Пахаев Александр Алексеевич
студент
Попов Федор Алексеевич
д-р техн. наук, профессор,
заместитель директора по ИТ
Бийский технологический институт (филиал)
ФГБОУ ВПО «Алтайский государственный
технический университет им. И.И. Ползунова»
г. Бийск, Алтайский край

ИССЛЕДОВАНИЕ И РЕАЛИЗАЦИЯ МЕТОДОВ РАСЧЕТА НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ

Аннотация: *в данной статье рассмотрены методы оценки надежности ПО, а также ряд дестабилизирующих факторов, оказывающих влияние на качество ПО. Проанализированы особенности для описания надежности ПО, отмечены свойства ПО, составляющие понятие надежности.*

Ключевые слова: *программирование, методы оценки надежности, ошибка, угроза, дестабилизирующие факторы.*

Цель программирования – описание процессов обработки данных (процессов). Согласно ИФИПа [1]: данные – это представление фактов и идей в формализованном виде, пригодном для передачи и переработке в некоем процессе, а информация – это смысл, который придается данным при их представлении. Обработка данных – это выполнение систематической последовательности действий с данными. Данные представляются и хранятся на т.н. носителях данных. Совокупность носителей данных, используемых при какой-либо обработке данных, будем называть информационной средой. Набор данных, содержащихся в какой-либо момент в информационной среде, будем называть состоянием этой информационной среды. Процесс можно определить, как последовательность сменяющих друг друга состояний некоторой информационной среды.

Описать процесс – означает определить последовательность состояний заданной информационной среды. Если мы хотим, чтобы по заданному описанию требуемый процесс порождался автоматически на каком-либо компьютере, необходимо, чтобы это описание было формализованным. Такое описание называется программой. С другой стороны, программа должна быть понятной и человеку, т.к. при разработке программ, и при их использовании часто приходится выяснять, какой именно процесс она порождает. Поэтому программа составляется на удобном для человека формализованном языке программирования, с которого она автоматически переводится на язык соответствующего компьютера с помощью другой программы, называемой транслятором. Человеку (программисту), прежде чем составить программу на удобном для него языке программирования, приходится проделывать большую подготовительную работу по уточнению постановки задачи, выбору метода ее решения, выяснению специфики применения требуемой программы, прояснению общей организации разрабатываемой программы и многое другое. Использование этой информации может существенно упростить задачу понимания программы человеком, поэтому весьма полезно ее как-то фиксировать в виде отдельных документов (часто не формализованных, рассчитанных только для восприятия человеком).

Обычно программы разрабатываются в расчете на то, чтобы ими могли пользоваться люди, не участвующие в их разработке (их называют пользователями). Для освоения программы пользователем помимо ее текста требуется определенная дополнительная документация. Программа или логически связанная совокупность программ на носителях данных, снабженная программной документацией, называется программным средством (ПС). Программа позволяет осуществлять некоторую автоматическую обработку данных на компьютере. Программная документация позволяет понять, какие функции выполняет та или иная программа ПС, как подготовить исходные данные и запустить требуемую программу в процесс ее выполнения, а также: что означают получаемые результаты

(или каков эффект выполнения этой программы). Кроме того, программная документация помогает разобраться в самой программе, что необходимо, например, при ее модификации.

Таким образом актуальность данной статьи состоит в том, что иногда возникает необходимость в оценке надежности программного обеспечения (ПО).

Можно считать, что продуктом технологии программирования является ПС, содержащее программы, выполняющие требуемые функции. Здесь под «программой» часто понимают правильную программу, т.е. программу, не содержащую ошибок. Однако понятие ошибки в программе трактуется в среде программистов неоднозначно. Согласно Майерсу [2] будем считать, что в программе имеется ошибка, если она не выполняет того, что разумно ожидать от нее пользователю. «Разумное ожидание» пользователя формируется на основании документации по применению этой программы. Следовательно, понятие ошибки в программе является существенно не формальным. В этом случае правильнее говорить об ошибке в ПС. Разновидностью ошибки в ПС является несогласованность между программами ПС и документацией по их применению. В работе [3] выделяется в отдельное понятие частный случай ошибки в ПС, когда программа не соответствует своей функциональной спецификации (описанию, разрабатываемому на этапе, предшествующему непосредственному программированию). Такая ошибка в указанной работе называется дефектом программы. Однако выделение такой разновидности ошибки в отдельное понятие вряд ли оправданно, так как причиной ошибки может оказаться сама функциональная спецификация, а не программа.

