НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ
« КПИ»

Организационные аспекты процесса тестирования

План

1.Классификация ошибок.

2. Связь ошибки с отказом.

3. Источники ошибок.

4. Организация процесса тестирования

Объектами тестирования могут быть компоненты, групп компонентов, подсистема и система. Для каждого из них формируется стратегия проведения тестирования. Если объект тестирования относится к белому или черному «ящикам», состав компонентов которого неизвестный, то тестирование проводится посредством вводом в него входных тестовых данных для получения выходных данных. Проектировщик тестов должен заглянуть внутрь «черного ящика» и исследовать детали процессов обработки данных, вопросы обеспечения защиты и восстановления данных, а также интерфейсы с другими программами и системами. Это способствует подготовке тестовых данных для проведения тестирования. Для некоторых типов объектов группа тестирования не может сгенерировать представительное множество тестовых наборов, которые демонстрировали бы функциональную правильность работы компоненты при всех их возможных наборах тестах.

Поэтому предпочтительным является метод «белого ящика», при котором можно использовать структуру объекта для организации тестирования по различным ветвям.

Например, можно выполнить тестовые наборы, которые проходят через все операторы или все контрольные точки компоненты для того, чтобы убедиться в правильности их работы.

Классификация ошибок. Международный стандарт ANSI/IEEE–729–83 разделяет все ошибки в разработке программ на следующие

Ошибка (error) – состояние программы, при котором выдается неправильные результаты, причиной которых являются изъяны (flaw) в операторах программы или в технологическом процессе ее разработки, что приводит к неправильной интерпретации исходной информации, а следовательно и к неверному решению.

Дефект (fault) в программе является следствием ошибок разработчика на любом из этапов разработки и может содержаться в исходных или проектных спецификациях, текстах кодов программ, эксплуатационной документация и т.п. Дефект обнаруживается в процессе выполнения программы.

Отказ (failure)– это отклонение программы от функционирования или невозможность программы выполнять функции, определенные требованиями и ограничениями и рассматривается как событие, способствующее переходу программы в неработоспособное состояние из–за ошибок, скрытых в ней дефектов или сбоев в среде функционирования

Отказ может быть результатом следующих причин:

– ошибочная спецификация или пропущенное требование, т.е. спецификация точно не отражает того, что предполагал пользователь;

– спецификация может содержать требование, которое невозможно выполнить на данной аппаратуре и программном обеспечении;

– проект программы может содержать ошибки (например, база данных спроектирована без защиты от несанкционированного доступа пользователя, а требуется защита);

– программа может быть неправильной, т.е. она выполняет несвойственный алгоритм или он сделан не полностью.

Таким образом, отказы как правило, являются результатами одной или более ошибок в программе, а также наличия разного рода дефектов.

Ошибки на этапах ЖЦ тестирования.

Приведенные типы ошибок распределяются по этапам ЖЦ и им соответствуют такие источники их возникновения:

– непреднамеренное отклонение разработчиков от рабочих стандартов или планов реализации;

– спецификации функциональных и интерфейсных требований выполнены без
соблюдения стандартов разработки и т.п., что приводит к нарушению функционирования программ;

– организации процесса разработки – несовершенна или недостаточное управление руководителем проекта ресурсами (человеческими, техническими, программными и т.д.) и вопросами тестирования и интеграции элементов проекта.

Рассмотрим этапы тестирования, определенные в соответствии с рекомендациями стандарта ISO/IEC 12207, и приведем типы ошибок, которые обнаруживаются на каждом из них.

Этап разработки требований.

При определении исходной концепции системы и определении исходных требований заказчика к системе возникают ошибки
аналитиков при спецификации верхнего уровня системы и построении концептуальной модели предметной области.

Характерными ошибками этого этапа являются:

– неадекватность описания спецификациями требований конечных пользователей;

– некорректность спецификации взаимодействия ПО со средой функционирования
или с пользователями;

– несоответствие требований заказчика к отдельным и общим свойствам ПО;

– некорректность описания функциональных характеристик;

– необеспеченность инструментальными средствами поддержки всех аспектов
реализации требований заказчика и др.

Этап проектирования.

Ошибки при проектировании компонентов могут возникать при описании алгоритмов, логики управления, структур данных, интерфейсов, логики моделирования потоков данных, форматов ввода–вывода и др. В основе этих ошибок лежат дефекты спецификаций аналитиков и ошибок проектировщиков. К ним относятся ошибки, связанные с :

– определением интерфейса пользователя со средой;

– описанием функций (неадекватность целей и задач компонентов, которые
обнаруживаются при проверке комплекса компонентов);

– определением процесса обработки информации и взаимодействия между процессами (результат некорректного определения взаимосвязей компонентов и процессов);

– некорректным заданием данных и их структур при описании отдельных компонентов и ПС в целом;

– некорректным описанием алгоритмов модулей;

– определением условий возникновения возможных ошибок в программе;

– нарушением принятых для проекта стандартов и технологий.

Этап кодирования.

