тестирования может входить как статическое тестирование, так и динамическое тестирование. Процессы динамического тестирования показаны в общих чертах на рисунке 5 и полностью описаны в ИСО/МЭК/ИИЭР 29119-2. Процессы статического тестирования описаны в других опубликованных стандартах, например ИИЭР 1028.

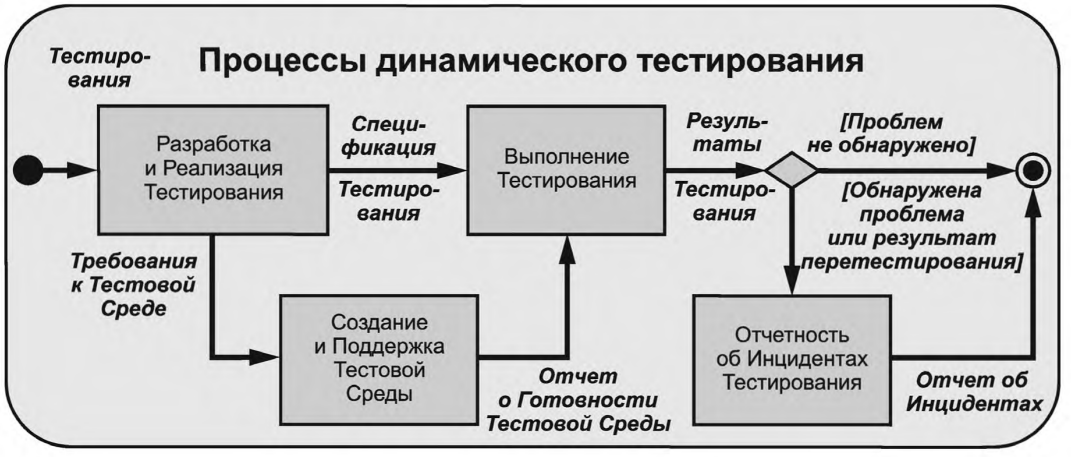


Рисунок 5 — Процессы динамического тестирования

Для получения дополнительной информации о любом из процессов тестирования, включая Организационный Процесс Тестирования, Процессы Менеджмента Тестирования и Процесс Динамического Тестирования, следует обратиться к ИСО/МЭК/ИИЭР 29119-2.

### 5.3 Общие процессы тестирования в жизненном цикле программного обеспечения

Программное обеспечение имеет ожидаемый жизненный цикл от начальной его концепции до возможного прекращения его использования. Тестирование программного обеспечения имеет место в более широком контексте разработки и сопровождения программного обеспечения. В 5.3 в качестве примера рассмотрен жизненный цикл разработки программного обеспечения и некоторые связи между его подпроцессами и процессами тестирования. В ИСО/МЭК 12207:2008 подробно рассмотрены жизненные циклы программного обеспечения. В ИСО/МЭК 15288 подробно рассмотрены процессы жизненного цикла системы. Пример жизненного цикла системы показан на рисунке 6.

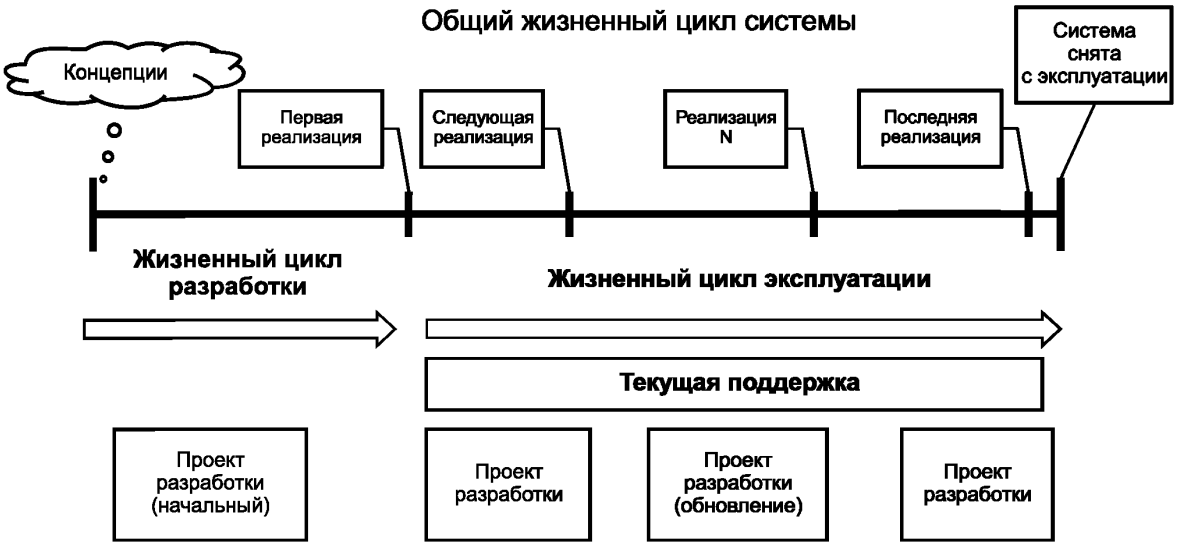


Рисунок 6 — Пример жизненного цикла программного обеспечения

Жизненный цикл программного обеспечения обычно состоит из нескольких жизненных подциклов. На рисунке 6 показан жизненный цикл программного обеспечения, зачастую состоящий из одного или более жизненных циклов разработки и одного или более жизненных циклов эксплуатации.

Период времени от концепции до первоначальной версии известен как жизненный цикл разработки, который является частью жизненного цикла программного обеспечения. Жизненный цикл разработки управляется и контролируется в проекте разработки.

С момента первого запуска система переходит в эксплуатацию (функционирование). Система остается в эксплуатации до момента прекращения использования. Этот период может варьироваться от нескольких часов до нескольких десятилетий. Период функционирования часто состоит из промежутков времени, в течение которых используется определенная версия системы, а в то же время разрабатывается для выпуска новая версия. Разработка любой новой версии должна рассматриваться как самостоятельный проект, что влечет за собой соответствующие требования тестирования. Текущее сопровождение обычно настраивается таким образом, чтобы обеспечить доступность и нормальное функционирование системы.

В отдельных случаях возможно тестирование функционирующей системы без соответствующего проекта разработки, например «пробные прогоны» тестов Аварийного Восстановления. В подобных ситуациях также применимы представленные в настоящем стандарте процессы.

Тестирование может выполняться для оценки удовлетворения бизнес-требований к приобретаемому программному обеспечению. Основы оценки и тестирования приобретаемого готового коммерческого программного обеспечения (COTS) можно найти в ИСО/МЭК 25051.

#### 5.3.1 Подпроцессы проекта разработки и их результаты

Разработка программного обеспечения и разработка системы обычно состоят из нескольких общих стандартных блоков. В индустрии программного обеспечения эти стандартные блоки обычно называют: «фазы», «этапы», «шаги», «уровни» или в общем случае — «подпроцессы разработки».

В подпроцессы разработки могут входить:
- инженерия требований;
- проектирование архитектуры;
- детальное проектирование;
- кодирование.

Конкретный подход к разработке системы, принятый в целом в организации или отдельном проекте, определяет организацию этих подпроцессов разработки. В проекте последовательной разработки анализ требований может быть самостоятельным начальным процессом. В проекте динамичной разработки требования будут определяться каждые несколько недель для каждой разрабатываемой части.

В каждом из подпроцессов разработки что-то производится. Это может быть высокоструктурированный подробный документ, неофициальное документированное или недокументированное решение.

В любом конкретном проекте разработки отдельные подпроцессы разработки могут быть выполнены только один раз, а могут повторяться столько раз, сколько необходимо. Общие подпроцессы разработки и связанные с ними рабочие продукты, подсистемы и завершенная программная система показаны на рисунке 7.

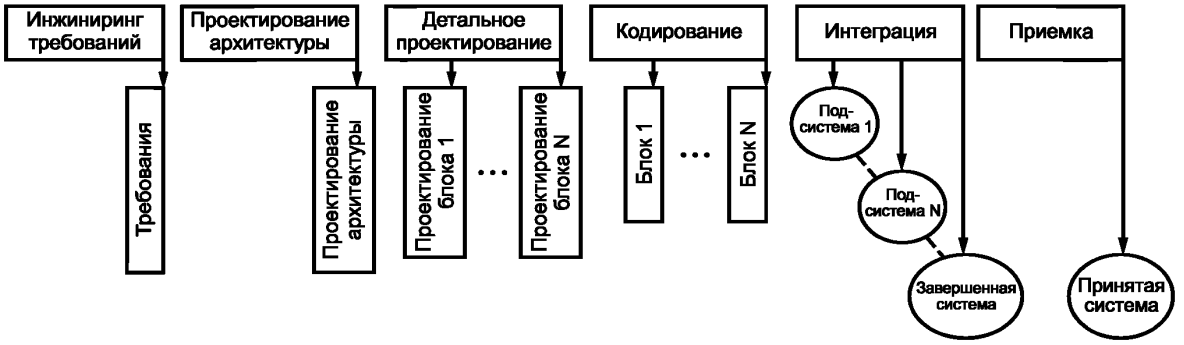


Рисунок 7 — Пример подпроцессов разработки

Каждый рабочий продукт, компонент программной системы и полная программная система — это потенциальный элемент тестирования.

Следует отметить, что определение модели разработки выходит за рамки применения настоящего стандарта. Фазы, показанные на рисунке 7, являются только примерами для иллюстрации применения тестирования при разработке.

### 5.3.2 Текущее сопровождение и его результаты

Текущее сопровождение имеет место в период функционирования жизненного цикла программного обеспечения, то есть после начальной разработки, и им можно управлять в контексте Управления Приложениями или структуры процессов Менеджмента Услуг ИТ. В проекте может быть предусмотрено непрерывное обслуживание, и этим он зачастую сильно отличается от проекта разработки, например финансовая модель может быть другой. Одна из распространенных финансовых моделей планирует количество бюджетных средств на сопровождение на определенный период. Это может быть закреплено в соглашении об уровне обслуживания (SLA) между потребителем и обслуживающей организацией.

Процесс текущего сопровождения обычно предусматривает поддержание приемлемого или согласованного уровня надежности и готовности системы. Это включает производство новых версий системы с исправлением обнаруженных при функционировании наиболее приоритетных дефектов и, возможно, внедрение приоритетных изменений функциональности. Такие версии могут реализовываться в виде «живой» системы либо на разовой основе по завершении исправлений и изменений, либо с заданной частотой, например каждые три месяца. Если корректировка по ходу обслуживания осуществляется на разовой основе, то реализация и внедрение необходимых исправлений и/или изменений обычно выполняются как можно быстрее. Если период корректировки по ходу обслуживания фиксирован, то реализуется столько необходимых исправлений и/или изменений, сколько возможно реализовать в этот период времени с учетом последовательности внедрения согласно перечню приоритетов. Основные результаты такого периода реализации показаны на рисунке 8.

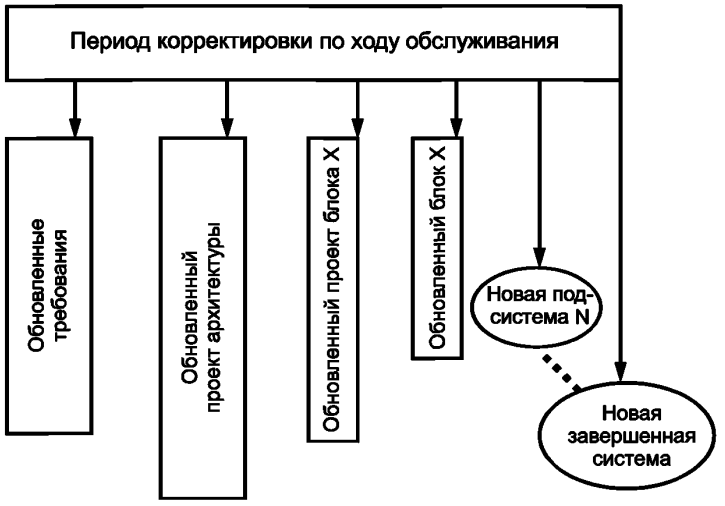


Рисунок 8 — Пример периода корректировки по ходу обслуживания

В зависимости от назначения реализации каждый из выходов, показанных на рисунке 8, может быть произведен за один период реализации, но может случиться и так, что потребуется несколько периодов. Например, может случиться так, что оперативная коррекция не повлияет на требования и проект, но затронет только код и завершенную систему, или может случиться, что все рабочие продукты будут затронуты несколько раз перед тем, как система будет готова к выпуску.

Периоды корректировки по ходу обслуживания повторяются по мере необходимости в период функционирования жизненного цикла системы.

**5.3.3 Процессы поддержки в жизненном цикле разработки программного обеспечения**

Процессы поддержки необходимы в организации для обеспечения поддержки жизненного цикла разработки программного обеспечения. Некоторые из этих процессов:

- обеспечение качества;
- менеджмент проектов;
- менеджмент конфигурации;
- совершенствование процессов.

5.3.3.1 Обеспечение качества и тестирование

Обеспечение Качества — это совокупность запланированных и систематических процессов и действий поддержки, необходимых для обеспечения надлежащего уровня уверенности в том, что процесс или рабочий продукт удовлетворяет установленным техническим требованиям или требованиям к качеству. Достигается это сочетанием методов, стандартов, инструментов и навыков, признанных как соответствующая практика. Процесс Обеспечения Качества использует результаты тестирования и другую информацию для анализа, оценки и информирования о любой проблеме (включая любой риск) в проектировании, планировании или выполнении процессов программной инженерии.

В ходе тестирования необходимо собирать показатели, поскольку они могут обеспечить информацию о качестве процессов тестирования и/или элементов тестирования и результативности их использования в каждом проекте.

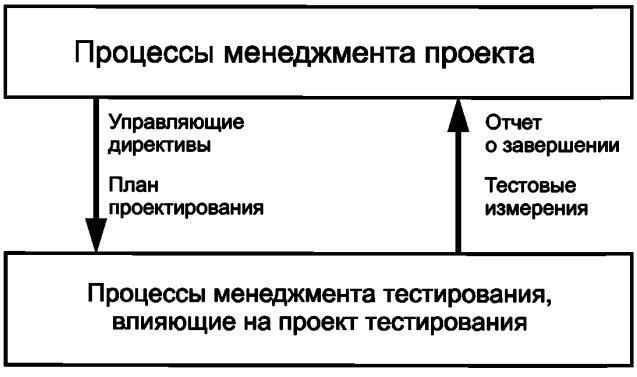


Рисунок 9 — Взаимосвязь всего проекта и проекта тестирования

5.3.3.2 Менеджмент проектов и тестирование

Менеджмент проектов относится к процессам поддержки, которые используются для планирования и контроля выполнения проекта, включая менеджмент проекта тестирования в составе всего проекта. Независимо от того, кто несет ответственность за отдельные процессы, процессы менеджмента проектов и менеджмента тестирования тесно связаны между собой, как показано на рисунке 9.