В связи с тем, что задание на ПС обычно формулируется не формально, а также из-за неформализованности понятия ошибки в ПС, нельзя доказать формальными методами (математически) правильность ПС. Нельзя показать правильность ПС и тестированием: как указал Дейкстра [4], тестирование может лишь продемонстрировать наличие в ПС ошибки. Поэтому понятие правильной ПС неконструктивно в том смысле, что после окончания работы над созданием ПС мы не сможем убедиться, что достигли цели.

Альтернативой правильного ПС является надежное ПС. Надежность ПС – это его способность безотказно выполнять определенные функции при заданных условиях в течение заданного периода времени с достаточно большой вероятностью [5]. При этом под отказом в ПС понимают проявление в нем ошибки [2]. Таким образом, надежная ПС не исключает наличия в ней ошибок – важно лишь, чтобы эти ошибки при практическом применении этого ПС в заданных условиях проявлялись достаточно редко. Убедиться, что ПС обладает таким свойством можно при его испытании путем тестирования, а также при практическом применении. Таким образом, фактически мы можем разрабатывать лишь надежные, а не правильные ПС.

Разрабатываемая ПС может обладать различной степенью надежности. Так же как в технике, степень надежности можно характеризовать [2] вероятностью работы ПС без отказа в течении определенного периода времени. Однако в силу специфических особенностей ПС определение этой вероятности наталкивается на ряд трудностей по сравнению с решением этой задачи в технике.

При оценке степени надежности ПС следует также учитывать последствия каждого отказа. Некоторые ошибки в ПС могут вызывать лишь некоторые неудобства при его применении, тогда как другие ошибки могут иметь катастрофические последствия, например, угрожать человеческой жизни. Поэтому для оценки надежности ПС иногда используют дополнительные показатели, учитывающие стоимость (вред) для пользователя каждого отказа.

Рассмотрим методы оценки надежности, начиная с классического.

Так как при основном соединении элементов работоспособное состояние системы имеет место при совпадении работоспособных состояний всех элементов, то вероятность этого состояния системы определяется произведением вероятностей работоспособных состояний всех элементов. Если состав системы представлен в виде n числа последовательно включенных элементов, то при вероятности безотказной работы каждого из элементов pi(t) вероятность безотказной работы системы (1).

$$P_C(t) = p_1(t)p_2(t) \dots p_n(t) = \prod_{i=1}^{n} p_i(t) \qquad (1)$$

При параллельном соединении элементов и при условии, что для работы системы хватает работы одного из включенных параллельно элементов, отказ системы – совместное событие, имеющее место при отказе всех параллельно включенных элементов. Если параллельно включены m элементов и вероятность отказа каждого из них $qj\ (t)\ =\ 1\text{-}pj\ (t)$, то вероятность отказа этой системы

$$Q_P(t) = q_1(t)q_2(t) \dots q_m(t) = \prod_{j=1}^{m} q_j(t) \qquad (2)$$

Если в состав структурной схемы надежности системы входит последовательное и параллельное соединение элементов, то расчет ее надежности можно произвести с использованием (1), (2).

Для определения значения средней наработки системы до отказа и других показателей надежности, необходимо знать законы распределения времени безотказной работы элементов (наработки до отказа) системы. Так как на участке нормальной эксплуатации с удовлетворительной точностью в качестве закона распределения времени безотказной работы элементов можно принять экспоненциальный закон, то при основном соединении элементов, если $p_i(t) = e^{-\lambda_i t}$, то выражение (1) будет иметь следующий вид:

$$P_C(t) = e^{-\lambda_1 t} e^{-\lambda_2 t} \dots e^{-\lambda_n t} = e^{-\lambda_C t}, \qquad (3)$$

где $\lambda_C = \lambda_1 + \lambda_2 + \cdots + \lambda_n = \sum_{i=1}^{n} \lambda_i$.

