УДК 621.396.6.019.3

О.В. Иванченко, Е.С. Солдатенко, Е.С. Владимирова

Севастопольский национальный технический университет, Севастополь

АРХИТЕКТУРНЫЙ ПОДХОД К ОЦЕНКЕ НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ СИСТЕМЫ УПРАВЛЕНИЯ КРИТИЧЕСКОЙ ИНФРАСТРУКТУРЫ

Широкое внедрение и применение информационных технологий в системах управления критическими инфраструктурами (КрИ) существенно повышает эффективность их использования по назначению. Оперативное решение достаточно сложных задач управления КрИ невозможно без использования соответствующего программного обеспечения (ПО). В то же время уроки, извлеченные в ходе анализа последствий наиболее крупных аварий и катастроф с участием различных инфраструктурных образований, свидетельствуют о недооценке вклада ПО в ухудшение общего уровня надежности. С целью устранения указанной проблемы предлагается использовать комплексный подход к оценке надежности ПО системы управления КрИ с учетом различных режимов ее эксплуатации. Разработанный подход базируется на архитектурном представлении процессов функционирования соответствующего программного модуля и эффективной реализации процедуры его тестового контроля.

Ключевые слова: *система управления критической инфраструктурой, программное обеспечение, архитектурный подход, непрерывная и дискретная марковские цепи, вероятность безотказной работы, коэффициент оперативной готовности.*

Постановка проблемы

Экономическое процветание современного общества сопровождается ростом сложности и масштабов задач, решаемых с использованием критических инфраструктур (КрИ).

Обеспечение эффективного применения критических инфраструктур по назначению во многом достигается за счет широкого внедрения информационных технологий (ИТ) в системах управления (СУ) КрИ различного уровня. Известно, что одной из важнейших составляющих СУ КрИ является программный модуль, реализованный в виде отдельного критического приложения. На сегодняшний день сложилась парадоксальная ситуация, когда программное обеспечение (ПО) выступает одним из действенных инструментов повышения функциональной безопасности и одновременно является источником новых специфических дефицитов надежности. Последствия этих негативных явлений проявляются в виде сбоев и отказов, которые должны быть спрогнозированы, выявлены, оценены и снижены до приемлемого уровня еще на ранних этапах жизненного цикла СУ КрИ [1].

В настоящее время серьезную озабоченность у представителей компаний, занимающихся применением КрИ по назначению, вызывают участившиеся сбои программного обеспечения, являющиеся основным источником отказов СУ [2, 3]. Это обстоятельство вызывает не меньшее беспокойство у разработчиков критических приложений, т.к. нарушается один из основных принципов создания безотказного ПО, согласно которого «на интервале использования по назначению после завершения этапа разработки количество серьезных неисправностей программного модуля должно существенно уменьшаться» [4].

Следует отметить, что значительная часть исследователей предпринимала попытки обосновать количественные оценки надежности программных компонентов при условии гарантированного обеспечения минимально допустимого уровня функциональной безопасности (ФБ) СУ КрИ. Большинство работ в этой области были посвящены обеспечению надежности ПО путем его резервирования еще на этапе проектирования [5, 6]. Некоторые авторы решали эту же задачу на этапе верификации ПО, пытаясь распределить надежность между компонентами программного модуля таким образом, чтобы поддержать предельно допустимый общесистемный уровень [7]. Однако, по мнению специалистов в сфере ФБ КрИ [8], несмотря на значительный объем затрачиваемых интеллектуальных и финансовых ресурсов, ни одна из предлагаемых моделей не отвечает следующим требованиям:

1) оценка системной надежности должна выполняться с учетом архитектуры ПО, исходя из необходимости определения степени влияния компонентов программного модуля на процессы, протекающие в СУ КрИ и возможные последствия в случае различных отказов, сбоев;

2) так как эффективное функционирование СУ КрИ в значительной степени зависит не только от правильной работы компонентов ПО критического приложения, но и от операционной среды, в которой реализован данный программный модуль, то в разрабатываемой модели должна учитываться надежность ОС;

3) поскольку в критических приложениях все чаще применяют различные механизмы обеспечения отказоустойчивости, то в предлагаемой модели надежности должны быть реализованы возможности по смягчению последствий сбоев, нарушений;

4) наконец, чтобы модель была полезной на практике, она должна быть гибкой и по возможности применяться пользователем для ответа на возникающие вопросы в области обеспечения надежности СУ КрИ.