Оценка, анализ рисков и планирование тестирующих действий должны составлять единое целое с планированием всего проекта. Поэтому для процесса менеджмента проекта тестирования план проекта, который является выходным информационным элементом процесса менеджмента проектов, является входом.

В ходе выполнения проекта тестирования измерения, полученные в результате конкретных тестирующих действий, анализируются менеджером тестирования и передаются менеджеру проекта для анализа в контексте проекта. Это может привести к изменениям в плане проекта, затрагивающим проект тестирования, к обновлению плана проекта и формированию соответствующих директив для проекта тестирования, необходимых для обеспечения гарантии, что проект тестирования держится под контролем.

По завершении подпроцесса тестирования или проекта тестирования менеджеру проекта представляется отчет о завершении, в котором суммированы ход и результаты подпроцесса тестирования или проекта тестирования.

5.3.3.3 Менеджмент конфигурации и тестирование

Менеджмент конфигурации — это другая совокупность процессов поддержки, с которыми тесно связано тестирование. Цель менеджмента конфигурации состоит в том, чтобы установить и поддерживать целостность рабочих продуктов. Для того чтобы выяснить, соответствует ли организация или проект предъявляемым требованиям, рекомендуется проверить систему менеджмента конфигурации организации или проекта перед ее практическим использованием.

Процессы менеджмента конфигурации включают в себя уникальную идентификацию, управляемое хранение, аудит реализации, контроль изменений и отчетность о состоянии отдельных рабочих продуктов, системных компонентов и систем в течение времени жизни продукта. Объект менеджмента конфигурации вызывают элементом конфигурации. Процессы менеджмента конфигурации управляются событиями, то есть все они инициируются независимо друг от друга в зависимости от требований других процессов.

Рабочие продукты процесса тестирования, которые могут быть переданы в менеджмент конфигурации, включают в себя:

- Организационные Спецификации Тестирования (например, Политика Тестирования, Организационная Стратегия Тестирования).
- Планы тестирования.
- Спецификации тестирования.
- Элементы конфигурации тестовой среды, такие как инструменты тестирования, тестовые данные (основа), драйверы и заглушки.

Все три уровня модели процесса тестирования, определенной в настоящем стандарте, могут взаимодействовать с менеджментом конфигурации. Элементы конфигурации, предоставленные процессу тестирования, являются рабочими продуктами потребности, необходимыми для процесса в качестве входа, и являются объектами менеджмента конфигурации. Элементы конфигурации, получаемые из процесса тестирования, являются рабочими продуктами, произведенными процессом тестирования, должны быть переданы под контроль менеджмента конфигурации.

Результатами Организационного Процесса Тестирования могут, например, быть Политика Тестирования и Организационная Стратегия Тестирования, которые передаются под контроль менеджмента конфигурации. Менеджер тестирования проекта может взять копию Плана Проекта, который является объектом менеджмента конфигурации, и использовать эту копию в качестве базы для Плана Тестирования Проекта, который далее передается под контроль менеджмента конфигурации. Выполняющие подпроцесс динамического тестирования могут взять копию спецификации требований, которая является объектом менеджмента конфигурации, и использовать эту копию в качестве базы для спецификации тестирования, которая далее передается под контроль менеджмента конфигурации.

Для обеспечения возможности воссоздать проблему для дальнейшего анализа рекомендуется (там, где это возможно), чтобы система менеджмента конфигурации была всеобъемлющей и устойчивой для того, чтобы тестирование могло быть повторено при тех же условиях, как прежде. Такая практика приемлема для того, чтобы исключить определенные типы тестирования, например поблочное тестирование, из требования повторяемости при условии, что исключение заявлено в Организационной Стратегии Тестирования или Плане Проекта.

Отчеты менеджмента конфигурации для процесса тестирования должны предоставить подробные результаты, которые могут потребоваться для анализа прогресса и состояния инцидентов.

5.3.3.4 Совершенствование процессов и тестирование

Совершенствование процессов осуществляет меры, направленные на такие изменения процессов организации, которые обеспечат более эффективное и результативное достижение бизнес-целей организации.

Процессы тестирования и процесс совершенствования процессов взаимодействуют двумя способами:

1 Процессы тестирования предоставляют информацию, на которой могут базироваться действия совершенствования процессов (помимо информации, полученной из других источников).

2 Сами процессы тестирования могут быть объектами совершенствования процессов.

В случае предоставления совершенствованию процессов информации тестированием обычно имеет место взаимодействие между процессом менеджмента тестирования, примененного на уровне проекта, и процессом совершенствования процессов. Тестовые измерения могут быть переданы непосредственно из процессов менеджмента тестирования. Процесс совершенствования процессов может также получить некоторые связанные с тестированием метрики из менеджмента конфигурации, если он присутствует и охватывает контроль изменений.

Совершенствование процессов тестирования может осуществляться в рамках улучшения всей организации или может быть в рамках совершенствования процессов тестирования независимо от всей организации. В любом случае для того, чтобы определить, какие процессы тестирования должны быть улучшены, как это можно сделать и какими должны быть усовершенствованные процессы, процессы совершенствования процессов должны получать информацию из разных источников. Результатом будут новые версии каждого из отобранных процессов тестирования, которые затем должны быть развернуты для соответствующих пользователей. Идентифицированные улучшения могут быть применены только для конкретного проекта или могут быть реализованы для всех проектов организации. Необходимо контролировать новые процессы тестирования, чтобы оценить, производят ли они ожидаемый эффект.

### 5.4 Тестирование на базе рисков

Исчерпывающая проверка программной системы невозможна, поэтому тестирование представляет собой действия выборочного обследования. Для подбора надлежащей выборки тестирования имеется множество понятий тестирования (например, практика, методы и типы), которые определяются и обсуждаются в настоящем стандарте. Ключевая предпосылка настоящего стандарта состоит в проведении оптимального в рамках заданных ограничений и контекста тестирования, с использованием основанного на рисках подхода.

Этот подход обеспечивает определение относительной стоимости различных стратегий тестирования с точки зрения снижаемых рисков для заинтересованных в завершенной системе сторон и с точки зрения разрабатывающей систему стороны. Выполнение тестирования на базе рисков обеспечивает во время тестирования наиболее тщательный анализ рисков с самым высоким приоритетом. В определении рисков функционирующих систем могут быть полезны другие стандарты, например ИСО/МЭК 16085 «Менеджмент рисков».

Риски могут быть классифицированы многими способами. Например, риски могут быть идентифицированы с точки зрения неудовлетворения нормативных и/или законных требований, невыполнения договорных обязательств, связанного с неуспешным ходом и завершением проекта (риски проекта), и необеспечения ожидаемого поведения рабочего продукта (риски продукта).

При проведении тестирования на базе рисков анализ рисков используется для их идентификации и ранжирования таким образом, чтобы выявленные в разрабатываемой системе риски (и в разработке этой системы) могли быть ранжированы, выстроены по приоритету и классифицированы, а впоследствии снижены.

#### 5.4.1 Использование тестирования на базе рисков в Организационном Процессе Тестирования

Организационная Политика Тестирования устанавливает контекст проведения тестирования в организации. При этом необходимо идентифицировать организационные риски, которые могут повлиять на процесс тестирования. Например, организации, производящей медицинское программное обеспечение, вероятно, придется соответствовать строгим стандартам качества. Организационная Политика Тестирования для этой организации может включать в себя требования соответствия связанным с безопасностью стандартам и таким образом попытаться смягчить воздействие того, что не удается реализовать в проекте.

Организационная Стратегия Тестирования должна учитывать подход к менеджменту рисков в процессе тестирования. В вышеупомянутом примере организации, производящей медицинское про-

граммное обеспечение, Организационная Стратегия Тестирования может разработать руководящие материалы о том, каким образом в организации должно выполняться тестирование, чтобы обеспечивать управление риском несоответствия нормативным стандартам. В дополнение к этому Организационная Стратегия Тестирования может предписать использование конкретного подхода к применению менеджмента рисков для использования в процессах менеджмента тестирования.

**5.4.2 Использование тестирования на базе рисков в процессах менеджмента тестирования**

Классифицированный профиль риска (идентифицированный набор рисков и их рангов) используется для того, чтобы определить, какое тестирование должно применяться в проекте. Это записано в стратегии тестирования, которая является частью плана тестирования. Оценки рисков, например, могут использоваться для определения строгости тестирования (подпроцессов тестирования, методов проектирования тестирования, критериев завершения тестирования). Расстановка приоритетов рисков может использоваться для формирования графика тестирования (Тестирование, связанное с высокоприоритетными рисками, обычно выполняется раньше). Тип риска может быть использован для выбора выполнения наиболее подходящей формы тестирования. Например, если имеется риск, выявленный в интерфейсах между компонентами, то необходимо интеграционное тестирование, а если есть выявленный риск того, что пользователи могут испытывать затруднения в использовании приложения, то необходимо выполнить тестирование удобства использования.

Хорошей практикой является привлечение к этим действиям максимально широкого круга заинтересованных сторон для обеспечения согласования идентификации максимального числа рисков и соответствующих действий по их смягчению (или, возможно, отказа от учета некоторых рисков).

Преимущество использования этого подхода состоит в том, что в любой момент риск поставки может быть просто доведен до заинтересованных сторон как остаточный риск.

Результаты выполнения тестирования выявленных рисков и также развитие бизнеса естественным образом приводят к изменениям с течением времени профиля риска, а, следовательно, тестирование на базе рисков необходимо рассматривать в качестве длительной деятельности.

**5.4.3 Использование тестирования на базе рисков в процессах динамического тестирования**

Профиль рисков используется для осуществления руководства всеми процессами динамического тестирования. В процессе проектирования тестирования риски продукта используются для того, чтобы подсказать тестеру, применение какого метода проектирования тестирования наиболее целесообразно. Он (профиль рисков) может быть использован для определения того, сколько раз необходимо произвести анализ тестирования в ходе процесса проектирования тестирования с приложением больших усилий в областях с более высокими рисками, чем в областях с низким риском. Риск может также быть использован для определения приоритетов наборов функций и, как только они получены, тестовых условий и контрольных примеров.

Выполнение динамического тестирования — это деятельность по снижению рисков. Выполнение контрольных примеров снижает риск, поскольку если тест пройден, то и вероятность появления риска обычно ниже и, таким образом, оценка риска снижается. Аналогичным образом, если тест завершился неудачей, то риск может увеличиться. Процесс отчетности об инцидентах тестирования может быть в таких случаях использован для определения того, привел ли неработающий контрольный пример к разрешению проблемы или требуется дальнейшее расследование.

**5.5 Подпроцесс тестирования**

Проект тестирования обычно представляет собой структуру, состоящую из некоторого числа подпроцессов тестирования, опирающихся на стратегию тестирования проекта. Подпроцесс тестирования может быть связан с фазой жизненного цикла программного обеспечения и/или фокусироваться на определенном атрибуте качества. В число подпроцессов тестирования могут быть включены как статическое тестирование, так и динамическое тестирование.

Каждый подпроцесс управляется посредством применения к нему процессов менеджмента тестирования. Подпроцесс тестирования начинается с планирования тестирования. Деятельность по мониторингу и управлению тестированием непрерывно осуществляется в ходе всего тестирования, запланированного для подпроцессов тестирования, а в соответствующих случаях информация, полученная из мониторинга, может послужить основанием для пересмотра планирования.

Планирование тестирования включает в себя идентификацию цели тестирования, области применения тестирования и рисков, связанных с рассматриваемым подпроцессом тестирования. Результат планирования руководит стратегией подпроцесса тестирования, включая статическое тестирование и динамическое тестирование, запланированные в подпроцессе тестирования. В отдельных

случаях то, что обрисовано в общих чертах в стратегии тестирования проекта, может быть использовано непосредственно в стратегии подпроцесса тестирования, но в других случаях необходимы конкретные решения для формирования подпроцесса тестирования на основе определенных рисков, которые должны быть учтены.

При определении подпроцесса тестирования важно отметить, какие функции элемента тестирования должны быть проверены, а какие нет. Этим обеспечивается четкость и правильность намерений относительно области применения тестирования.

Подпроцесс тестирования, обычно, включает в себя не один раунд динамического тестирования или статического тестирования. В случае если определено только по одному раунду каждого типа тестирования, то может возникнуть необходимость повторить раунд, например, для того, чтобы удовлетворить критериям завершения тестирования. Повторные процессы динамического тестирования обычно называются повторным тестированием и регрессионным тестированием. Повторное тестирование и регрессионное тестирование может быть выполнено для определения того, что изменения, внесенные в рабочий продукт для устранения дефектов, решили проблемы и не привели к возникновению новых дефектов. Дополнительная информация приводится в 5.5.5. Пример общего подпроцесса тестирования показан на рисунке 10.

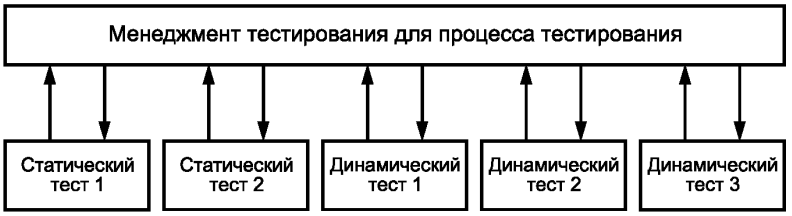


Рисунок 10 — Пример общего подпроцесса тестирования

Число подпроцессов тестирования в проекте тестирования зависит от стратегии тестирования и фаз жизненного цикла, определенных для всего проекта. Оно не зависит от жизненного цикла разработки (то есть последовательный или эволюционный цикл разработки не определяет требуемое число подпроцессов тестирования).

Примеры подпроцессов тестирования детально представлены в приложении D.

Цель тестирования, элемент тестирования, базис тестирования и риски специфичны для подпроцесса тестирования и определяют выбор тестирующих действий, выполняемых в подпроцессе тестирования, а также применяемые методы проектирования тестирования. Примеры целей тестирования, элементов тестирования, базиса тестирования и методов проектирования тестирования представлены далее.