Так, при основном соединении элементов, имеющих экспоненциальный закон распределения времени безотказной работы, закон распределения времени безотказной работы системы также будет экспоненциальным, в соответствии с этим имеем:

$$F_C(t) = 1\text{-}e^{-\lambda_C t};\ f_C(t) = \lambda_C e^{-\lambda_C t};\tau_C = 1/\lambda_C;\sigma_C = 1/\lambda_C \qquad (4)$$

При резервном соединении m элементов, которые носят экспоненциальный закон распределения времени безотказной работы, вероятность отказа группы параллельно включенных элементов:

$$Q_P(t) = \left(1\text{-}e^{-\lambda_1 t}\right)\left(1\text{-}e^{-\lambda_2 t}\right) \dots \left(1\text{-}e^{-\lambda_m t}\right) = \prod_{j=1}^{m}(1\text{-}e^{-\lambda_j t}) \qquad (5)$$

Если все элементы равнонадежны и, $\lambda_1 = \lambda_2 = \lambda_m = \lambda_j = \lambda$ то

$$Q_P(t) = \left(1\text{-}e^{-\lambda t}\right)^m;P_P(t) = 1\text{-}\left(1\text{-}e^{-\lambda t}\right)^m.$$

Следовательно, при резервном соединении элементов экспоненциальный закон распределения времени безотказной работы не сохраняется.

Во большинстве случаев рассмотренный выше способ расчета надежности не может быть использован, потому что не всегда схема надежности содержит последовательно-параллельное соединение элементов.

Переходим к следующему методу: методу перебора состояний.

Расчету надежности любой системы независимо от используемого метода предшествует определение 2-ух непересекающихся множеств состояний элементов, которые соответствуют работоспособному и неработоспособному состояниям системы. Каждое из этих состояний описывается набором элементов, которые находятся в работоспособном и неработоспособном состояниях. Поскольку при независимых отказах вероятность каждого из состояний определяется произведением вероятностей нахождения элементов в соответствующих состояниях, то при числе состояний, равном m, вероятность работоспособного состояния системы

$$P = \sum_{j=1}^{m} \prod_{I_j} p_i \prod_{k_j} q_k; \quad (6)$$

вероятность отказа

$$Q = 1 - \sum_{j=1}^{m} \prod_{I_j} p_i \prod_{k_j} q_k, \quad (7)$$

где m – общее число работоспособных состояний, в каждом j-ом из которых число исправных элементов равно l, а вышедших из строя – k.

Стоит отметить, что в данном методе суммируются только работоспособные состояния системы.

Существенным недостатком метода перебора состояний является то, что даже при сравнительно простой структуре его применение сопряжено с громоздкими выкладками [6; 7].

Метод разложения относительно особого элемента основывается на использовании формулы полной вероятности. В сложной системе выделяется особый элемент, образующие полную группу всех возможных состоянии H_i

$$\sum_{i=1}^{n} P\{H_i\} = 1$$

Если анализируемое состояние системы А, то его вероятность

$$P\{A\} = \sum_{i=1}^{n} P\{H_i\} P\left\{\frac{A}{H_i}\right\} = \sum_{i=1}^{n} P_i\{A\} \quad (8)$$

Второй сомножитель в (8) определяет вероятность состояния А при условии, что особый элемент находится в состоянии H_i. Рассмотрение H_i-го состояния особого элемента как безусловного позволяет упростить структурную схему надежности и свести ее к последовательно-параллельному соединению элементов.

Сопоставление обоих методов расчета надежности показывает, что выделение особого элемента с последующим анализом упрощенных структурных схем заметно сокращает выкладки [7].

В некоторых случаях для анализа надежности сложной системы бывает достаточным определение граничных оценок надежности сверху и снизу.

При оценке вероятности безотказной работы сверху определяют минимальные наборы работоспособных элементов (путей), которые обеспечивают работоспособное состояние системы. При формировании пути, считая, что все элементы неработоспособны, последовательным переводом элементов в работоспособное состояние производится подбор вариантов соединений элементов, которые обеспечивают наличие цепи. Всё это составляет метод минимальных путей и сечений.

Если исключение любого элемента из набора приводит к отказу пути, то набор элементов образует минимальный путь. Отсюда, в пределах одного пути элементы расположены в основном соединении, а сами пути подключаются параллельно.

При определении минимальных сечений осуществляется подбор минимального числа элементов, вызывающий отказ системы при переводе из работоспособного состояния в неработоспособное. При корректном подборе элементов се-

чения возврат любого из элементов в работоспособное состояние восстанавливает работоспособное состояние системы. Так как отказ каждого из сечений приводит к отказу системы, следовательно, первые соединяются последовательно. В пределах каждого сечения элементы соединяются параллельно, потому что для работы системы достаточно всего лишь наличия работоспособного состояния любого из элементов сечения.

Следовательно, любая система преобразуется в структуру с параллельно-последовательным или последовательно-параллельным соединением элементов при составлении минимальных путей и сечений [6; 7].