Целью статьи является разработка подхода по оценке надежности программного обеспечения критического приложения, учитывающего особенности его архитектурного построения и реализации процессов тестового контроля для различных режимов использования систем управления КрИ по назначению.

1. Аналитико-стохастическая модель надежности системы управления КрИ

Поддержание КрИ в готовности к использованию по назначению во многом зависит от оперативного решения задач функционального и тестового контроля составляющих инфраструктурного образования. Поэтому в рамках единого методологического подхода возникает необходимость построения математических моделей надежности элементов КрИ с различными вариантами контроля информационно-технического состояния и режимами эксплуатации.

Предлагаемый подход базируется на реализации принципа аналитико-стохастического моделирования (АСМ) [9] для интересующих нас случаев. В качестве исследуемой СУ КрИ рассмотрим наземный сегмент системы управления космическими аппаратами (КосАп), архитектура которого представлена на рис. 1.

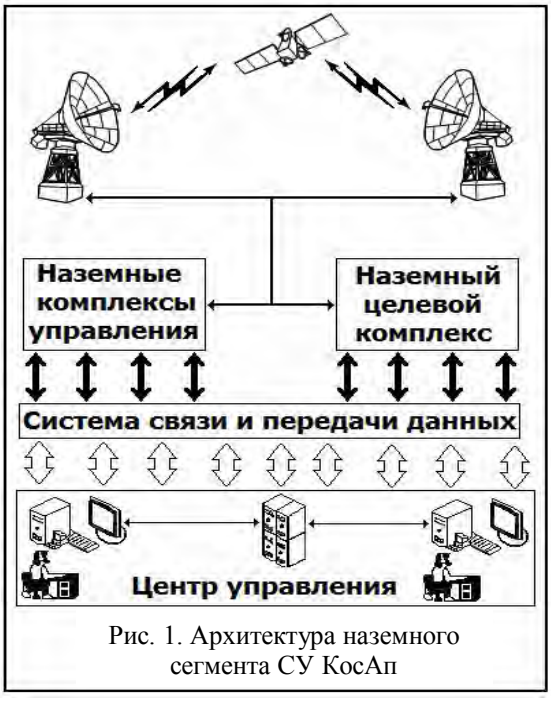

Рис. 1. Архитектура наземного сегмента СУ КосАп

В соответствии с установленным уровнем формализации представим процесс функционирования СУ КосАп в виде 2-х фазного процесса (рис. 2), каждая фаза которого характеризуется следующими режимами эксплуатации:

1) первая фаза – режим ожидания применения СУ КосАп по назначению (характеризуется интервалом ожидания применения по назначению, т.е. $t \in (0; t_{ож})$);

2) вторая фаза – режим применения по назначению СУ КосАп (характеризуется интервалом собственно применения по назначению, т.е. $t \in (t_{ож}; t_{ож} + \Delta t_{пр})$).

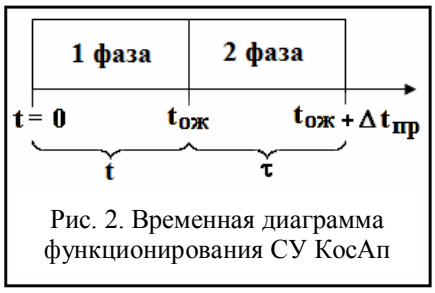

Рис. 2. Временная диаграмма функционирования СУ КосАп

Анализ опыта эксплуатации и применения по назначению СУ КосАп свидетельствует о целесообразности введения следующих ограничений и допущений:

1) продолжительности интервалов t и τ (рис. 2) являются предсказуемыми неслучайными величинами;

2) в качестве основной модели безотказности СУ КосАп рассматривается экспоненциальный закон распределения [9];

3) готовность системы управления в режиме ожидания в момент времени $t \in (0; t_{ож})$ определяется безотказностью и восстанавливаемостью технологического оборудования СУ КосАп [10];

4) переход ко второй фазе (рис. 2) осуществляется после завершения контроля информационно-технического состояния (КИТС) СУ КосАп или решения системой расчетных контрольных задач;

5) надежность системы управления на интервале применения по назначению τ в основном зависит от отказоустойчивости программного модуля СУ КосАп [8].

Особенности эксплуатации СУ КосАп на интервале ожидания применения по назначению $t \in [0; t_{ож}]$ позволяют выделить следующие информационно-технические состояния (ИТС): S_1 – РС системы управления КосАп; S_2 – состояние сбоя соответствует разрушению основного массива и (или) частичному разрушению резерва; S_3 – состояние возвращения соответствует восстановлению СУ КосАп после сбоя (возвращение в РС с использова-

нием резервных копий и (или) предысторий); S_4 – состояние КИТС системы управления КосАп; S_5 – состояние решения расчетных контрольных задач соответствует наличию в системе дополнительного неразрушаемого РС массива на интервале между КИТС.