**5.5.1 Цели тестирования**

Тестирование выполняется, чтобы достигнуть одной и большего числа целей. Цели тестирования, охватываемые настоящим стандартом, включают в себя:

- предоставление информации для действий менеджмента рисков;
- предоставление информации о качествах продукта;
- оценку того, оправдал ли продукт надежды заинтересованной стороны;
- оценку того, были ли дефекты корректно устранены без неблагоприятных побочных эффектов;
- оценку корректной реализации изменений без неблагоприятных побочных эффектов;
- оценку выполнения требований (то есть нормативных, проектных, договорных и т. д.).

Тестирование завершается в случае достижения цели тестирования для функции или набора функций. Тип проверяемой функции определяет, какого рода тестирование необходимо. Функции имеют показатели качества, которые должны соответствовать требованиям. Состав показателей качества для функции или набора функций позволяет тестеру определять, какие типы тестирования могут быть использованы для достижения цели тестирования.

Вполне вероятно, что лишь некоторые цели тестирования приемлемы для конкретных подпроцессов тестирования или типов элементов тестирования. Определение соответствующих целей тестирования для элемента тестирования может помочь в выборе применения корректных подпроцессов тестирования. Например, при тестировании готового коммерческого продукта цель тестирования для оценки того, что дефекты устранены, не может быть принята, поскольку ожидается, что поставщик уже завершил надежное тестирование для поиска и устранения дефектов. Поэтому в таких условиях умест-

но использовать подпроцесс приемочных испытаний для достижения цели тестирования, то есть удостовериться в том, что изменения реализованы без неблагоприятных побочных эффектов.

**5.5.2 Элемент тестирования**

Тестирование выполняется на элементе тестирования против того, что ожидается от элемента тестирования. Это ожидание в соответствии с 5.5.4 называется базисом тестирования. Элемент тестирования — это результат действия, такого процесса, как менеджмент, разработка, обслуживание, самого тестирования или другого процесса поддержки.

Примеры элементов тестирования включают в себя:

- связанные с кодом элементы тестирования:
- исполнимый компонент программного обеспечения;
- подсистему;
- полную систему;
- связанные с документацией элементы тестирования:
- план, например план проекта, план тестирования или план менеджмента конфигурации;
- спецификацию требований;
- архитектурный проект;
- детальный проект;
- исходный код;
- руководство, например руководство пользователя или руководство по установке;
- спецификации тестирования и процедуры тестирования.

Последний элемент вышеприведенного списка означает, что, несмотря на то, что спецификации тестирования и процедуры тестирования разрабатываются тестерами, они также должны быть подвергнуты тестированию (статическому тестированию), поскольку они также могут быть не без дефектов.

**5.5.3 Тестирование показателей качества**

ИСО/МЭК 25010 «Модели качества систем и программных продуктов» определяет модель качества программного обеспечения. Эта модель включает в себя восемь показателей качества, которые определяют атрибуты качества элемента тестирования. Тестирование представляет собой действие, которое измеряет важные показатели качества конкретного элемента тестирования. К этим восьми показателям качества относятся:

- функциональная пригодность: степень, с которой продукт или система обеспечивают выполнение функции в соответствии с заявленными и подразумеваемыми потребностями при использовании при указанных условиях;
- уровень производительности: производительность относительно суммы использованных при определенных условиях ресурсов;
- совместимость: способность продукта, системы или компонента обмениваться информацией с другими продуктами, системами или компонентами и/или выполнять требуемые функции при совместном использовании одних и тех же аппаратных средств или программной среды;
- удобство использования: степень, с которой продукт или система могут быть использованы определенными пользователями для достижения конкретных целей с эффективностью, результативностью и удовлетворенностью в заданном контексте использования;
- надежность: степень выполнения системой, продуктом или компонентом определенных функций при указанных условиях в течение установленного периода времени;
- защищенность: степень защищенности информации и данных, обеспечиваемая продуктом или системой путем ограничения доступа людей, других продуктов или систем к данным в соответствии с типами и уровнями авторизации;
- сопровождаемость: результативность и эффективность, с которыми продукт или система могут быть модифицированы предполагаемыми специалистами по обслуживанию;
- переносимость: степень простоты эффективного и рационального переноса системы, продукта или компонента из одной среды (аппаратных средств, программного обеспечения, операционных условий или условий использования) в другую.

Для проверки показателя качества может потребоваться реализация подпроцесса тестирования. Например, планирование и выполнение тестирования для измерения показателя качества защищенности могут потребовать реализации подпроцесса тестирования защищенности (тестирования защищенности).

Каждый из вышеперечисленных показателей качества имеет ряд дочерних характеристик, которые для обеспечения полного представления показателя качества могут быть протестированы. Следует также помнить, что не все показатели качества применимы ко всем системам, например пере-

носимость не может быть важна для одноразовой встроенной системы. Необходимо обратить внимание на то, что приведенный выше перечень показателей качества не всегда является исчерпывающим, в отдельных случаях может потребоваться определение соответствующих дополнительных показателей качества для конкретного элемента тестирования.

В практике тестирования принято разделение тестирования на тестирование функционального показателя качества, называемое «функциональным тестированием», и тестирование других показателей качества, называемое «нефункциональным тестированием». Тип тестирования, используемого для определения показателя качества, отличного от функциональной пригодности, обычно называют нефункциональным типом тестирования и к нему можно отнести такие типы тестирования, как нагрузочное тестирование, стрессовое тестирование, тестирование на возможность проникновения, тестирование удобства использования и т. д.

При разработке Плана Тестирования менеджер по тестированию должен учесть все показатели качества. Акцент, сделанный на тестирование любого отдельного показателя качества и его дочерних характеристик, возможно, изменится в зависимости от таких аспектов, как:

- профиль риска разрабатываемой системы; например, критичное к защищенности приложение может требовать особой надежности;
- отрасль промышленности, для которой разрабатывается система; например, банковское приложение может требовать особой защищенности.

**5.5.4 Базис тестирования**

В настоящем стандарте термин «Базис тестирования» использован для совокупности знаний (в любой форме), исходя из которой может планироваться, разрабатываться, прогнозироваться, реализовываться и управляться тестирование элемента. Базис тестирования может включать в себя выбор режима тестирования, критерии успешного/неуспешного прохождения, входы, среду, практики и методики. Помимо прочего, природа базиса тестирования варьируется по фазам жизненного цикла разработки.

Примерами базиса тестирования являются:

- ожидания по формату и содержанию документации, обычно в форме стандартов и/или контрольных списков;
- ожидания потребителя/пользователя по программной системе, новой или уже существующей, обычно спецификаций требований в письменной форме. Они могут быть представлены как функциональные/нефункциональные описания с употреблением глагола «должен», содержащие варианты использования, истории пользователя или другие формы неформально или формально записанные требования. Сюда могут быть включены нормативные требования, которые должны соблюдаться для определенных типов продуктов, например, для критичного к безопасности программного обеспечения для фармацевтической промышленности или для транспортных систем, таких как поезд или самолет;
- опыт тестера или экспертов в другой предметной области по работе с функциями, необходимыми пользователям, или с историей продукта;
- ожидания по прямым и/или косвенным интерфейсам между компонентами программной системы и/или по сосуществованию компонентов программной системы, обычно в форме проекта архитектуры в виде схем и/или формального письменного определения протокола;
- ожидания по реализации компонентов программной системы в коде, обычно в форме детального проекта.

Следует обратить внимание на то, что один и тот же элемент может быть элементом тестирования в одном подпроцессе тестирования и базисом тестирования в другом подпроцессе тестирования. Спецификация требований может, например, быть элементом тестирования подпроцесса статического тестирования и базисом тестирования дальнейшего подпроцесса тестирования системы.

Требования можно разделить на две основные категории, а именно:

- функциональные требования — определение того, что элемент должен сделать согласно показателю качества функциональной пригодности, представленному в ИСО/МЭК 25010 «Модели качества систем и программного обеспечения»;
- нефункциональные требования — определение того, как хорошо функциональность должна проявиться и вести себя согласно другим показателям качества, представленным в ИСО/МЭК 25010 «Модели качества систем и программного обеспечения». Нефункциональные требования частично или полностью связаны с функциональностью, и обычно функциональные требования связаны с соответствующими группами нефункциональных требований или с отдельными из них.

**5.5.5 Повторное и регрессионное тестирование**

Повторное тестирование проверяет, исправило ли разрешение инцидента первичную проблему. Повторное тестирование часто выполняется простым повтором контрольного примера, который при-

вел к неожидаемому результату и который в свою очередь привел к идентификации первичной проблемы. Однако для эффективного повторного тестирования могут быть определены и проанализированы новые тестовые условия, а также разработаны новые контрольные примеры.

Регрессионное тестирование представляет собой выборочное тестирование ранее протестированной системы или компонента, для того чтобы удостовериться, что изменения не вызвали непреднамеренных побочных эффектов и что система или компонент все еще соответствуют своим исходным требованиям. Цель регрессионного тестирования состоит в том, чтобы установить факт того, что изменения не привели к возникновению дефектов в неизмененных частях системы.

При планировании тестирования необходимо учесть как регрессионное тестирование, так и повторное тестирование, поскольку обычно для завершения подпроцесса тестирования требуется и то и другое. Необходимо в графике тестирования предусмотреть время для обоих действий.

**5.5.6 Методика проектирования тестирования**

Существуют методики проектирования тестирования как для статического, так и для динамического тестирования. Цель методики проектирования тестирования заключается в оказании помощи тестерам элементов тестирования в максимально эффективном и продуктивном нахождении дефектов. В статическом тестировании это обычно достигается идентификацией очевидных дефектов («проблем») в документальных элементах тестирования или аномалий в исходном коде. Динамическое тестирование выявляет дефекты, вызывая отказы исполнимых элементов тестирования.

5.5.6.1 Методы проектирования статического тестирования

Статическое тестирование может включать в себя различные действия, например статический анализ кода, анализ прослеживаемости перекрестных ссылок документа, анализ для поиска проблем и предоставления другой информации о программных продуктах.

Статическое тестирование является одной из форм тестирования. Оно является частью стратегии тестирования и как таковое попадает в область применения настоящего стандарта. Существуют стандарты, где описаны методы, которые можно использовать для выполнения статического тестирования. В данном разделе рассматривается понятие статического тестирования и его связь с другими стандартами.

Методики анализа и процессы, используемые в разработке программного обеспечения, определены в ИИЭР 1028:2008. Действия верификации и валидации описаны в ИИЭР 1012 (см. приложение А). Основная цель любого из методов проектирования статического тестирования состоит в том, чтобы найти дефекты, но у них есть и вторичные цели, например сквозной контроль может использоваться для обмена знаниями, а технический анализ — для достижения консенсуса. Инспекция помимо прочего может служить цели предотвратить будущие дефекты. Выбор типа(ов) методики проектирования статического тестирования в любой конкретной ситуации зависит не столько от типа элемента тестирования, сколько от учитываемых рисков (чем выше риск, тем более формальная методика проектирования статического тестирования рекомендуется) и вторичных целей методики проектирования статического тестирования.

Хотя статическое тестирование часто необходимо в рамках выполнения тестирования программного обеспечения проекта или организации, оно рассмотрено только в серии стандартов ИСО/МЭК/ИИЭР 29119 «Тестирование программного обеспечения» (более подробно в настоящем стандарте) и в вышеупомянутых публикациях. Методы и процессы статического тестирования не рассматриваются в других стандартах серии ИСО/МЭК/ИИЭР 29119 «Тестирование программного обеспечения».

5.5.6.2 Методы проектирования динамического тестирования

Методы проектирования тестирования для динамического тестирования — это методы идентификации тестовых условий, элементов тестового покрытия и впоследствии контрольных примеров, которые будут выполняться на элементе тестирования. Методы проектирования динамического тестирования классифицированы на три основные категории на основе того, как получаются входные данные для тестирования. Эти категории методов основаны на спецификации, структуре и опыте. В ИСО/МЭК/ИИЭР 29119-4 подробно описан каждый из методов проектирования динамического тестирования.

Основанные на спецификации методы проектирования тестирования используются для получения контрольных примеров из базиса тестирования, определяющего ожидаемое поведение элемента тестирования. При использовании этих методов входные данные для тестирования контрольного примера и ожидаемый результат получаются из базиса тестирования. Выбор, какие из основанных на спецификации методов проектирования тестирования использовать в каждой конкретной ситуации, зависит от природы базиса тестирования и/или элемента тестирования, и от присущих рисков. Примерами основанных на спецификации методов проектирования тестирования, охватываемых

ИСО/МЭК/ИИЭР 29119-4, являются Анализ Граничных Значений, Тестирование Изменения Состояния и Тестирование Таблицы Решений.

Основанные на структуре методы проектирования тестирования используются для получения контрольных примеров из структурной характеристики, например структуры исходного кода или структуры меню. Если эти методы применяются к исходному коду приложения, то ожидаемые результаты для контрольных примеров получаются из базиса тестирования. Выбор, какие из основанных на структуре методов проектирования тестирования использовать в каждом конкретном случае, зависит от природы базиса тестирования и от присущих рисков. Эти методы определены и подробно описаны в ИСО/МЭК/ИИЭР 29119-4 «Методы тестирования». Примерами основанных на структуре методов проектирования тестирования, охватываемых ИСО/МЭК/ИИЭР 29119-4, являются Тестирование Ветвей, Тестирование Условия и Тестирование Потока Данных.

### 5.6 Методики тестирования

#### 5.6.1 Введение

Основанный на рисках подход к тестированию, представленный в 5.4, широко применяется и является с точки зрения серии стандартов ИСО/МЭК/ИИЭР 29119 фундаментальным подходом. Существует множество различных методов планирования и реализации тестирования проектов. Традиционная практика была вынуждена базироваться на тестировании выполнения требований получать перед выполнением теста контрольные примеры вручную и использовать смесь ручного и автоматизированного управления выполнением теста. Использование тестирования на базе рисков не означает, что эти практики нельзя использовать в планировании тестирования, претендующего на соответствие с ИСО/МЭК/ИИЭР 29119-2. Выбор стратегии тестирования определяется множеством рисков, таких как риски организации, риски проекта и элемента тестирования. Например, организация может подвергаться риску нарушения контракта, если она не гарантирует, что проверяется каждое требование. Поэтому применение основанной на требованиях практики может быть способом управления этим организационным риском. В данном разделе представлен ряд методик тестирования, каждая из которых может использоваться как составная часть стратегии тестирования, создаваемой в соответствии с ИСО/МЭК/ИИЭР 29119-2. Как правило, изолированное применение любой из этих методик тестирования маловероятно, и она может использоваться в качестве составной части большей стратегии тестирования.