При произвольных функциях распределения времени безотказной работы и восстановления надежность систем анализируют путем дискретизации времени с заданием на каждом его интервале вероятностей перехода системы из одного состояния в другое. Данный метод называют методом переходных вероятностей. При постоянстве направлений переходов системы из одного состояния в другое и допущении об ординарности, независимости и стационарности потока отказов система может быть отнесена к Марковским системам с дискретным временем.

Отличительным свойством Марковских систем является то, что вероятность перехода системы в любое из возможных состояний, число которых ограничено, не зависит от предшествующих состоянии, а зависит только от предыдущего.

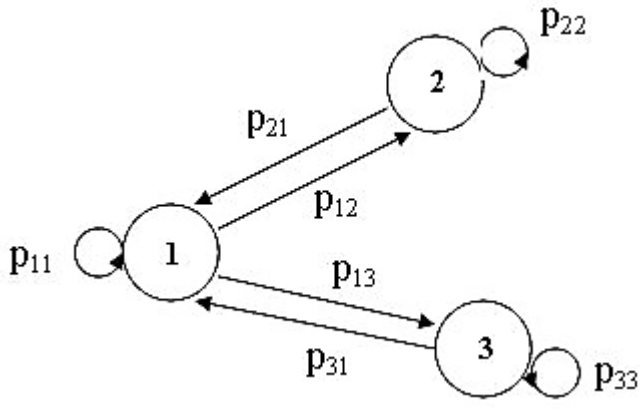

Рис. 1. Размеченный граф состояний восстанавливаемой системы

Надежность таких систем описывается системой алгебраических уравнений, число которых соответствует числу возможных состояний системы. Для их составления используется ориентированный граф состояний (рис.1), вершины

которого соответствуют возможным состояниям системы, а ребра характеризуют направление и вероятность перехода из одного состояния в другое [6; 7].

Последний метод оценки надежности (метод переходных интенсивностей0) использует для расчетов экспоненциальное распределение.

Экспоненциальное распределение с удовлетворительной точностью описывает функционирование технических систем и их элементов на участке нормальной эксплуатации. Применение экспоненциального распределения для описания процесса восстановления позволяет представить анализируемые системы в виде Марковских систем с непрерывным временем и использовать для анализа их надежности систему линейных дифференциальных уравнений первого порядка (при ординарных независимых отказах).

Экспоненциальное распределение описывает процессы в системах без предыстории, поскольку изменение вероятности их нахождения в том или ином состоянии за интервал t зависит только от длительности временного интервала.

В общем случае число дифференциальных уравнений определяется числом возможных состояний системы, которое (как для систем с дискретным временем) должно быть ограничено.

Теория надежности, претерпела развитие для описания технических систем (в том числе технические средства АС). Отказы происходят в связи с разрушением и старением компонентов, причем для восстановления требуется ремонт, регулировка, замена компонентов или техническое средство. Разрушение и старение не является свойственным ни для программного обеспечения (ПО) системы в целом, ни для отдельных программ. Однако все-таки, возможен перенос части понятий, терминов и методов надежности и на ПО (принимая при этом определенную условность такого подхода).

При разработке ПО возникает ряд причин, которые приводят к появлению ошибок:

– ошибки в переносе программ на носители;

– неправильный выбор методов защиты программ;

– неправильное составление общей структуры ПО и взаимосвязи программ;

– неправильное понимание программистом алгоритма и др.

К сожалению, отладка ПО не способна полностью устранить все ошибки, потому что число возможных сочетаний входных данных и состояний системы при ее функционировании настолько велико, что все возможные ветви прохождения программ заранее проверить практически невозможно. Поэтому поток моментов проявления ошибок ПО при функционировании АС носит исключительно случайный характер: ошибки появляются в различные случайные моменты времени, когда программа выходит на участок с ошибкой.

Существуют два подхода к выбору показателей надежности ПО. С одной стороны, существует возможность использования обычных показателей надежности, такие как вероятность отсутствия ошибок за время t; среднее время между ошибками; среднее время восстановления ПО после прекращения функционирования и тому подобное. Данные показатели описывают появление ошибок ПО во времени, поэтому целесообразно использовать их для ПО, которое непрерывно эксплуатируется. Для программ, которые используются нерегулярно (только в случаях необходимости), можно применять такие показатели, как вероятность успешного выполнения одного прогона программы, вероятность решение произвольной задачи из потока реальных задач с помощью данного ПО.

