**Способы обеспечения качества программных продуктов.**

**Введение.**

Всем, кто когда-либо работал с программным обеспечением производства Microsoft, IBM, Novell или другой компании, приходилось решать проблемы различной степени сложности, вызванные сбоями в работе ПО. Производство современного ПО происходит на фоне высоких требований, предъявляемых к качеству создаваемых программ и значительной сложности выполняемых ими функций. Для обеспечения надежности программных продуктов приходится выявлять ошибки на всех этапах проектирования, начиная с этапа анализа требований и заканчивая устранением ошибок на стадии сопровождения. И если на этапе анализа требований стремятся исключить логические ошибки той деятельности, под которую пишется ПО, то на остальных этапах от целого ряда ошибок стремятся избавится комплексом мер, начиная от декомпозиций и заканчивая “Идеальной” технологией программирования - технология, которая по некоторому достаточно неформальному описанию объекта программирования автоматически генерирует текст синтаксически и семантически корректной программы.

Компании-производители пытаются повысить качество программных продуктов с помощью тестирования. Существуют тестовые методики, использующие специальные драйверы, автоматически проверяющие промежуточные версии разрабатываемого продукта на различном аппаратном обеспечении. Другой способ проверки - бета-тестирование. В этом случае разработчики программного обеспечения разрешают пользователям попробовать предварительные версии продуктов.

**Способы обеспечения качества программных продуктов.**

**Стандартизация.**

Непрерывный рост требований к качеству ПС стимулирует создание и активное применение международных стандартов и регламентированных технологий, автоматизирующих основные процессы их жизненного цикла, начиная с инициирования проекта. Это привело к существенному изменению в последние годы объектов, методологии и культуры в области создания и развития ПС и БД.

МС ИСО серии 9000 определяют и регламентируют создание, развитие, применение и сертификацию систем качества любых предприятий, независимо от их назначения. Они содержат и развивают основное положение, что "поставщик должен разработать и поддерживать в рабочем состоянии документально оформленную систему качества как средство, обеспечивающее соответствие продукции установленным требованиям заказчика". В них зафиксировано право заказчика на инспекцию системы качества предприятия поставщика до заключения контракта на поставку продукции.

В последние годы сформировалась комплексная система управления качеством продукции TQM (Totaly Quality Management), которая концептуально близка к предшествующей более общей системе на основе стандартов ИСО серии 9000. Система ориентирована на удовлетворение требований потребителя, на постоянное улучшение процессов производства или проектирования, на управление процессами со стороны руководства предприятия на основе фактического состояния проекта. Основные достижения TQM состоят в углублении и дифференциации требований потребителей по реализации процессов, их взаимодействию и обеспечению качества продукции. Системный подход поддержан рядом специализированных инструментальных средств, ориентированных на управление производством продукции. Поэтому эта система пока не находит применения в области обеспечения качества жизненного цикла программных средств.

Применение этого комплекса может служить основой для систем обеспечения качества программных средств, однако требуется корректировка, адаптация или исключение некоторых положений стандартов применительно к принципиальным особенностям технологий и характеристик этого вида продукции. Кроме того, при реализации систем качества ПС необходимо привлечение ряда стандартов, формально не относящихся к этой серии и регламентирующих показатели качества, жизненный цикл, верификацию и тестирование, испытания, документирование и другие особенности ЖЦ комплексов программ.

В России в области обеспечения жизненного цикла и качества сложных комплексов программ существует и применяется очень небольшая группа устаревших стандартов серий ГОСТ 19.ХХХ и ГОСТ 34.ХХХ. Предприятия, выполняющие государственные заказы при создании ПС для внутреннего применения, вынуждены использовать эти стандарты. Однако в экспортных заказах зарубежные клиенты требуют соответствия технологии проектирования, производства и качества продукции современным международным стандартам. Это противоречие особенно обострилось в государственных заказах Минобороны России, результаты которых одновременно используются и внутри, и вне страны.

По требованиям Минобороны России для применения международных стандартов в таких заказах их необходимо перевести на русский язык, адаптировать, юридически утвердить и издать в качестве ГОСТ Р, на что требуется несколько лет. Однако при поставках за рубеж многие стандарты допускается практически применять на языке оригинала или в виде перевода до утверждения Госстандартом России, что вполне удовлетворяет зарубежных заказчиков, но не допускается отечественными органами сертификации для продукции внутри страны.

