УДК 681.34

К ВОПРОСУ ОЦЕНКИ НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С МНОГОУРОВНЕВОЙ АРХИТЕКТУРОЙ

[1]Тынченко В.В., [2]Царев Р.Ю.

[1] ФГБОУ ВПО «Сибирский государственный аэрокосмический университет имени академика М.Ф. Решетнева», Красноярск, Россия (660037, Красноярск, пр. им. газ. «Красноярский рабочий», 31), e-mail: 051301@mail.ru;
[2] ФГАОУ ВПО «Сибирский федеральный университет», Красноярск, Россия (660074, Красноярск, ул. Академика Киренского, 26Б), e-mail: tsarev.sfu@mail.ru

Применение программного обеспечения в критичных областях науки и производства требует гарантии высокого уровня его надежности. Анализ эффективности методов и средств обеспечения требуемого уровня надежности предполагает наличие математического аппарата, способного представить объективную оценку параметров надежности. В данной работе представлены аналитические выражения, позволяющие оценить ряд параметров надежности программного обеспечения с учетом его архитектуры и взаимосвязи программных компонентов. В статье рассмотрены условные и безусловные вероятности сбоя различных компонентов программного обеспечения, представлены формулы для расчета среднего времени простоя, среднего времени появления сбоя, вероятности безотказной работы. Предложенный подход проиллюстрирован решением практического примера оценки надежности программного обеспечения с двухуровневой архитектурой. Кроме этого, показано, что надежность может быть повышена за счет введения избыточных версий компонентов программного обеспечения.

Ключевые слова: надежность, архитектура, программное обеспечение, мультиверсионное программирование

TOWARD THE PROBLEM OF EVALUATION OF THE RELIABILITY OF SOFTWARE WITH MULTIPLE LEVEL ARCHITECTURE

[1]Tynchenko V.V., [2]Tsarev R.Y.

[1]Siberian state aerospace university named after academician M. F. Reshetnev, Krasnoyarsk, Russia (660037, Krasnoyarsk, avenue named after newspaper «Kranoyarskii rabochii», 31), e-mail: 051301@mail.ru;
[2]Siberian Federal University, Krasnoyarsk, Russia (660074, Krasnoyarsk, street A. Kirenskogo, 26B), e-mail: tsarev.sfu@mail.ru

Application of software in critical areas of science and industry requires the guarantees of high level of the reliability. Analysis of the effectiveness of methods and tools that provide required level of the reliability assumes the existence of mathematical apparatus allowing the objective evaluation of reliability parameters. This paper presents the analytical expressions allowing the software reliability parameters assessment taking into account software architecture and interrelationship of software components. The article describes the conditional and unconditional probability of failures of various software components. It presents equations for the calculation such reliability parameters as mean time to repair, mean time to failure, and probability of failure-free operation. The proposed approach is illustrated by an example of the evaluation of reliability of software with bi-level architecture. Additionally, it is shown that the software reliability can be improved by introducing redundant versions of software components.

Keywords: reliability, architecture, software, mutiversion programming

Одним из важнейших показателей качества программного обеспечения является его надежность. Надежность определяется как способность безотказно выполнять определенные функции при заданных условиях в течение заданного периода времени с достаточно большой вероятностью [5]. Применительно к сегодняшнему представлению о программном обеспечении следует признать тот факт, что при разработке программ можно стремиться только к минимизации появления ошибок, а не к их полному отсутствию. Связано это с тем,

что зачастую невозможно выполнить тестирование на всем множестве разнородных данных и отследить все траектории обработки информации в программном обеспечении [3].

До сих пор надежность программ связывалась с методами отладки и тестирования, степенью и качеством отлаженности программ, что вполне отвечает смысловому содержанию понятия надежности. Однако с появлением проблем при разработке больших и сложных программ встает вопрос об увеличении надежности программ на уровне архитектуры, что позволит, не повышая надежности отдельных компонентов, повысить надежность программного обеспечения в целом, избежать распространения ошибки по компонентам, а также ускорит процессы поиска, анализа и устранения последствий сбоя в процессе эксплуатации [6].

Анализ архитектурной надежности программного обеспечения включает уровни, соответствующие различным компонентам и их взаимосвязи. В зависимости от того, где произошел сбой, длительность отказа и его влияние на надежность программного обеспечения различны. Сбой может происходить на различных уровнях архитектуры, в модуле, процессе, интерфейсе компонента [4]. Число архитектурных уровней в модели архитектуры зависит от проекта программного обеспечения.

В данной работе приведены аналитические выражения расчета различных показателей надежности программного обеспечения и приведен пример оценки надежности двухуровнего программного обеспечения.