Граф переходов для первой фазы функционирования СУ КосАп изображен на рис. 3.

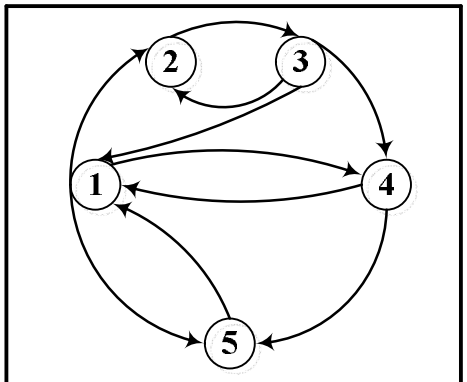

Рис. 3. Граф переходов для первой фазы функционирования СУ КосАп

Дальнейшие рассуждения базируются на представлении процесса изменения ИТС системы управления КосАп как марковского.

Исходя из этого, построим опорную модель в виде непрерывной марковской цепи, на основе применения которой выполним расчеты показателей надежности СУ КосАп. Задачу решим, используя метод Рунге-Кутта [11], для начальных условий в момент времени $t = 0$, когда $P_1(0) = 1$, $\forall P_i(0) = 0$, где $i = 2,...,5$.

В соответствии с изображенным графом (рис. 3) запишем систему дифференциальных уравнений Колмогорова-Чепмена в виде [12, 9]

$$\begin{cases} \dfrac{dP_1(t)}{dt} = (-\lambda_{12} - \lambda_{14} - \lambda_{15})P_1(t) + \lambda_{31}P_3(t) + \\ \qquad + \lambda_{41}P_4(t) + \lambda_{51}P_5(t); \\ \dfrac{dP_2(t)}{dt} = \lambda_{12}P_1(t) - \lambda_{23}P_2(t) + \lambda_{32}P_3(t); \\ \dfrac{dP_3(t)}{dt} = \lambda_{23}P_2(t) - (\lambda_{31} + \lambda_{32} + \lambda_{34})P_3(t); \\ \dfrac{dP_4(t)}{dt} = \lambda_{14}P_1(t) + \lambda_{34}P_3(t) - (\lambda_{41} + \lambda_{45})P_4(t); \\ \dfrac{dP_5(t)}{dt} = \lambda_{15}P_1(t) + \lambda_{45}P_4(t) - \lambda_{51}P_5(t), \end{cases} \quad (1)$$

где $\sum_{i=1}^{5} P_i(t) = 1.$

Результаты расчета стационарного коэффициента готовности $K_г(t) = P_1(t)$ системы управления космическими аппаратами в виде зависимости $K_г(T_o, t)$ для среднего времени восстановления $T_в = 0{,}5$ ч представлены на рис. 4.

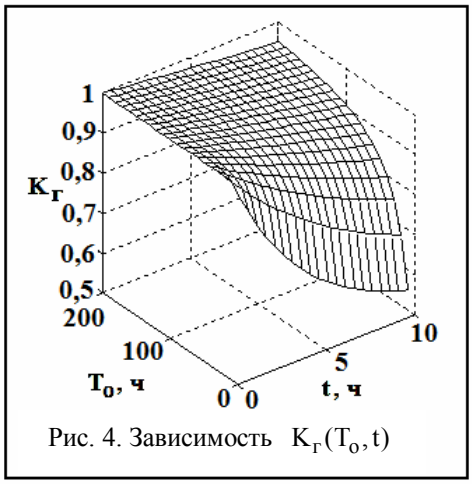

Рис. 4. Зависимость $K_г(T_o, t)$

2. Марковская модель надежности ПО критического приложения

На следующем этапе АСМ строится марковская модель надежности ПО критического приложения, которая адекватно описывает процессы, протекающие в системе управления КосАп во второй фазе ее функционирования. Примером для иллюстрации выбрано критическое приложение, написанное на языке программирования С, используемое в интересах Европейского космического агенства (ЕКосАп). Это ПО включает около 10000 линий кода С [8]. Программный модуль состоит из трех основных подмодулей:

1) подмодуль синтаксического анализа (ПМСА);

2) вычислительный подмодуль (ПМВТ);

3) подмодуль форматирования (ПМФТ).