К настоящему моменту ряд отечественных руководителей убедился, что для обеспечения высокого качества, надежности функционирования и безопасности применения сложных комплексов программ целесообразно выделять специалистов, ответственных за соблюдение технологии создания и развития программ, за обеспечение и контроль качества, а также за надежность и безопасность проекта ПС в целом и его компонентов. Обеспечение качества должно реализовываться специалистами в ЖЦ программных средств на основе использования современной методологии, технологического инструментария, стандартов и нормативных документов. Для систематической координированной борьбы с угрозами качеству необходимо проводить исследования конкретных факторов, влияющих на качество функционирования и безопасность применения программ со стороны, реально существующих и потенциально возможных дефектов в создаваемых комплексах программ.

Разработка и сопровождение сложных ПС на базе современных технологий позволяет предупреждать и устранять наиболее опасные системные и алгоритмические ошибки на ранних стадиях проектирования, а также использовать неоднократно проверенные в других проектах программные и информационные компоненты высокого качества. Для обнаружения и устранения ошибок проектирования все этапы разработки и сопровождения ПС должны быть поддержаны методами и средствами систематических, автоматизированных верификации, тестирования и испытаний. При разработке ПС целесообразно применять различные методы, эталоны и виды тестирования, каждый из которых ориентирован на обнаружение, локализацию или диагностику определенных типов дефектов.

Удостоверение достигнутого качества функционирования сложных критических ПС и процессов их жизненного цикла должно базироваться на сертификатах, выданных аттестованными проблемно-ориентированными испытательными лабораториями. Сертификация систем качества предприятий - разработчиков ПС по МС ИСО серии 9000 - позволяет заказчикам и покупателям выбирать из них наиболее надежных партнеров для реализации информационных систем, способных гарантировать высокое качество поставляемых и используемых комплексов программ. Применение сертифицированных систем качества предприятий не только гарантирует высокое, устойчивое качество проектирования и обеспечение жизненного цикла ПС, но позволяет во многих случаях не проводить или сокращать сертификацию конечного продукта. Базой такой сертификации предприятий, разрабатывающих программные средства, может служить комплекс стандартов, нормативных и инструктивных документов, представленных на схеме. Краткое содержание стандартов ИСО изложено в, а полные тексты в.

Необходимость анализа и развития сертификации программ как самостоятельной проблемы обусловлена специфическими стандартами и особенностями их жизненного цикла. К ним относятся, с одной стороны, объективная необходимость удостоверения качества и потребительских свойств продукции, активизация деятельности в этом направлении на международном уровне. С другой, отсутствие в государственных и международных стандартах количественных требований к информационным системам, широкое многообразие классов и видов программ и баз данных, обусловленное различными функциями ИС, предопределяет формальные трудности, связанные с процедурами доказательства соответствия ПС и БД условиям контрактов и требованиям потребителей. Поэтому основой сертификации должны быть детальные и эффективные методики испытаний систем качества и конкретных ПС, специально разработанные тестовые задачи и генераторы для их формирования, а также квалификация и авторитет испытателей. Для этого заказчики должны выбирать подрядчиков - исполнителей своих проектов, имеющих системы обеспечения качества программных средств для ИС и сертификаты, удостоверяющие реализацию и применение системы качества предприятием-разработчиком.

В отечественных ИС все больше применяются программные компоненты зарубежных фирм, которые также не могут быть абсолютно гарантированы от проявления дефектов проектирования, программирования и документации. Для обеспечения требуемого качества функционирования комплексов программ с использованием импортных компонентов следует закупать только лицензионно-чистые продукты, поддерживаемые гарантированным сопровождением конкретных поставщиков. Эти компоненты должны сопровождаться полной эксплуатационной и технологической документацией, сертификатом соответствия и комплектами тестов. В контрактах на закупку должны специально фиксироваться обязательства поставщиков по сопровождению и замене версий ПС при выявлении дефектов или совершенствовании функций. Все версии зарубежных ПС следует проверять на качество функционирования в конкретном окружении ИС путем повторных испытаний или отдельными проверками, подтверждающими зарубежные сертификат и эксплуатационную документацию.

