Стандартный глоссарий терминов, используемых в тестировании программного обеспечения

Версия 2.0 (от 4 декабря 2008)

Подготовлен ‘Glossary Working Party’
International Software Testing Qualifications Board

Редактор: Erik van Veenendaal (Нидерланды)
Редактор перевода: Александр Александров (Россия)

Авторы

Rex Black (США)
Sigrid Eldh (Швеция)
Isabel Evans (Великобритания)
Dorothy Graham (Великобритания)
Julian Harty (Великобритания)
David Hayman (Великобритания)
Juha Itkonen (Финляндия)
Vipul Kocher (Индия)
Fernando Lamas de Oliveira (Португалия)
Tilo Linz (Германия)
Peter Morgan (Великобритания)
Thomas Muller (Швейцария)
Avi Ofer (Израиль)
Dale Perry (США)
Horst Pohlmann (Германия)
Meile Posthuma (Нидерланды)
Erkki Poyhonen (Финляндия)
Maaret Pyhajarvi (Финляндия)
Andy Redwood (Великобритания)
Stuart Reid (Великобритания)
Piet de Roo (Нидерланды)
Steve Sampson (Великобритания)
Shane Saunders (Великобритания)
Hans Schaefer (Норвегия)
Jurrien Seubers (Нидерланды)
Dave Sherratt (Великобритания)
Mike Smith (Великобритания)
Andreas Spillner (Германия)
Richard Taylor (Великобритания)
Geoff Thompson (Великобритания)
Stephanie Ulrich (Германия)
Matti Vuori (Финляндия)
Gearrel Welvaart (Нидерланды)
Pete Williams (Великобритания)

Авторы перевода

Андрей Конушин (Россия)
Алексей Александров (Россия)
Александр Александров (Россия)

История изменений

Версия 1.3, 31 мая 2007	
Добавлены новые термины: - action word driven testing - bug tracking tool - coverage measurement tool - modeling tool - monkey testing - scripted testing - specification-based technique - stress testing tool - structure-based technique - unit test framework - white box technique	*Измененные термины:* - basic block - control flow graph - defect management tool - independence of testing - project risk - risk-based testing - test comparator - test process
Версия 2.0, 2 декабря 2007	
Добавлены новые термины: - attack - buffer - buffer overflow - bug taxonomy - classification tree - control flow analysis - continuous representation - cost of quality - defect based technique - defect based test design technique - defect taxonomy - error seeding tool - Failure Mode, Effect and Criticality Analysis (FMECA) - false-fail result - false-pass result - false-negative result - false-positive result - fault attack - fault seeding - fault seeding tool - hazard analysis - hyperlink - hyperlink tool - load profile - operational acceptance testing - operational profile - orthogonal array - orthogonal array testing - pairwise testing - performance profiling - pointer - procedure testing - process improvement - production acceptance testing	*Измененные термины:* - bebugging - error seeding - Failure Mode and Effect Analysis (FMEA) - Fault Tree Analysis (FTA) - modified multiple condition testing - process cycle test - root cause - specification-based technique - stress testing - test charter

- qualification - reliability growth model - retrospective meeting - risk level - risk type - root cause analysis - safety critical system - software attack - Software Failure Mode and Effect Analysis (SFMEA) - Software Failure Mode Effect and Criticality Analysis (SFMECA) - Software Fault Tree Analysis (SFTA) - software life cycle - staged representation - system of systems - test design - test estimation - test implementation - Test Maturity Model Integration (TMMi) - test progress report - test rig - test schedule - test session - wild pointer	
Версия 2.0 перевод на русский язык, 4 декабря 2008	
Глоссарий переведен на русский язык.	

Содержание

Предисловие

При создании этого глоссария рабочая группа изучила варианты, комментарии и всевозможные мнения представителей промышленности, коммерции и правительственных органов и организаций с целью создать международный стандарт тестирования, который будет принят как можно большим количеством областей. Полное понимание вряд ли, если вообще когда-нибудь, может быть достигнуто без создания документа подобного рода. Вклад в этот глоссарий был получен из сообществ тестировщиков множества стран: Австралии, Бельгии, Финляндии, Германии, Индии, Израиля, Нидерландов, Норвегии, Португалии, Швеции, Швейцарии, Великобритании, США и России.

Многие тестировщики использовали BS 7925-1 с момента его первой публикации в 1998 году. Также он послужил основой для квалификации Information Systems Examination Board (ISEB) как базового (Foundation), так и профессионального (Practitioner) уровней. Стандарт изначально был разработан с акцентом на компонентное тестирование, но с момента его публикации было получено много комментариев и предложений касательно новых определений, как для улучшения, так и для расширения стандарта, для покрытия более широкого диапазона тестирования программного обеспечения. Многие из этих предложенных изменений были включены в эту новую версию глоссария по тестированию. Он будет использован как ссылочный документ для международной системы сертификации тестировщиков программного обеспечения International Software Testing Qualifications Board (ISTQB).

1.Введение

Много времени и усилий было потрачено впустую внутри и между промышленными, коммерческими, правительственными, профессиональными и научными учреждениями, когда неоднозначность была результатом неспособности адекватно различить такие термины как «покрытие операторов» и «покрытие альтернатив»; «набор тестов», «спецификация теста» и «план тестирования», а также схожие термины, которые формируют понятийную основу различных секторов общества. Кроме того, часто профессиональное или техническое использование этих терминов вариативно.

2. Цель

Данный документ описывает принципы, термины и определения, призванные облегчить взаимодействие в области тестирования программного обеспечения и связанных областях.

3. Структура документа

Глоссарий организован в виде совокупности упорядоченных по алфавиту определений. Некоторым терминам было отдано предпочтение среди многочисленных синонимов - в этом случае определение выбранного термина включает ссылки на синонимы. Например «структурное тестирование» ссылается на «тестирование методом белого ящика». Для синонимов используется указатель «См.»

Кроме ссылок на синонимы, в Глоссарии используются перекрестные ссылки «См. также», помогающие пользователю быстро перейти к нужному термину. Перекрестные ссылки «См. также» применяются для отсылок к менее широкому термину от более широкого, и к терминам, пересекающимися своими значениям.

4. Нормативные ссылки.

На момент публикации редакция данного глоссария является актуальной. Все стандарты периодически пересматриваются, и участники соглашения, основанного на этом Стандарте, поощряются для исследования возможности применения наиболее свежих редакций нижеперечисленных стандартов. Члены IEC и ISO поддерживают списки актуальных Международных стандартов.

- BS 7925-2:1998 Software Component Testing

- DO-178B:1992 Software Considerations in Airborne Systems and Equipment Certification, Requirements and Technical Concepts for Aviation (RTCA SC167)

- IEEE 610.12:1990 Standard Glossary of Software Engineering Terminology

- IEEE 829:1998 Standard for Software Test Documentation

- IEEE 1008:1993 Standard for Software Unit Testing

- IEEE 1012:1986 Standard for Verification and Validation Plans

- IEEE 1028:1997 Standard for Software Reviews and Audits

- IEEE 1044:1993 Standard Classification for Software Anomalies

- IEEE 1219:1998 Software Maintenance

- ISO/IEC 2382-1:1993. Data processing - Vocabulary - Part 1: Fundamental terms

- ISO 9000:2000 Quality Management Systems – Fundamentals and Vocabulary (ГОСТ Р ИСО 9000-2001 Системы менеджмента качества – Основные положения и словарь)

- ISO/IEC 9126-1:2001 Software Engineering – Software Product Quality – Part 1: Quality characteristics and sub-characteristics (ГОСТ 9126. Информационная технология. Оценка программного продукта. Характеристики качества и руководящие указания по их применению)

- ISO/IEC 12207:1995. Information Technology – Software Life Cycle Processes. (ГОСТ Р ИСО МЭК 12207-99. Информационные технологии. Процессы жизненного цикла программного обеспечения)

- ISO/IEC 14598-1:1996. Information Technology – Software Product Evaluation - Part 1: General Overview

5. Торговые марки

В этом документе использованы следующие торговые марки:

- CMM и CMMI являются зарегистрированными торговыми марками Университета Карнеги-Меллон;

- TMap, TPA и TPI являются зарегистрированными торговыми марками Sogeti Nederland BV;

- TMM является зарегистрированным знаком услуг Института Технологий Иллинойса;

- TMMi является зарегистрированной торговой маркой TMMi Foundation.

6. Определения

CASE: Аббревиатура от Computer Aided Software Engineering (Автоматизация проектирования программного обеспечения).

CAST: Аббревиатура от Computer Aided Software Testing (Автоматизация тестирования программного обеспечения). См. также а*втоматизация тестирования*.

COTS: Аббревиатура от Commercial Off-The-Shelf Software (коммерческое готовое программное обеспечение). См. также *готовое программное обеспечение*.

LCSAJ: Последовательность линейного кода с переходами, состоящая из трех элементов (условно определенная номерами строк исходного кода): начало линейной последовательности выполняемых операторов, конец линейной последовательности и целевая строка кода, получающая управление после конца линейной последовательности.

V-модель (V-model): Модель, описывающая процессы жизненного цикла разработки программного обеспечения с момента составление спецификации требований до этапа сопровождения. V-модель показывает интеграцию процессов тестирования в каждую фазу цикла разработки программного обеспечения.

A

Абстрактный тестовый сценарий (abstract test case): См. *тестовый сценарий высокого уровня*.

Автоматизация выполнения тестов (test execution automation): Использование программного обеспечения (например, средств захвата/воспроизведения) для контроля выполнения тестов, сравнения полученных результатов с эталонными, установки предусловий тестов и других функций контроля тестирования и организации отчетов.

Автоматизация тестирования (test automation): Использование программного обеспечения для осуществления или помощи в проведении определенных тестовых процессов, например, управление тестированием, проектирование тестов, выполнение тестов и проверка результатов.

Автоматизированное тестирование (scripted testing): Выполнение тестов, реализуемое при помощи заранее записанной последовательности тестов.

Автоматизированное тестовое обеспечение (automated testware): Тестовое обеспечение, используемое в автоматизированном тестировании, например, инструментальные сценарии.

Автоматизированный сценарий тестирования (test script): Обычно относится к спецификации процедуры тестирования (автоматизированной).

Активация путей (path sensitizing): Составление набора входных значений для обеспечения выполнение определенного пути.

Альтернатива (decision): Точка программы, в которой управление имеет два или более альтернативных путей. Узел с двумя или более связями для разделения ветвей.

Альфа-тестирование (alpha testing): Моделируемое или действительное эксплуатационное тестирование потенциальными пользователями/заказчиками или

независимой командой тестирования на стороне разработчиков, но вне разрабатывающей организации. Альфа-тестирование часто применяется к коробочному программному обеспечению в качестве внутреннего приёмочного тестирования.

Анализ влияния (impact analysis): Оценка изменений в документации разработки и тестирования, а также компонентов с целью внесения данных изменений в определенные требования.

Анализ граничных значений (boundary value analysis): Разработка тестов методом черного ящика, в котором тестовые сценарии проектируются на основании граничных значений. См. также *граничное значение.*

Анализ дерева недочетов (Fault Tree Analysis (FTA)): Метод, используемый для анализа причин недочетов (дефектов). Методика визуально моделирует для вскрытия специфических недочетов то, как логические связи между отказами, человеческими ошибками и внешними событиями могут сочетаться.

Анализ дерева недочетов программного обеспечения (Software Fault Tree Analysis (SFTA)): См. *анализ дерева недочетов (FTA).*

Анализ мутаций (mutation analysis): Метод определения законченности набора тестов путем измерения степени, с которой набор тестов может отличить программу от ее незначительных вариаций.

Анализ первопричины (root cause analysis): Анализ, направленный на идентификацию первопричин дефектов. При применении мер к устранению первопричины, можно надеяться на минимизацию частоты появления дефектов определенного типа.

Анализ покрытия (coverage analysis): Измерение достигнутого покрытия по отношению к заданному элементу покрытия во время выполнения теста в соответствии с предопределенными критериями. Проводится для определения, необходимо ли дополнительное тестирование, и если да, то какие тестовые сценарии нужны.

Анализ потока данных (data flow analysis): Вид статического анализа, основанный на определении и использовании переменных.

Анализ потока управления (control flow analysis): Вид статического анализа, основанный на представлении последовательностей событий (путей) в процессе выполнения компонента или системы.

Анализ причинно-следственных связей (cause-effect analysis): См. *отображение причинно-следственных связей*.

Анализ рисков (risk analysis): Процесс оценки идентифицированных рисков для вычисления их вероятности и влияния.

Анализ случайности (hazard analysis): Метод, используемый для характеристики элементов риска. Результат анализа случайности определяют методы, используемые в разработке и тестировании системы. См. также *анализ рисков.*

Анализ тестируемости (testability review): Детальная проверка базиса тестирования с целью определения, является ли он достаточно качественным, чтобы выступать в роли первоисточника для процесса тестирования. [TMap]

Анализ Тестовых Точек (Test Point Analysis (TPA)): Метод оценки затрат на тестирование на основе формулы, опирающийся на анализ функциональных точек. [TMap]

Анализ типов отказов и эффекта (Failure Mode and Effect Analysis (FMEA)): систематический подход для определения и анализа рисков идентификации возможных типов отказов и попытка их предотвращения. См. также *анализ типов отказов, эффекта и критичности*.

Анализ типов отказов и эффекта программного обеспечения (Software Failure Mode and Effect Analysis (SFMEA)): См. *анализ типов отказов и эффекта*.

Анализ типов отказов, эффекта и критичности (Failure Mode, Effect and Criticality Analysis (FMECA)): расширение FMEA; в дополнение к основному FMEA, включает анализ критичности, используемый для отображения вероятности типов отказов по отношению к критичности их последствий. Результат отражает тип отказа с относительно высокой вероятностью и критичностью последствий, позволяя предпринять корректирующие действия там, где они будут иметь наибольшую ценность. См. также *анализ типов отказов и эффекта*.

Анализ типов отказов, эффекта и критичности программного обеспечения (Software Failure Mode Effect, and Criticality Analysis (SFMECA)): См. *анализ типов отказов, эффекта и критичности (FMECA).*

Анализ функциональных точек (Function Point Analysis (FPA)): Метод, помогающий при оценке размера функциональности информационной системы. Оценка не зависит от технологии. Оценка может быть использована как основа оценки производительности, расчета необходимых ресурсов и контроля проекта.