На рис. 5 в соответствии с [13] представлена архитектурная модель данного ПО, которая фактически трансформируется в граф состояний приводимой дискретной марковской цепи (ДМЦ), используемой в качестве опорной модели.

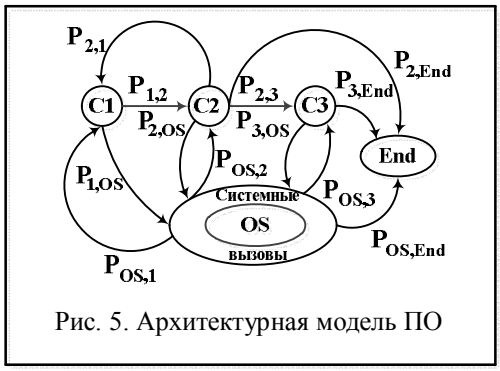

Рис. 5. Архитектурная модель ПО

Для реализации аналитико-стохастического моделирования рассмотрим следующие состояния, описывающие процесс функционирования программного модуля критического приложения: С1 – состояние функционирования ПМСА; С2 – состояние функционирования ПМВТ; С3 – состояние функционирования

ПМФТ; OS – состояние функционирования операционной системы (ОС) и системного обмена с подмодулями C; End – состояние завершения вычислительного процесса по управлению АС.

Значения переходных вероятностей для приводимой ДМЦ представлены в табл. 1 [13]. Моделирование выполняется для начальных условий в момент времени $t=0$, когда $P_{OS}(0)=1$, $P_{C1}(0)=0$, $P_{C2}(0)=0$, $P_{C3}(0)=0$, $P_{End}(0)=0$.

Таблица 1

Матрица переходных вероятностей

p_{ij}	C1	C2	C3	OS	End
C1	0,236	3E-4	0	0,77	7,8E-5
C2	0	0,632	0,029	0,34	5,0E-4
C3	0	0	9,9E-4	0,99	9,1E-4
OS	0,405	0,014	0,581	0	0
End	0	0	0	0	1

В общем виде матрица переходных вероятностей $P=[p_{ij}]$ может быть записана таким образом:

$$P=\begin{pmatrix} Q & C \\ 0 & I \end{pmatrix}, \qquad (2)$$

где Q – субстохастическая матрица размерностью $(n-\ell)$ на $(n-\ell)$; I – единичная матрица размерностью $\ell=1$ (т.е. размерность матрицы соответствует количеству поглощающих состояний); 0 – матрица размерностью ℓ на $(n-\ell)$, все элементы которой равны нулю; C – матрица размерностью $(n-\ell)$ на ℓ. Тогда, фундаментальная матрица M для k-го числа переходов из состояния s_i в состояние s_j может быть представлена в виде [13,14]

$$M=(I-Q)^{-1}=I+Q+Q^2+\ldots+Q^k=\sum_{k=0}^{\infty}Q^k. \qquad (3)$$

После несложных преобразований фундаментальная матрица (3) приводится к виду $M=I+MQ$. Обозначим переходы из состояния s_i в состояние s_j через $X_{i,j}$. Будем полагать, что ожидаемое (прогнозируемое) число переходов соответствует значению математического ожидания $\nu_{i,j}=E[X_{i,j}]$ или $m_{i,j}$. После чего ожидаемое число переходов из начального состояния $i=1$ в состояние j запишем как $\nu_{1,j}=m_{1,j}$. Далее в фундаментальной матрице можно выстроить диагональ из элементов [15]

$$M_D=\begin{cases} m_{i,j}, & \forall i=j, \\ 0, & i\neq j, \end{cases} \qquad (4)$$

и определить

$$\sigma^2=M(2M_D-I)-M_2, \qquad (5)$$

где $M_2=[m_{i,j}^2]$, $D[X_{i,j}]=\sigma_{i,j}^2$.

Используя подход, изложенный в [8], мы можем определить вероятность правильного функционирования ПО как

$$E[R]=\prod_i^n E[R_i^{X_{1,i}}]=\left(\prod_i^{n-1}R_i^{E[X_{1,i}]}\right)R_n, \qquad (6)$$

где $R_i^{E[X_{1,i}]}$ – вероятностный показатель функционирования i-й компоненты ПО за число посещений $E[X_{1,i}]$; R_n – вероятностный показатель завершения функционирования n-го компонента ПО.