**Обеспечение надежности на этапе кодирования и компиляции программного обеспечения.**

Разработка любого программного средства может быть представлена как процесс, состоящий из ряда последовательных преобразований одного описания решаемой задачи в другое, начиная от постановки задачи и заканчивая программой, реализованной в кодах конкретной ЭВМ.

Все время существования программного средства от зарождения идеи по его созданию, до завершения его эксплуатации, обычно определяют как жизненный цикл. Укрупнено можно выделить пять наиболее важных этапов жизненного цикла программного средства (ЖЦ ПС): спецификацию (10%), проектирование (10%), кодирование (10%), отладку (20%) и сопровождение (50%). В скобках записаны относительные затраты ресурсов на создание ПС.

По затратам времени, человеческих и машинных ресурсов все эти этапы не одинаковы. Наиболее “дорогими”, в этом смысле, являются этапы, связанные с поиском ошибок в программах. Затраты ресурсов на них могут быть равными, или даже превосходить совокупные затраты ресурсов на остальных этапах. В стандарте DOD‑STD‑2167‑A около 30% требований, документов и соответствующих им процессов непосредственно связаны с отладкой, тестированием и испытаниями программ. Данный стандарт является обязательным при выполнении заказов Министерства обороны США.

Эти затраты быстро увеличиваются при возрастании требований к качеству ПС. По оценкам, приведенным в работе, на долю устранения ошибок и сопровождение ПС приходится почти 75% всех затрат. Следует учитывать, что значительная часть работ, выполняемых на этапе сопровождения, связана с поиском и устранением оставшихся в программе ошибок.

Ретроспектива развития методов и средств автоматизации программирования в этом отношении говорит сама за себя. В модульном программировании акцент делается на разбиение программы на модули таким образом, чтобы данные (обрабатываемые модулем) были спрятаны в нем. Эта доктрина, известная как “принцип ограничения доступа к данным”, в значительной степени повысила модифицируемость и эффективность порождаемого кода.

Эволюция техники модульного программирования привела к появлению объектно-ориентированного стиля программирования, который во многом унифицировал процесс создания ПС. К достоинствам этого метода относится то, что в нем более полно реализуется технология структурного программирования, облегчается процесс создания сложных иерархических систем, появляется удобная возможность создания пользовательских библиотек объектов в различных областях применения.

В 80-х годах исследование причин неудач при реализации больших программных проектов показало, что число ошибок в спецификациях на программы значительно превышает их количество в программных кодах. Так около 56% ошибок допускаются на этапе формулировки требований к программе при этом расходуется в среднем 82% всех усилий, затраченных коллективом на устранение ошибок проекта. В то время как на этапе кодирования программ допускается соответственно 7% ошибок и тратится 1% усилий на их ликвидацию. В это время формулируется тезис о том, что целью программирования является не порождение программы как таковой, а создание технологических условий, когда разрабатываемое программное обеспечение легко адаптируется к новым обстоятельствам и новому пониманию решаемой задачи. Р. Хемминг так формулирует этот тезис: “Здравая вычислительная практика требует постоянного исследования изучаемой задачи не только перед организацией вычислений, но также в процессе его развития и особенно на той стадии, когда полученные числа переводятся обратно и истолковываются на языке первоначальной задачи”.

Перечисленные выше причины привели в середине 80-х годов к осознанию необходимости реализации интегрированного окружения поддержки всего жизненного цикла ПС и, в первую очередь, этапа проектирования ПС, что обусловило появление инструментальных средств автоматизации проектирования программных систем (CASE-технологий).

Первоначально CASE-средства были ориентированы на решение задач автоматизированного сбора информации по предметной области и проектированию будущего ПС, что позволяет экономить время при создании ПС за счет более тщательного анализа исходных требований и лучшего начального планирования программы. Впоследствии в CASE-средствах 2-го поколения полностью или частично были автоматизированы такие важные составляющие жизненного цикла ПС как моделирование информации предметной области; программирование; тестирование, отладка ПС и измерение качества; поддержка документирования; сопровождение.

Применение CASE-инструментов позволяет в значительной степени снизить трудоемкость создания ПС, а в отдельных случаях заменить программирование автоматическим синтезом программ.