Этот раздел описывает применяемые в настоящее время различные методики тестирования для того, чтобы продемонстрировать некоторые возможности тестирования, доступные во время планирования тестирования.

#### 5.6.2 Основанное на требованиях тестирование

Главная цель основанного на требованиях тестирования состоит в обеспечении гарантии, что во время тестирования требования к элементу тестирования были рассмотрены (то есть «покрыты») для определения того, отвечает ли элемент тестирования требованиям конечного пользователя. Это тестирование используется также для сбора и предоставления заинтересованным сторонам другой ценной информации жизненного цикла, такой как идентифицированные в элементе тестирования дефекты. При применении этой практики требования используются для того, чтобы предоставлять информацию, необходимую для принятия решений по планированию, проектированию и реализации тестирования и выполнению процессов. Следует отметить, что основанное на требованиях тестирование может быть, в частности, использовано для получения результатов верификации требований, определенной в ИСО/МЭК 12207.

В основанном на требованиях тестировании преимущественно используется тестирование по сценарию. Контрольные примеры пишутся перед выполнением теста в ходе процесса Разработки и Реализации Тестирования. Далее контрольные примеры выполняются в ходе процесса Выполнения Теста согласно расписанию, определенному в Процедуре Тестирования. Затем производится анализ результатов выполнения теста для последующего усовершенствования тестирования, которое может потребовать дополнительной работы по проектированию тестирования и сценария контрольного примера. Контрольные примеры, создаваемые в ходе такого дополнительного проектирования тестирования, документируются и проводятся далее в жизненном цикле выполнения теста.

Основанное на требованиях тестирование может поддерживаться другими методиками тестирования, если эти методики помогают продемонстрировать, что требования тестируются надлежащим образом. Для обеспечения гарантии выполнения всех требований дополнительно к использованию основанного на требованиях тестирования с целью разрешения других рисков могут быть применены и другие методы (например, тестирование на базе опыта). Необычным является требование отсутствия

регрессивных дефектов, оставшихся в поставленном продукте. Однако в этом случае можно применить исследовательское тестирование как способ разрешения этого риска регрессии.

Целесообразность применения основанного на требованиях тестирования зависит от контекста. Если требования не полны или не согласованы, то и результирующее тестирование может иметь те же недостатки. Даже если требования хорошо определены, есть опасность того, что бюджетные ограничения и ограничения времени могут не позволить проверить все требования. В случае если требования определены с дополнительной информацией об их относительных приоритетах, эта информация может быть использована для определения приоритетов тестирования (в этом случае основанный на рисках подход может быть использован для назначения более высокого приоритета высокоприоритетному требованию). На практике тестеры при выполнении основанного на требованиях тестирования часто используют дополнительную информацию таким образом, чтобы самые важные (соответствующие самым высоким рискам) требования были протестированы наиболее тщательно и раньше других.

**5.6.3 Модельное тестирование**

В тестировании широко используется понятие модели, представляющей ожидаемое поведение элемента тестирования, которое является базисом тестирования. Эта модель может быть представлена в форме требований на естественном языке, ментальных образов, математических уравнений, графических нотаций (например, диаграммы изменения состояний, диаграммы UML) или матрицы (например, таблицы решений), или совокупностью этих форм. Традиционно тестер использует модель, чтобы вручную получить входы и ожидаемые результаты тестирования при выполнении тестирования на основе спецификации, а также ожидаемые результаты при выполнении тестирования на основе структуры (входы тестирования в этом случае получаются в результате анализа структуры элемента тестирования). Далее тестирование выполняется либо вручную, либо с использованием инструментов тестирования.

Модельное тестирование использует принципиально иную практику, хотя в ее основании лежит модель ожидаемого поведения. Особенность модельного тестирования заключается в требовании от модели формальности и подробности, достаточных для получения из модели необходимой для тестирования информации. Подобная информация может включать в себя планы тестирования, проект, процедуры, входы, риски и/или прогнозы. Преимущества использования модельного тестирования заключаются в генерации информации тестирования, повышенном уровне автоматизации в жизненном цикле тестирования (от планирования до документирования) и ранней идентификации некоторых видов ошибок. Следствием повышенного уровня автоматизации является факт того, что за относительно короткое время могут быть автоматически сгенерированы, выполнены и проверены миллионы контрольных примеров.

Типичными для использования модельного тестирования условиями являются важные приложения, отказ от которых может привести к большому ущербу и требуемое поведение которых поддается моделированию в форме, позволяющей использовать инструменты модельного тестирования (нужно также наличие необходимого для создания и поддержки модели квалифицированного персонала). Графическая нотация UML часто используется в качестве базы этих моделей, хотя используются и другие нотации. Другим соображением в пользу использования модельного тестирования является то, что при эксплуатации предполагается сопровождение приложения на регулярной основе, а изменение модели проще, чем обновление автоматизированных сценариев тестирования (каждый раз после обновления модели инструмент модельного тестирования может далее генерировать и запускать множество тестов при относительно низкой стоимости).

Использование модели может обеспечить преимущества и в других областях. Например, автоматическая генерация исходного кода на основе модели может уменьшить количество дефектов определенных типов, создаваемых во время разработки (например, опечаток в коде и дефектов, которые могут быть обнаружены модельной проверкой). Это облегчает процесс тестирования за счет сокращения числа дефектов в элементах тестирования, попадающих в тестирование из процесса разработки.

**5.6.4 Математическое тестирование**

Математические методики тестирования могут использоваться для планирования, проектирования, выбора данных и установки входных условий в случае, если возможно достаточно детальное определение пространства входа или выхода элемента тестирования. В своем подходе математические методы используют различные области математики. Эти методы могут уменьшить величину субъективной систематической ошибки выбора контрольного примера и назначения приоритетов.

На практике в математическом тестировании используется множество методик, однако наиболее часто используют методики проектирования тестирования, описанные в ИСО/МЭК/ИИЭР 29119-4:

- комбинаторное тестирование;
- случайный выбор контрольного примера.

Дополнительно статическое моделирование может использоваться для определения тестового покрытия (не рассматривается в ИСО/МЭК/ИИЭР 29119-4), например:

- типологическая выборка;
- Fuzz-тестирование;
- кластерная выборка;
- выборка экспертной оценки.

Математические методики тестирования могут также использоваться для объективной выборки на базе математических инструментов и методов из пространства контрольных примеров.

Для использования математических методик тестирования в общем случае необходимы автоматизированные инструменты для обработки большого количества сгенерированных входов.

**5.6.5 Основанное на опыте тестирование**

Основанное на опыте тестирование базируется на следующих аспектах:

- предыдущий опыт тестирования;
- знание определенного программного обеспечения и систем;
- знание проблемной области;
- метрики из предыдущих проектов (внутри организации и в отрасли).

В ИСО/МЭК/ИИЭР 29119-4 описан основанный на опыте метод проектирования тестирования «Предположение об ошибках». Другие методики проектирования тестирования, основанные на опыте, включают в себя «Исследовательское тестирование», «Программные атаки» и «Свободное тестирование», но не ограничены этими методиками. «Исследовательское тестирование» и «Программные атаки» не входят в ИСО/МЭК/ИИЭР 29119-4 в качестве методов проектирования тестирования, поскольку эти методики допускают использование различных методов проектирования тестирования.

Исследовательское тестирование сочетает в себе действия, описанные в ИСО/МЭК/ИИЭР 29119-2 как процессы «Разработки и Реализации Тестирования» и процессы «Выполнения Теста» для динамичного достижения целей тестирования. Обычно при использовании исследовательского тестирования контрольные примеры не разрабатываются и не документируются заранее, а в решении, что и как должно тестироваться следующим, тестер руководствуется интуицией, любознательностью и результатами предыдущих тестирований.

Исследовательское тестирование, предположение об ошибках и свободное тестирование — это методики тестирования, которые не связаны с большими объемами документации (например, процедурами тестирования), необходимой для выполнения тестирования. Что касается использования сценария, то в своей большей части методики, основанные на опыте тестирования, используются без предварительного сценария. Использование таких методов может обеспечить только адаптированное соответствие ИСО/МЭК/ИИЭР 29119-2.

**5.6.6 Тестирование по сценарию и тестирование без сценария**

В таблице 1 приведены преимущества и недостатки тестирования по сценарию и тестирования без сценария. Необходимо отметить, что в проекте эти два метода не исключают друг друга, а чаще всего соединяются в гибридной методике на базе уровня риска элемента тестирования.

Основным фактором при принятии решения об использовании тестирования по сценарию, без сценария или гибрида того и другого является профиль риска элемента тестирования. Например, на практике гибридная методика может использовать в одном и том же проекте тестирование по сценарию для проверки элементов тестирования с высоким риском и тестирование без сценария для проверки элементов тестирования с низким риском.

Таблица 1 — Сравнение методов выполнения теста по сценарию и без сценария

| Вид тестирования | Преимущества | Недостатки |
|---|---|---|
| Тестирование по сценарию | Тестирование повторимо; контрольные примеры могут быть выполнены снова, обеспечивая таким образом хорошие возможности для действий верификации и валидации.<br>Контрольные примеры в виде сценария могут быть прослежены в обратном направлении явно до требований, позволяя документировать тестовое покрытие в виде матрицы прослеживаемости. | Обычно более длительно и дороже, чем тестирование без сценария; однако если контрольные примеры в виде сценария допускают повторное использование, то это может приводить к экономии средств в течение длительного времени.<br>Контрольные примеры, определенные до выполнения тестирования, в меньшей степени способны адаптироваться к системе как таковой. |

*Окончание таблицы 1*

| Вид тестирования | Преимущества | Недостатки |
|---|---|---|
| Тестирование по сценарию | Случаи тестирований могут быть сохранены в виде допускающих повторное использование артефактов для текущих и будущих проектов, возможно уменьшив время, необходимое для будущей Разработки и Реализации Тестирования | Возможно, меньше стимулов для исполнителей тестирования, поскольку большая часть аналитической работы была завершена заранее. Это может вызвать ослабление внимания тестеров, не учитывающих детали тестирования во время выполнения теста |
| Тестирование без сценария | Тестеры не ограничены сценарием и могут следовать за идеями, возникшими при выполнении теста в режиме реального времени.<br>Тестеры могут адаптировать «Разработку и Реализацию Тестирования» и «Выполнение Тестирования» к поведению системы в режиме реального времени.<br>Элемент тестирования может быть быстро исследован | Тестирование обычно неповторимо.<br>Тестер должен в случае необходимости применить большой спектр методов проектирования тестирования, следовательно, более опытные тестеры обычно обладают большими способностями в обнаружении дефектов, чем менее опытные.<br>Тестирования без сценария не предоставляют записей о ходе тестирования или предоставляют их очень мало; таким образом, без использования инструментальных средств протоколирования трудно оценить процесс выполнения динамического тестирования |

### 5.7 Автоматизация в тестировании

Многие задачи и действия, описанные в ИСО/МЭК/ИИЭР 29119-2 как процессы Менеджмента Тестирования и Динамического Тестирования, а также многие аспекты методов тестирования, представленные в ИСО/МЭК/ИИЭР 29119-4, могут быть автоматизированы. Автоматизация тестирования может использоваться для поддержки одной из методик тестирования, представленных в 5.6 (например, Модельное Тестирование почти всегда базируется на использовании при выполнении теста автоматизированных инструментов). Автоматизация тестирования требует использования программного обеспечения, обычно называемого инструментами тестирования. Несмотря на то что обычно автоматизированным тестированием считается выполнение тестирования по сценарию, а не выполнение тестером теста вручную, многие из дополнительных задач и действий тестирования могут поддерживаться инструментами на базе программного обеспечения. В списке представлены некоторые примеры областей, в которых действуют инструменты тестирования:

- менеджмент контрольных примеров;
- мониторинг тестирования и управления;
- генерация тестовых данных;
- статический анализ;
- генерация контрольных примеров;
- выполнение контрольных примеров;
- реализация и обслуживание тестовой среды;
- тестирование на основе сеанса.

Имеется большое разнообразие доступных инструментов автоматизации тестирования. Они могут разрабатываться в организации или быть получены извне как коммерческий продукт либо как программное обеспечение с открытым доступом.

### 5.8 Управление дефектами

Менеджеры по тестированию проекта часто ответственны за управление дефектами в проекте (также называемое управлением инцидентами). Управление дефектами регламентируется ИСО/МЭК/ИИЭР 29119-2 и используется тестерами в расследовании и документировании инцидентов тестирования, а также в повторном тестировании дефектов в случае необходимости. Другие аспекты управления дефектами не рассмотрены непосредственно в серии стандартов ИСО/МЭК/ИИЭР 29119. Понятия и процессы управления дефектами описаны в ИСО/МЭК 12207, ИСО/МЭК 20000 и ИИЭР 1044.

**Приложение А**
**(справочное)**

## Роль тестирования в верификации и валидации

На рисунке А.1 показана классификация действий верификации и валидации. Верификации и валидации могут быть подвергнуты системы, аппаратные и программные продукты. Эти виды действий определены и детализированы в ИИЭР 1012 и ИСО/МЭК 12207. Большая часть верификации и валидации осуществляется посредством тестирования. Серия стандартов ИСО/МЭК/ИИЭР 29119 рассматривает динамическое и статическое тестирование программного обеспечения (непосредственно или через ссылки), частично охватывая модели верификации и валидации. Исчерпывающее рассмотрение модели верификации и валидации выходит за рамки серии стандартов ИСО/МЭК/ИИЭР 29119, но тестеру важно понимать, как тестирование вписывается в эту модель.