Разложив (6) в ряд Тейлора, получим [16]

$$E[R]=\left[\prod_i^{n-1}\left(R_i^{m_{1,i}}+\frac{1}{2}(R_i^{m_{1,i}})(\log R_i)^2\sigma_{1,i}^2\right)\right]R_n, \qquad (7)$$

где $m_{1,i}=E[X_{1,i}]$, $D[X_{1,i}]=\sigma_{1,i}^2$.

При рассмотрении (7) необходимо учитывать, если $X_{1,n}=1$, то $m_{1,n}=1$, $\sigma_{1,n}^2=0$.

Для завершения архитектурного описания модели добавим состояние, учитывающее функционирование ОС. Предположим, что вероятностный показатель W правильного функционирования ОС известен. Тогда по аналогии с (7) будет справедливо следующее выражение:

$$W'=E[W^{X_{1,OS}}]=W^{m_{1,OS}}+\frac{1}{2}\left(W^{m_{1,OS}}\right)\times \\ \times(\log W)^2\sigma_{1,OS}^2, \qquad (8)$$

где $m_{1,OS}$, $\sigma_{1,OS}^2$ – математическое ожидание и дисперсия числа посещений ОС, соответственно.

Аналогично запишем выражение для остальных компонентов ПО:

$$R'=E[R_i^{X_{1,i}}]=R_i^{m_{1,i}}+\frac{1}{2}\left(R_i^{m_{1,i}}\right)\times \\ \times(\log R_i)^2\sigma_{1,i}^2, \qquad (9)$$

а выражение (7) представим в упрощенном виде как

$$E[R]=\left[\prod_i^{n-1}R_i'\right]R_nW'. \qquad (10)$$

В [8] ОС рассматривается как абсолютно надежная, правильно функционирующая система, для которой $W'\approx 1$.

Результаты моделирования, полученные с использованием специализированной программной угилиты SREPT (Software Reliability Estimation and Prediction Tool), представлены в табл. 2.

Основное соотношение для оценочной модели безотказности ПО критического приложения записывается следующим образом:

$$P(\tau_i)=\exp\left\{-\int_0^{\tau_i}\lambda_i(t)dt\right\}=\exp\{-\lambda_i\tau_i\}, \qquad (11)$$

где τ_i – время реализации пользовательских функций для i-го подмодуля C.

Таблица 2

Основные временные
и системные характеристики ПО

Исследуемый параметр		Величина
1. Временные характеристики, ч		
Продолжительность реализации пользовательских функций (для подмодулей С)		
С1	С2	С3
0,01128	0,00248	0,0001251
Продолжительность реализации системных вызовов (для ОС)		0,10589
2. Пользовательские и системные характеристики		
Среднее количество пользовательских функций (для модуля С)		407,4
Среднее количество системных вызовов (для ОС)		5442

3. Модель роста надежности ПО критического приложения (модель Гоела-Окумото)

В соотношении (11) значения интенсивности отказов λ_i определяют с использованием известных моделей роста надежности (МРН) ПО критического приложения [17]. Как правило, МРН ПО используются для получения статистической оценки функции интенсивности отказов $\lambda(t)$ программных модулей критических приложений. Основными целями моделирования роста надежности ПО являются:

– оценка продолжительности интервала безотказной работы программного обеспечения;

– оценка фактического уровня надежности программного обеспечения.

Определяя значение выбранного показателя безотказности (например, вероятность безотказной работы, средняя наработка на отказ, интенсивность отказов), в качестве которого в том числе может рассматриваться количество выявленных дефектов, мы тем самым выполняем оценку уровня надежности ПО.

В качестве МРН ПО предлагается [8] использовать модель Гоела-Окумото, базирующуюся на следующих допущениях [17 – 19]:

1. Наработки между отказами (сбоями) за время выполнения программного модуля распределены по экспоненциальному закону.

2. Прогнозируемое число отказов (сбоев) за фиксированное время выполнения программного модуля t распределено по закону Пуассона. Функция прогнозируемого числа отказов (сбоев) $\lambda(t)$ подчиняется граничным условиям $\lambda(0)=0$, $\lim\limits_{t\to\infty}\lambda(t)=N$, где N – конечное число.

3. Количество выявленных отказов (сбоев) ПО на интервале времени $[t;t+\Delta t]$, где $\Delta t\to 0$, пропорционально ожидаемому числу необнаруженных неисправностей $N-\lambda(t)$ и определяется как

$$\frac{d\lambda(t)}{dt}=\kappa\left(N-\lambda(t)\right),$$

где κ – коэффициент пропорциональности.