Таким образом, развитие методов автоматизации разработки ПС происходит на различных основах (модульное программирование, объектно-ориентированный подход, логическое программирование, CASE-технологии), которые так или иначе развивают концепции структуризации в программировании. Структуризация способствует проведению эффективной декомпозиции проекта, что позволяет получать как целостное представление о ПС, так и его деталях. Однако, несмотря на многочисленные разработки в этой области, в целом проблема автоматизации разработки ПС остается нерешенной по многим причинам как методологического, так и практического характера.

В последнее время в связи с совершенствованием технических средств отображения информации утверждается новый графический подход к решению проблемы автоматизации разработки ПС, основанный на идее привлечения визуальных форм представления программ, в большей степени соответствующих образному способу мышления человека. Применение графических методов обещает кардинально повысить производительность труда программиста. Кроме того, графическая форма записи по сравнению с текстовым представлением программ обеспечивает более высокий уровень их структуризации, соблюдение технологической культуры программирования, предлагает более надежный стиль программирования.

Одним из основных факторов повышения эффективности и надежности программирования можно считать придание образности формам спецификации данных и описания алгоритма. В этом смысле главный недостаток существующих технологий программирования заключается в преимущественно текстовых формах представления основных компонент программы, что делает программу невыразительной и чрезвычайно затрудняет ее восприятие человеком.

В настоящее время принципиально изменилась роль информационных технологий в обществе. С одной стороны, программные продукты представляют собой достаточно дорогостоящий товар. С другой стороны, ужесточились требования к качеству ПС, поскольку последние широко используются в таких сферах деятельности человека, которые традиционно относят к группе риска. Это аэрокосмические технологии, энергетика, связь, бизнес и т.д.

Современные информационные технологии играют исключительно важную роль в интеллектуализации общества, позволяют активизировать и эффективно использовать информационные ресурсы общества. Это приводит к необходимости создания эффективных методов достижения заданного качества разрабатываемых ПС в условиях ограниченности ресурсов, выделяемых на разработку.

Высокое качество ПС достигается либо методами безошибочного программирования (“пассивными” методами), либо путем выявления и устранения ошибок (“активными” методами).

Современные методы безошибочного программирования основываются на реализации процессов автоматизации всех этапов жизненного цикла ПС от проектирования и кодирования программ до документирования и их сопровождения. К таким средствам относятся: CASE-средства, объектно-ориенированное программирование, методы логического программирования. Особое место занимают методы визуального программирования, поскольку приближение формы представления программы и способов ее кодирования к образному способу мышления человека в значительной степени сокращает число ошибок, допускаемых человеком при разработке программ и повышает надежность программирования.

Активные методы повышения надежности ПС совершенствуются за счет развития средств автоматизации тестирования программ. Сложность ПС и высокие требования по их надежности требуют выработки принципов структурного построения сложных программных средств, обеспечивающих гибкость модификации ПС и эффективность их отладки. К таким принципам в работе относят:

- модульность и строгую иерархию в структурном построении программ;

- унификацию правил проектирования, структурного построения и взаимодействия компонент ПС;

- унификацию правил организации межмодульного интерфейса;

- поэтапный контроль полноты и качества решения функциональных задач.

**Тестирование программного обеспечения.**

Многие организации, занимающиеся созданием программного обеспечения, до 50% средств, выделенных на разработку программ, тратят на тестирование, что составляет миллиарды долларов по всему миру в целом. И все же, несмотря на громадные капиталовложения, знаний о сути тестирования явно не хватает и большинство программных продуктов неприемлемо ненадежно даже после «основательного тестирования».

О состоянии дел лучше всего свидетельствует тот факт, что большинство людей, работающих в области обработки данных, даже не может правильно определить слово «тестирование», и это на самом деле главная причина неудач.

«Тестирование — процесс, подтверждающий правильность программы и демонстрирующий, что ошибок в программе нет.» Основной недостаток подобного определения заключается в том, что оно совершенно неправильно; фактически это почти определение антонима слова «тестирование». Человек с некоторым опытом программирования уже, вероятно, понимает, что невозможно продемонстрировать отсутствие ошибок в программе. Поэтому определение описывает невыполнимую задачу, а так как тестирование зачастую все же выполняется с успехом, по крайней мере с некоторым успехом, то такое определение логически некорректно. Правильное определение тестирования таково: Тестирование — процесс выполнения программы с намерением найти ошибки.