Однако при применении понятий классической теории надежности к ПО стоит учесть особенности и отличия этих объектов от традиционных технических систем, для которых изначально разрабатывалась теория надежности:

– доминирующие факторы, определяющие надежность программ – дефекты и ошибки проектирования и разработки, и второстепенное значение имеет физическое разрушение программных компонент при внешних воздействиях;

– необходимость физической замены программных компонент и их относительно редкое разрушение, приводит к принципиальному изменению понятий сбоя и отказа программ и к их подразделению по длительности восстановления относительно некоторого допустимого времени простоя для функционирования информационной системы;

– понятия и методы теории надежности применимы не для всех видов программ – их можно использовать только к ПО, которое функционирует в реальном времени и непосредственно взаимодействует с внешней средой;

– в дальнейшем одна и та же ошибка не может повториться после её исправления в программе. Более того, ошибки, которые выявлены в ПО одной из множества однотипных систем, во всех таких системах обычно исправляются. Поток ошибок ПО нестационарный, потому что по мере выявления каждой ошибки параметр их потока уменьшается. Отказы ТС носят повторяющийся характер (по одной и той же причине); один и тот же отказ и этого, и иных аналогичных средств может повториться вновь после восстановления (по той же причине). Поток отказов ТС в установившемся режиме с тем или иным приближением можно принять стационарным;

– непредсказуемость времени, места и вероятности проявления ошибок и дефектов, а также их редкое обнаружение при реальной эксплуатации достаточно-надежных программных средств, не позволяет эффективно использовать традиционные методы априорного расчета показателей надежности сложных систем, которые ориентированы на стабильные, измеряемые значения надежности составляющих компонент.

С учетом перечисленных особенностей для описания надежности ПО могут быть использованы специальные показатели, которые характерны только для ПО и отражающие, главным образом, качество выполнения ПО. Эти показатели позволяют оценить следующие свойства ПО, составляющие понятие «надежность ПО»:

1. Корректность – статическое свойство программы, которое определяется как отсутствие в ней ошибок. Корректность программ обеспечивает отладка (проверка) на множестве исходных данных, которые регламентирует документация.

2. Устойчивость – динамическое свойство программы, характеризующее её способность выдавать правильные результаты при аппаратных, информационных и эргатических воздействиях. При этом можно выделить 2 вида устойчивости:

1. Консервативность – способность программы при наличии возмущений, которые не позволяют правильно решить задачу, перевести вычислительную систему в состояние отказа, из которого с минимальными потерями можно выполнить процедуру рестарта.

2. Устойчивость программ обеспечивают структурной, информационной, временной и алгоритмической избыточностью.

Толерантность – способность программы продолжать свою работу и выдавать правильные результаты при наличии перечисленных воздействий [7].

Для систематической, координированной борьбы с ошибками, отказами, сбоями необходимо исследовать факторы, которые оказывают влияние на надежность ПО. При решении задач надежности можно представить следующий структурный состав программного обеспечения современных АСУ:

– объектный код программ, которые исполняются вычислительными средствами в процессе функционирования ПО;

– информация, которая выдается потребителям и на исполнительные механизмы, является результатом обработки исходных данных и информации, которая накоплена в базе данных;

– динамический вычислительный процесс обработки данных, автоматизированной подготовки решений и выработки управляющих воздействий;

– информация, которая накоплена в базах данных, отражает объекты внешней среды, и процессы ее обработки.

Вышеперечисленные компоненты ПО являются в некотором роде объектами уязвимости, на которые действуют различные дестабилизирующие факторы, подразделяющиеся на внутренние (уязвимости присущие самим объектам) и внешние, которые обусловлены средой функционирования этих объектов (рисунок 2).

Степень влияния всех внутренних дестабилизирующих факторов, а также некоторых внешних угроз на надежность ПС определяется в наибольшей степени качеством технологий проектирования, разработки, сопровождения и документирования ПС и их основных компонентов [3, с. 22, 23].