4. За период выполнения программного модуля $t\in[0;t_n]$ количество отказов (сбоев), наблюдаемых на каждом из непересекающихся интервалов времени $(0,t_1),(t_1,t_2),\ldots,(t_{n-1},t_n)$, где $t_1<t_2<\ldots<t_n$, является независимой величиной.

5. Ошибка, послужившая причиной возникновения отказа (сбоя) ПО, устраняется мгновенно без ее повторного проявления.

Принимая во внимание указанные допущения, функция интенсивности отказов (сбоев) для модели Гоела-Окумото может быть записана в следующем виде:

$$\lambda(t)=E_E e^{-\beta t}, \qquad (12)$$

где $E_E=\kappa N$ – ожидаемое значение интенсивности отказов (сбоев) ПО за конечное время его выполнения t; β – скорость снижения интенсивности отказов (сбоев) ПО за счет реализации механизма отказоустойчивости; κ – коэффициент пропорциональности, устанавливающий связь между количеством отказов и временем выполнения ПО. График функциональной зависимости $\lambda(t)$, определяемой согласно соотношения (12), представлен на рис. 6.

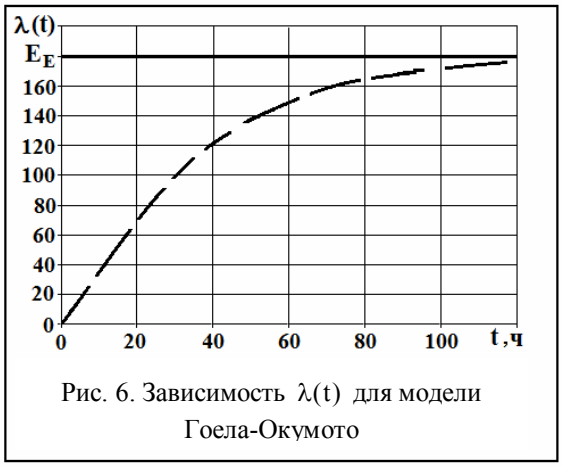

Рис. 6. Зависимость $\lambda(t)$ для модели Гоела-Окумото

В табл. 3 представлены результаты статистического оценивания параметров модели Гоела-Окумото, полученные в [8] на этапе клиентского использования ПО, архитектура которой представлена на рис. 5.

Таблица 3

Статистические оценки
параметров модели Гоела-Окумото

С1		С2		С3	
N	κ	N	κ	N	κ
13	0,0546	11	0,0946	5	0,0534

4. Оценка уровня надежности системы управления критическими инфраструктурами

На завершающем этапе АСМ выполним оценку уровня надежности СУ КрИ, архитектура которой представлена на рис. 1.

Для этого определим значения стационарного коэффициента оперативной готовности (КОГ), используя известное соотношение [10, 12]

$$K_{ог}(t,\tau) = K_г(t)P(\tau), \qquad (13)$$

где $K_г(t)$ – стационарный коэффициент готовности (КГ) наземного сегмента СУ КосАп; $P(\tau)$ – вероятность безотказной работы (ВБР) ПО критического приложения.

В соотношении (13) значения $K_г(t)$ определяют как характеристику надежности СУ КосАп для первой фазы ее функционирования (рис. 2) с применением системы дифференциальных уравнений Колмогорова-Чепмена (1).

Соответственно, значение $P(\tau)$ вычисляют как показатель надежности программного обеспечения критического приложения для второй фазы функционирования СУ КосАп (рис. 2).

Результаты расчета $K_г(t)$ представлены на рис. 4. Значения ВБР определяют как

$$P(\tau) = \prod_{s=1}^{\xi} P_s(\tau_s), \qquad (14)$$

где $P_s(\tau_s)$ – вероятность безотказной работы s-й составляющей программного модуля С за время реализации τ_s пользовательских функций s-м подмодулем; ξ – количество подмодулей, содержащихся в программном модуле С.

Далее вычисляют значения ВБР программного обеспечения по формуле (14), используя в качестве исходных данных как временные характеристики программного модуля (табл. 2), так и статистические оценки параметров модели Гоела-Окумото (табл. 3).

Значения стационарного коэффициента оперативной готовности определяют в соответствии с соотношением (13). График зависимости $K_{ог}(T_о, t)$ для указанных исходных данных представлен на рис. 7.

Обобщенные результаты оценки уровня надежности, определяемого значениями средней наработки между отказами (сбоями) $T_о$ (интенсивности отказов (сбоев) λ_{12}) и среднего времени восстановления $T_в$, компонентов СУ КосАп для фиксированных значений стационарного КОГ представлены в табл. 4.