Невозможно гарантировать отсутствие ошибок в нетривиальной программе; в лучшем случае можно попытаться показать наличие ошибок. Если программа правильно ведет себя для солидного набора тестов, нет основании утверждать, что в ней нет ошибок; со всей определенностью можно лишь утверждать, что не известно, когда эта программа не работает. Конечно, если есть причины считать данный набор тестов способным с большой вероятностью обнаружить все возможные ошибки, то можно говорить о некотором уровне уверенности в правильности программы, устанавливаемом этими тестами.

Психологические эксперименты показывают, что большинство людей, поставив цель (например, показать, что ошибок нет), ориентируется в своей деятельности на достижение этой цели. Тестовик подсознательно не позволит себе действовать против цели, т. е. подготовить тест, который выявил бы одну из оставшихся в программе ошибок. Поскольку мы все признаем, что совершенство в проектировании и кодировании любой программы недостижимо и поэтому каждая программа содержит некоторое количество ошибок, самым плодотворным применением тестирования будет найти некоторые из них. Если мы хотим добиться этого и избежать психологического барьера, мешающего нам действовать против поставленной цели, наша цель должна состоять в том, чтобы найти как можно больше ошибок. Сформулируем основополагающий вывод:

Если ваша цель — показать отсутствие ошибок, вы. их найдете не слишком много. Если же ваша цель — показать наличие ошибок, вы найдете значительную их часть.

Надежность невозможно внести в программу в результате тестирования, она определяется правильностью этапов проектирования. Наилучшее решение проблемы надежности — с самого начала не допускать ошибок в программе. Однако вероятность того, что удастся безупречно спроектировать большую программу, бесконечно мала. Роль тестирования состоит как раз в том, чтобы определить местонахождение немногочисленных ошибок, оставшихся в хорошо спроектированной программе. Попытки с помощью тестирования достичь надежности плохо спроектированной программы совершенно бесплодны.

Тестирование оказывается довольно необычным процессом (вот почему оно и считается трудным), так как этот процесс разрушительный. Ведь цель проверяющего (тестовика) — заставить программу сбиться. Он доволен, если это ему удается; если же программа на его тесте не сбивается, он не удовлетворен.

Еще одна причина, по которой трудно говорить о тестировании — это тот факт, что о нем известно очень немногое. Если сегодня мы располагаем 5% тех знании о проектировании и собственно программировании (кодировании), которые будут у нас к 2000 г., то о тестировании нам известно менее 1%.

Хотя в тестировании можно выделить несколько различных процессов, такие термины, как тестирование, отладка, доказательство, контроль и испытание, часто используются как синонимы и, к сожалению, для разных людей имеют разный смысл. Хотя стандартных, общепринятых определений этих терминов нет, попытка сформулировать их была предпринята на симпозиуме по тестированию программ. Классификацию различных форм тестирования мы начнем с того, что дадим эти определения, слегка их дополнив и расширив их список.

Тестирование (testing), как мы уже выяснили,—процесс выполнения программы (или части программы) с намерением (или целью) найти ошибки.

Доказательство (proof) — попытка найти ошибки в программе безотносительно к внешней для программы среде. Большинство методов доказательства предполагает формулировку утверждений о поведении программы и затем вывод и доказательство математических теорем о правильности программы. Доказательства могут рассматриваться как форма тестирования, хотя они и не предполагают прямого выполнения программы. Многие исследователи считают доказательство альтернативой тестированию — взгляд во многом ошибочный; более подробно это обсуждается в гл. 17.

Контроль (verification) — попытка найти ошибки, выполняя программу в тестовой, или моделируемой, среде.

Испытание (validation) — попытка найти ошибки, выполняя программу в заданной реальной среде.

Аттестация (certification) — авторитетное подтверждение правильности программы, аналогичное аттестации электротехнического оборудования Underwriters Laboratories. При тестировании с целью аттестации выполняется сравнение с некоторым заранее определенным стандартом.