Анализатор (analyzer): См. *статический анализатор*.

Анализатор кода (code analyzer): См. *статический анализатор кода*.

Анализируемость (analyzability): Способность программного продукта быть проверенным на отсутствие отказов или их причин, а также частей, которые нужно проверить вследствие изменений.[ISO 9126] См. также с*опровождаемость*.

Аналитический отчет о тестировании (test evaluation report): Документ, создаваемый в конце процесса тестирования и подводящий итог тестовым активностям и результатам. Также в нем содержится оценка процесса тестирования и вынесенный опыт.

Аномалия (anomaly): Любое состояние, которое не соответствует ожидаемому, основанному на спецификации требований, проектной документации, пользовательской документации, стандартов и т.п., или исходя из чьего-либо восприятия или опыта. Аномалии могут быть найдены во время (но не только) рецензирования, тестирования, анализа, сборки или использования программных продуктов или соответствующей документации [IEEE1044]. См. также *помеха, дефект, отклонение, ошибка, недочет, отказ, инцидент, проблема*.

Атака (attack): Направленная и нацеленная попытка оценить качество, главным образом надежность, объекта тестирования за счет попыток вызвать определенные отказы.

Атака на недочеты (fault attack): См. *атака*.

Аудит (audit): Независимая оценка программных продуктов или процессов с целью установления соответствия стандартам, рекомендациям, спецификациям и/или процедурам, основанным на объективных критериях, включающих документы, которые определяют:
(1). Форму или содержание продуктов для производства;
(2). Процесс, согласно которому продукты будут произведены;
(3). Как будет измеряться соответствие стандартам или рекомендациям. [IEEE 1028]

Аудит конфигурации (configuration auditing): Функция проверки состава библиотек элементов конфигурации, например на соответствие стандартам. [IEEE 610]

Б

Базис тестирования (test basis): Документ, на основании которого определяются требования к компоненту или системе. Документация, на которой базируются тестовые сценарии. Если правка данного документа может быть осуществлена только в процессе формальной процедуры внесения изменения, то такой базис тестирования называется замороженным базисом тестирования. [TMap]

Базовая версия (baseline): Спецификация или программный продукт, который был формально отрецензирован или согласован, в последствии используется как базовая версия для дальнейшей разработки, и который может быть изменен только в процессе формального контроля процесса изменений. [согласно IEEE 610]

Базовый блок (basic block): Последовательность одной или более упорядоченных выполняемых операторов, которые не содержат ветвей. Примечание: узел на графе потока управления представляет собой базовый блок.

Базовый набор тестов (basis test set): Набор тестовых сценариев полученных на основании внутренней структуры компонента или спецификации, предназначенный для убеждения в 100% достижении заданных критериев покрытия.

Безопасность (safety): Способность программного продукта при использовании оговоренным образом оставаться в рамках приемлемого риска причинения вреда здоровью, бизнесу, программам, собственности или окружающей среде. [ISO 9126]

Бета-тестирование (beta testing): Эксплуатационное тестирование потенциальными и/или существующими клиентами/заказчиками на внешней стороне никак не связанными с разработчиками, с целью определения действительно ли компонент или система удовлетворяет требованиям клиента/заказчика и вписывается в бизнес-процессы. Бета-тестирование часто проводится как форма внешнего приёмочного тестирования готового программного обеспечения для того чтобы получить отзывы рынка.

Буфер (buffer): Устройство или область памяти, используемые для временного хранения данных с целью компенсации разницы в скорости потока данных, времени или частоты событий, или объемов данных, которые могут быть обработаны устройствами или процессами, участвующими в передаче или использовании данных. [Согласно IEEE 610]

В

Валидация (validation): Доказанное объективными результатами исследования подтверждение того, что требования для конкретного определенного использования приложения были выполнены. [ISO 9000]

Ведущий специалист по тестированию (test leader): См. *руководитель тестирования.*

Верификация (verification): Доказанное объективными результатами исследования подтверждение того, что определенные требования были выполнены. [ISO 9000]

Вертикальная трассируемость (vertical traceability): Отслеживание требований через уровни разработки к компонентам.

Ветвь (branch): Базовый блок, который может быть выбран для выполнения, основываясь на логической структуре программы, в которой доступен один из двух или более альтернативных путей, например, case, jump, go to, if then-else.

Веха (milestone): Точка в течение времени проекта, к которой заранее определенные (промежуточные) поставки и результаты должны быть готовы.

Возможность взаимодействия (interoperability): способность программного продукта взаимодействовать с одним или более заданными компонентами или системами [ISO 9126] См. также *функциональность.*

Воспроизводимость теста (test reproduceability): Атрибут теста, показывающий, что результаты теста одинаковы при каждом выполнении этого теста.

Восстанавливаемость (recoverability): Способность программного продукта восстанавливать требуемый уровень работоспособности и рабочие данные, пострадавшие в результате ошибки. [ISO 9126] Также см. *надежность.*

Восходящее тестирование (bottom-up testing): Последовательный подход к интеграционному тестированию, при котором компоненты нижнего уровня тестируются первыми и потом используются для облегчения тестирования компонентов более высокого уровня. Этот процесс повторяется до тех пор, пока компонент на самом верху иерархии не будет протестирован. См. также *интеграционное тестирование.*

Вход (input): Переменная (хранимая внутри или вне компонента), считываемая компонентом.

Входное значение (input value): Экземпляр входа. См. также *вход.*

Входной тест (intake test): Специальный тип теста "на дым" для принятия решения, готов ли компонент или система для дальнейшего детального тестирования. Обычно начинается в начале фазы тестирования. См. также *тест "на дым".*

Входные данные теста (test input): Данные, получаемые объектом тестирования из внешнего источника во время проведения тестирования. В роли внешнего источника может выступать аппаратное обеспечение, программное обеспечение или человек.

Выборочное тестирование (random testing): Разработка тестов методом черного ящика, в котором тестовые сценарии выбираются для соответствия функциональному разрезу, обычно с помощью алгоритма псевдо-случайного выбора. Этот метод может использоваться для тестирования таких нефункциональных атрибутов, как надежность и производительность.

Выполнение теста (test execution): Процесс запуска теста на исследуемом компоненте или системе, приводящий к реальным результатам.

Выполнимый путь (feasible path): Путь, для которого существует набор входных значений и предусловий, позволяющих ему быть выполненным.

Выполняемый оператор (executable statement): Оператор, который при компиляции переводится в объект кода, и который будет выполнен процедурно при выполнении программы, а также может выполнять операции над данными.

Выходное значение (output value): Экземпляр выходных данных. См. также *выходные данные.*

Выходные данные (output): Переменная (хранимая внутри компонента или вне его), выданная компонентом.

Г

Гибкое тестирование (agile testing): Способ тестирования для проектов, использующих гибкие методологии, такие как экстремальное программирование (XP), рассматривающий процесс разработки как потребителя процесса тестирования и делающий упор на парадигму раннего тестирования. См. также *разработка на основе тестов.*

Гиперссылка (hyperlink): Указатель на веб-странице, ведущий на другие веб-страницы.

Главный план тестирования (master test plan): План тестирования, обычно охватывающий несколько уровней тестирования. См. также *план тестирования.*

Горизонтальная трассируемость (horizontal traceability): Трассировка требований к уровню тестирования по отношению к уровням документации (например, план тестирования, спецификация проектирования теста, спецификация тестовых сценариев и спецификация процедуры тестирования или автоматизированный сценарий тестирования).

Готовое программное обеспечение (off-the-shelf software): Программное обеспечение, разработанное для широкого рынка, т.е. для большого числа клиентов, и поставляемое большинству в одинаковой конфигурации.

Граничное значение (boundary value): Входное или выходное значение, которое находится на грани эквивалентной области или на наименьшем расстоянии от обеих сторон грани, например, минимальное или максимальное значение области.

Граф потока управления (control flow graph): Абстрактное представление всех возможных последовательностей событий (путей) в процессе выполнения компонента или системы.

График тестирования (test schedule): Список задач, действий или событий в процессе тестирования, определяющий даты и/или время их начала и завершения, и их взаимозависимости.

Группа контроля изменений (change control board): См. *группа контроля конфигурации.*

Группа контроля конфигурации (configuration control board (CCB)): Группа людей, ответственных за оценку и утверждение или неутверждение предложенных изменений в элементах конфигурации, а также за обеспечение внесения предложенных изменений. [IEEE 610]

Грязное тестирование (dirty testing): См. *негативное тестирование.*

Д

Дерево классификации (classification tree): Дерево, показывающее иерархично упорядоченные эквивалентные области, которое используется для разработки тестовых сценариев в методе дерева классификации. См. также *метод дерева классификации.*

Дефект (defect): Изъян в компоненте или системе, который может привести компонент или систему к невозможности выполнить требуемую функцию, например неверный оператор или определение данных. Дефект, обнаруженный во время выполнения, может привести к отказам компонента или системы.

Диаграмма причинно-следственных связей (cause-effect graph): Графическое представление входных данных и/или сигналов (причин) и связанных выходных данных (следствий), которое может быть использовано для разработки тестовых сценариев.

Диаграмма состояний (state diagram): Диаграмма, иллюстрирующая состояния, которые может принимать компонент или система, и показывающая ситуации или события, приводящие к переходу из одного состояния в другое. [IEEE 610]

Дикий указатель (wild pointer): Указатель, указывающий к точке, находящийся вне диапазона, определенного для указателя или не существующей. См. *указатель.*

Динамический анализ (dynamic analysis): Процесс оценки поведения, например производительности памяти, загрузки ЦПУ системы или компонента во время выполнения. [IEEE 610]

Динамическое сравнение (dynamic comparison): Сравнение фактического и ожидаемого результатов, производимое во время работы программного обеспечения, например с помощью инструмента выполнения тестов.

Динамическое тестирование (dynamic testing): Тестирование, проводимое во время выполнения программного обеспечения, компонента или системы.

Домен (domain): Набор, из которого могут быть выбраны корректные входные и/или выходные данные.

Доступность (availability): Уровень готовности и доступности компонента или системы при необходимости их использования. Часто выражается в процентах. [IEEE 610]

Драйвер (driver): Компонент программного обеспечения или средство тестирования, которое заменяет компонент, обеспечивающий управление и/или вызов компонента или системы. [TMap]

Е

Ежедневная сборка (daily build): Действия, в ходе которых система ежедневно (обычно ночью) компилируется и собирается целиком, так что целостная система доступна в любое время и включает все последние изменения.

Ж

Жизненный цикл программного обеспечения (software life cycle): Период времени, начинающийся с момента появления концепции программного обеспечения и заканчивающийся тогда, когда использование программного обеспечения более невозможно. Жизненный цикл программного обеспечения обычно включает в себя

следующие этапы: концепт, описание требований, дизайн, реализация, тестирование, инсталляция и наладка, эксплуатация и поддержка и, иногда, этап вывода из эксплуатации. Данные фазы могут накладываться друг на друга или проводиться итерационно.

З

Заблокированный тестовый сценарий (blocked test case): Тестовый сценарий, который не может быть выполнен вследствие невыполнения предусловий.

Завершение тестирования (test closure): Во время фазы завершения тестирования собираются данные обо всех завершенных процессах с целью объединения опыта, тестового обеспечения, фактов и чисел. Фаза завершения тестирования состоит из архивирования тестового обеспечения и оценки процесса тестирования, включающей в себя подготовку аналитического отчета о тестировании. См. также *процесс тестирования.*

Заглушка (stub): Минимальная или специализированная реализация программного компонента. Использующаяся для подмены компонента, от которого зависит разработка или тестирование другого компонента системы. [IEEE 610]

Заданные входные данные (specified input): Входные данные, для которых результат описывается спецификацией.

Заказное программное обеспечение (bespoke software, custom software): Программное обеспечение, разработанное специально для группы пользователей или заказчиков. Противоположность - *готовое программное обеспечение*.

Заменяемость (replaceability): Способность программного продукта к использованию его вместо другого программного продукта для тех же самых целей и в том же самом окружении [ISO 9126] См. также *переносимость.*

Замороженный базис тестирования (frozen test basis): Документ базиса тестирования, который может быть изменён только посредством формального процесса контроля изменений. См. *базовая версия.*

Запись теста (test recording): См. *протоколирование тестирования.*

Защищенность (security): Свойства программного продукта, отражающие его способность не допускать неавторизированный доступ, случайный или умышленный, к программам и данным. См. *функциональность.*

Значение цикломатической сложности (cyclomatic number): См. *цикломатическая сложность.*

Зрелость (maturity): (1) Возможности организации в части эффективности ее процессов и применяемых методик. См. также *Модель Зрелости Процессов Разработки Программного Обеспечения*, *Модель Зрелости Тестирования*. (2) Возможность программного продукта избегать отказа как результата дефектов в программном обеспечении. [ISO 9126] См. также *надежность*.

И

Идентификация конфигурации (configuration identification): Элемент управления конфигурацией, состоящий из выбора элементов конфигурации для системы и

фиксирования их функциональных и физических характеристик в технической документации. [IEEE 610]

Изменяемость (changeability): Способность программного продукта быть измененным определенным образом при необходимости. [ISO 9126]. См. также *сопровождаемость.*

Измерение (measurement): Процесс присвоения числа или категории сущности для описания атрибута этой сущности. [ISO 14598]

Измеритель (instrumenter): Программный инструмент для оснащения средствами контроля.

Изоляционное тестирование (isolation testing): Тестирование отдельных компонентов в изоляции от окружающих компонентов в окружении компонентов, которые при необходимости эмулируются заглушками и драйверами.