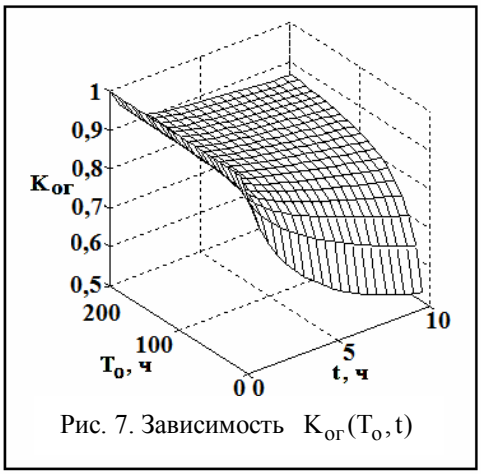

Рис. 7. Зависимость $K_{ог}(T_о, t)$

Таблица 4

Оценки показателей надежности системы управлении космическими аппаратами

Продолжительность эксплуатации, ч	КОГ	$T_в$, ч	λ_{12}, 1/ч
1	0,95	0,5	0,005
2,5	0,92		
5	0,88		
7,5	0,7		
10	0,55		

Полученные результаты (табл. 4) могут быть использованы для обоснования рациональных сроков использования по назначению СУ КрИ с учетом факторов безопасного, отказоустойчивого функционирования технологического оборудования и ПО критического приложения.

Выводы

Применение архитектурного подхода для моделирования процессов использования по назначению ПО критического приложения предоставляет широкие возможности по решению задач обеспечения эффективного функционирования СУ КрИ при приоритете их безопасности и надежности. Предлагаемый авторами подход позволяет выполнить оценку фактического уровня надежности системы управления критической инфраструктуры с учетом режимов эксплуатации ее компонентных составляющих и отказоустойчивости программного обеспечения. Разработанная архитектурно-ориентированная модель была использована для определения значений стационарного коэффициента готовности наземного сегмента СУ КосАп.

Дальнейшие перспективы применения архитектурного подхода связаны с оптимизацией расхода тестового ресурса программных модулей критического приложения и обоснования предельных величин показателей функциональной безопасности СУ КрИ.

Список литературы

1. Мищенко В.О. CASE-оценка критических программных систем. В 3-х томах. Т. 1. Качество / В.О. Мищенко, О.В. Поморова, Т.А. Говорущенко / Под ред. Харченко В.С. – Х.: Нац. аэрокосмический ун-т "Харьк. авиац. ин-т", 2012. – 201 с.

2. Gray J. A Census of Tandem System Availability Between 1985 and 1990 / J. Gray Technical report // Tandem Computers Inc., 1990, (90.1) (Part number 33579). – 267 p.

3. Voas J. Predicting how badly "good" software can behave / J. Voas,F. Charron, G. McGraw, K. Miller, M. Friedman // IEEE Software. 1997, 14(4), p. 73 – 83.

4. Software aging analysis Of-Off The Shelf Software Items, Salvatore Orlando, Ph.D Thesis, 2007, p. 236.

5. Naruemon Wattanapongsakorn. Reliability optimization models for embedded systems with multiple applications / Steven P. Levitan // IEEE Transactions on Reliability, vol. 53, no. 3, Sep 2004.

6. Rice F. Wanda. Simplifying the solution of redundancy allocation problems / Cassady C. Richard and Wise R. Tracy // Proc. of the Annual Reliability & Maintainability Symposium (RAMS '99). – P. 190-194, 1999.

7. Michael R. Lyu. Optimal Allocation of Test Resources for Software Reliability Growth Modeling in Software Development / Sampath Rangarajan and Aad P.A. van Moorsel // IEEE Transactions on Reliability, vol. 51, no. 2, June 2002.

8. Pietrantuono R. Software Reliability and Testing Time Allocation: An Architecture-Based Approach / R. Pietrantuon, S. Russo, Kishor S. Trivedi // IEEE Transactions on Software Engineering, vol. 36, no. 3, 2010, p. 323 – 337.

9. Безопасность критических инфраструктур: математические и инженерные методы анализа и обеспечения / Под ред. Харченко В.С. – МОН Украины, Нац. аэрокосм. ун-т им. Н.Е. Жуковского «ХАИ», 2011. – 641 с.

10. Теоретические основы проектирования информационно-управляющих систем космических аппаратов / В.В. Кульба, Е.А. Микрин, В.В. Павлов, В.Н. Платонов, Ин-т проблем упр. им. В.А. Трапезникова РАН – М.: Наука, 2006. – 579 с.