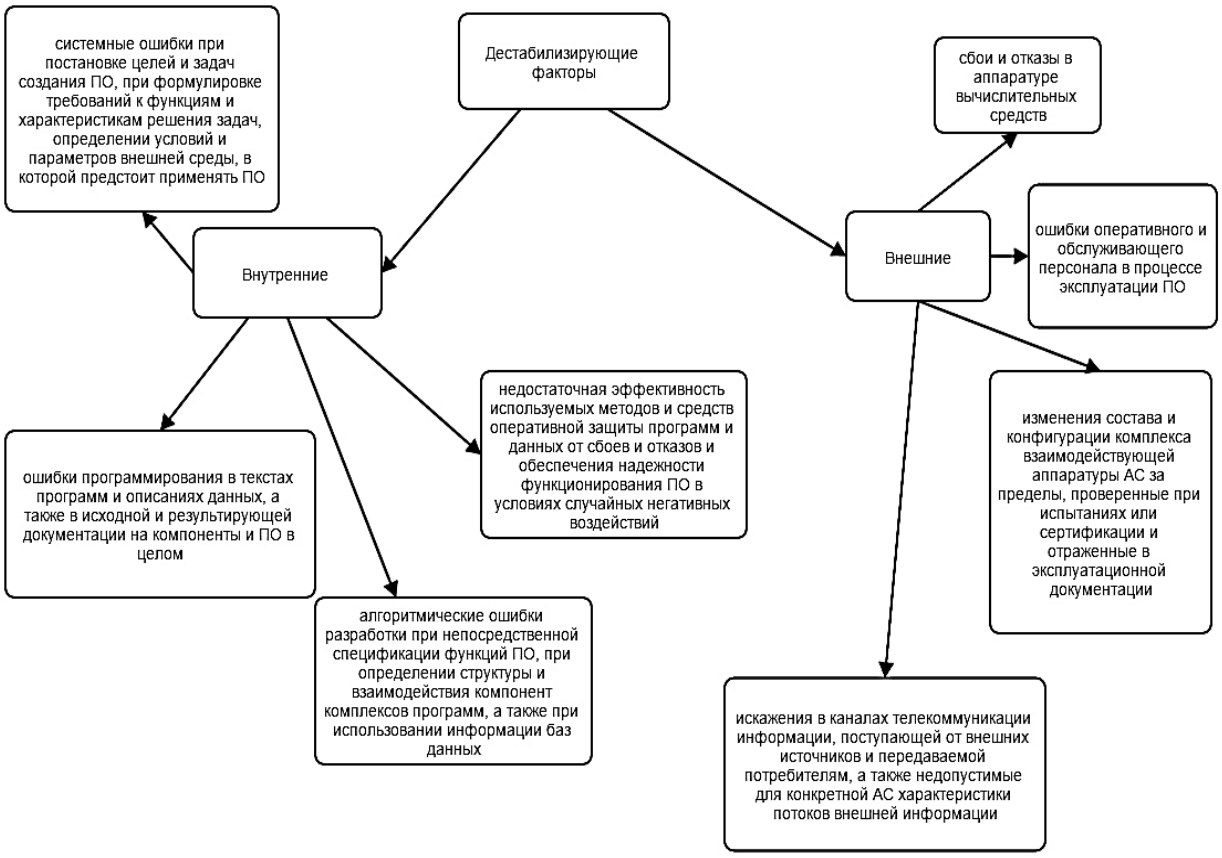

Рис. 2. Дестабилизирующие факторы

Таким образом, в данной статье были рассмотрены методы оценки надежности и дестабилизирующие факторы, влияющие на качество ПО.

Список литературы

1. Гоулд И.Г. Терминологическая работа IFIP (Международная федерация по обработке информации) и ICC (Международный вычислительный центр) / И.Г. Гоулд, Дж.С. Тутилл // Журн. вычисл. матем. и матем. физ. – 1965. – №2. – С. 377–386.

2. Майерс Г. Надежность программного обеспечения / Г. Майерс – М.: Мир, 1980.

3. Соммервилл Иан. Инженерия программного обеспечения, 6-е изд. / Иан Соммервилл. Пер. с англ. – М.: Вильямс, 2002. – 624 с.

4. Дейкстра Э. Заметки по структурному программированию / У. Дал, Э. Дейкстра, К. Хоор. Структурное программирование. – М.: Мир, 1975. – С. 7–97.

5. Criteria for Evalution of Software. – ISO TC97/SC7 #367 (Supersedes Document #327).

6. Феллер В. Введение в теорию вероятностей и ее приложения / В.Феллер. – М.: Мир, 1967.

7. Петров В.Н. Информационные системы / В.Н. Петров – СПб.: Питер, 2002. – 688 с.

8. Надежность программного обеспечения [Электронный ресурс]. – Режим доступа: http://fan-5.ru/best/best-175124.php