Изучаемость (learnability): Способность программного продукта быть изученным пользователем для работы с этим приложением. [ISO 9126] См. также *практичность.*

Именованный тестовый сценарий (concrete test case): См. *тестовый сценарий низкого уровня.*

Имитатор (simulator): Устройство, компьютерная программа или система, используемая в тестировании, работающая или ведущая себя аналогично заданной при тех же входных данных [IEEE 610, DO178b]. См. также э*мулятор.*

Имитация (simulation): Моделирование выбранных поведенческих характеристик одной физической или теоретической системы другой системой. [ISO 2382/1]

Индикатор производительности (performance indicator): Высокоуровневая метрика эффективности и/или производительности, использующаяся для направления и контроля прогрессивной разработки. Например, смещение сроков для разработки программного обеспечения. [CMMI]

Индикатор производительности тестов (test performance indicator): Высокоуровневая метрика эффективности и/или продуктивности, использующаяся для управления и контроля при поэтапной разработке тестов, например, процент выявления дефектов

Инкрементная модель разработки (incremental development model): Модель жизненного цикла разработки, в которой проект разделен на серию приращений, каждое из которых добавляет часть функциональности в общих требованиях проекта. Требования приоритизированы и внедряются в порядке приоритетов. В некоторых (но не во всех) версиях этой модели жизненного цикла каждый подпроект следует «мини V-модели» со своими собственными фазами проектирования, кодирования и тестирования.

Инкрементное тестирование (incremental testing): Тестирование, при котором компоненты или системы интегрируются и тестируются по одному или вместе до тех пор, пока все компоненты или системы не интегрированы и не протестированы.

Инспектор (inspector): См. *рецензент.*

Инспекция (inspection): Тип равноправного анализа, основанный на визуальной проверке документов для поиска ошибок. Например, нарушение стандартов разработки и несоответствие документации более высокого уровня. Наиболее формальная методика

рецензирования и поэтому всегда основывается на документированной процедуре. [IEEE 610, IEEE 1028]. См. также *равноправный анализ.*

Инструмент выполнения тестов (test execution tool): Тип тестового инструмента, который позволяет исполнять другое программное обеспечение с использованием автоматического сценария тестирования, например - захват/воспроизведение. [Fewster and Graham]

Инструмент динамического анализа (dynamic analysis tool): Инструмент, обеспечивающий информацией о состоянии кода программного обеспечения во время его выполнения. Эти инструменты наиболее часто используются для поиска пустых указателей, проверки вычислений указателя, а также для отслеживания распределения, использования и освобождения памяти и определения утечек памяти.

Инструмент записи/воспроизведения (record/playback tool): См. *средство захвата/воспроизведения.*

Инструмент измерения покрытия (coverage measurement tool): См. *инструмент покрытия.*

Инструмент моделирования (modelling tool): Средство, поддерживающее валидацию моделей программного обеспечения или системы [Graham]

Инструмент мониторинга (monitoring tool): См. *монитор.*

Инструмент отладки (debugging tool): Инструментарий, используемый программистами для воспроизведения отказов, исследования состояния программ и поиска соответствующего дефекта. Отладчики позволяют программистам исполнять программу пошагово для останова на любом операторе программы и для установки и проверки программных переменных.

Инструмент отслеживания дефектов (defect tracking tool): См. *инструмент управления дефектами.*

Инструмент подготовки тестовых данных (test data preparation tool): Тип тестового инструмента, который позволяет осуществлять выборки данных из имеющихся баз данных, или же создавать, генерировать, обрабатывать и редактировать данные для использования в тестировании.

Инструмент подсева недочетов (fault seeding tool): Инструмент для подсева (т.е. намеренной вставки) недочетов в компонент или в систему.

Инструмент подсева ошибок (error seeding tool): См. *инструмент подсева недочетов.*

Инструмент покрытия (coverage tool): Средство, обеспечивающее объективное измерение того, какие структурные элементы (например, операторы или ветви) были проверены наборами тестов.

Инструмент проектирования тестов (test design tool): Инструмент, упрощающий проектирование теста при помощи генерации входных данных тестов на основе спецификаций, которые могут находиться в хранилище инструмента CASE (например, инструмент управления требованиями); тестовое условие, хранящихся в памяти самого инструмента, или же на основе кода.

Инструмент рецензирования (review tool): Инструмент, помогающий в процессе рецензирования. Типичными функциями являются: возможность планирования и контроля процесса рецензирования, обеспечение передачи данных, поддержка совместного рецензирования и общий репозиторий для сбора показателей и составления отчетности.

Инструмент статического анализа (static analysis tool): См. *статический анализатор.*

Инструмент стрессового тестирования (stress testing tool): Инструмент поддержки стрессового тестирования.

Инструмент тестирования (test tool): Программный продукт, поддерживающий одну или несколько задач тестирования, таких как планирование и контроль, специфицирование, создание изначальных файлов и данных, выполнение и анализ тестов. [TMap] См. *CAST.*

Инструмент тестирования защищенности (security testing tool): Инструментарий поддержки тестирования характеристик защищенности и уязвимости.

Инструмент тестирования производительности (performance testing tool): Инструмент для помощи в проведении тестирования производительности, обычно имеющий две основные функции: генерация нагрузки и измерения тестовых операций. Генерация нагрузки может имитировать множественных пользователей или же большие объемы данных. Во время выполнения, с определенных операций снимаются и протоколируются замеры времени отклика. Инструменты тестирования производительности обычно выдают отчеты на основе протокола тестирования и графики нагрузки относительно времени отклика.

Инструмент управления дефектами (defect management tool): Инструмент, обеспечивающий фиксирование дефектов и изменений, а также поддержку их состояний. Часто имеет процессно-ориентированные возможности для поддержки и контроля распределения, исправления и повторной проверки дефектов, а также возможности отчетности. См. также *инструмент управления инцидентами.*

Инструмент управления инцидентами (incident management tool): Инструмент, который обеспечивает запись и отслеживание статуса инцидентов. Часто имеют возможности, ориентированные на технологический процесс, для записи и контроля распределения, исправления и повторного тестирования инцидентов, а также возможности отчетности. См. также *инструмент управления дефектами.*

Инструмент управления тестированием (test management tool): Инструмент, помогающий в управлении тестированием и контроле процесса тестирования. Обычно включает в себя такие функции как: управление тестовым обеспечением, планирование графика тестов, протоколирование результатов, отслеживание прогресса, управление инцидентами и составление отчетов о тестировании.

Инструмент управления требованиями (requirements management tool): Инструмент, обеспечивающий запись самих требований, их атрибуты (например, приоритет, ответственных сотрудников) и аннотации, и облегчающий управление изменениями и трассируемость уровней требований. Некоторые инструменты управления требованиями также предоставляют средства статического анализа, такие как проверка на непротиворечивость и на нарушения нормативов, изначально заданных в требованиях.

Интеграционное тестирование (integration testing): Тестирование, выполняемое для обнаружения дефектов в интерфейсах и во взаимодействии между интегрированными компонентами или системами. См. также *тестирование интеграции компонентов, системное интеграционное тестирование*.

Интеграционное тестирование в малом (integration testing in the small): См. *тестирование интеграции компонентов.*

Интеграционное тестирование в целом (integration testing in the large): См. *системное интеграционное тестирование.*

Интеграция (integration): Процесс интегрирования компонентов или систем в бОльшую структуру.

Интегрированная модель зрелости процессов программного обеспечения (Capability Maturity Model Integration (CMMI)): Система, описывающая ключевые элементы эффективного процесса разработки и поддержки продукта. CMMI включает в себя передовой опыт планирования, проектирования и управления разработкой и поддержкой продукта. CMMI является наследницей CMM. [CMMI]. См. также *модель зрелости процессов программного обеспечения (CMM).*

Интегрированная Модель Зрелости Тестирования (Test Maturity Model Integrated (TMMi)): Пятиступенчатая структура улучшения процесса тестирования, связанная с интегрированной моделью зрелости процессов программного обеспечения (CMMI) и описывающая ключевые элементы эффективного процесса тестирования.

Интегрированная среда модульного тестирования (unit test framework): Инструмент, предоставляющий окружение для модульного тестирования или компонентного тестирования, в котором может быть протестирован как в изоляции, так и с соответствующими заглушками и драйверами. Этот инструмент также предоставляет возможность отладки. [Graham]

Инфраструктура тестирования (test infrastructure): Артефакты, необходимые для проведения тестирования, такие как тестовое окружение, инструменты тестирования, офисное окружение и процедуры.

Инцидент (incident): Любое событие, требующее исследования. [IEEE 1008]

Инцидент программного обеспечения (software test incident): См. *инцидент.*

Использование ресурсов (resource utilization): Способность использования программным продуктом соответствующего количества ресурсов определенного типа (например, объема оперативной и памяти второго уровня, размера временных файлов и т.д.) во время работы в установленных условиях. [ISO 9126] См. *эффективность*.

Исследовательское тестирование (exploratory testing): Неформальный метод проектирования тестов, при котором тестировщик активно контролирует проектирование тестов в то время, как эти тесты выполняются, и использует полученную во время тестирования информацию для для проектирования новых и улучшенных тестов. [Bach]

Исход (outcome): См. *результат*.

Исход теста (test outcome): См. *результат*.

Исход условия (condition outcome): Приведение условия к оценке Истина или Ложь.

Исчерпывающее тестирование (exhaustive testing): Методика тестирования, в которой набор тестов включает в себя все комбинации входных данных и предусловий.

Итеративная модель разработки (iterative development model): Модель жизненного цикла разработки, в которой проект разделен обычно на большое количество итераций. Итерация это полный цикл разработки, завершающийся выпуском (внутренним или внешним) рабочего продукта, являющегося частью конечного разрабатываемого продукта, который разрастается от итерации к итерации.

Итоговый митинг (retrospective meeting): Митинг в конце проекта, во время которого участники проекта оценивают проект и извлеченный из него опыт, который может быть использован в следующем проекте.

Итоговый отчет о тестировании (test summary report): Документ, подводящий итог задачам и результатам тестирования, также содержащий оценку соответствующих объектов тестирования относительно критериев выхода. [IEEE 829]

К

Качество программного обеспечения (software quality): Сумма функциональности и свойств программного продукта, влияющих на его способность удовлетворить сформулированные или подразумеваемые потребности.

Класс эквивалентности (equivalence class): См. *эквивалентная область.*

Классификация дефектов (defect taxonomy): Иерархическая система категорий, разработанная для помощи в классификации дефектов.

Классификация помех (bug taxonomy): См. *классификация дефектов.*

Ключевой показатель производительности (key performance indicator): См. *индикатор производительности.*

Код (code): Компьютерные инструкции и определения данных, выраженные программным языком или в форме выходных данных сборщика, компилятора или иного транслятора. [IEEE 610]

Компаратор (comparator): См. *тестовый компаратор.*

Компилятор (compiler): Программное средство, переводящее программы, выраженные на языке высокого уровня, в их эквиваленты на машинном языке. [IEEE 610]

Комплект тестов (test set): См. *набор тестов.*

Компонент (component): Наименьший элемент программного обеспечения, который может быть протестирован отдельно.

Компонентное тестирование (component testing): Тестирование отдельных компонентов программного обеспечения [Согласно IEEE 610].

Конечный автомат (finite state machine): Вычислительная модель, состоящая из ограниченного числа состояний и переходов между этими состояниями, возможно сопутствующими действиями. [IEEE 610]

Контроль версий (version control): См. *контроль конфигурации.*

Контроль изменений (change control): См. *контроль конфигурации.*

Контроль конфигурации (configuration control): Элемент управления конфигурацией, состоящий из оценки, координации, утверждения или отклонения, а также внесения изменений в элементы конфигурации после формального обоснования идентификации конфигурации. [IEEE 610]

Контроль рисков (risk control): Процесс, в результате которого выносятся решения и принимаются защитные меры для уменьшения рисков до определенного уровня или поддержанию рисков в оговоренных рамках.

Контроль тестирования (test control): Задача управления тестированием, связанная с разработкой и применением комплекса корректирующих мер для возвращения тестирования проекта в график при выявлении отклонений от плана. См. *управление тестированием.*

Конфигурационное тестирование (configuration testing): См. *тестирование переносимости.*

Конфигурация (configuration): структура компонента или системы, определяемая числом, типом и взаимосвязанностью составляющих частей.

Концепция тестирования (test charter): Изложение целей тестирования и, возможно, идей относительно процесса тестирования. Используются в исследовательском тестировании. См. также *исследовательское тестирование*.

Координатор (moderator): Лидер и главное лицо, ответственное за инспекцию или иной процесс рецензирования.

Коробочное программное обеспечение (commercial off-the-shelf software): См. *готовое программное обеспечение*.

Критерии входа (entry criteria): Набор общих и специфичных условий для продолжения процесса с определенной задачей, например, фаза тестирования. Цель критериев входа - предотвращение начала задачи, которое может потребовать больше (бесполезных) усилий, чем на устранение не пройденных критериев входа. [Gilb and Graham]

Критерии выхода (exit criteria): Набор общих и специфичных условий, согласованных заранее с заинтересованными сторонами, для того, чтобы процесс мог официально считаться завершенным. Цель критериев выхода - предотвращение возможности, когда задание считается завершенным, однако еще существуют отдельные незавершенные части задания. Критерии выхода используются для отчетности, а также планирования того, когда остановить тестирование. [Gilb and Graham].

Критерии завершения (completion criteria): См. *критерии выхода.*

Критерии приёмки (acceptance criteria): Критерии выхода, которым должны соответствовать компонент или система, для того, чтобы быть принятыми пользователем, заказчиком или другим уполномоченным лицом. [IEEE 610]

Критерий возобновления тестирования (resumption criteria): Критерии, выполнение которых должно вызвать возобновления тестирования после его приостановки. [IEEE 829]

Критерий завершения тестирования (test completion criteria): См. *критерии выхода.*

Критерий приостановки (suspension criteria): Условия, при выполнении которых (временно) приостанавливается тестирование, полностью или частично. [IEEE 829]

Критерий прохождения/непрохождения (pass/fail criteria): Правила для определения того, прошел ли элемент тестирования или свойство тест или нет. [IEEE 829]

Критичность (severity): Важность воздействия конкретного дефекта на разработку или функционирование компонента или системы. [IEEE 610]

Л

Лист Оценки Практичности Программного Обеспечения (Software Usability Measurement Inventory (SUMI)): Методика тестирования практичности использования программного продукта (например, удовлетворенности пользователя системой или компонентом), основанная на анкетировании. [Veenendaal]

Логический тестовый сценарий (logical test case): См. *тестовый сценарий высокого уровня*.