Параметры надежности программного обеспечения

Параметры надежности зависят от типа архитектуры программного обеспечения, числа архитектурных уровней, компонентов и зависимостей среди компонентов [2]. Безусловные и условные вероятности сбоя программного обеспечения различны для различного числа компонентов. Безусловная вероятность сбоя зависит от количества кода в компонентах (например, может быть пропорционально количеству строк в программе или может зависеть от отношения числа измененных к общему числу строк в программе). Безусловная вероятность сбоя может также быть оценена как частота восстановлений компонента независимо от восстановления других компонентов. Условная вероятность сбоя зависит от количества кода в компонентах, зависимости компонентов и связи между компонентами. Например, условная вероятность сбоя может быть выше для компонентов, зависящих от большого числа других компонентов (при статической зависимости компонентов) или связанных с большим числом других компонентов (при динамической зависимости компонентов). Условная вероятность сбоя может также быть оценена как частота восстановления компонента, вызванного восстановлением других компонентов.

Среднее время простоя системы в архитектуре зависит от условных и безусловных вероятностей сбоев на всех уровнях архитектуры и от среднего времени доступа, анализа и восстановления сбойных компонентов [10]. Положим, что время устранения сбоя равно времени, которое требуется для доступа, анализа, восстановления. Это означает, что время восстановления меньше, чем время устранения сбоя. Среднее время простоя системы вычисляется для всех архитектурных уровней и всех компонентов на каждом уровне [7].

Для каждого архитектурного уровня программного обеспечения вероятность использования каждого компонента умножается на вероятность сбоя компонента и на сумму средних времен анализа, доступа и восстановления для этого компонента [1]. Поскольку сбойный компонент может вызывать сбои в зависящих от него компонентах, как на других уровнях архитектуры, так и на том же самом уровне, то для каждого отдельного уровня архитектуры и для всех компонентов условная вероятность появления сбоя умножается на сумму относительных времен доступа, анализа и восстановления этих компонентов. Аналогично, для одного уровня и для всех компонентов условная вероятность появления сбоя умножается на сумму относительных времен доступа, анализа и восстановления этих компонентов. Среднее время простоя системы равно:

$$MTTR = \sum_{i=1}^{F}[PU_i \times PF_i \times [(TA_i + TC_i + TE_i) + \sum_{j=1, j\neq i}^{F}[PL_{ji} \times [(TA_j + TC_j + TE_j) +$$

$$+ \sum_{l, j\in D_l}[PL_{lj} \times (TA_l + TC_l + TE_l)]]] + \sum_{k, i\in D_k} PL_{ki} \times [(TA_k + TC_k + TE_k) + \quad (1)$$

$$+ \sum_{m=1, m\neq j}^{F}[PL_{mk} \times [(TA_m + TC_m + TE_m) + \sum_{l, m\in D_l} PL_{lm} \times (TA_l + TC_l + TE_l)]]]]],$$

где F – общее число компонентов в архитектуре программного обеспечения; M – число уровней архитектуры программного обеспечения; D_i – непересекающиеся множества компонентов на уровне i, $i \in \{1, \ldots, M\}$; PU_i – вероятность использования компонента i, $i \in \{1, \ldots, F\}$; PF_i – вероятность сбоя в компоненте i, $i \in \{1, \ldots, F\}$; PL_{ij} – условная вероятность сбоя в компоненте i при сбое в компоненте j, $i \in \{1, \ldots, F\}$, $j \in \{1, \ldots, F\}$; TA_i – относительное время доступа к компоненту i, $i \in \{1, \ldots, F\}$; TC_i – относительное время анализа сбоя в компоненте i, $i \in \{1, \ldots, F\}$; TE_i – относительное время устранения сбоя в компоненте i, $i \in \{1, \ldots, F\}$.

Среднее время появления сбоя зависит от условных и безусловных вероятностей сбоев во всех компонентах на всех архитектурных уровнях и от относительного времени использования компонентов, в которых сбоя не происходит [1, 10]. Среднее время сбоя вычисляется для всех архитектурных уровней и всех компонентов на каждом архитектурном

уровне. Для каждого архитектурного уровня программного обеспечения вероятность использования компонента умножается на вероятность того, что каждый компонент будет работать без сбоев в течение относительного времени его использования. Кроме этого, для каждого отдельного архитектурного уровня и для всех компонентов условная вероятность работы без сбоев умножается на относительное время использования этих компонентов. Среднее время появления сбоя равно:

$$MTTF = \sum_{i=1}^{F}[PU_i \times (1-PF_i) \times [TU_i + \sum_{j=1, j\neq i}^{F}[(1-PL_{ji}) \times [TU_j + \sum_{l, j\in D_l}[(1-PL_{lj}) \times TU_l]]]] + \\ + \sum_{k, i\in D_k}[(1-PL_{ki}) \times [TU_k + \sum_{m=1, m\neq j}^{F}[(1-PL_{mk}) \times [TU_m + \sum_{l, m\in D_l}(1-PL_{lm}) \times TU_l]]]]]]. \quad (2)$$