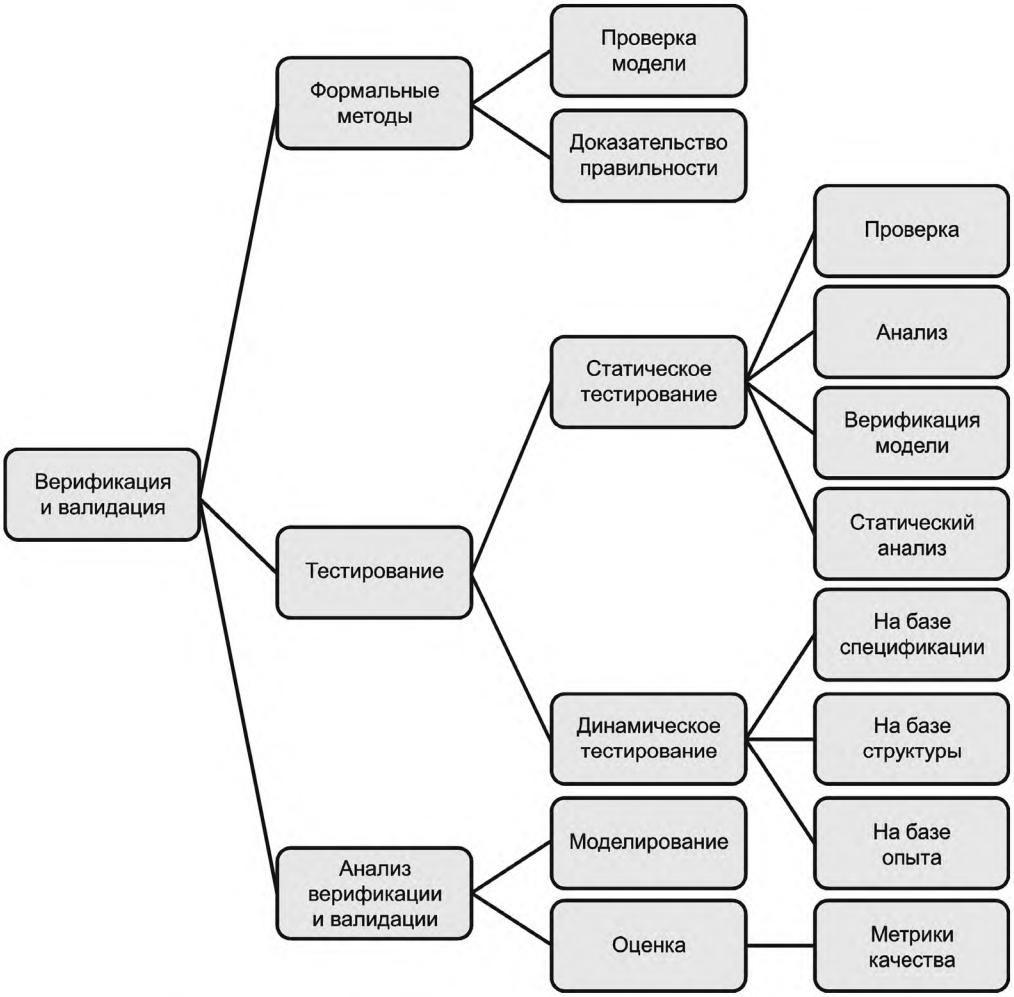


Рисунок А.1 — Иерархия действий верификации и валидации

**Приложение В**
**(справочное)**

## Метрики и показатели

### В.1 Метрики и показатели

Основная цель тестирования заключается в предоставлении информации, необходимой для управления рисками. Чтобы контролировать и управлять тестированием, а также предоставлять своевременную информацию заинтересованным сторонам, необходимы эффективные измерения процесса тестирования. Для измерения процесса тестирования нужно определить, какая информация должна быть предоставлена, как ее получить и как она должна быть представлена. Таким образом, для всех действий тестирования необходимо определить и использовать метрики, а также предоставить показатели измерений, как для продуктов, так и для процессов.

Некоторые примеры метрик, которые могут использоваться в тестировании:

- остаточный риск; число смягченных рисков/число идентифицированных рисков;
- накопленные открытые и закрытые дефекты; число выявленных за день дефектов по сравнению с числом устраненных за день дефектов;
- прогресс контрольных примеров; число выполненных контрольных примеров/число запланированных для выполнения контрольных примеров;
- процент обнаружения дефектов; число дефектов, выявленных в тестировании/общее числе выявленных дефектов (в целом).

**Приложение С**
**(справочное)**

## Тестирование в различных моделях жизненного цикла

### С.1 Общие сведения

Это приложение дает примеры того, как тестирование может вписаться в различные модели жизненного цикла разработки программного обеспечения. Для каждого конкретного проекта идентифицируются необходимые этапы разработки и/или действий, что и определяет, как относительно друг друга они будут выполняться в течение жизненного цикла разработки. Совокупность этапов разработки и их взаимосвязь называют «моделью жизненного цикла разработки» или «моделью жизненного цикла».

В течение многих лет в индустрии программного обеспечения используется ряд моделей жизненного цикла. Типичными моделями жизненного цикла являются (список неисчерпывающий):

- динамичная;
- эволюционная;
- последовательная (т. е. каскадная модель).

Выполняемые во время разработки действия примерно одни и те же для всех моделей жизненного цикла; основные различия заключаются в определении предметов действий, количестве, характере подготавливаемой документации и частоте, с которой они повторяются в течение жизненного цикла разработки.

Примечание — Стандартизация различных моделей жизненного цикла не является целью настоящего стандарта, в намерения которого входит оказание поддержки тем, кто реализует эти жизненные циклы, в создании условий для тестирования.

### С.2 Динамичная разработка и тестирование

#### С.2.1 Принципы динамичной разработки

Динамичная разработка обычно соответствует нижеперечисленным принципам:

- разработка по этапам — результатом каждого цикла являются полезные и применимые продукты;
- цикличность разработки — в ходе разработки допускается развитие требований (то есть изменение и добавление);
- разработка ориентирована на людей — опора на качество проектной группы (например, разработчиков и тестеров), а не на точно определенные процессы; обеспечение самоуправления быстрой и динамичной команды; ежедневное взаимодействие группы разработчиков с бизнес-заинтересованными сторонами;
- инженерно-техническое совершенство — достигается посредством дисциплинированного подхода к разработке, интеграции и тестированию продуктов.

Существует множество методик и схем динамичной разработки, включая: Экстремальное программирование (XP), Scrum, Crystal, Kanbanи Feature-Driven Development. В то время как все они используют различные подходы, они следуют принципам динамичной разработки, изложенным в манифесте динамичной разработки (Agile Manifesto, см. http//:agilemanifesto.org/). Поскольку в рамках настоящего стандарта невозможно рассмотреть его использование при реализации всех методов и схем динамичной разработки, то в качестве примера в настоящем стандарте будет использована схема метода Scrum. Метод Scrum не является методологией разработки (то есть он не предлагает лучших методов описания требований или записи кода), а определен как схема менеджмента проектов, в которой разработчиками используется динамичная методология (часто используется XP).

Проект Scrum состоит из множества итераций, называемых спринтами. Результатом каждого спринта обычно является новая функциональность, которая может быть предоставлена пользователям, как показано на рисунке С.1. Подобная новая функциональность может представлять собой улучшение предшествующей функциональности или дополнение ее новыми возможностями. Продолжительность каждого спринта обычно составляет от одной до четырех недель. Зачастую количество спринтов в начале проекта неизвестно, поскольку в начале типичных динамичных проектах требования не определены полностью. В ходе проекта требования развиваются. Требования потребителя собираются в Портфеле Продукта обычно в виде истории пользователей.

Модель процесса тестирования, определенная в настоящем стандарте, применима для тестирования, производимого в проектах, выполняемых соответственно модели динамичной разработки.

#### С.2.2 Менеджмент тестирования в динамичной разработке

Организационная Стратегия Тестирования должна отражать факт соответствия разработки модели динамичной разработки. В организации, где разработка выполняется с использованием в различных проектах разных моделей разработки, в том числе и динамичной, как правило, имеется определенная Организационная Стратегия Тестирования для динамичной разработки. Организационная Стратегия Тестирования проектов для динамичной разработки должна использовать специфические понятия динамичной разработки, такие как «портфель продук-

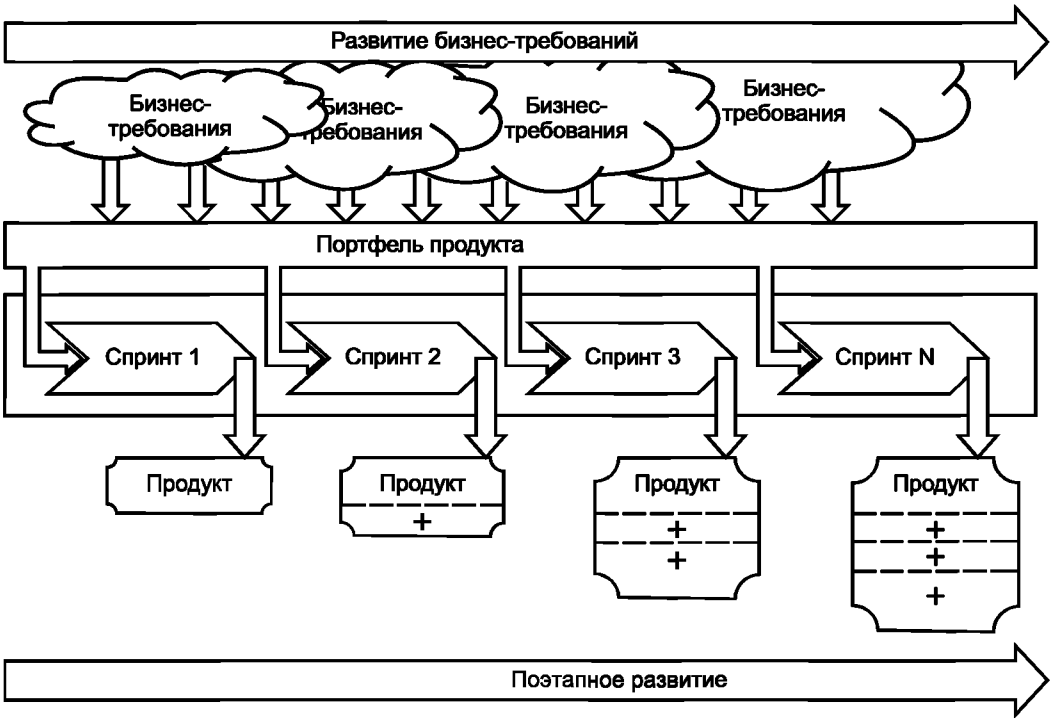


Рисунок С.1 — Проект Scrum в качестве примера жизненного цикла проекта (динамичная разработка)

та», «спринт», «ежедневная встреча», но помимо этого содержание любой Организационной Стратегии Тестирования зависит, прежде всего, от профиля рисков для соответствующих проектов и продуктов (несмотря на это, тип используемой модели разработки может создать дополнительные типы риска, которые также должны быть учтены в Стратегии Тестирования).

Проектом динамичной разработки часто управляет менеджер проектов, а спринты продвигаются мастером Scrum (эти роли выполняются одним и тем же человеком). Менеджмент тестирования в динамичном проекте осуществляется как интегрированная часть управления портфелем продукта, конкретными спринтами и ежедневными встречами.

В начале разработки спринта команда Scrum и потребитель (владелец продукта) приходят к соглашению, какие из историй пользователя из портфеля продукта должны быть реализованы в этом спринте. Отобранные истории включаются в портфель спринта. Затем команда разрабатывает план спринта, планируя разработку и тестирующие действия и присваивая роли и обязанности членам команды. Динамичная разработка осуществляется посредством тех же общих процессов, присущих любой разработке и тестированию продукта, как это показано на примере спринта Scrum на рисунке С.2.

Результат спринта демонстрируется потребителю на презентации спринта, где у команды есть возможность показать заинтересованным сторонам, что они создали. Заключительное действие в спринте — это ретроспектива спринта, в ходе которой команды рассматривают, насколько хорошо спринт выполнен, и определяют, где и какие улучшения для будущих спринтов можно сделать, то есть совершенствование процессов встроено в структуру Scrum.

В ходе спринта в начале каждого дня проводятся ежедневные встречи Scrum, на которых отмечается, что было сделано и делается в этот день, а также с какими проблемами может столкнуться команда. Эти встречи позволяют мастеру Scrum определить, какие препятствия необходимо преодолеть, чтобы обеспечить наиболее эффективно максимальный прогресс команды.

Ключевыми факторами в динамичном проекте являются управление риском дефектов регрессии (поскольку каждый спринт основывается на предыдущих спринтах) и управление изменяющейся природой требований и их влиянием на артефакты тестирования. Обычно для управления риском регрессии используется автоматизация тестирования, а для управления влиянием отсутствия подробных требований может быть применено исследовательское тестирование.

### С.2.3 Подпроцессы тестирования в динамичной разработке

Действия тестирования являются неотъемлемой частью проекта динамичной разработки. Как показано на рисунке С.2, тестирование выполняется на постоянной основе всюду по ходу спринта. Подпроцессы тестирования могут быть определены и выполняться с использованием процессов, представленных в настоящем стандарте, для тестирования историй пользователя и развития разрабатываемой системы.