Ложный негативный результат (false-negative result): См. *ложный отчет о пройденном тесте.*

Ложный отчет о непройденном тесте (false-fail result): Результат теста, по которому было сообщено об ошибке, хотя на самом деле дефекта в объекте тестирования нет.

Ложный отчет о пройденном тесте (false-pass result): Результат теста, который не смог определить присутствие дефекта, реально существующего в объекте тестирования.

Ложный позитивный результат (false-positive result): См. *ложный отчет о непройденном тесте.*

М

Маскирование дефектов (defect masking): Случай, когда один дефект препятствует нахождению другого. [IEEE 610]

Маскирование недочетов (fault masking): См. *маскирование дефектов.*

Мастер установки (installation wizard): Программное обеспечение на любом носителе, которое помогает устанавливающему провести процесс установки. Обычно мастер выполняет процесс установки, информирует о результате установки и даёт возможность выбора вариантов установки.

Масштабируемость (scalability): Способность программного продукта к модернизации с целью удовлетворения возрастающей нагрузки. [Gerrard]

Мера (measure): Число или категория, присвоенная атрибуту сущности путем проведения измерения. [ISO 14598]

Мертвый код (dead code): См. *недостижимый код.*

Метод белого ящика (white-box technique): См. *разработка тестов методом белого ящика.*

Метод выполнения тестов (test execution technique): Метод, выбранный для фактического выполнения тестов, ручной или же автоматизированный.

Метод дерева классификации (classification tree method): Разработка тестов методом черного ящика, в которой тестовые сценарии, описанные средствами дерева классификации, разрабатываются для проверки комбинаций выборок входных и/или выходных подмножеств. [Grochtmann]

Метод проектирования тестов (test design technique): Метод, используемый для создания и/или выбора тестовых сценариев.

Метод разработки спецификаций теста (test specification technique): См. *метод проектирования тестов.*

Метод разработки тестов на основе спецификации (specification-based technique): См. р*азработка тестов методом черного ящика.*

Метод разработки тестов на основе спецификации (specification-based test design technique): См. *разработка тестов методом черного ящика.*

Метод разработки тестовых сценариев (test case design technique): См. *метод проектирования тестов.*

Метод тестирования (test technique): См. *метод проектирования тестов.*

Метод тестирования "большой взрыв" (big-bang testing): Вид интеграционного тестирования, в котором элементы программного или аппаратного обеспечения, или и то и другое, собираются в компонент или в целую систему сразу, а не по этапам. [Согласно IEEE 610] См. также *интеграционное тестирование.*

Метод черного ящика (black box technique): См. *разработка тестов методом черного ящика.*

Метод на основе дефектов (defect based technique): См. *метод создания тестовых сценариев на основе дефектов.*

Метод на основе опыта (experienced-based technique): См. *метод создания тестов на основе опыта.*

Метод проектирования функциональных тестов (functional test design technique): Процедура для получения и/или выбора тестовых сценариев, основыванная на анализе спецификации функциональности компонента или системы без ссылки на внутреннюю структуру. См. также *разработка тестов методом черного ящика*.

Метод создания тестов на основе опыта (experienced-based test design technique): Процедура получения и/или выбора тестовых сценариев, основанная на опыте, знаниях и интуиции тестировщика.

Метод создания тестовых сценариев на основе дефектов (defect based test design technique): Процедура получения и/или выбора тестовых сценариев, ориентированных на одну или более категорию дефектов, с разработкой тестов исходя из знаний об определенной категории дефектов. См. также *классификация дефектов.*

Метод, основанный на структуре (structure-based technique): См. *разработка тестов методом белого ящика.*

Метрика (metric): Шкала измерений и метод, используемый для измерений [ISO 14598].

Метрика покрытия Чау (Chow's coverage metrics): См. *покрытие N-переходов* [Chow].

Множественное условие (multiple condition): См. *составное условие.*

Модель Зрелости Процессов Разработки Программного Обеспечения (Capability Maturity Model (CMM)): Пятиуровневая система, описывающая ключевые элементы эффективного процесса разработки программного обеспечения. Модель зрелости включает в себя передовой опыт планирования, проектирования и управления процессами разработки и поддержки программного обеспечения. [CMM] См. также *интегрированная модель зрелости процессов программного обеспечения*.

Модель Зрелости Тестирования (Test Maturity Model (TMM)): Пятиступенчатая структура улучшения процесса тестирования, связанная с моделью зрелости процессов программного обеспечения (CMM) и описывающая ключевые элементы эффективного процесса тестирования.

Модель роста надежности (reliability growth model): Модель, показывающая рост надежности во время тестирования компонента или системы, вызванный исправлением дефектов, вызывавших отказы.

Модифицированное покрытие множественных условий (modified multiple condition coverage): См. *покрытие определений условий.*

Модифицированное покрытие условия альтернативы (modified condition decision coverage): См. *покрытие определений условий.*

Модифицированное тестирование множественных условий (modified multiple condition testing): См. *тестирование определений условий.*

Модифицированное тестирование условия покрытия (modified condition decision testing): См. *тестирование определений условий.*

Модуль (module, unit): См. *компонент.*

Модульное тестирование (module testing, unit testing): См. *компонентное тестирование.*

Монитор (monitor): Программный инструмент или аппаратное устройство, запущенное параллельно с тестируемым компонентом или системой и осуществляющее наблюдение, запись и/или анализ поведения этого компонента или системы. [IEEE 610]

Мониторинг тестирования (test monitoring): Задача управления тестированием, связанная с периодической проверкой статуса тестирования проекта. Составляемые отчеты содержат сравнение реального состояния с запланированным. См. *управление тестированием.*

H

Набор сценариев тестирования (test case suite): См. *набор тестов.*

Набор тестов (test suite): Комплект тестовых наборов для исследуемого компонента или системы, в котором обычно постусловие одного теста используется в качестве предусловия для последующего.

Нагрузочный тест (load test): Тип тестирования производительности, проводимый с целью оценки поведения компонента или системы при возрастающей нагрузке, например количестве параллельных пользователей и/или операций, а также определения какую нагрузку может выдержать компонент или система. См. также *тестирование производительности, стрессовое тестирование*.

Надежность (reliability): Способность программного продукта функционировать при заданных условиях на протяжении определенного периода времени, или для определенного количества операций. [ISO 9126]

Не пройденный (fail): Тест считается не пройденным, если фактический результат не соответствует ожидаемому результату.

Негативное тестирование (negative testing): Тестирование, нацеленное на демонстрацию того, что система или компонент не работают. Негативное тестирование относится в большей степени к позиции тестировщика, нежели к определенному подходу к тестированию или метод проектирования тестов, например - тестирование с некорректными входными значениями или тестирование обработки исключений.

Недостижимый код (unreachable code): Код, который не может быть достигнут и исполнен.

Недостижимый путь (infeasible path): Путь, который не может быть проверен любым набором возможных входных значений.

Недочет (fault): См. *дефект*.

Независимость тестирования (independence of testing): Разделение ответственностей, которое позволяет выполнять объективное тестирование. [DO-178b]

Непрерывное представление (continuous representation): Структура модели зрелости процессов программного обеспечения, в которой уровни зрелости предусматривают рекомендуемый порядок применения подходов к улучшению процессов в заданных областях процессов. [CMMI]

Непрохождение теста (test fail): См.*не пройденный.*

Несоответствие (non-conformity): Невыполнение определенного требования. [ISO 9000]

Неформальное рецензирование (informal review): Рецензирование, которое не основано на формальной (документированной) процедуре.

Нефункциональное тестирование (non-functional testing): Тестирование атрибутов компонента или системы, не относящихся к функциональности, то есть надежность, эффективность, практичность, сопровождаемость и переносимость

Нефункциональные требования (non-functional requirement): Требования, относящиеся не к функциональности, а к таким атрибутам как надежность, эффективность, практичность, сопровождаемость и переносимость

Нефункциональный метод разработки тестов (non-functional test design techniques): Процедура разработки или выбора тестовых сценариев для нефункционального тестирования, основанная на анализе спецификации компонента или системы в отрыве от его внутренней структуры. См. *разработка тестов методом черного ящика*.

Нормативное тестирование (regulation testing): См. *тестирование соответствия.*

O

Область входных значений (input domain): Набор, из которого могут быть выбраны верные значения . См. также *домен.*

Область выходных данных (output domain): Множество, из которого могут быть выбраны допустимые выходные значения. См. также *домен*.

Обработка исключений (exception handling): Поведение компонента или системы в случае неправильных входных данных, введенных человеком или от другого компонента или системы, либо из-за внутреннего отказа.

Объект тестирования (test object): Компонент или система, которые должны быть протестированы. См. *элемент тестирования.*

Объемное тестирование (volume testing): Тестирование, при котором система испытывается на больших объемах данных. См. *тестирование использования ресурсов.*

Ожидаемый исход (expected outcome): См. *ожидаемый результат*.

Ожидаемый результат (expected result): Поведение компонента или системы при установленных условиях, которое определенно спецификацией или другими источниками.

Оператор (statement): Сущность языка программирования, обычно являющаяся минимальным неделимым исполняемым блоком.

Оператор исходного кода (source statement): См. *оператор.*

Определение данных (data definition): Выполняемый оператор, где переменной присваивается значение.

Определение рисков (risk identification): Процесс идентификации рисков с использованием таких методик как мозговой штурм, контрольные списки и история отказов.

Оракул (oracle): См. *тестовый оракул.*.

Ортогональный массив (orthogonal array): Двумерный массив, построенный со специальными математическими свойствами такими, что при выборе двух любых столбцов массива, каждому члену массива соответствует пара комбинаций.

Оснащение средствами контроля (instrumentation): Вставка дополнительного кода в программу для сбора информации о поведении программы во время выполнения. Например, для измерения покрытия кода

Отказ (failure): Отклонение компонента или системы от ожидаемого выполнения, эксплуатации или результата. [Fenton]

Отклонение (deviation): См. *инцидент.*

Отладка (debugging): Процесс поиска, анализа, и устранения причин отказов в программном обеспечении.

Отладчик (debugger): См. *инструмент отладки.*

Отображение причинно-следственных связей (cause-effect graphing): Разработка тестов методом черного ящика, в котором тестовые сценарии разрабатываются на основе диаграмм причинно-следственных связей. [BS 7925/2]

Отчет о дефекте (defect report): Документ, содержащий отчет о любом недостатке в компоненте или системе, который может привести компонент или систему к невозможности выполнить требуемую функцию. [IEEE 829]

Отчет о передаче элемента тестирования (test item transmittal report): См. *сопроводительная записка.*

Отчет о помехе (bug report): См. *отчет о дефекте.*

Отчет о проблеме (problem report): См. *отчет о дефекте.*

Отчет о тестировании (test report): См. *итоговый отчет о тестировании.*

Отчет о ходе тестирования (test progress report): Документ, подводящий итог задачам и результатам, составляемый с определенной периодичностью с целью сравнения прогресса тестирования с базовой версией (такими, как исходный план тестирования) и извещения о рисках и альтернативах, требующих решения руководства.

Отчет об отклонении (deviation report): См. *отчет по инциденту.*

Отчет по инциденту (incident report): Документ, описывающий событие, которое произошло, например, во время тестирования, и которое необходимо исследовать. [IEEE 829]

Отчет по инциденту программного обеспечения (software test incident report): См. *отчет по инциденту.*

Оценка (evaluation): См. *тестирование.*

Оценка затрат на тестирование (test estimation): Рассчитанная аппроксимация (например, затраченные усилия, дата завершения, связанные затраты, число тестовых сценариев, и т.д.), результаты которой могут использоваться даже когда входные данные неполные, неопределенные или неточные.

Оценка функциональности (functionality testing): Процесс тестирования для определения функциональных возможностей программного продукта.

Ошибка (error): Действие человека, которое приводит к неправильному результату. [IEEE 610]

П

Память (storage): См. *использование ресурсов.*

Пара "определение-использование" (definition-use pair): Ассоциация определений переменных и их использования. Переменные используются, в частности, для вычислений (например, умножение) или указания пути выполнения (в качестве предиката).

Парное программирование (pair programming): Подход к разработке программного обеспечения, при котором кода (при разработке или тестировании) пишется двумя программистами за одним компьютером. По сути это подразумевает непрекращающиеся рецензии кода.

Парное тестирование (pair testing): Два человека (двое тестировщиков, разработчик и тестировщик, или конечный пользователь и тестировщик), работающих вместе над поиском дефектов. Обычно они работают за одним компьютером, в течении работы передавая управление друг другу.

Первопричина (root cause): Источник дефекта, при удалении которого частота подобных дефектов сокращается, или же подобные дефекты исчезают полностью. [CMMI]

Передовой опыт (best practice): Лучший или инновационный метод, который может улучшить производительность организации в заданных условиях, обычно признается как "лучший" другими подобными организациями.

Переменная (variable): Элемент памяти, доступный для программного продукта через его имя.

Переносимость (portability): Легкость, с которой программный продукт может быть перенесен с одной аппаратного или программного окружения в другое. [ISO 9126]

Переполнение буфера (buffer overflow): Дефект доступа к памяти вследствие попытки процесса сохранить данные, превосходящие ограничения фиксированной длины буфера, приводящий к перезаписыванию соседних областей памяти или вызыванию исключения переполнения. См. также *буфер.*

Переход состояний (state transition): Переход между двумя состояниями компонентам или системы.

План тестирования (test plan): Документ, описывающий цели, подходы, ресурсы и график запланированных тестовых активностей. Он определяет объекты тестирования, свойства для тестирования, задания, ответственных за задания, степень независимости каждого тестировщика, тестовое окружение, метод проектирования тестов, определяет используемые критерии входа и критерии выхода и причины их выбора, а также любые риски, требующие планирования на случай чрезвычайных обстоятельств. [IEEE 829]

План тестирования проекта (project test plan): См. *главный план тестирования.*

План фазы тестирования (phase test plan): План тестирования, обычно описывающий одну фазу тестирования. См. *план тестирования.*

Планирование тестирования (test planning): Работа по составлению и поддержанию актуальности плана тестирования.

Плотность дефектов (defect density): Количество дефектов, обнаруженных в компоненте или системе, поделенное на размер компонента или системы (выраженный в стандартных единицах измерения, например строках кода, числе классов или функций).