На данном этапе возникают ошибки, которые являются результатом дефектов проектирования, ошибок программистов и менеджеров процесса разработки и отладки. Причиной ошибок являются:

– бесконтрольность в допустимости значений входных параметров, индексов массивов, параметров циклов, выходных результатов, деления на 0 и др.;

– неправильная обработка нерегулярных ситуаций при анализе кодов возврата от вызываемых подпрограмм, функций и др.;

– нарушение стандартов кодирования (плохие комментарии, нерациональное
выделение модулей и компонент и др.);

– использование одного имени для обозначения разных объектов или разных имен для обозначения одного объекта, плохая мнемоника имен;

– несогласованное внесение изменений в программу разными разработчиками и др.

Этап тестирования.

На этом этапе ошибки допускаются тестировщиками, а также программистами при выполнении технологии сборки и тестирования, выбора тестовых наборов и сценариев тестирования и др. Отказы в программном обеспечении, вызванные такого рода ошибками, должны выявляться, устраняться и не отражаться на статистике ошибок компонент и программного обеспечения в целом.

Этап сопровождения. *На этапе сопровождения причиной ошибок являются*

недоработки и дефекты эксплуатационной документации, малые показатели

модифицируемости и удобочитаемости, а также некомпетентность лиц, ответственных за сопровождение и/или усовершенствование ПО. В зависимости от сущности вносимых изменений на этом этапе могут возникать практически любые ошибки, аналогичные ранее перечисленным ошибкам на предыдущих этапах.

Все ошибки, которые возникают в программах, принято подразделять на следующие классы:

– логические и функциональные ошибки;

– ошибки вычислений и времени выполнения;

– ошибки ввода–вывода и манипулирования данными;

Логические ошибки являются причиной нарушения логики алгоритма, внутренней несогласованности переменных и операторов, а также правил программирования.

Функциональные ошибки являются следствием неправильно определенных функций, нарушения порядка их применения или отсутствия полноты их реализации и т.д.

Ошибки вычислений возникают по причине неточности исходных данных и реализованных формул, погрешностей методов, неправильного применения операций вычислений или операндов. Ошибки времени выполнения связаны с не обеспечением требуемой скорости обработки запросов или времени восстановления программы.

Ошибки ввода–вывода и манипулирования данными являются следствием некачественной подготовки данных для выполнения программы, сбоев при занесении их в базах данных или при выборке из нее.

Ошибки интерфейса относятся к ошибкам взаимосвязи отдельных элементов друг с другом, что проявляется при передаче данных между ними, а также при
взаимодействии со средой функционирования.

Ошибки объема относятся к данным и являются следствием того, что реализованные методы доступа и размеры баз данных не удовлетворяют объемам информации системы или интенсивности ее обработки.

Приведенные основные классы ошибок свойственны разным типам компонентов ПО и проявляются они в программах по–разному. Так, при работе с БД возникают
ошибки представления и манипулирования данными, логические ошибки в задании
прикладных процедур обработки данных и др. В программах вычислительного
характера преобладают ошибки вычислений, а в программах управления и обработки – логические и функциональные ошибки. В ПО, состоящем из множества разноплановых программ реализации разных функций, могут содержаться ошибки разных типов и т.д.
Ошибки интерфейсов и нарушение объема характерны для любого ПО.
Анализ типов ошибок в программах является необходимым условием создания планов тестирования и методов тестирования для обеспечения правильности ПО.
На современном этапе развития средств поддержки разработки ПО (CASE–технологии, объектно–ориентированные методы и средства проектирования моделей и программ) проводится такое проектирование, при котором ПО защищается от наиболее типичных ошибок и тем самым предотвращается появление программных дефектов.

Связь ошибки с отказом. Наличие ошибки в программе, как правило, приводит к отказу ПО при его функционировании. Для анализа причинно–следственных связей "ошибка–отказ" существуют следующие действия:

– идентификация изъянов в технологиях проектирования и программирования;

– взаимосвязь изъянов процесса проектирования и допускаемых человеком ошибок;

– классификация отказов, изъянов и возможных ошибок, а также дефектов на каждом этапе разработки;

– сопоставление ошибок человека, допускаемых на определенном этапе разработки, и дефектов в объекте, как следствий ошибок спецификации проекта, моделей программ и т.д.);

– проверка и защита от ошибок на всех этапах ЖЦ, а также обнаружение дефектов на каждом этапе разработки;

– сопоставление дефектов и отказов в ПО для разработки системы взаимосвязей и методики локализации, сбора и анализа информации об отказах и дефектах;

– разработка подходов к документированию процессов и испытания ПО.

Конечная цель причинно–следственных связей "ошибка–отказ" заключается в определении методов и средств тестирования и обнаружения ошибок определенных классов, а также критериев завершения тестирования на множестве наборов данных; в определении путей совершенствования организации процесса разработки, тестирования и сопровождения ПО.