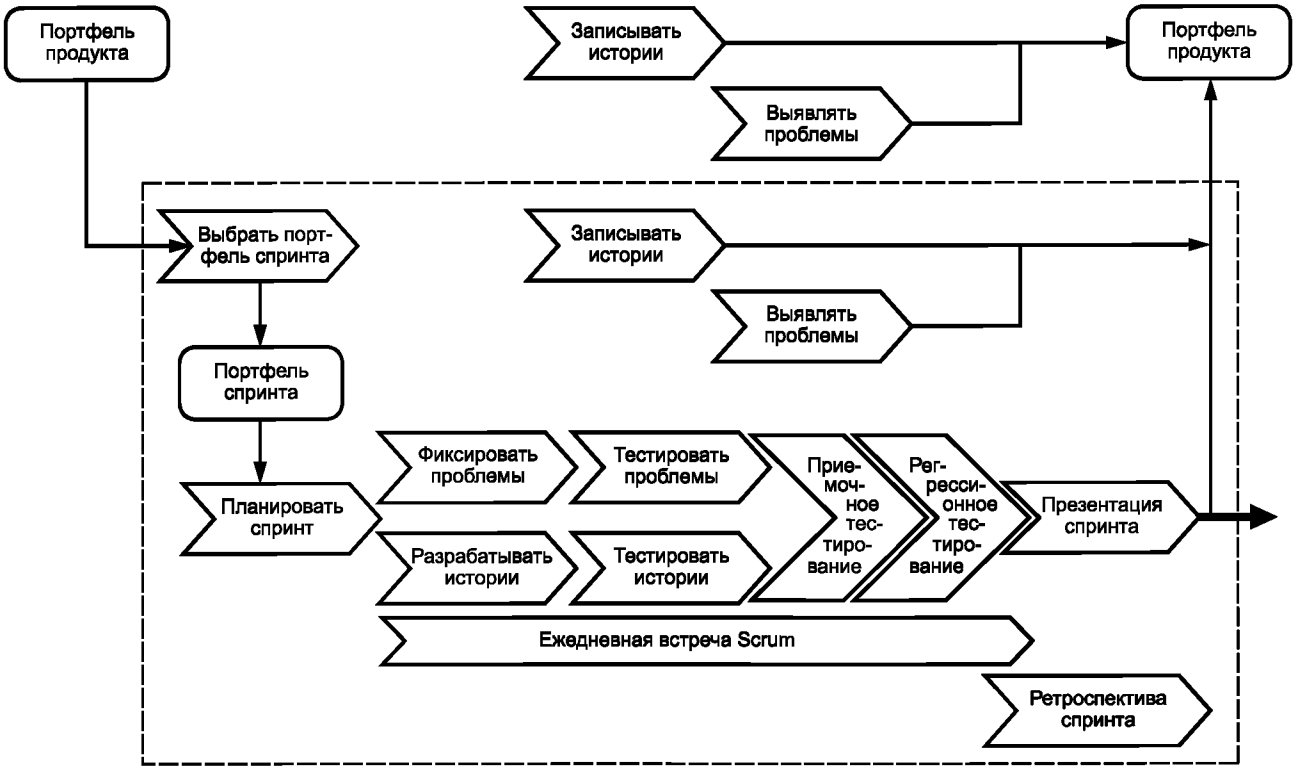


Рисунок С.2 — Пример динамичного жизненного цикла спринта

Типичными методиками тестирования, используемыми командой спринта, являются:

- разработка через тестирование (TDD); это разработка, в которой тесты для кода разрабатываются перед разработкой кода. Тесты базируются на пользовательской истории и могут быть разработаны совместно тестером и разработчиком. Такие тестирования обычно реализуются с использованием автоматизированных инструментов покомпонентного тестирования, что позволяет рассматривать разработку через тестирование как форму программирования;
- тестирование автоматизированных сборок и непрерывная интеграция; это случай, когда система постоянно обновляется и регрессивно тестируется по мере регистрации кода. Используется для обеспечения своевременной идентификации и коррекции проблем регрессии и интеграции;
- тестирование всех показателей качества системы (то есть как функциональных, так и нефункциональных). Выполняется на основании историй пользователя и всех существующих требований высокого уровня. Тестирование системы обычно сопровождается приемочными испытаниями, в которых должны участвовать конечные пользователи для гарантии того, что обеспеченная функциональность удовлетворяет их потребности;
- регрессионное тестирование, как правило, требуется для того, чтобы определить отсутствие у любых изменений в текущем спринте неблагоприятных побочных эффектов на существующие функции и возможности продукта.

В конце «идеального» спринта функции готовы к употреблению пользователями; это означает, что все вышеупомянутое тестирование выполняется в спринте, как показано на рисунке С.2. На практике многие проекты считаются слишком трудными, что приводит к принятию компромиссных вариантов, таких как выполнение тестирования параллельным действием со смещением или выполнение тестирования в специализированном фокусируемом на тестировании спринте.

### С.3 Последовательная разработка и тестирование

#### С.3.1 Последовательные принципы разработки

Последовательные модели жизненного цикла возникли ранее других и широко используются в настоящее время. Базовая (оригинальная) последовательная модель известна как каскадная модель и представляет собой упорядоченную последовательность этапов разработки, предшествующих фазе тестирования, с заключительной операционной фазой в конце.

Последовательная модель жизненного цикла характеризуется отсутствием явного повторения фаз за исключением тех случаев, когда абсолютная необходимость повторения продиктована обратной связью от последующих фаз.

Модель процесса тестирования, определенная в настоящем стандарте, может быть применена к тестированию разработки, соответствующей последовательной модели жизненного цикла.

**С.3.2 Менеджмент тестирования в последовательной разработке**

В Организационной Стратегии Тестирования должно быть отражено, что разработка соответствует последовательной модели разработки. В ведущей разработку организации, если там используется сочетание разных моделей разработки для различных проектов, включая последовательную, обычно существует одна или несколько конкретных организационных стратегии тестирования для используемых моделей разработки. Организационная Стратегия Тестирования должна использовать терминологию, применяемую в соответствующем типе проекта, в противном случае, содержание Организационной Стратегии Тестирования будет зависеть от соответствующего профиля риска для проектов и продуктов, а не от типа используемой модели разработки.

Последовательный проект управляется менеджером проекта. Для большинства проектов также определены роли менеджера по развитию и менеджера по тестированию. В зависимости от размера проекта эти роли могут выполняться различными людьми или несколько ролей могут выполняться одним и тем же человеком.

Планы последовательной разработки создаются для всего проекта, хотя в ходе проекта они должны развиваться. Проекты последовательной разработки могут быть относительно велики, и в начале проекта детальное планирование всего проекта может быть невозможно. Необходима также и разработка Плана Тестирования Проекта. Этот план обычно оформляется в виде отдельного документа, но может быть и частью общего плана проекта. Возможна разработка отдельных планов подпроцессов тестирования при условии, что выполнение этих подпроцессов тестирования предусмотрено Планом Тестирования Проекта. В последовательной разработке планы обычно документируются формально.

**С.3.3 Подпроцессы тестирования в последовательной разработке**

Подпроцессы тестирования, определенные для эволюционной разработки в С.3, имеют также большое значение для тестирования в последовательном проекте, хотя они выполняются только один раз (один проход через последовательную модель разработки).

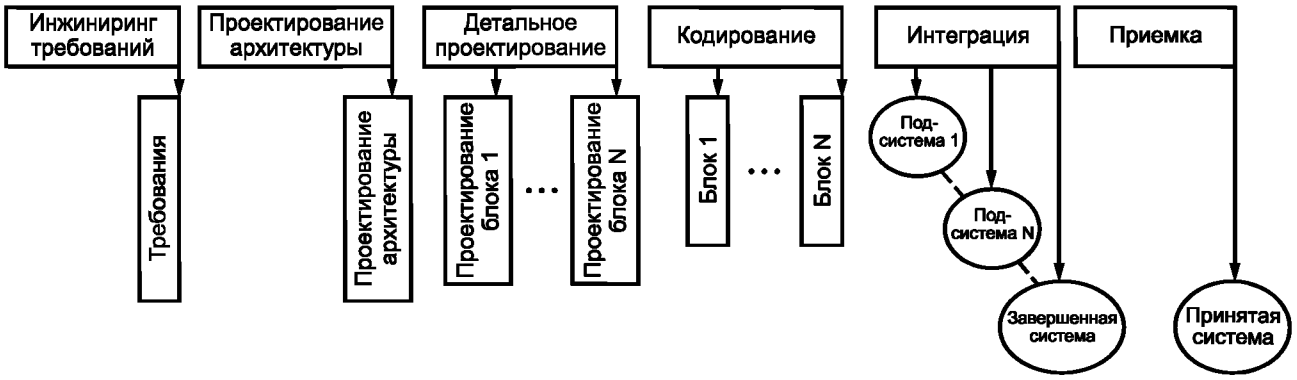


Рисунок С.3 — Пример подпроцесса тестирования в последовательном жизненном цикле разработки

Следует обратить внимание на то, что рисунок С.3 выполнен не в масштабе — относительный размер показанных подпроцессов тестирования не соответствует фактическим размерам подпроцессов тестирования с точки зрения, например, затраченных времени, усилий или средств.

Определяются тестирование Архитектуры, тестирование Детального Проектирования и тестирование Исходного Кода. Для каждого из них соответствующий этап разработки может быть закончен на основе результатов завершения тестирования, то есть после того, как все детально разработанные элементы были проверены индивидуально.

Фаза кодирования включает в себя два различных подпроцесса тестирования. Первым из них является подпроцесс тестирования исходного кода, представляющий собой статическое тестирование исходного кода, а вторым — покомпонентное тестирование, то есть динамическое тестирование исполнимого или интерпретируемого кода. Эти подпроцессы тестирования могут использоваться совместно, и динамическое тестирование компонента может зависеть от успешного статического тестирования исходного кода.

Конечным результатом фазы интеграции является завершенная система. На этом этапе может начинаться выполнение действий тестирования системы, предполагая, что они уже были запланированы и подготовлены на основе требований.

Подпроцессы тестирования, показанные здесь в качестве примера, описаны далее в приложении Е.

**С.4 Эволюционная разработка и тестирование**

**С.4.1 Эволюционные принципы разработки**

Эволюционная разработка основывается на двух основных принципах: итерации и инкрементной разработке. Итерация позволяет разработчикам создавать систему в виде совокупности более мелких частей и для улуч-

шения следующей итерации использовать обратную связь, как от разработанного продукта, так и от методов разработки. Итерация по сравнению с традиционными последовательными циклами позволяет раньше рассмотреть более высокие риски, обеспечивая таким образом возможность увеличения время их анализа. При инкрементной разработке результаты каждой итерации предоставляются потребителю, означая, что в каждой из поставок серии пользователи получают систему с увеличивающейся функциональностью.

Итерация состоит из всех или некоторых стандартных действий разработки. В случае если результат итерации будет поставлен пользователям, итерация должна включать в себя фазу приемки, в противном случае фаза приемки выполняется только в последней итерации.

На рисунке С.3 показан последовательный жизненный цикл разработки. Эволюционный жизненный цикл разработки может быть представлен как множество дискретных последовательных жизненных циклов, каждый из которых добавляет разработанной системе дополнительную функциональность.

Модель процесса тестирования, определенная в настоящем стандарте, может быть применена к тестированию разработки, соответствующей эволюционной модели разработки.

**С.4.2 Менеджмент тестирования в эволюционной разработке**

В Организационной Стратегии Тестирования должно быть отражено, что разработка соответствует эволюционной модели разработки. В ведущей разработку организации, использующей сочетание разных моделей разработки для различных проектов, включая эволюционную, обычно существует одна или несколько конкретных организационных стратегий тестирования для используемых моделей разработки. Организационная Стратегия Тестирования должна использовать терминологию, используемую соответствующим типом проекта, но помимо этого содержание любой Организационной Стратегии Тестирования зависит прежде всего от профиля рисков для соответствующих проектов и продуктов (несмотря на то, что тип используемой модели разработки может создать дополнительные типы риска, которые также должны быть учтены в Стратегии Тестирования).

Эволюционный проект управляется менеджером проекта. Для большинства проектов определены роли менеджера по развитию и менеджера по тестированию. В зависимости от размера проекта эти роли могут выполняться разными людьми или же несколько ролей могут выполняться одним и тем же человеком.

В эволюционной разработке планы создаются как для всего проекта, так и для каждой итерации. План Тестирования Проекта оформляется в виде отдельного документа, однако он может быть и частью общего плана проекта. Для определения тестирования итерации может быть разработан меньший план тестирования итерации. В эволюционной разработке планы могут быть документированы более или менее формально, но, как правило, они поддерживаются инструментами документирования и/или планирования. Планы Тестирования итераций и связанные с ними планы подпроцессов тестирования часто используются повторно с необходимыми от итерации к итерации коррекциями.

Прогресс тестирования контролируется в ходе итерации, необходимые корректирующие действия предпринимаются на постоянной основе и доводятся до заинтересованных сторон в обновленных Планах Тестирования.

**С.4.3 Подпроцессы тестирования в эволюционной разработке**

В ходе каждой итерации могут быть проверены как рабочие продукты, так и завершенная система. Подпроцессы тестирования могут быть определены и выполняться с использованием процессов, представленных в настоящем стандарте, для тестирования исполнимых рабочих продуктов и производимой системы.

Первый этап разработки в этом примере — инженерия требований, где определяются бизнес-требования, функциональные/нефункциональные и системные требования. На рисунке С.3 во избежание загромождения картины показан только подпроцесс тестирования, связанный с первой фазой (фазой тестирования требований). Тестирование требований, поскольку они определены, можно рассматривать как подпроцесс, состоящий из статического тестирования. Формальность этого подпроцесса тестирования зависит от профиля риска для продукта, но поскольку требования являются основой выполнения итераций, то методика проектирования статического тестирования должна быть выбрана более формально.

Подобные подпроцессы тестирования могут быть определены для двух стадий проектирования, а также для этапа кодирования. В примере модели разработки, показанном на рисунке С.3, в подпроцессы тестирования входят:

- тестирование архитектуры;
- тестирование детального проектирования;
- тестирование исходного кода.

Эти подпроцессы тестирования обычно сопровождают большую часть соответствующего этапа разработки, концентрируются на одном типе элемента тестирования, например требованиях, и включают в себя различные способы проверки элемента тестирования, например пошаговый разбор и технический анализ. Подпроцесс тестирования требований не будет завершен до тех пор, пока согласно плану подпроцесса тестирования не будут проверены все представленные требования. Завершение подпроцесса, как правило, является более или менее формальным признаком завершения этапа инженерии требований, однако это зависит от результата подпроцесса тестирования требований.

Таким же образом, не будут завершены подпроцессы тестирования проекта и исходного кода до тех пор, пока согласно плану подпроцесса тестирования не будут проверены все разработанные элементы тестирования, а соответствующий этап разработки будет зависеть от результатов подпроцесса тестирования.

На рисунке С.3 показан подпроцесс приемочного испытания. Действия планирования и подготовки в подпроцессе приемочного испытания могут быть начаты, как только риск существенного изменения в требованиях для данной итерации снизится до степени, оправдывающей запуск подпроцесса тестирования. Проект тестирования будет выявлять дефекты в требованиях и таким образом вносить свой вклад в информационное представление требований. Подобные подпроцессы тестирования могут быть определены и для других этапов разработки, где возможно динамическое тестирование. В примере модели разработки, показанном на рисунке С.3, в состав подпроцессов могли бы входить следующие подобные подпроцессу приемки подпроцессы:

- покомпонентное тестирование;
- интеграционное тестирование;
- тестирование системы.

В примере на рисунке С.3 показано также тестирование производительности. В зависимости от профиля риска для системы можно определить конкретные подпроцессы тестирования для покрытия определенных категорий требований или определенных областей системы, определяемых требованиями. Такие конкретные подпроцессы тестирования могут выполняться на многих этапах разработки, а следовательно, могут иметь множество элементов тестирования и связанных с ними базисов тестирования, обязанностей по тестированию, методов, сред, целей тестирования, критериев завершения и планов. Однако в рассмотренном примере подпроцесс тестирования фокусируется только на требованиях к производительности и то, как они описываются, разрабатываются и реализуются в системе.

Все вышеперечисленные типы тестирования могут быть выполнены для каждой последующей итерации.