Плотность недочетов (fault density): См. *плотность дефектов.*

Поведение (behavior): Отклик компонента или системы на набор входных значений и предусловий.

Поведение во времени (time behavior): См. *производительность.*

Повторное тестирование (re-testing): Тестирование, во время которого исполняются тестовые сценарии, выявившие ошибки во время последнего запуска, для подтверждения успешности исправления этих ошибок.

Подветвь (subpath): Последовательность исполняемых операторов в коде компонента.

Подсев недочетов (fault seeding): Процесс намеренного внесения известных дефектов в дополнение к тем, что уже существуют в компоненте или в системе, для целей отслеживания уровня обнаружения и устранения, а также оценивания количества оставшихся в системе дефектов. [IEEE 610]

Подсев ошибок (error seeding): См. *подсев недочетов.*

Подтверждающее тестирование (confirmation testing): См. *повторное тестирование.*

Подход к тестированию (test approach): Реализация стратегии тестирования для определенного проекта. Обычно включает в себя заключения, сделанные на основе цели (тестирования) проекта и анализе рисков, стартовые точки процесса тестирования, применяемые методики разработки тестов, критерии выхода, типы тестирования, которые должны быть произведены.

Покрытие (coverage): Уровень, выражаемый в процентах, на который определенный элемент покрытия был проверен набором тестов.

Покрытие LCSAJ (LCSAJ coverage): Процент LCSAJ компонента, которые были проверены набором тестов. 100% покрытие LCSAJ предполагает 100% покрытие альтернатив

Покрытие N-переходов (N-switch coverage): Процент последовательностей N+1 переходов, выполненных набором тестов. [Chow]

Покрытие альтернатив (decision coverage): Процент результатов альтернативы, который был проверен набором тестов. Стопроцентное покрытие решений подразумевает стопроцентное покрытие ветвей и стопроцентное покрытие операторов.

Покрытие ветвей (branch coverage): Процент ветвей, которые были выполнены набором тестов. 100% покрытие ветвей подразумевает 100% покрытие альтернатив и 100% покрытие операторов.

Покрытие граничных значений (boundary value coverage): Процент граничных значений, который был проверен набором тестов.

Покрытие кода (code coverage): Метод анализа, определяющий, какие части программного обеспечения были проверены (покрыты) набором тестов, а какие нет, например, покрытие операторов, покрытие альтернатив или покрытие условий.

Покрытие комбинаций условий (condition combination coverage): См. *покрытие множественных условий.*

Покрытие комбинаций условий ветвей (branch condition combination coverage): См. *покрытие множественных условий.*

Покрытие множественных условий (multiple condition coverage): Процент комбинаций всех исходов одиночных условий в рамках одного оператора, который был проверен набором тестов. Стопроцентное покрытие множественных условий означает стопроцентное покрытие определений условий

Покрытие операторов (statement coverage): Процентное отношение операторов, исполняемых набором тестов, к их общему количеству.

Покрытие определений условий (condition determination coverage): Процент всех одиночных исходов условий, независимо влияющих на результаты альтернативы, которые были проверены набором сценариев тестирования. 100% покрытие определений условий подразумевает 100% покрытие условий альтернатив

Покрытие потока данных (data flow coverage): Процент пар "определение-использование", который был проверен набором тестов.

Покрытие путей (path coverage): Процент путей, которые были пройдены в процессе выполнения набора тестов. 100% покрытие путей обеспечивает 100% покрытие LCSAJ.

Покрытие условий (condition coverage): Процент исходов условий, которые были проверены набором тестов. 100% покрытие условий требует, чтобы каждое отдельное условие в каждом выражении решения было проверено как Истина и Ложь.

Покрытие условий альтернатив (decision condition coverage): процент всех исходов условий и покрытий альтернатив, который был проверен набором тестов. Стопроцентное покрытие условий решения подразумевает стопроцентное покрытие условий и стопроцентное покрытие альтернатив.

Покрытие условий ветвей (branch condition coverage): См. *покрытие условий.*

Покрытие эквивалентного разбиения (equivalence partition coverage): Процент эквивалентных областей, который был проверен набором тестов.

Политика тестирования (test policy): Документ высокого уровня, описывающий принципы, подход и основные цели организации в отношении тестирования.

Полное тестирование (complete testing): См. *исчерпывающее тестирование.*

Пользовательский тест (user test): Тест, во время которого реальные пользователи включаются в процесс оценки практичности компонента или системы.

Пользовательское приёмочное тестирование (user acceptance testing): См. *приёмочное тестирование.*

Пользовательское тестирование на основе сценариев (user scenario testing): См. *тестирование по сценариям использования.*

Помеха (bug): См. *дефект*.

Понятность (understandability): Способность программного продукта предоставить пользователю возможность сделать вывод о пригодности продукта, и о том, каким образом данный продукт может быть использован для выполнения конкретных задач. [ISO 9126] См. также *практичность.*

Попарное тестирование (pairwise testing): Разработка тестов методом черного ящика, в которой тестовые сценарии разрабатываются таким образом, чтобы выполнить все возможные отдельные комбинации каждой пары входных параметров. См. *тестирование ортогональных массивов*.

Поставка (deliverable): Любой (рабочий) продукт, который должен быть поставлен кому-либо, отличному от автора (рабочего) проекта.

Постусловие (postcondition): Условия окружения и состояния, которые должны быть выполнены после выполнения теста или процедуры тестирования.

Поток данных (data flow): Абстрактное представление последовательности и возможных изменений состояния объектов данных, при котором состояние объекта это: создание, использование либо уничтожение. [Beizer]

Поток управления (control flow): Последовательность событий (путей) в процессе выполнения компонента или системы.

Практичность (usability): Понятность, легкость в изучении и использовании и привлекательность программного продукта для пользователя при условии использовании в заданных условиях эксплуатации. [ISO 9126]

Предположение об ошибках (error guessing): Метод проектирования тестов, когда опыт тестировщика используется для предугадывания того, какие дефекты могут быть в тестируемом компоненте или системе в результате сделанных ошибок, а также для разработки тестов специально для их выявления.

Предсказанный исход (predicted outcome): См. *ожидаемый результат.*

Предтестирование (pretest): См. *входной тест.*

Предусловие (precondition): Условия окружения и состояния, которые должны быть выполнены перед началом выполнения определенного теста или процедуры тестирования.

Привлекательность (attractiveness): Способность программного продукта быть привлекательным для пользователя. [ISO 9126] См. также *практичность*.

Приёмка (acceptance): См. *приёмочное тестирование.*

Приемлемость (suitability): Способность программного продукта предоставлять подходящий функционал для определенных задач и целей конечного пользователя. [ISO 9126] См. *функциональность.*

Приёмочное тестирование (acceptance testing): Формальное тестирование по отношению к потребностям, требованиям и бизнес процессам пользователя, проводимое с целью определения соответствия системы критериям приёмки. [Согласно IEEE 610]

Приоритет (priority): Степень важности, присваиваемая объекту. Например, дефекту.

Приспособляемость (adaptability): Способность программного продукта быть адаптированным к различным заданным окружениям без применения действий или средств, кроме тех, которые предназначены для этих целей для данного программного продукта.[ISO 9126] См. также *переносимость*.

Проблема (problem): См. *дефект*.

Проверенный (exercised): Можно сказать, что программный элемент был проверен тестовым сценарием, если входные данные приводят к выполнению этого элемента, например, такого как оператор, альтернатива или иной структурный элемент.

Проверяющий (checker): См. *рецензент.*

Прогон теста (test run): Выполнение теста на определенной версии объекта тестирования

Программная атака (software attack): См. *атака.*

Программное обеспечение (software): Компьютерные программы, алгоритмы и, зачастую, документация и данные, относящиеся к функционированию компьютерной системы. [IEEE 610]

Программный измеритель (program instrumenter): См. *измеритель*.

Проект (project): Проект - это уникальный набор координируемых и контролируемых задач с датами начала и окончания, предпринимаемый для достижения цели в соответствии с определенными требованиями, включающими в себя ограничения по времени, стоимости и ресурсам. [ISO 9000]

Проектирование теста (test design): (1). См. *спецификация проектирования теста.*
(2). Процесс перевода общих требований к тестированию в конкретные тестовые условия и тестовые сценарии.

Производительность (performance): Степень, с которой система или компонент выполняет заложенные в нее функции в установленных рамках на время обработки и пропускную способность. [IEEE 610] См. также *эффективность.*

Производственное приёмочное тестирование (production acceptance testing): См. *эксплуатационное приёмочное тестирование*.

Пройден (pass): Тест считается пройденным, если его фактические результаты соответствуют ожидаемым результатам.

Простейшее сравнительное тестирование (elementary comparison testing): Разработка тестов методом черного ящика, в котором тестовые сценарии создаются для проверки комбинаций входных данных, используя концепцию покрытия определений условий. [TMap]

Просчет (mistake): См. *ошибка.*

Протокол прогона теста (test run log): См. *протокол тестирования.*

Протокол тестирования (test log): Хронологический протокол деталей, относящихся к выполнению тестов. [IEEE 829]

Протоколирование тестирования (test logging): Процесс записи информации о выполненных тестах в протокол тестирования.

Профилирование производительности (performance profiling): Определение пользовательских профилей в тестировании производительности, нагрузочном или стрессовом тестировании. Профили должны отражать ожидаемое или реальное использование, основываясь на функциональный разрез компонента или системы и, соответственно, ожидаемой рабочей нагрузки. См. *профиль нагрузки, функциональный разрез*.

Профиль нагрузки (load profile): Характеристика нагрузки, которой тестируемая система или компонент могут подвергнуться в процессе эксплуатации. Профиль нагрузки определяется количеством виртуальных пользователей, которые выполняют определенный набор операций в заданный промежуток времени в соответствии с выбранным заранее функциональным разрезом. См. также *функциональный разрез*.

Прохождение теста (test pass): См. *прохождение.*

Процедура тестирования (test procedure): См. *спецификация процедуры тестирования.*

Процент Выявления Дефектов (Defect Detection Percentage (DDP)): Количество дефектов, выявленных в фазе тестирования, поделенное на число дефектов, найденных в этой фазе тестирования, а также во всех последующих фазах.

Процент Обнаружения Недочетов (Fault Detection Percentage (FDP)): См. *процент выявления дефектов.*

Процесс (process): Комплекс взаимосвязанных активностей, преобразующих входы в выходы. [ISO 12207]

Процесс тестирования (test process): Фундаментальный процесс тестирования охватывает планирование тестирования, анализ и дизайн тестов, внедрение и выполнение тестов, оценку достижения критериев выхода и отчетность, а также работы по завершению тестирования.

Псевдоотладка (bebugging): См. *подсев недочетов*. [Abbott]

Псевдослучайный (pseudo-random): Последовательность, кажущейся случайной, но на самом деле созданная на основе некоторой заранее заданной последовательности.

Путь (path): Последовательность событий (например, исполняемых операторов) в компоненте или системе от точки входа до точки выхода.

Путь аудита (audit trail): Путь, по которому исходная информация на входе процесса (например, данные) может быть на основе выходных результатов процесса, принимая выходные данные процесса в качестве стартовой точки. Это облегчает анализ дефектов и позволяет провести аудит процесса. [По TMap]

Путь потока управления (control flow path): См. *путь*.

P

Работоспособность (operability): Способность программного обеспечения быть доступным в использовании и управлении для пользователя. [ISO 9126] См. также *практичность*.

Рабочее окружение (operational environment): Аппаратные и программные продукты, установленные на стороне пользователя или клиента, где тестируемый компонент или система будут использоваться. Программное обеспечение может включать в себя операционную систему, СУБД и другие приложения.

Равноправный анализ (peer review): Рецензирование разрабатываемого программного продукта, проводящееся сотрудниками компании-разработчика с целью нахождения дефектов и внесение улучшений. Примерами рецензирования являются: инспекция, технический анализ и разбор.

Разбор (walkthrough): Пошаговый разбор, проводимый автором документа для сбора информации и обеспечения одинакового понимания содержания документа. [Freedman and Weinberg, IEEE 1028] См. *равноправный анализ.*

Разработка на основе тестов (test driven development): Прием разработки программного обеспечения, при котором вначале разрабатываются тестовые сценарии, тестирование зачастую автоматизируется, и после этого разрабатывается то программное обеспечение, которое будет использовать эти тестовые сценарии.

Разработка тестов методом белого ящика (white box test design technique): Процедура разработки или выбора тестовых сценариев на основании анализа внутренней структуры компонента или системы.

Разработка тестов методом черного ящика (black box test design technique): Процедура создания и/или выбора тестовых сценариев, основанная на анализе функциональной или нефункциональной спецификации компонента или системы без знания внутренней структуры.

Расписание выполнения тестов (test execution schedule): Схема выполнения тестовых процедур. Тестовые процедуры включаются в расписание выполнения тестов исходя из контекста тестирования и в том порядке, в котором они должны выполняться.

Реализация теста (test implementation): Процесс разработки и расстановки приоритетов процедурам тестирования, создание тестовых данных и, опционально, подготовка тестовой обвязки и написание автоматических процедур тестирования.

Регистратор (recorder): См. *секретарь.*

Регистрация инцидентов (incident logging): Запись деталей любого инцидента, произошедшего, например, во время тестирования.

Регрессионное тестирование (regression testing): Тестирование уже протестированной программы, проводящееся после модификации для уверенности в том, что процесс модификации не внес или не активизировал ошибки в областях, не подвергавшихся изменениям. Проводится после изменений в коде программного продукта или его окружения.

Результат (result): Результат выполнения теста. В результат могут входить все виды информации на экране, изменения в данных, отчеты, отправляемые сообщения. См. также *фактический результат, ожидаемый результат.*

Результат альтернативы (decision outcome): Результат альтернативы (который, таким образом, определяет ветви, которые должны быть выбраны).

Результат теста (test result): См. *результат.*

Рецензент (reviewer): Участник рецензирования, производящий идентификацию и описание отклонений, выявленных в рецензируемом продукте или проекте. Рецензенты могут выбираться таким образом, чтобы представлять различные точки зрения и выполнять различные роли в процессе рецензирования.