Коэффициент надежности (вероятности безотказной работы) программного обеспечения может быть оценен согласно следующей формуле:

$$R_S = \sum_{i=1}^{F} PU_i \times R_i, \quad (3)$$

где R_i – коэффициент (вероятности безотказной работы) надежности компонента i.

Надежность программного обеспечения можно повысить, если использовать избыточные версии компонент, например согласно мультиверсионной методологии [8, 9]. При мультиверсионном формировании состава модулей надежность каждого компонента определяется как:

$$R_i = 1 - \prod_{k\in Z_i}(1-PF_{ik}),$$

где PF_{ik} – вероятность сбоя компонента из соответствующего ему множества версий Z_i.

Таким образом, при увеличении надежности некоторых компонент, т.е. при уменьшении вероятности сбоя в этих компонентах, уменьшается среднее время простоя системы и увеличивается среднее время появления сбоя.

Оценка надежности программного обеспечения с двухуровневой архитектурой

Рассмотрим пример оценки показателей надежности программного обеспечения, архитектура которого имеет два уровня: уровень интерфейсов (компоненты 1–3) и уровень модулей (компоненты 4–12). Соответственно, определим два непересекающихся множества компонент D_i:

$$D_1 = \{1, 2, 3\},$$

$$D_2 = \{4, 5, 6, 7, 8, 9, 10, 11, 12\}.$$

Архитектура рассматриваемого программного обеспечения может быть представлена посредством матрицы инцидентности графа:

$$G_{ij} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Исходные данные о компонентах данного программного обеспечения представлены в таблице 1.

Таблица 1

Данные о компонентах

Компонент	C_i	PU_i	PF_i	TA_i	TC_i	TE_i	TU_i
1	300	0,2	0,05	1	3	2	400
2	300	0,1	0,05	1	3	2	100
3	300	0,1	0,05	1	3	2	150
4	400	0,05	0,2	2	1	4	40
5	200	0,05	0,05	2	1	4	40
6	100	0,6	0,05	2	1	4	40
7	200	0,4	0,05	2	1	4	40
8	100	0,7	0,05	2	1	4	40
9	200	0,3	0,05	2	1	4	40
10	100	0,08	0,2	2	1	4	40
11	200	0,07	0,2	2	1	4	40
12	100	0,15	0,05	2	1	4	40

Условная вероятность сбоя в компоненте *i* при сбое в компоненте *j* для данного программного обеспечения может быть представлена следующим образом:

$$PL_{ij} = \begin{pmatrix} 1 & 0,6 & 0,7 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 \\ 0 & 1 & 0 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 & 0,1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0,1 & 0,1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

На основе представленных данных согласно выражениям (1)–(3) могут быть получены значения показателей надежности рассматриваемого программного обеспечения:

$$MTTR = 0,62, \quad MTTF = 1288,94, \quad R_S = 0,9152.$$

Представленные значения показателей надежности рассчитаны для архитектуры программного обеспечения без избыточности. Однако они могут быть улучшены благодаря введению дополнительных версий критичных по надежности компонентов программного обеспечения согласно мультиверсионной методологии [8, 9]. Положим, что повышенные требования по надежности предъявляются к компонентам 4, 10 и 11, поскольку именно они реализуют наиболее критичные функции. Введение избыточных версий увеличивает стоимость программного обеспечения, которая может быть рассчитана по формуле:

$$C_S = \sum_{i=1}^{M} \sum_{j \in Z_i} C_j.$$

Различные варианты архитектур мультиверсионного программного обеспечения, а также показатели надежности и стоимость приведены в таблице 2.

Таблица 2

Варианты архитектур программного обеспечения

	Вариант 1	Вариант 2	Вариант 3	Вариант 4	Вариант 5
MTTR	0,42	0,31	0,45	0,36	0,30
MTTF	1536,04	1558,67	1519,02	1550,18	1558,99
R_S	0,9238	0,9440	0,9312	0,9392	0,9524
C_S	2700	2800	2600	3100	3200

В таблице 3 представлено количество версий отдельных компонентов в различных вариантах архитектур программного обеспечения с избыточностью.