В связи с постоянным расширением возможностей продукта для каждой итерации необходимо описанное выше всеобъемлющее регрессионное тестирование. Для каждой итерации должен быть определен подпроцесс регрессионного тестирования. Подпроцесс регрессионного тестирования может включать в себя регрессионное тестирование всех элементов, расширяемых в итерации, включая требования, проект, исходный код и систему, или же выбор элементов может быть сделан в зависимости от профиля риска. Для обеспечения большей динамичности в регрессионном тестировании от итерации к итерации можно для каждого типа элемента определить конкретный подпроцесс регрессионного тестирования.

**Приложение D**
**(справочное)**

## Примеры подпроцессов тестирования в деталях

### D.1 Общие сведения

В данном приложении приводятся примеры подпроцессов тестирования. Здесь приводится всего лишь несколько примеров в отличие от конкретного проекта тестирования, где могут потребоваться другие определенные подпроцессы.

Примеры подпроцессов тестирования:

- приемочные испытания;
- тестирование детального проектирования;
- интеграционное тестирование;
- тестирование производительности;
- регрессионное тестирование;
- повторное тестирование;
- тестирование истории;
- тестирование системы;
- покомпонентное тестирование.

Следует обратить внимание на то, что регрессионное тестирование и повторное тестирование включены как конкретные подпроцессы тестирования. В определенных случаях они могут входить в состав любого другого подпроцесса.

В описание каждого подпроцесса тестирования входят:

- цель подпроцесса тестирования;
- запланированный состав подпроцесса тестирования — статические и/или процессы динамического тестирования, которые будут выполняться.

Для каждого процесса статического или динамического тестирования, запланированного в составе подпроцесса тестирования, описание включает:

- цель тестирования;
- элемент(ы) тестирования;
- базис тестирования;
- применимые процессы тестирования;
- предлагаемую методику (методики) проектирования тестирования, если это применимо.

Следует обратить внимание на то, что представленные примеры являются всего лишь примерами. В каждой фактической ситуации выбор должен осуществляться в соответствии с профилем риска для продукта.

### D.2 Подпроцесс приемочного испытания

Этот пример представляет подпроцесс тестирования, связанный с фазой приемки жизненного цикла разработки.

| | |
|---|---|
| Цель подпроцесса тестирования: | Продемонстрировать потребителю приемлемость завершенной системы с точки зрения указанных им требований. |
| Запланированный состав подпроцесса тестирования: | Динамическое тестирование 1. Генеральная репетиция.<br>Динамическое тестирование 2. Презентация. |

**Динамическое тестирование 1. Генеральная репетиция**

| | |
|---|---|
| Цель тестирования: | Гарантировать, что заключительное выполнение в присутствии потребителя будет успешно. |
| Элемент тестирования: | Завершенная система. |
| Базис тестирования: | Требования пользователя, руководства пользователя, документация бизнес-процессов. |
| Процессы динамического тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. |
| Метод(ы) проектирования тестирования: | Тестирование по сценарию использования, другие методы в зависимости от природы требований. |

Разработка и реализация контрольных примеров могут быть начаты по достижении стабильности требований пользователя. Хотя перед приемочными испытаниями предполагается, что система работает, большинство организаций перед окончательной официальной презентацией системы в присутствии потребителя подготавливает и выполняет тестирование, известное под названием «Генеральная репетиция». При этом могут быть заплани-

рованы повторное тестирование и процессы регрессионного тестирования для устранения каких-либо «последних» дефектов, выявленных в результате этого тестирования.

**Динамическое тестирование 2. Презентация**

| | |
|---|---|
| Цель тестирования: | Представить завершенную систему потребителю. |
| Элемент тестирования: | Завершенная система. |
| Базис тестирования: | Нет. |
| Процессы динамического тестирования: | Установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. |
| Метод(ы) проектирования тестирования: | Нет. |

После заключительной «Генеральной репетиции» необходимо привести тестовую среду в исходное состоянии, иначе выполнение теста будет происходить соответственно процедурам и среде, подготовленным в ходе «Генеральной репетиции».

### D.3 Подпроцесс тестирования детального проектирования

Этот пример представляет подпроцесс тестирования, связанный с фазой детального проектирования жизненного цикла разработки.

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о качестве детального проектирования. |
| Запланированный состав подпроцесса тестирования: | Статическое тестирование 1. Документация элемента детального проектирования.<br>Статическое тестирование 2. Удобство использования элемента детального проектирования (неудобство использования системы!).<br>Статическое тестирование 3. Полнота элемента детального проектирования. |

В фазе детального проектирования в соответствии с архитектурой разрабатываются отдельные элементы проекта. Каждый из них может быть подвергнут статическому тестированию посредством определенного для этого подпроцесса тестирования. Таким образом, объектами планирования статического тестирования могут быть элементы тестирования, определенные как конкретные элементы детального проектирования. Подпроцесс тестирования детального проектирования завершается только тогда, когда завершатся все запланированные тестирования (или в зависимости от обстоятельств фиксируется отказ).

**Статическое тестирование 1. Документация элемента детального проектирования**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о том, как документирован элемент детального проектирования. |
| Элемент тестирования: | Элемент детального проектирования. |
| Базис тестирования: | Внутренние и/или внешние инструкции, определяющие правила документирования детального проектирования. |
| Метод(ы) проектирования тестирования: | Технический анализ или проверка. |

**Статическое тестирование 2. Удобство использования элемента детального проектирования**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о полноценности элемента детального проектирования с точки зрения, например, кодирования или тестирования. |
| Элемент тестирования: | Элемент детального проектирования. |
| Базис тестирования: | Нет. |
| Метод(ы) проектирования тестирования: | Пошаговый разбор, например, с программистами или с тестерами. |

**Статическое тестирование 3. Полнота элемента детального проектирования**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о полноте элемента детального проектирования. |
| Элемент тестирования: | Элемент детального проектирования. |
| Базис тестирования: | Информация о прослеживаемости для проекта и/или требований уровня выше. |
| Метод(ы) проектирования тестирования: | Технический анализ. |

### D.4 Подпроцесс интеграционного тестирования

Этот пример представляет подпроцесс тестирования, связанный с фазой интеграции в жизненном цикле разработки, когда компоненты (протестированные) постепенно интегрируются.

Во время фазы интеграции интегрируются два из множества произведенных подобных элементов тестирования. Динамическое тестирование в данном случае является общим и может быть выполнено для двух любых

интегрируемых компонентов, начиная от самых первых двух интегрируемых компонентов до тех пор, пока последние два компонента не интегрированы в полную систему. Подпроцесс интеграционного теста завершается только тогда, когда завершатся все запланированные тестирования (или в зависимости от обстоятельств фиксируется отказ).

Возможны различные уровни интеграции. Это может быть интеграция компонентов на исходном коде в больший компонент, образующий завершенную полную систему, интеграция подсистем разных типов (аппаратных средств, программного обеспечения, данных, учебных материалов и т. д.) в одну полную систему или интеграция полноценных систем, объединяющая их в больший компонент, входящий в завершенную систему систем. Везде принципы одинаковы, хотя формальность зависит от профиля риска.

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о взаимодействии интегрированных компонентов. |
| Запланированный состав подпроцесса тестирования: | Статическое тестирование. Прямой интерфейс.<br>Динамическое тестирование 1. Прямой интерфейс.<br>Динамическое тестирование 2. Косвенный интерфейс.<br>Динамическое тестирование 3. Сосуществование. |

**Статическое тестирование. Прямой интерфейс**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о прямом интерфейсе между двумя интегрированными компонентами, например, в форме списка параметров. |
| Элемент тестирования: | Исходный код интерфейса между интегрируемыми компонентами. |
| Базис тестирования: | Архитектура. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. |
| Метод(ы) проектирования тестирования: | Технический анализ или проверка в зависимости от профиля риска. |

**Динамическое тестирование 1. Прямой интерфейс**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о прямом интерфейсе между двумя интегрированными компонентами, например, в форме списка параметров. |
| Элемент тестирования: | Интерфейс между интегрируемыми компонентами. |
| Базис тестирования: | Архитектура. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. |
| Метод(ы) проектирования тестирования: | Сообразно обстоятельствам. |

**Динамическое тестирование 2. Косвенный интерфейс**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о косвенном интерфейсе между двумя интегрированными компонентами, например, через базу данных. |
| Элемент тестирования: | Интегрированный компонент. |
| Базис тестирования: | Архитектура. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. Может быть возможность повторно использовать процедуры тестирования из ранее выполненных подпроцессов (компонент). Если это верно, то разработка и реализация тестирования могут быть минимизированы или опущены. |
| Метод(ы) проектирования тестирования: | Сообразно обстоятельствам. |

**Динамическое тестирование 3. Сосуществование**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о сосуществовании интегрированного компонента (или полной системы) с другими существующими в среде системами. |
| Элемент тестирования: | Интегрированный компонент (или полная система) и существующие в среде системы. |
| Базис тестирования: | Архитектура. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. Может быть возможность повторно использовать процедуры тестирования из других тестов. Если это верно, то разработка и реализация тестирования могут быть минимизированы или опущены. |
| Метод(ы) проектирования тестирования: | Сообразно обстоятельствам и возможно дополнение тестированием на базе опыта и/или исследовательское тестированием. |

### D.5 Подпроцесс тестирования производительности

Этот пример представляет подпроцесс тестирования, фокусирующийся на производительности системы.

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о выполнении требований к производительности для системы. |
| Запланированный состав подпроцесса тестирования: | Статическое тестирование 1. Документация требований к производительности.<br>Статическое тестирование 2. Полнота требований к производительности.<br>Статическое тестирование 3. Архитектура с точки зрения производительности.<br>Статическое тестирование 4. Детальное проектирование с точки зрения производительности.<br>Динамическое тестирование 1. Применимая подсистема с точки зрения производительности.<br>Динамическое тестирование 2. Завершенная система с точки зрения производительности. |

Этот подпроцесс тестирования не привязан к какой-либо определенной фазе жизненного цикла разработки. Тестирование может быть запланировано для выполнения по мере завершения разработки подходящего элемента тестирования. Подготовка к статическому тестированию может быть начата, как только завершено планирование разработки элемента тестирования, а исследование, последующая обработка и анализ могут быть реализованы, как только элемент тестирования объявлен готовым к статическому тестированию. Разработка и реализация динамического тестирования может начинаться, как только стабилизируется базис тестирования, а выполнение может быть начато, как только элемент тестирования объявлен готовым.

Данный раздел представляет пример возможных подпроцессов тестирования с точки зрения атрибутов качества. Подобные же подпроцессы тестирования могут быть определены, например, для функциональности, управляемости и переносимости.

**Статическое тестирование 1. Документация требований к производительности**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о качестве требований к производительности. |
| Элемент тестирования: | Совокупность требований к производительности. |
| Базис тестирования: | Внутренние и/или внешние инструкции по документированию требований к производительности, например, с точки зрения тестируемости. |
| Метод(ы) проектирования тестирования: | Технический анализ или проверка (следует помнить, что проверке должен предшествовать неформальный или технический анализ, чтобы гарантировать определенную завершенность требований, которые будут проверены). |

**Статическое тестирование 2. Полнота требований к производительности**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о полноте требований к производительности (все функциональные требования должны иметь соответствующие требования к производительности). |
| Элемент тестирования: | Все требования. |
| Базис тестирования: | Информация о вертикальной прослеживаемости между функциональными требованиями и требованиями к производительности. |
| Метод(ы) проектирования тестирования: | Технический анализ. |

**Статическое тестирование 3. Архитектура с точки зрения производительности**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о том, каким образом требования к производительности включены в архитектуру. |
| Элемент тестирования: | Архитектура. |
| Базис тестирования: | Требования к производительности и соответствующие инструкции. |
| Метод(ы) проектирования тестирования: | Пошаговый разбор, технический анализ или проверка (следует помнить, что проверке должен предшествовать неформальный или технический анализ, чтобы гарантировать определенную завершенность требований, которые будут проверены). |

**Статическое тестирование 4. Детальное проектирование с точки зрения производительности**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о том, как требования к производительности были учтены в детальном проектировании. |
| Элемент тестирования: | Один или более соответствующих элементов детального проектирования. |
| Базис тестирования: | Требования к производительности и соответствующие инструкции. |

| | |
|---|---|
| Метод(ы) проектирования тестирования: | Пошаговый разбор, технический анализ или проверка (следует помнить, что проверке должен предшествовать неформальный или технический анализ, чтобы гарантировать определенную завершенность требований, которые будут проверены). |

**Динамическое тестирование 1: Применимая подсистема с точки зрения производительности**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о производительности определенной подсистемы. |
| Элемент тестирования: | Одна или более соответствующие подсистемы. |
| Базис тестирования: | Требования к производительности и возможно идентифицированные риски производительности. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. Некоторые процедуры тестирования могут быть использованы повторно для интеграционного тестирования. |
| Метод(ы) проектирования тестирования: | Применимые методы. |

Эти тестирования обычно выполняются в ходе подпроцесса интеграционного теста.

**Динамическое тестирование 2. Завершенная система с точки зрения производительности**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о производительности абсолютно интегрированной системы. |
| Элемент тестирования: | Завершенная система. |
| Базис тестирования: | Требования к производительности и возможно идентифицированные риски производительности. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. Некоторые процедуры тестирования могут быть использованы повторно для тестирования системы. |
| Метод(ы) проектирования тестирования: | Применимые методы. |

Это тестирование обычно выполняется в ходе подпроцесса тестирования системы.

### D.6 Подпроцесс регрессионного тестирования

Этот пример представляет общий подпроцесс тестирования, который будет выполняться после реализации изменений в элементе, связанном с элементом тестирования, и/или изменений среды, в которой работает элемент тестирования. Подпроцесс тестирования должен использоваться для элемента тестирования, ранее прошедшего одно или более тестирований и, как определено, не затронутого реализованными изменениями.

Подпроцесс тестирования может быть выполнен как часть любого другого подпроцесса тестирования и для любого элемента тестирования. Выбор элемента тестирования зависит от характера изменений и профиля риска. Необходимое число регрессионных тестов в подпроцессе тестирования зависит от первоначального качества элемента тестирования и критериев завершения тестирования.