Рецензирование (review): Оценка состояния продукта или проекта с целью установления расхождений с запланированными результатами и для выдвижения предложений по улучшению. Примерами рецензирования могут служить: управленческое рецензирование, неформальное рецензирование, технический анализ, инспекция и разбор

Риск (risk): Фактор, который может привести к негативным последствиям в будущем, обычно выражается через вероятность и влияние.

Риск продукта (product risk): Риск, непосредственно связанный с объектом тестирования. См. *риск.*

Риск проекта (project risk): Риск, относящийся к управлению и контролю проектом или тестированием в проекте. Например: нехватка персонала, жесткие сроки окончания, изменения требований, и т.д. См. *риск.*

Руководитель инспектирования (inspection leader): См. *координатор.*

Руководитель тестирования (test manager): Человек, ответственный за управление ресурсами и работами по тестированию, а также за экспертизу тестового объекта. Направляет, контролирует, администрирует, планирует и регулирует экспертизу тестового объекта.

Руководство по установке (installation guide): Прилагаемые инструкции на любом носителе, которые помогают устанавливающему провести процесс установки. Это может быть инструкция по установке, инструкция по установке "шаг за шагом", мастер установки или любое другое описание процесса.

Ручная имитация работы программы (desk checking): Тестирование программного обеспечения или спецификаций посредством ручной эмуляции их исполнения на испытательном стенде. См. также *статический анализ.*

С

Свободное рецензирование (ad hoc review): См. *неформальное рецензирование.*

Свободное тестирование (ad hoc testing): Тестирование, выполняемое неформально; без формальной подготовки тестов, формальных методов проектирования тестов, определения ожидаемых результатов и руководства по выполнению тестирования.

Свойство (feature): Атрибут компонента или системы, который определён или подразумевается в документации требований (например, надежность, практичность или ограничения проекта). [IEEE 1008]

Свойство программного обеспечения (software feature): См. *свойство.*

Секретарь (scribe): Человек, записывающий в протоколе каждый упомянутый дефект или улучшение, во время собрания, посвященного рецензирования. Секретарь должен обеспечить разборчивость и понятность протокола собрания.

Сертификация (certification): Процесс подтверждения того, что компонент, система или лицо отвечает предъявляемым требованиями, например, посредством сдачи экзамена.

Сессия тестирования (test session): Непрерывный промежуток времени, во время которого выполняются тесты. В исследовательском тестировании каждая сессия тестирования основывается на концепции тестирования, но тестировщики также могут исследовать новые возможности или проблемы во время сессии. Тестировщик создает и использует тестовый сценарий на лету и записывает динамику. См. *исследовательское тестирование.*

Синтаксическое тестирование (syntax testing): Разработка тестов методом черного ящика, в которой тестовые сценарии строятся на основе области определения входящих и/или выходных значений.

Система (system): Совокупность компонентов, объединенная для выполнения определенной функции или набора функций. [IEEE 610]

Система с особыми требованиями к обеспечению безопасности (safety critical system): Система, сбой или некорректная работа которой может привести к человеческой гибели или ущербу здоровью, бизнесу, программам, собственности или окружающей среде.

Система систем (system of systems): Многокомпонентные распределенные системы, работающие в компьютерных сети в разных уровней и в разных доменах, объединяемые общей базой знаний и ориентированные на решение глобальных междисциплинарных проблем и целей.

Системное интеграционное тестирование (system integration testing): Тестирование интеграции систем и пакетов программ, тестирование интерфейсов связи с внешними системами (интернет и т.д.).

Системное тестирование (system testing): Процесс тестирования системы в целом с целью проверки того, что она соответствует установленным требованиям. [Hetzel]

Сложность (complexity): Уровень, на котором компонент или система спроектированы и/или имеют внутреннюю структуру сложную для понимая, поддержки и проверки. См. также *цикломатическая сложность.*

Смягчение рисков (risk mitigation): См. *контроль рисков.*

Совершенствование Процесса Тестирования (Test Process Improvement (TPI)): Непрерывная структура совершенствования процесса тестирования, описывающая ключевые элементы эффективного процесса тестирования, фокусируясь в первую очередь на системном и приёмочном тестировании.

Соответствие (compliance): Способность программного продукта соответствовать стандартам, соглашениям или правилам законодательства и другим подобным предписаниям. [ISO 9126]

Сопроводительная записка (release note): Документ, идентифицирующий объекты для тестирования, их конфигурацию, текущий статус и прочую необходимую информацию, предоставляемую разработчиками тестировщикам и иным заинтересованным лицам в начале этапа выполнения тестов. [IEEE 829]

Сопроводительный отчет (item transmittal report): См. *сопроводительная записка.*

Сопровождаемость (maintainability): Легкость изменения программного продукта для исправления дефектов, для соответствия новым требованиям, с целью облегчения последующего сопровождения или для адаптации к изменившемуся окружению [ISO 9126]

Сопровождение (maintenance): Модификация программного продукта после его поставки с целью исправления дефектов, улучшения производительности или других характеристик или для адаптации продукта к изменившемуся окружению [IEEE 1219]

Составное условие (compound condition): Два или более одиночных условия, объединенных посредством логических операторов (И, ИЛИ, Исключающее ИЛИ), например 'A>B И C>1000'.

Сосуществование (co-existence): Способность программного продукта сосуществовать с другим независимым программным обеспечением в общем окружении, разделяя общие ресурсы. [ISO 9126] См. также *переносимость*.

Спецификация (specification): Документ, (в идеале - исчерпывающе, однозначно и доступно) описывающий требования, дизайн, поведение или иные характеристики компонента или системы. Зачастую в спецификацию включаются процедуры контроля исполнения

Спецификация компонента (component specification): Описание функций компонента в терминах его выходных значений для заданных входных значений при определенных условиях, а также требуемого нефункционального поведения (например, использование ресурсов).

Спецификация проектирования теста (test design specification): Документ, описывающий тестовое условие (элементы покрытия) для элемента тестирования, детализованный подход к тестированию, и идентифицирующий соответствующие тестовые сценарии высокого уровня. [IEEE 829]

Спецификация процедуры тестирования (test procedure specification): Документ, описывающий последовательность действий при выполнении теста. Также известен как ручной сценарий тестирования. [IEEE 829]

Спецификация теста (test specification): Документ, состоящий из спецификации проектирования теста, спецификации тестовых сценариев и/или спецификации процедуры тестирования.

Спецификация тестовых сценариев (test case specification): Документ, описывающий комплект тестовых сценариев - цель, входы, тестовые операции, ожидаемые результаты и предусловия выполнения для объекта тестирования.

Сравнение результатов выполнения (post-execution comparison): Сравнение реальных и ожидаемых результатов, производимое после окончания работы программного обеспечения.

Сравнительное тестирование (back-to-back testing): Тестирование, при котором два или больше варианта компонента или системы выполнены с одинаковыми входными значениями, а результаты сравнены и проанализированы в случае различия.

Средство захвата/воспроизведения (capture/playback tool): Тип инструмента выполнения тестов, в котором входная информация записывается во время ручного

тестирования с целью создания автоматизированных тестовых сценариев, которые могут быть выполнены позже (т.е. повторены). Эти средства часто используют для поддержки автоматизированного регрессионного тестирования.

Средство захвата/повтора (capture/replay tool): См. *средство захвата/воспроизведения.*

Средство защиты (security tool): Инструмент, обеспечивающий защиту приложения.

Средство отслеживания помех (bug tracking tool): См. *инструмент управления дефектами.*

Средство проверки гиперссылок (hyperlink tool): Инструмент, применяемый для проверки наличия на веб-сайте неверных гиперссылок.

Средство управления конфигурацией (configuration management tool): Средство, обеспечивающее поддержку идентификации и контроля элементов конфигурации, их статуса в разрезе изменений и версий, а также выпуска базовых версий, состоящих из элементов конфигурации.

Стабильность (stability): Способность программного продукта избегать непредвиденных последствий модификации программного кода. [ISO 9126] См. *сопровождаемость.*

Стадия тестирования (test stage): См. *уровень тестирования.*

Статистическое тестирование (statistical testing): Методика разработки тестов, в которой модель статистического распределения вероятности входящих значений используется для создания репрезентативных сценариев тестирования. См. *тестирование функционального разреза*.

Статический анализ (static analysis): Анализ программных артефактов, таких как требования или программный код, проводимый без исполнения этих программных артефактов.

Статический анализ кода (static code analysis): Анализ исходного кода, производимый без его исполнения.

Статический анализатор (static analyzer): Инструмент, обеспечивающий статический анализ.

Статический анализатор кода (static code analyzer): Инструмент, обеспечивающий статический анализ кода. Данный инструмент проверяет свойства исходный кода, такие как соответствие стандартам оформления кода, параметры качества или отклонения потоков данных.

Статическое тестирование (static testing): Тестирование компонента или системы на уровне спецификации или реализации без исполнения кода программного продукта, например рецензирование или статический анализ кода.

Стоимость качества (cost of quality): Общая стоимость затрат на задачи обеспечения и проблемы качества, часто разделяемая на стоимость предотвращения, стоимость оценки, стоимость внутренних отказов и стоимость внешних отказов.

Стороннее приёмочное тестирование (site acceptance testing): Приёмочное тестирование пользователями или заказчиком на своей стороне. Проводится с целью

определить как, так и удовлетворение надобностей пользователей или заказчика данной системой или компонентом. Обычно включает в себя проверку, как программного обеспечения, так и технической базы.

Стратегия тестирования (test strategy): Высокоуровневое описание уровней тестирования, которые должны быть выполнены, и тестирования, входящего в эти уровни, для организации или программы из одного или более проектов.

Стрессовое тестирование (stress testing): Вид тестирования производительности, оценивающий систему или компонент на граничных значениях рабочих нагрузок или за их пределами, или же в состоянии ограниченных ресурсов, таких как память или доступ к серверу. [IEEE 610] См. *тестирование производительности, нагрузочное тестирование.*

Структурное покрытие (structural coverage): Метрики покрытия, основанные на внутренней структуре компонента или системы.

Структурное тестирование (structural testing): См. *тестирование методом белого ящика.*

Структурный метод разработки тестов (structural test design technique): См. *разработка тестов методом белого ящика.*

Структурный разбор (structured walkthrough): См. *разбор.*

Ступенчатое представление (staged representation): Структура модели, в которой достижение цели серии процессных областей устанавливает уровень зрелости. Каждый уровень является необходимым основанием для последующих уровней. [CMMI]

Сценарий выполнения (test scenario): См. *спецификация процедуры тестирования.*

Сценарий использования системы (use case): Последовательность операций во взаимодействии пользователя и системы со значимым результатов

T

Таблица причинно-следственных решений (cause-effect decision table): См. *таблица решений.*

Таблица решений (decision table): Таблица, отражающая комбинации входных данных и/или причин с соответствующими выходными данными и/или действиям (следствиям), которая может быть использована для проектирования тестовых сценариев.

Таблица состояний (state table): Таблица, показывающая конечные переходы для каждого состояния вследствие каждого возможного события, как для корректных, так и для некорректных переходов.

Тест (test): Набор из одного или нескольких тестовых сценариев. [IEEE 829]

Тест "на дым" (smoke test): Выборка из общего числа запланированных тестовых сценариев, покрывающая основную функциональность компонента или системы. Проводится с целью удостовериться, что базовые функции программы в целом работают корректно, без углубления в детали. Ежедневная сборка и тест "на дым" являются передовыми практическими методами. См. *входной тест.*

Тест полноты (confidence test): См. *тест "на дым".*

Тест работоспособности (sanity test): См. *тест "на дым".*

Тестирование (testing): Процесс, содержащий в себе все активности жизненного цикла, как динамические, так и статические, касающиеся планирования, подготовки и оценки программного продукта и связанных с этим результатов работ с целью определить, что они соответствуют описанным требованиям, показать, что они подходят для заявленных целей и для определения дефектов.

Тестирование "сверху вниз" (top-down testing): Инкрементальный подход к интеграционному тестированию, в котором компоненты из верхнего уровня иерархии объектов тестируются в первую очередь, с использованием заглушек вместо компонентов более низкого уровня. Протестированные компоненты используются для тестирования компонентов более низкого уровня и данный процесс повторяется до тех пор, пока не будет протестированы компоненты самого низшего уровня. См. *интеграционное тестирование.*

Тестирование LCSAJ (LCSAJ testing): Разработка тестов методом белого ящика, в котором тестовые сценарии разрабатываются для проверки LCSAJ.

Тестирование N-переходов (N-switch testing): Вид тестирования таблицы переходов, в котором тестовые сценарии разрабатываются для выполнения всех правильных последовательностей N+1 переходов. [Chow]. См. *тестирование таблицы переходов.*

Тестирование алгоритма [TMap] (algorithm test [TMap]): См. *тестирование ветвей.*

Тестирование безопасности (safety testing): Тестирование программного продукта с целью с целью определить его безопасность.

Тестирование бизнес-циклов (process cycle test): Разработка тестов методом черного ящика, в которой тестовые сценарии разрабатываются для выполнения бизнес-процедур и процессов. [TMap]. См. *тестирование процессов.*

Тестирование в период сопровождения (maintenance testing): Тестирование изменений в действующей системе или влияния изменений в окружении на действующую систему.

Тестирование в условиях эксплуатации (field testing): См. *бета-тестирование.*

Тестирование ветвей (branch testing): Рразработка тестов методом белого ящика, в которой тестовые сценарии проектируются для выполнения ветвей.

Тестирование ветвями (thread testing): Вариант тестирования интеграции компонентов, в котором нарастающая интеграция компонентов производится аналогично реализации подклассов требований, в отличие от интеграции компонентов согласно уровням иерархии.

Тестирование возможности взаимодействия (interoperability testing): Процесс тестирования для определения возможности взаимодействия программного продукта. См. также *оценка функциональности*.

Тестирование восстанавливаемости (recoverability testing): Процесс тестирования, исследующий восстанавливаемость программного продукта. См. также *тестирование надежности.*

Тестирование граничных значений (boundary value testing): См. *анализ граничных значений.*

Тестирование документации (documentation testing): Тестирование качества документации, например руководства пользователя или руководства по установке.