Таблица 3

Количество версий компонентов в различных вариантах архитектур

	Вариант 1	Вариант 2	Вариант 3	Вариант 4	Вариант 5
Компонент 1	1	1	1	1	1
Компонент 2	1	1	1	1	1
Компонент 3	1	1	1	1	1
Компонент 4	2	1	1	2	2
Компонент 5	1	1	1	1	1
Компонент 6	1	1	1	1	1
Компонент 7	1	1	1	1	1
Компонент 8	1	1	1	1	1
Компонент 9	1	1	1	1	1
Компонент 10	2	2	1	1	2
Компонент 11	1	2	2	2	2
Компонент 12	1	1	1	1	1

Анализируя полученные результаты, можно отметить, что, вводя избыточные версии всего трех компонентов: 4, 10 и 11, при максимальном количестве версий, равном двум, повышаются значения показателей надежности всего программного обеспечения. В частности, вероятность безотказной работы увеличивается с 0,9152 до 0,9524. Если максимальное количество версий указанных компонентов увеличить до трех, то вероятность безотказной работы достигнет значения 0,9884.

Заключение

Предложенные в статье аналитические выражения позволяют выполнить оценку и анализ надежности программного обеспечения с учетом его архитектуры, числа архитектурных уровней, компонентов программного обеспечения и зависимостей между ними. В работе рассмотрены основные показатели надежности: среднее время простоя, среднее время появления сбоя, вероятность безотказной работы. Представлен пример оценки показателей надежности программного обеспечения с двухуровневой архитектурой. Показана возможность повышения надежности программного обеспечения за счет введения дополнительных версий компонентов. Предлагаемый математический аппарат целесообразно использовать при анализе и синтезе программного обеспечения, применяемого в критичных областях.

Список литературы

1. Буторов В.В. Оценка надежности клиент-серверных приложений корпоративной системы управления предприятием / В.В. Буторов, С.В. Тынченко, Р.Ю. Царев // Фундаментальные исследования. – 2015. – № 5. – Ч. 3. – С. 488–492.

2. Кузнецов А.С. Многоэтапный анализ архитектурной надежности и синтез отказоустойчивого программного обеспечения сложных систем: монография / А.С.

Кузнецов, С.В. Ченцов, Р.Ю. Царев. – Красноярск: Сибирский федеральный университет, 2013. – 143 с.
3. Любицын В.Н. Необходимость разработки надежного программного обеспечения как вызов современности / В.Н. Любицын // Вестник ЮУрГУ. Серия: Компьютерные технологии, управление, радиоэлектроника. – 2012. – № 23. – С. 26–29.
4. Модель анализа надежности распределенных вычислительных систем / Р.Ю. Царев, А.Н. Пупков, М.А. Огнерубова, М.В. Сержантова, Н.А. Бесчастная // Вестник СибГАУ. – 2013. – Вып. 1 (47). – С. 86–91
5. Павловская О.О. Статические методы оценки надежности программного обеспечения / О.О. Павловская // Вестник ЮУрГУ. Серия: Компьютерные технологии, управление, радиоэлектроника. – 2009. – № 26 (159). – С. 35–37.
6. Практическая реализация надежностного анализа архитектуры программной системы / Е.В. Гражданцев, М.А. Русаков, О.И. Завьялова, Р.Ю. Царев // Вестник СибГАУ. – 2008. – Вып. 1 (18). – С. 37–40.
7. Царев Р.Ю. К проблеме оценки надежности сложных программных систем / Р.Ю. Царев, А.В. Штарик, Е.Н. Штарик // Журнал Сибирского федерального университета. Серия: Техника и технологии. – 2015. – Т. 8. – № 1. – С. 33–47.
8. Царев Р.Ю. Методология многоатрибутивного формирования мультиверсионного программного обеспечения сложных систем управления и обработки информации: монография / Р.Ю. Царев; Краснояр. гос. аграр. ун-т. – Красноярск, 2011. – 210 с.
9. Avizienis A., Chen L. On the implementation of N-version programming for software fault-tolerance during program execution (1977) In Proc. IEEE Comput Soc Int Comput Software & Appl Conf, COMPSAC '77, pp. 149–155.
10. Hac A. Using a software reliability model to design a telecommunications software architecture (1991) IEEE Transactions on Reliability, 40 (4), pp. 488–494.

Рецензенты:

Носков М.В., д. ф.-м.н., профессор, заместитель директора по научной работе Института космических и информационных технологий Сибирского федерального университета, г. Красноярск.

Ченцов С.В., д.т.н., профессор, зав. кафедрой «Системы автоматики, автоматизированное управление и проектирование» Сибирского федерального университета, г. Красноярск.