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставлять информацию о состоянии элемента тестирования каждый раз при реализации изменений (связанных или не связанных с элементом тестирования). |
| Запланированный состав подпроцесса тестирования: | Повторное выполнение предыдущих статических исследований или соответствующих уже выполненных динамических тестирований. |

**Регрессионный тест**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о качестве измененного элемента тестирования на неизменных элементах тестирования. |
| Элемент тестирования: | Рассматриваемый элемент тестирования. |
| Базис тестирования: | Сообразно обстоятельствам. |
| Подробные процессы тестирования: | Установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. Разработка и реализация тестирования не требуются, так как это тестирование выполняется с использованием выбранных из ранее прошедших процедур тестирования. |
| Метод(ы) проектирования тестирования: | Сообразно обстоятельствам. |

**Статическое тестирование 1. Документация требований**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о том, как документированы требования. |
| Элемент тестирования: | Отобранные требования (либо все, либо часть). |

| | |
|---|---|
| Базис тестирования: | Внутренние и/или внешние инструкции, например, относительно определенного стиля документирования требований и/или относительно соответствующей информации, такой как уникальная идентификация, приоритет и инициатор. |
| Метод(ы) проектирования тестирования: | Технический анализ или проверка (следует помнить, что проверке должен предшествовать неформальный или технический анализ, чтобы гарантировать определенную завершенность требований, которые будут проверены). |

**Статическое тестирование 2. Удобство использования требований**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о полноценности требований к проекту или тестированию. |
| Элемент тестирования: | Отобранные требования (либо все, либо часть). |
| Базис тестирования: | Нет. |
| Метод(ы) проектирования тестирования: | Пошаговый разбор, например, с разработчиками или с тестерами. |

**Статическое тестирование 3. Полнота требований**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о полноте ряда требований. |
| Элемент тестирования: | Отобранные требования (либо все, либо часть). |
| Базис тестирования: | Инструкции и/или информация о прослеживаемости для требований высокого уровня. |
| Метод(ы) проектирования тестирования: | Технический анализ. |

### D.7 Подпроцесс повторного тестирования

Этот пример представляет общий подпроцесс тестирования, который будет выполняться после реализации изменений в элементе, вызванных дефектом, найденным при предыдущем выполнении теста, то есть для элемента тестирования, ранее уже проходившего тест.

Подпроцесс тестирования может быть выполнен как часть любого другого подпроцесса тестирования и для любого элемента тестирования.

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о дефекте, который, как доложено, был устранен. |
| Запланированный состав подпроцесса тестирования: | Повторное выполнение ранее не прошедшего соответствующего статического или динамического тестирования. |

**Повторное тестирование:**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о качестве измененного элемента тестирования. |
| Элемент тестирования: | Рассматриваемый элемент тестирования. |
| Базис тестирования: | Сообразно обстоятельствам. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста; отчетность об инцидентах тестирования. Разработка и реализация тестирования не требуются, так как это тестирование выполняется, используя процедуры тестирования, которые ранее не завершились успехом. |
| Метод(ы) проектирования тестирования: | Сообразно обстоятельствам. |

### D.8 Подпроцесс тестирования совокупности истории

Этот пример представляет подпроцесс тестирования, связанный с выбором для обработки в конкретном спринте портфеля на базе текущего портфеля проекта.

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о качестве совокупности историй для обработки в определенном спринте. |
| Запланированный состав подпроцесса тестирования: | Статическое тестирование. Возможность реализации историй. |

**Статическое тестирование. Возможность реализации историй**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о совокупности отобранных историй. |
| Элемент тестирования: | Отобранные истории. |
| Базис тестирования: | Инструкции, например, относительно понимания историй и оценки времени разработки, а также зависимости и непротиворечивости историй. |

| | |
|---|---|
| Метод(ы) проектирования тестирования: | Неформальный анализ или технический анализ. |

### D.9 Подпроцесс тестирования истории

Этот пример представляет подпроцесс тестирования, связанный с завершением реализации истории перед включением истории в ежедневную сборку (или в блок для включения в новую сборку).

Во время обработки историй из портфеля спринта производится множество подобных элементов тестирования, каждый из которых является реализацией истории. Поэтому подпроцесс тестирования истории может покрыть тестирование одной или более историй, в зависимости от того, сколько историй было реализовано перед сборкой. Конкретные реализованные истории тестируются похожими способами. В этом случае используется общее динамическое тестирование, которое и может быть выполнено для любой истории. Подпроцесс тестирования истории завершается только тогда, когда завершены все запланированные тестирования (или в зависимости от обстоятельств фиксируется отказ).

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о реализованной истории перед включением ее в сборку. |
| Запланированный состав подпроцесса тестирования: | Статическое тестирование: атрибуты качества исходного кода.<br>Динамическое тестирование: тестирование истории, исследовательское тестирование. |

**Статическое тестирование. Атрибуты качества исходного кода**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о качестве исходного кода. |
| Элемент тестирования: | Созданный или затронутый реализацией истории исходный код. |
| Базис тестирования: | Внутренние и/или внешние инструкции, например, относительно определенного стиля записи кода и/или необычных особенностей кодирования, таких как использование переменной перед ее объявлением. |
| Метод(ы) проектирования тестирования: | Технический анализ или статический анализ. |

**Динамическое тестирование. Тестирование истории**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о качестве реализации истории. |
| Элемент тестирования: | История, реализованная в среде изолированно от общей сборки. |
| Базис тестирования: | История. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. |
| Метод(ы) проектирования тестирования: | Соответствующие методики, дополненные методиками, необходимыми для требуемого покрытия. |

### D.10 Подпроцесс тестирования системы

Этот пример представляет подпроцесс тестирования, связанный с фазой завершения разработки системы. Он может быть также связан с фазой, определенной как фаза созревания, которая будет завершена перед объявлением готовности системы к приемочным испытаниям.

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о качестве полной системы. |
| Запланированный состав подпроцесса тестирования: | Динамическое тестирование. Тестирование системы. |

**Динамическое тестирование. Тестирование системы**

| | |
|---|---|
| Цель тестирования: | Оценить качество полной системы после интеграции. |
| Элемент тестирования: | Полная система. |
| Базис тестирования: | Системные требования. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. |
| Метод(ы) проектирования тестирования: | Разбиение эквивалентности, анализ граничных значений, тестирование изменения состояния и метод дерева классификации сообразно обстоятельствам. |

Цель тестирования системы состоит в том, чтобы найти в функциональности системы дефекты по сравнению с требованиями к программной системе.

Следует предвидеть, что для достижения критериев завершения тестирования системы потребуется множество подпроцессов повторного тестирования и подпроцессов регрессионного теста.

### D.11 Подпроцесс покомпонентного тестирования

Этот пример представляет подпроцесс тестирования, связанный с фазой кодирования жизненного цикла разработки.

Во время фазы кодирования производится множество подобных элементов тестирования, которые являются компонентами исходного кода и могут быть либо непосредственно интерпретируемыми компонентами, либо

скомпилированы и связаны в исполнимые компоненты. В связи с этим общий подпроцесс покомпонентного тестирования может охватить подобным образом тестирование всех или некоторых конкретных из этих компонентов. В этом примере использовано общее динамическое тестирование, которое может быть выполнено для любого компонента. Подпроцесс покомпонентного тестирования завершается только тогда, когда завершены все запланированные тестирования (или в зависимости от обстоятельств фиксируется отказ).

| | |
|---|---|
| Цель подпроцесса тестирования: | Предоставить информацию о качестве компонента. |
| Запланированный состав подпроцесса тестирования: | Динамическое тестирование. Тестирование компонента. |

**Динамическое тестирование. Тестирование компонента**

| | |
|---|---|
| Цель тестирования: | Предоставить информацию о качестве компонента. |
| Элемент тестирования: | Компонент в изоляции от других компонентов в системе (может потребовать драйверов и заглушек). |
| Базис тестирования: | Детальное проектирование для компонента, включая прослеживаемость к требованиям. |
| Подробные процессы тестирования: | Разработка и реализация тестирования; установка и поддержка тестовой среды; выполнение теста и отчетность об инцидентах тестирования. |
| Метод(ы) проектирования тестирования: | Соответствующие методики, дополненные методиками, необходимыми для требуемого покрытия. |

**Приложение Е**
**(справочное)**

## Роли и обязанности в тестировании

### Е.1 Роли в тестировании

Существует множество различных наименований профессиональных ролей в тестировании, поэтому настоящий стандарт не предоставляет исчерпывающий список различных профессиональных ролей и обязанностей для глобального использования в тестировании. Вместо этого в общих чертах представлены роли, которые могут выполняться человеком, ответственным за реализацию некоторого аспекта процесса тестирования, определенного в настоящем стандарте. За каждую из нижеперечисленных ролей могут нести ответственность несколько человек.

**Стратег тестирования**

Устанавливает и обеспечивает соответствие Организационному Процессу Тестирования.

**Менеджер тестирования**

Разрабатывает и управляет Процессом Менеджмента Тестирования и гарантирует его соответствие. Менеджер тестирования также планирует и управляет Процессами Динамического Тестирования.

**Тестер**

Получает результаты тестирования и выполняет процессы, относящиеся к Процессам Динамического Тестирования.

Очевидно, что в реальности процессы, определенные в настоящем стандарте, будут выполняться несколькими людьми с различными должностями.

### Е.2 Обмен информацией в тестировании

Тестеры должны общаться с людьми соответствующих уровней организации, а также с заинтересованными сторонами внешних организаций (то есть с разработчиками элемента тестирования, спонсорами продукта, командой поддержки и персоналом продажи и маркетинга). Статус тестирования должен своевременно освещаться в соответствии с графиком проектных работ. Обмен информацией может осуществляться в письменной форме и базироваться на таких документах, как Организационная Стратегия Тестирования, Планы Тестирования, Отчеты о Ходе Тестирования и Отчеты о Завершении Тестирования. Письменная форма может дополняться устными презентациями, как это часто бывает в более формальной последовательной или эволюционной разработке. В менее формальных режимах разработки, таких как динамичная разработка, обмен информацией, может преимущественно осуществляться в устной форме и подкрепляться письменной формой документов по мере необходимости.

### Е.3 Независимость в тестировании

Тестирование должно быть максимально объективным. Чем ближе аналитик тестирования к автору элемента тестирования, тем труднее обеспечить объективность. Общеизвестно, что автору найти дефекты в своей собственной работе намного труднее, чем обнаружить те же самые дефекты независимому тестеру. Независимость в оценке продукта общепринята во многих отраслях, например, издательство, где оценку выполняет редактор; производство, где есть управление качеством; жилищное строительство, где есть строительная инспекция.

Далее приводится список в порядке увеличения уровня независимости между автором и тестером:

a) автор проверяет свой собственный продукт;

b) тестирование разработано и выполнено человеком, не являющимся автором, но с той же ответственностью, что и автор, и который является членом той же организационной единицы, что и автор, подчиняющимся тому же менеджеру;

c) тестирование разработано и выполнено тестером, являющимся членом той же организационной единицы, что и автор, подчиняющимся тому же менеджеру;

d) тестирование разработано и выполнено хотя и внутренними тестерами, но независимыми от разрабатывающей организационной единицы;

e) тестирование разработано и выполнено тестерами, привлеченными внешней организацией (консультантами), но работающими в той же организации, что и автор;

f) тестирование разработано и выполнено тестерами внешней организации (тестирование третьей стороной).

Основная задача состоит в том, чтобы в рамках ограничений времени, бюджета, качества и риска проекта достигнуть максимальной независимости между разработчиками тестирования и разработчиками элементов тес-

тирования. Организационная Стратегия Тестирования организации должна определить необходимую степень независимости, и это должно найти отражение в Плане Тестирования Проекта и конечных планах подпроцессов. Условия высоких рисков обычно приводят к более высокой степени независимости. Стандарт ИИЭР верификации и валидации программного обеспечения ИИЭР 1012—2004 определяет понятия независимости в действиях верификации и валидации, включая тестирование.

Степень независимости для различных подпроцессов тестирования обычно различна. В динамическом покомпонентном тестировании наблюдается самый низкий уровень независимости (то есть отсутствие независимости), хотя такая же степень независимости обычно считается неприемлемой в экспертных оценках (статическое тестирование, произведенное разработчиками).

В случае тестирования проекта динамичной разработки концепция интегрированной команды разработчиков и тестеров обычно означает, что более высоких уровней независимости, возможно, трудно достигнуть. В такой ситуации необходимо позаботиться о том, чтобы обеспечить максимальную независимость.

## Библиография


[1] BS 7925-1:1998, Software testing — Vocabulary
[2] BS 7925-2:1998, Software testing — Software component testing
[3] CRISPIN, L. and GREGORY, J. 2009. Agile Testing: A Practical Guide for Testers and Agile Teams. Pearson Education
[4] IEC 60300-3-9:1995, Risk analysis of technological systems
[5] IEEE Std 610.12—1995, IEEE Standard Glossary of Software Engineering Terminology
[6] IEEE Std 829—2008, IEEE Standard for Software and System Test Documentation
[7] IEEE Std 1008—1987, IEEE Standard for Software Unit Testing
[8] IEEE Std 1012—2012, IEEE Standard for Software Verification and Validation
[9] IEEE Std 1028—2008, IEEE Standard for Software Reviews and Audits
[10] ISO/IEC 12207:2008, Systems and software engineering — Software life cycle processes
[11] ISO/IEC 15026-3:2011, Information technology — System and software integrity levels
[12] ISO/IEC 16085:2006, Systems and software engineering — Life cycle processes — Risk management
[13] ISO/IEC/IEEE 24765:2010, Systems and software engineering — Vocabulary
[14] ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE
[15] ISO/IEC 25010:2011, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models
[16] ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing
[17] International Software Testing Qualifications Board (ISTQB), Standard glossary of terms used in Software Testing [online]. 2010. Updated 1 April 2010 [viewed 11 April 2011]. Available from: gost.e-ecolog.ru
[18] KOEN, B. V., 1985. Definition of the Engineering Method. American Society for Engineering Education



УДК 006.034:004.054:006.354 ОКС 35.080

Ключевые слова: тестирование программного обеспечения, процесс планирования тестирования, подпроцесс, План Тестирования, Организационная Стратегия Тестирования, риск, менеджмент, валидация, верификация

Редактор *Л.В. Коретникова*
Технический редактор *В.Н. Прусакова*
Корректор *Е.Д. Дульнева*
Компьютерная верстка *Л.А. Круговой*

Сдано в набор 19.05.2016. Подписано в печать 06.06.2016. Формат 60×84 1/8. Гарнитура Ариал.
Усл. печ. л. 6,05. Уч.-изд. 5,48. Тираж 34 экз. Зак. 1404.
Подготовлено на основе электронной версии, предоставленной разработчиком стандарта.

Издано и отпечатано во ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.


gost.e-ecolog.ru gost.e-ecolog.ru

gost.e-ecolog.ru