11. Численные методы / Н.С. Бахвалов, Н.П. Жидков, Г.М. Кобельков, Классический университетский учебник – М.: Бином. Лаборатория знаний, 2008. – 640 с.

12. Острейковский В.А. Теория надежности: учеб. для вузов / В.А. Острейковский. – М.: Высшая школа, 2003. – 463 с.

13. Pasquini A. Sensitivity of reliability-growth models to operational profile errors vs testing accuracy / A.N. Crespo, and P. Matrella // IEEE Transactions on Reliability, vol. 45, no.4, pp. 531 – 540, 1996.

14. Trivedi K.S. Probability and Statistics with Reliability, Queuing and Computer Science Applications / John Wiley and Sons, 2001. – 356 p.

15. Chin-Yu Huang. An Assessment of Testing-Effort Dependent Software Reliability Growth Models / Sy-Yen Kuo, Michael R. Lyu // IEEE Trans. On Reliability, vol. 56, no. 2, June 2007.

16. Sharma V.S. Quantifying software performance, reliability and security: An architecture-based approach / V.S. Sharma, K.S. Trivedi // The journal of systems and software, vol. 80, issue 4, pp. 493 – 509, 2007.

17. Application of Goel-Okumoto Model in Software Reliability Measurement [Электронный ресурс] – Special Issue of International Journal of Computer Applications (0975 – 8887) on Issues and Challenges in Networking, Intelligence and Computing Technologies – ICNICT 2012.

18. Musa J.D. A logarithmic Poisson execution time model for software reliability measurement / J.D. Musa, K. Okumoto // Proc. 7th Int. Conference Software Engineering, Orlando, Florida, March 26-29, 1984. – P. 230 – 238.

19. Goel A.L. Time-Dependent Error-Detection Rate Model for Software Reliability and Other Performance Measures / A.L. Goel, K. Okumoto // IEEE Trans. Reliability 28, 1979. – P. 206 – 211.

Поступила в редколлегию 11.12.2013

Рецензент: д-р техн. наук, проф. В.С. Харченко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.

АРХІТЕКТУРНИЙ ПІДХІД ДО ОЦІНКИ НАДІЙНОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ УПРАВЛІННЯ КРИТИЧНОЇ ІНФРАСТРУКТУРИ

О.В. Іванченко, О.С. Солдатенко, О.С. Владімірова

Широке впровадження та застосування інформаційних технологій в системах управління критичними інфраструктурами (КІ) суттєво підвищує ефективність їхнього застосування за призначенням. Оперативне вирішення достатньо складних завдань управління КІ неможливе без використання відповідного програмного забезпечення (ПЗ). В той же час уроки, які були отримані під час аналізу наслідків найбільш крупних аварій і катастроф за участю різних інфраструктурних утворювань, свідчать про недооцінку вкладу ПЗ в погіршення загального рівня надійності. З метою усунення вказаної проблеми пропонується застосовувати комплексний підхід до оцінки надійності ПЗ системи управління КІ з врахуванням різних режимів її експлуатації. Розроблений підхід базується на архітектурному представленні процесів функціонування відповідного програмного модуля та ефективній реалізації процедури його тестового контролю.

Ключові слова: *система управління критичної інфраструктури, програмне забезпечення, архітектурний підхід, безперервний та дискретний марковські ланцюги, ймовірність безвідмовної роботи, коефіцієнт оперативної готовності.*

ASSESSMENT OF RELIABILITY SOFTWARE OF THE CRITICAL INFRASTRUCTURE'S MANAGEMENT SYSTEM: AN ARCHITECTURE-BASED APPROACH

O.V. Ivanchenko, E.S. Soldatenko, E.S. Vladimirova

Wide introduction and application of information technologies in management systems of the critical infrastructures (CIs) significantly increases the effectiveness of their intended use. Operative solutions rather complex of CIs management tasks are impossible without appropriate software (SW). At the same time, the lessons learned during the analysis of the effects of most major accidents and disasters involving various infrastructure entities indicate about underestimation of the contribution to the deterioration of the general reliability SW's level. In order to eliminate this problem are propose to use a comprehensive approach to assessing the reliability of CI's software management systems, taking into account different modes of its operation. The developed approach is based on an architectural representation of the functioning of corresponding software module and the effective implementation of the procedures for its control test.

Keywords: *critical infrastructures management system, software, architectural approach, Continuous Time Markov Chain, Discrete Time Markov Chain, probability of uptime, rate of operational readiness.*