Тестирование доступности (accessibility testing): Тестирование, которое определяет степень легкости, с которой пользователи с ограниченными способностями могут использовать систему или ее компоненты. [Gerrard]

Тестирование дуг (arc testing): См. *тестирование ветвей.*

Тестирование защищенности (security testing): Тестирование с целью оценить защищенность программного продукта. См. также *оценка функциональности.*

Тестирование интеграции компонентов (component integration testing): Тестирование, выполняемое для выявления дефектов в интерфейсах и взаимодействии между интегрированными компонентами.

Тестирование интерфейса (interface testing): Тип интеграционного тестирования, связанный с тестированием интерфейсов между компонентами или системами.

Тестирование использования памяти (storage testing): См. *тестирование использования ресурсов.*

Тестирование использования ресурсов (resource utilization testing): Процесс тестирования, исследующий использование ресурсов программным продуктом. См. *тестирование эффективности.*

Тестирование комбинаций уловий ветвей (branch condition combination testing): См. *покрытие множественных условий.*

Тестирование комбинаций условий (condition combination testing): См. *тестирование множественных условий.*

Тестирование масштабируемости (scalability testing): Тестирование с целью оценить масштабируемость программного продукта.

Тестирование методом белого ящика (white box testing): Тестирование, основанное на анализе внутренней структуры компонента или системы.

Тестирование методом конечных состояний (finite state testing): См. *тестирование таблицы переходов.*

Тестирование методом черного ящика (black box testing): Тестирование, функциональное или нефункциональное, без знания внутренней структуры компонента или системы.

Тестирование миграции (migration testing): См. *тестирование преобразования.*

Тестирование множественных условий (multiple condition testing): разработка тестов методом белого ящика, в котором тестовые сценарии разрабатываются для проверки комбинаций исходов одиночных условий (в рамках одного оператора).

Тестирование мутаций (mutation testing): См. *сравнительное тестирование.*

Тестирование на основе архитектуры (design-based testing): Подход к тестированию, в котором тестовые сценарии разрабатываются на основе архитектуры и/или подробного проекта компонента или системы (например, тестирование интерфейсов между компонентами или системами).

Тестирование на основе бизнес-процессов (business process-based testing): Метод тестирования, в котором тестовые сценарии проектируются на основании описаний и/или знаниях бизнес-процессов.

Тестирование на основе данных (data driven testing): методика написания автоматизированных тестовых сценариев, при которой входные тестовые данные и ожидаемые результаты хранятся в таблицах, таким образом, что отдельный сценарий может выполнить все тесты в таблице. Тестирование на основе данных часто используется для поддержки средств исполнения тестов, таких как средство захвата/воспроизведения. [Fewster и Graham] См. также *тестирование на основе ключевых слов*.

Тестирование на основе ключевых слов (keyword driven testing): Артефакты, создаваемые во время процесса тестирования и требующиеся для планирования, разработки и выполнения тестов. Например: документация, сценарии, входы, ожидаемые результаты, процедуры установки и удаления, файлы, базы данных, окружение и любое другое дополнительное программное обеспечение или инструменты, используемые в тестировании. [Fewster and Graham]

Тестирование на основе рабочих слов (action word driven testing): См. *тестирование на основе ключевых слов.*

Тестирование на основе спецификации (specification-based testing): См. *тестирование методом черного ящика.*

Тестирование на основе структуры (structurebased testing): См. *тестирование методом белого ящика.*

Тестирование на основе сценариев (scenario testing): см. *тестирование по сценариям использования.*

Тестирование на основе требований (requirements-based testing): Подход к тестированию, при котором тестовые сценарии разрабатываются на основе целей и условий тестирования, вытекающих из требований; то есть тесты, проверяющие определенный функционал или оценивающие нефункциональные атрибуты системы, такие как надежность или практичность

Тестирование на соответствие стандартам (standards testing): См. *тестирование соответствия.*

Тестирование надежности (reliability testing): Процесс тестирования, исследующий надежность программного продукта.

Тестирование недействительных значений (invalid testing): Тестирование, использующее входные значения, которые должны быть отклонены компонентом или системой. См. также *устойчивость к ошибкам.*

Тестирование операторов (statement testing): Разработка тестов методом белого ящика, в котором наборы тестов составляются с целью исполнения операторов.

Тестирование определений условий (condition determination testing): Разработка тестов методом белого ящика, в котором тестовые сценарии разрабатываются для проверки одиночных исходов условий, которые независимо влияют на результат альтернатив.

Тестирование ортогональных массивов (orthogonal array testing): Систематический подход к тестированию всех парных комбинаций переменных с использованием ортогональных массивов. Такой подход значительно уменьшает количество комбинаций переменных при проверке всех парных комбинаций. См. также *попарное тестирование*.

Тестирование, основанное на коде (code-based testing): См. *тестирование методом белого ящика.*

Тестирование, основанное на логике (logic-driven testing): См. *тестирование методом белого ящика.*

Тестирование переносимости (portability testing): Процесс тестирования с целью определить переносимость программного продукта.

Тестирование по сценариям использования (use case testing): Разработка тестов методом черного ящика, в котором тестовые сценарии создаются для выполнения сценариев использования.

Тестирование покрытия логики (logic-coverage testing): См. *тестирование методом белого ящика*.[Myers]

Тестирование потока данных (data flow testing): Разработка тестов методом белого ящика, в котором тестовые сценарии проектируются для проверки пары "определение-использование" для переменных.

Тестирование практичности (usability testing): Тестирование с целью определения степени понятности, легкости в изучении и использовании, привлекательности программного продукта для пользователя при условии использования в заданных условиях эксплуатации. [ISO 9126]

Тестирование преобразования (conversion testing): Тестирование программного обеспечения, применяемого для преобразования данных существующих систем для использования в заменяющих системах.

Тестирование программы (program testing): См. *компонентное тестирование.*

Тестирование прозрачного ящика (glass box testing): См. *тестирование методом белого ящика.*

Тестирование производительности (performance testing): Процесс тестирования с целью определить производительность программного продукта. См. также *тестирование эффективности.*

Тестирование путей (path testing): Разработка тестов методом белого ящика, в котором тесты создаются для проверки пути.

Тестирование разработки (development testing): Формальное или неформальное тестирование, проводимое во время реализации компонента или системы, обычно в рабочей среде разработчиков. [IEEE 610]

Тестирование регенерации (recovery testing): См. *тестирование восстанавливаемости.*

Тестирование альтернатив (decision testing): Разработка тестов методом белого ящика, в котором тестовые сценарии проектируются для проверки результатов альтернативы.

Тестирование связей (link testing): См. *тестирование интеграции компонентов.*

Тестирование сегментов (partition testing): См. *эквивалентное разбиение*. [Beizer]

Тестирование совместимости (compatibility testing): См. *тестирование возможности взаимодействия.*

Тестирование совместного доступа (concurrency testing): Тестирование с целью определить, как выполнение двух или более действий в один период времени (последовательно или параллельно) обрабатывается компонентом или системой. [Согласно IEEE 610]

Тестирование соответствия (compliance testing): Процесс тестирования для определения соответствия компонента или системы.

Тестирование соответствия (conformance testing): См. *тестирование соответствия.*

Тестирование сопровождаемости (maintainability testing): Процесс тестирования для определения сопровождаемости программного продукта.

Тестирование таблицы переходов (state transition testing): Разработка тестов методом черного ящика, в котором сценарии тестирования строятся на основе выполнения корректных и некорректных переходов состояний. См. *тестирование N-переходов.*

Тестирование таблицы решений (decision table testing): Разработка тестов методом черного ящика, в котором тестовые сценарии проектируются для проверки комбинаций входных данных и/или причин, отраженных в таблице решений. [Veenendaal]. См. также *таблица решений.*

Тестирование удобства эксплуатации (serviceability testing): См. *тестирование сопровождаемости.*

Тестирование условий (condition testing): Разработка тестов методом белого ящика, в котором тестовые сценарии разрабатываются для проверки исходов условий.

Тестирование условий альтернатив (decision condition testing): Разработка тестов методом белого ящика, в котором тестовые сценарии проектируются для исходов условий и результатов альтернатив.

Тестирование устойчивости (robustness testing): Процесс тестирования, исследующий устойчивость программного продукта.

Тестирование функционального разреза (operational profile testing): Статистическое тестирование, использующее модель системных операций (кратковременные операции) и вероятность их типичного использования. [Musa]

Тестирование целостности базы данных (database integrity testing): Тестирование методов и процессов, применяемых для доступа и управления данными, для удостоверения в том, что методы, процессы и правила доступа работают верно, а также,

что во время доступа к базе данных данные не повреждены или неожиданно удалены, обновлены или созданы.

Тестирование целостности данных (data integrity testing): См. *тестирование целостности базы данных.*

Тестирование эффективности (efficiency testing): Процесс тестирования для установления эффективности программного продукта.

Тестирование, основанное на рисках (risk-based testing): Подход к тестированию с целью минимизирования уровня проектных рисков и информирования заинтересованных лиц о текущем состоянии рисков с начальных стадий проекта. Подразумевает под собой управление процессом тестирования, исходя из идентифицированных рисков продукта.

Тестирование процессов (procedure testing): Тестирование, нацеленное на подтверждение того, что компонент или система функционируют в соответствии с новыми или имеющимися пользовательскими бизнес- или технологическими процессами.

Тестировщик (tester): Опытный специалист, принимающий участие в тестировании компонента или системы.

Тестируемость (testability): Способность программного продукта предоставлять возможность для тестирования внесенных изменений. [ISO 9126] См. *сопровождаемость.*

Тестовая запись (test record): См. *протокол тестирования.*

Тестовая обвязка (test harness): Тестовое окружение, включающее в себя заглушки и драйверы, необходимые для проведения теста.

Тестовая ситуация (test situation): См. *тестовое условие.*

Тестовое обеспечение (testware): Артефакты, создаваемые во время процесса тестирования и требующиеся для планирования, разработки и выполнения тестов. Например: документация, сценарии, входы, ожидаемые результаты, процедуры установки и удаления, файлы, базы данных, окружение и любое другое дополнительное программное обеспечение или инструменты, используемые в тестировании. [Fewster and Graham]

Тестовое окружение (test environment): Окружение, включающее в себя аппаратное обеспечение, измерительную аппаратуру, имитаторы, программный инструментарий и прочие инструменты, необходимые для проведения теста. [IEEE 610]

Тестовое покрытие (test coverage): См. *покрытие.*

Тестовое сравнение (test comparison): Процесс выявления разницы между реальными результатами, выдаваемыми исследуемым компонентом или системой, и ожидаемым результатом теста. Тестовое сравнение может производиться как во время выполнения теста (динамическое сравнение), или же по окончании выполнения теста.

Тестовое требование (test requirement): См. *тестовое условие.*

Тестовое условие (test condition): Объект или событие в компоненте или системе, которое должно быть проверено одним или несколькими тестовыми наборами. Например: функция, транзакция, свойство, атрибут качества или структурный элемент.

Тестовые данные (test data): Данные, которые существуют (например, в базе данных) на начало выполнения теста и влияют на работу, или же испытывают влияние со стороны тестируемой системы или компонента.

Тестовый генератор (test generator): См. *инструмент подготовки тестовых данных.*

Тестовый драйвер (test driver): См. *драйвер.*

Тестовый инцидент (test incident): См. *инцидент.*

Тестовый компаратор (test comparator): Инструмент тестирования, осуществляющий автоматическое сравнение реального и ожидаемого результата.

Тестовый отчет по инциденту (test incident report): См. *отчет по инциденту.*

Тестовый оракул (test oracle): Источник, при помощи которого можно определить ожидаемые результаты для сравнения с реальными результатами, выдаваемыми тестируемой системой. В роли тестового оракула могут выступать уже имеющаяся система (для эталонного тестирования), руководство пользователя, профессиональные знания специалиста, однако им не может быть программный код. [Adrion]

Тестовый стенд (test bed, test rig): См. *тестовое окружение.*

Тестовый сценарий (test case): Набор входных значений, предусловий выполнения, ожидаемых результатов и постусловий выполнения, разработанный для определенной цели или тестового условия, таких как выполнение определенного пути программы или же для проверки соответствия определенному требованию. [IEEE 610]

Тестовый сценарий высокого уровня (high level test case): Тестовый сценарий без конкретных (уровня реализации) значений входных данных и ожидаемых результатов. Использует логические операторы, а экземпляры реальных значений еще не определены и/или доступны. См. также тестовый сценарий низкого уровня

Тестовый сценарий низкого уровня (low level test case): Тестовый сценарий с конкретными (уровня реализации) значениями входных данных и ожидаемых результатов. Логические операторы из тестовых сценариев высокого уровня заменяются реальными значениями, которые соответствуют целям этих логических операторов. См. также *тестовый сценарий высокого уровня.*

Тестопригодные требования (testable requirements): Степень выраженности требований в терминах, допускающих начало работы над разработкой тестов (и, впоследствии, над тестовыми сценариями) и выполнение тестов для определения соответствия заявленным требованиям. [IEEE 610]

Технический анализ (technical review): Обсуждение, имеющее целью выработать единый подход к техническому процессу, и проводимое равноправными участниками. [Gilb and Graham, IEEE 1028] См. также *равноправный анализ.*

Тип отказа (failure mode): Физическое или функциональное проявление типа отказа. Например, система в состоянии отказа может быть характеризирована медленным выполнением операций, неправильным выводом или полным прерыванием выполнения. [IEEE 610]

Тип риска (risk type): Определенная категория рисков по отношению к типам тестирования, способным смягчить или контролировать это категорию рисков. Например, риск неправильного понимания взаимодействия с пользователем может быть смягчен при помощи тестирования практичности

Тип тестирования (test type): Группа процессов тестирования, направленных на тестирование компонента или системы с определенной целью, например, функциональное тестирование, тестирование практичности, регрессионное тестирование и т.д. Один и тот же тип тестирования может встречаться в одном или нескольких уровнях тестирования или фазах тестирования. [TMap]

Типовое программное обеспечение (standard software): См. *готовое программное обеспечение*.

Точка входа (entry point): Первый выполняемый оператор в компоненте.

Точка выхода (exit point): Последний выполняемый оператор в компоненте.