Отладка (debugging) не является разновидностью тестирования. Хотя слова «отладка» и «тестирование» часто используются как синонимы, под ними подразумеваются разные виды деятельности. Тестирование — деятельность, направленная на обнаружение ошибок; отладка направлена на установление точной природы известной ошибки, а затем — на исправление этой ошибки. Эти два вида деятельности связаны — результаты тестирования являются исходными данными для отладки.

Тестирование модуля, или автономное тестирование (module testing, unit testing) — контроль отдельного программного модуля, обычно в изолированной среде (т. е. изолированно от всех остальных модулей). Тестирование модуля иногда включает также математическое доказательство.

Тестирование сопряжении (integration testing) — контроль сопряжении между частями системы (модулями, компонентами, подсистемами).

Тестирование внешних функций (external function testing) — контроль внешнего поведения системы, определенного внешними спецификациями.

Комплексное тестирование (system testing) — контроль и/или испытание системы по отношению к исходным целям. Комплексное тестирование является процессом контроля, если оно выполняется в моделируемой среде, и процессом испытания, если выполняется в среде реальной, жизненной.

Тестирование приемлемости (acceptance testing) — проверка соответствия программы требованиям пользователя.

Тестирование настройки (installation testing) — проверка соответствия каждого конкретного варианта установки системы с целью выявить любые ошибки, возникшие в процессе настройки

**Бета - тестирование программного обеспечения.**

Другой способ проверки - бета-тестирование. В этом случае разработчики программного обеспечения разрешают пользователям попробовать предварительные версии продуктов. Однако большинство разработчиков, с утверждают, что внешние бета-тестеры не выявляют такого большого количества ошибок.

**Выводы.**

Быстрое увеличение сложности и размеров современных комплексов программ при одновременном повышении ответственности выполняемых функций резко повысило требования со стороны пользователей к их качеству, надежности функционирования и безопасности применения. Для каждого проекта ПС, выполняющего ответственные функции, должны разрабатываться и применяться система качества, специальные планы и программа, методология и инструментальные средства, обеспечивающие требуемые качество, надежность и безопасность функционирования. Для удовлетворения высоких требований к функционированию необходимы выделение из ЖЦ ПС задач и работ по обеспечению качества программ, а также обучение и концентрация усилий разработчиков на анализе и обосновании рентабельности выбранной методологии и методов разработки комплексов программ.

Широкий спектр требований к качеству, в зависимости от назначения и области применения ПС, приводит к необходимости адаптации стандартов, регламентирующих системы качества предприятий-разработчиков. Последовательная детализация рекомендаций базовых стандартов должна доводиться до формирования должностных инструкций специалистам, образуя в совокупности иерархический комплекс нормативных документов системы качества предприятия, обеспечивающий жизненный цикл сложных программных средств.

Только скоординированное, комплексное применение в проектах ПС с начала проектирования современных методов и стандартов позволяет достигать высокого качества, необходимого для использования ПС в распределенных критических и сложных системах обработки информации. Необходимо убедить руководителей проектов, заказчиков и разработчиков в том, что тщательно регламентированное и достаточно полное системное проектирование ПС и БД на основе современных методов и международных стандартов выгодно с позиции сокращения ошибок и повышения качества сложных комплексов программ.

**Список литературы**

Крайер Э. Успешная сертификация на соответствие нормам ИСО серии 9000: Пер. с нем. - М.: ИздАТ, 1999.

2. Сборник действующих международных стандартов ИСО серии 9000. Т. 1-3. - М.: ВНИИКИ, 1998.

3. Encyclopedia of Software Engineering. Vol.1 A-N; Vol. 2 O-Z. Editor - In - Chief John J. Marciniak. John Wiley & Sons.Inc. 1995.

4. Глудкин О.П. и др. Всеобщее управление качеством: Учебник для вузов. - М.: Радио и связь, 1999.

5. Гличев А.В. Основы управления качеством продукции. - М.: МАИ, 1998.

6. Круглов М.Г., Шишков Г.М. Управление качеством TQM. - М.: МГТУ "Станкин", 1999.

7. Липаев В.В. Сертификация систем качества предприятий, разрабатывающих программные средства для информационных систем, на соответствие стандартам серии ИСО 9000 // Информационные технологии. - 1999. - ╧ 12.