Приведем следующую классификацию типов отказов:
– аппаратный, при котором общесистемное ПО не работоспособно;
– информационный, вызванный ошибками во входных данных и передаче данных по каналам связи, сбое устройств ввода (следствие аппаратных отказов);
– эрготический, вызванный ошибками оператора при его взаимодействии с машиной (этот отказ является вторичным отказам и может привести к информационному или функциональному отказам);
– программный при наличии ошибок в компонентах и др.
Некоторые ошибки могут быть следствием недоработок при определении требований, проекта, генерации выходного кода или документации, С другой стороны, они порождаются в процессе разработки программы или при разработке интерфейсов отдельных элементов программы (нарушение порядка параметров, меньше или больше параметров и т.п.). На рис. показаны подобные случаи ошибочных ситуаций и причин их появления в каждой разработке.

Источники ошибок. Ошибки могут быть порождены в процессе разработки проекта, компонент, кода и документации. Как правило, они обнаруживаются при выполнении или сопровождении программного обеспечения в самых неожиданных и разных ее точках.

Некоторые ошибки в программе могут быть следствием недоработок при определении требований, проекта, генерации кода или документации. С другой стороны, ошибки порождаются в процессе разработки программы или интерфейсов ее элементов (например, при нарушении порядка задания параметров связи – меньше или больше, чем требуется и т.п.).

Причиной появления ошибок зачастую является: непонимание требований заказчика;

неточная спецификация требований в документах проекта и др. Это приводит к тому, что реализуются некоторые функции системы, которые будут работать не так, как предлагает заказчик. В связи с этим проводится совместное обсуждение непонимания некоторых деталей требований для их уточнения заказчика и разработчика.

Команда разработчиков системы может также изменить синтаксис и семантику описания системы. Однако некоторые ошибки могут быть не обнаружены (например, неправильно заданы индексы или значения переменных этих операторов).

Исходя из того, что каждая организация по разработке ПО (особенно общесистемного назначения), сталкивается с проблемами нахождения ошибок, ей приходится классифицировать типы обнаруживаемых ошибок и определять свое отношение к этим ошибкам.

На основе многолетней деятельности в области создания ПО разные фирмы создали свою классификацию ошибок, основанную выявлении причин их появления в процессе разработки, функциях, в областях появлении ошибок в ПО. Известно много различных подходов к классификации ошибок, рассмотрим некоторые из них.

Фирма IBM разработала подход к классификации ошибок, называемый

ортогональной классификацией дефектов. Подход предусматривает разбиение ошибок по категориям с ответственностью разработчиков за них. Схема классификации является продукто– и организационно–независимой и может применятся ко всем стадиям разработки ПО разного назначения. Табл.1 дает список ошибок согласно данной классификации. В ней разработчику предоставляется возможность идентифицировать не только типы ошибок, но и места, где пропущены или совершенны ошибки, а также неинициализированная переменная или инициализированной переменной присвоено неправильное значение.

Организация процесса тестирования

Все способы тестирования ПС объединяются базой данных, где помещаются
результаты тестирования системы. В ней содержатся все компоненты, тестовые контрольные данные, результаты тестирования и информация о документировании процесса тестирования.

База проекта поддерживается специальными инструментальными средствами типа CASE, которые обеспечивают ведение анализа ПрО, сборку данных об их объектах, потоках данных и тому подобное. База проекта хранит также начальные и эталонные данные, которые используются для сопоставления данных, накопленных в базе с данными, которые получены при тестировании.

При тестировании выполняются разные виды расчета характеристик этого процесса и способы планирования и управления:

1. Расчет продолжительности выполнения функций путем сбора средних значений о скорости выполнения операторов без выполнения программы на машине. Выявляются компоненты, которые требуют большого времени выполнения в реальной среде.

2. Управления выполнением состоит в организации подбора тестов проверки, их выполнении, селекции результатов тестирования и проведении сопоставления с эталонными значениями. Результаты процесса отображаются на дисплеи, например, в графической форме (пути прохождения по графу программы), в виде последовательности диаграмм UML, а также в виде информации об отказах и ошибках или конкретных значениях исходных параметров программы. Эти данные анализируются разработчиками для формулирования выводов о направлениях дальнейшей проверки правильности программы или их завершение.

3. Планирование тестирования предназначено для распределения сроков работ по тестированию, распределения тестировщиков по отдельным видам работ и составления ими тестов проверки работы системы. Практически определяется стратегия и пути тестирования. В диалоге запрашиваются реальные значения процесса выполнения или выдачи структуры о разветвления вершин графа и параметров циклов. Проверенные циклы, как правило, изымаются из путей выполнения программы. При планировании
путей выполнения создаются соответствующие тесты, критерии и входные значения.

4. Документирование результатов тестирования в соответствии с действующим
стандартом ANSI/IEEE 829, включает описание:

– задач, назначение и содержание ПС, а также описание функций соответственно требованиям заказчика;

– технологии разработки системы;

– планов тестирования различных объектов, необходимых ресурсов, соответствующих специалистов для проведения тестирования и технологических способов;

– тестов, контрольных примеров, критериев и ограничений оценки результатов
программного продукта, а также процесса тестирования;

Спасибо за внимание!