Точность (accuracy): Способность программного продукта обеспечивать правильные или согласованные результаты или действия с необходимым уровнем точности. [ISO 9126] См. также *оценка функциональности.*

Трассируемость (traceability): Способность идентифицировать связанные объекты в документации и программном обеспечении, например, требования со связанными с ними тестами. См. также *горизонтальная трассируемость, вертикальная трассируемость*.

Требование (requirement): Условия или возможности, необходимые пользователю для решения определенных задач или достижения определенных целей, которые должны быть достигнуты для выполнения контракта, стандартов, спецификации, или других формальных документов. [IEEE 610]

У

Указатель (pointer): Объект, описывающий местонахождение другого объекта. Например, объект, определяющий адрес следующей записи о сотруднике в очереди обработки. [IEEE 610]

Управление дефектами (defect management): Процесс распознавания, исследования, принятия действий и устранения дефектов. Он включает в себя фиксирование дефектов, их классификацию и выявления последствий. [IEEE 1044]

Управление инцидентами (incident management): Процесс распознавания, исследования, принятия действий и устранения инцидентов. Включает в себя протоколирование инцидентов, их классификацию и определение влияния. [IEEE 1044]

Управление конфигурацией (configuration management): Наука, применяющая техническое и административное руководство и контроль для: идентификации и документирования функциональных и физических характеристик элемента конфигурации, контроля изменений этих характеристик, записи и отчетности о состоянии процесса внедрения изменений, а также проверки совместимости с заданными требованиями. [IEEE 610]

Управление проблемами (problem management): См. *управление дефектами.*

Управление рисками (risk management): Систематическое использование процедур и практик с целью идентификации, анализа, определения приоритетов и контроля рисков.

Управление тестированием (test management): Планирование, оценка, мониторинг и контроль тестовых активностей, обычно выполняемые руководителем тестирования.

Управленческое рецензирование (management review): Систематическая оценка процесса приобретения, поддержки, разработки, эксплуатации или сопровождения программного обеспечения, выполняемая руководством или от лица руководства, контролирующего ход работ, определяющего состояние планов и графиков, подтверждающего требования и их место в системе, или оценивающего эффективность управленческих подходов для достижения целей. [IEEE 610, IEEE 1028]

Уровень отказов (failure rate): Отношение количества отказов данной категории к заданной единице измерения, например, отказ в единицу времени, количество отказов в транзакциях, количество отказов к количеству запусков. [IEEE 610]

Уровень риска (risk level): Важность риска определяется по его характеристикам: влияние и вероятность. Уровень риска может быть использован для определения интенсивности тестирования. Уровень риска может быть выражен как качественно (например: высокий, средний, низкий), так и количественно.

Уровень тестирования (test level): Объединенная и совместно управляемая группа задач тестирования. Уровень тестирования связан с областями ответственности в проекте. Примерами уровней тестирования могут служить компонентное тестирование, интеграционное тестирование, системное тестирование и приёмочное тестирование. [TMap]

Уровневый план тестирования (level test plan): План тестирования, обычно относящийся к одному уровню тестирования. См. также *план тестирования.*

Условие (condition): Логическое выражение, которое может принимать значения Истина или Ложь, например A>B. См. также *тестовое условие.*

Условие ветви (branch condition): См. *условие.*

Усовершенствование процессов (process improvement): Программа действий, нацеленная на улучшение производительности и зрелости организационных процессов, и результат такой программы. [CMMI]

Устанавливаемость (installability): Способность программного обеспечения быть установленным в определенном окружении. [ISO 9126]. См. также *переносимость.*

Установочное тестирование (installability testing): Процесс тестирования устанавливаемости программного продукта. См. также *тестирование переносимости.*

Устойчивость (robustness): Уровень, до которого компонент или система может функционировать корректно при наличии некорректных входных данных или функционирования в стрессовых условиях. [IEEE 610] См. также *устойчивость к ошибкам, устойчивость к недочетам.*

Устойчивость к ошибкам (error tolerance): Способность системы или компонента продолжать нормально функционировать, несмотря на присутствие неправильных входных данных. [IEEE 610]

Устойчивость к недочетам (fault tolerance): Способность программного продукта поддерживать определённый уровень производительности в случае программных недочетов (дефектов) или нарушении установленного интерфейса взаимодействия. [ISO 9126] См. также *надежность, устойчивость*.

Утечка памяти (memory leak): Дефект в программной логике выделения динамической памяти, являющийся причиной невозможности освободить выделенную память после того, как программа закончила ее использовать, и в конечном счете приводящий к отказу программы из-за недостатка памяти.

Учет статусов (status accounting): Составная часть управления конфигурациями, заключающаяся в сохранении и предоставлении отчетов по информации, необходимой для эффективного управления конфигурациями. Эта информация включает в себя перечень утвержденных идентификаторов конфигурации, статусы предложенных изменений конфигурации, и статусы реализации утвержденных изменений. [IEEE 610]

Ф

Фаза выполнения тестов (test execution phase): Период в цикле разработки программного обеспечения, во время которого программный продукт или его компоненты запускаются, и программный продукт оценивается с точки зрения выполнения предъявленных к нему требований. [IEEE 610]

Фаза тестирования (test phase): Определенный набор задач, объединенных в контролируемую фазу проекта, например, задачи выполнения уровня тестирования. [Gerrard]

Фактический исход (actual outcome): См. *фактический результат.*

Фактический результат (actual result): Наблюдаемое или генерируемое поведение компонента или системы во время тестирования.

Формальное рецензирование (formal review): Рецензирование, характеризирующееся документированными процедурами и требованиями, например, инспекция

Функциональная интеграция (functional integration): Подход к интеграции, который объединяет компоненты или системы для получения как можно раньше начальной рабочей функциональности. См. также *интеграционное тестирование.*

Функциональное тестирование (functional testing): Тестирование, основанное на анализе спецификации функциональности компонента или системы. См. также *тестирование методом черного ящика.*

Функциональное требование (functional requirement): Требование, определяющее функцию, которую компонент или система должны выполнять. [IEEE 610]

Функциональность (functionality): Способность программного продукта обеспечивать функции, которые соответствуют установленным и предполагаемым потребностям, при использовании ПО в определенных условиях. [ISO 9126]

Функциональный разрез (operational profile): Представление особого множества задач, выполняемых компонентом или системой, возможно опирающихся на поведение пользователя при взаимодействии с компонентом или системой, с указанием вероятности

их появления. Описываемые задачи чаще логические, нежели физические, и могут выполняться на разных машинах в независимые периоды времени.

Х

Хаотическое тестирование (monkey testing): Тестирование случайным выбором из большого диапазона входов, случайным нажатием кнопок, без соотнесения с тем, как в реальности будет использоваться система.

Характеристики качества программного обеспечения (software quality characteristic): См. *атрибут качества*.

Характеристики программного обеспечения (software product characteristic): См. *атрибут качества.*

Ц

Целостность (consistency): Уровень однородности, стандартизованности и отсутствия противоречивости в документах или частях компонента или системы. [IEEE 610]

Цель тестирования (test objective, test target): Причина или цель разработки и выполнения теста, набор критериев выхода.

Цикл тестирования (test cycle): Выполнение процесса тестирования для одной однозначно определяемой версии тестируемого объекта.

Цикломатическая сложность (cyclomatic complexity): число независимых путей в программе. Цикломатическая сложность определяется как: L - N + 2P, где L - число ребер/связей графа; N - число вершин графа; P - число несвязанных частей графа (например, граф вызова и подпрограмма). [McCabe]

Ш

Широкополосный предсказатель (Wide Band Delphi): Методика оценки затрат на тестирование на базе экспертной оценки, ставящая целью точную оценку с помощью коллективного опыта членов команды.

Шкала измерений (measurement scale): Шкала, ограничивающая тип анализа данных, который может быть осуществлен над ней. [ISO 14598]

Э

Эвристическая оценка (heuristic evaluation): Статическая методика тестирования практичности для определения соответствия интерфейса пользователя признанным принципам практичности (также называемая "эвристика").

Эквивалентная область (equivalence partition): Часть области входных или выходных данных, для которой поведение компонента или системы, основываясь на спецификации, считается одинаковым.

Эквивалентное разбиение (equivalence partitioning): Разработка тестов методом черного ящика, в которой тестовые сценарии создаются для проверки элементов эквивалентной области. Как правило, тестовые сценарии разрабатываются для покрытия каждой области как минимум один раз.

Эксплуатационное приемочное тестирование (operational acceptance testing): Эксплуатационное тестирование в фазе приемочного тестирования, обычно выполняемое пользователем и/или администратором в среде, имитирующей реальные условия рабочего окружения, фокусируясь на функциональных аспектах. Например, восстанавливаемость, поведение ресурсов, устанавливаемость и техническое соответствие. См. также *эксплуатационное тестирование*.

Эксплуатационное тестирование (operational testing): Тестирование, проводимое для оценки компонента или системы в его рабочем окружении. [IEEE 610]

Элемент конфигурации (configuration item): Сочетание аппаратного и/или программного обеспечения, предназначенное для управления конфигурацией и рассматриваемое в процессе управления конфигурацией как отдельный объект. [IEEE 610]

Элемент покрытия (coverage item): Сущность или свойство, используемые как базис для тестового покрытия, например эквивалентные области или операторы в коде.

Элемент тестирования (test item): Отдельный элемент, который должен быть протестирован. Обычно имеется один тестовый объект и несколько элементов тестирования. См. *объект тестирования.*

Эмулятор (emulator): Устройство, компьютерная программа или система, которая принимает те же самые входные данные и выдаёт те же самые выходные данные, что и данная система [IEEE 610]. См. также *имитатор*.

Эталонный тест (benchmark test): (1). Стандарт, согласно которому может производиться измерение или сравнение.
(2). Тест, который может использоваться для сравнения компонентов или систем друг с другом или на соответствие стандарту, указанному в (1). [Согласно IEEE 610]

Этап требований (requirements phase): Период в жизненном цикле программного обеспечения, в течении которого определяются и документируются требования к программному продукту. [IEEE 610]

Эффект зондирования (probe effect): Эффект, который оказывает измеряющий инструмент (например, инструмент тестирования производительности или монитор) на измеряемую систему. Для примера: производительность может быть несколько хуже в момент использования инструмента тестирования производительности

Эффективность (efficiency): Способность программного обеспечения обеспечивать необходимую производительность, относительно количества ресурсов используемых при установленных условиях. [ISO 9126]

Я

Язык описания сценариев (scripting language): Язык программирования, на котором пишутся автоматизированные сценарии тестирования для инструмента выполнения тестов (например: средства захвата/воспроизведения).

Дополнение А (Справочное)

Список источников, использованных при создании этого глоссария:

[Abbott] J. Abbot (1986), Software Testing Techniques, NCC Publications.

[Adrion] W. Adrion, M. Branstad and J. Cherniabsky (1982), Validation, Verification and Testing of Computer Software, in: Computing Surveys, Vol. 14, No 2, June 1982.

[Bach] J. Bach (2004), Exploratory Testing, in: E. van Veenendaal, The Testing Practitioner – 2nd edition, UTN Publishing, ISBN 90-72194-65-9.

[Beizer] B. Beizer (1990), Software Testing Techniques, van Nostrand Reinhold, ISBN 0-442-20672-0.

[Chow] T. Chow (1978), Testing Software Design Modelled by Finite-Sate Machines, in: IEEE Transactions on Software Engineering, Vol. 4, No 3, May 1978.

[CMM] M. Paulk, C. Weber, B. Curtis and M.B. Chrissis (1995), The Capability Maturity Model, Guidelines for Improving the Software Process, Addison-Wesley, ISBN 0-201-54664-7.

[CMMI] M.B. Chrissis, M. Konrad and S. Shrum (2004), CMMI, Guidelines for Process Integration and Product Improvement, Addison Wesley, ISBN 0-321-15496-7.

[Fenton] N. Fenton (1991), Software Metrics: a Rigorous Approach, Chapman & Hall, ISBN 0-53249-425-1.

[Fewster and Graham] M. Fewster and D. Graham (1999), Software Test Automation, Effective use of test execution tools, Addison-Wesley, ISBN 0-201-33140-3.

[Freedman and Weinberg] D. Freedman and G. Weinberg (1990), Walkthroughs, Inspections, and Technical Reviews, Dorset House Publishing, ISBN 0-932633-19-6.

[Gerrard] P. Gerrard and N. Thompson (2002), Risk-Based E-Business Testing, Artech House Publishers, ISBN 1-58053-314-0.

[Gilb and Graham] T. Gilb and D. Graham (1993), Software Inspection, Addison-Wesley, ISBN 0-201-63181-4.

[Graham] D. Graham, E. van Veenendaal, I. Evans and R. Black (2007), Foundations of Software Testing, Thomson Learning, ISBN 978-1-84480-355-2.

[Grochtmann] M. Grochtmann (1994), Test Case Design Using Classification Trees, in: Conference Proceedings STAR 1994.

[Hetzel] W. Hetzel (1988), The complete guide to software testing – 2nd edition, QED Information Sciences, ISBN 0-89435-242-3.

[McCabe] T. McCabe (1976), A complexity measure, in: IEEE Transactions on Software Engineering, Vol. 2, pp. 308-320.

[Musa] J. Musa (1998), Software Reliability Engineering Testing, McGraw-Hill Education, ISBN 0-07913-271-5.

[Myers] G. Myers (1979), The Art of Software Testing, Wiley, ISBN 0-471-04328-1.

[TMap] M. Pol, R. Teunissen, E. van Veenendaal (2002), Software Testing, A guide to the TMap Approach, Addison Wesley, ISBN 0-201-745712.

[Veenendaal] E. van Veenendaal (2004), The Testing Practitioner – 2nd edition, UTN Publishing, ISBN 90-72194-65-9.

Дополнение Б (Способы обсуждения данного глоссария)

Замечания по этому документу приветствуются, поскольку они позволят улучшить глоссарий для удовлетворения нужд сообщества тестировщиков.

В комментарий необходимо включать следующую информацию:

- Ваше имя и контактные данные;
- Номер версии глоссария (в настоящее время 2.0);
- Ссылка на комментируемую часть глоссария;
- Дополнительная информация, такая как причина предлагаемого изменения или ссылка на использование термина.

Вы можете предложить свои замечания электронной почтой на адрес konushin@istqb.org.