DOI 10.37882/2223-2966.2025.02.09

МОДЕЛИРОВАНИЕ НАДЕЖНОСТИ СИСТЕМЫ ОБОРУДОВАНИЯ, ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ВЗАИМОДЕЙСТВИЯ ОБОРУДОВАНИЯ И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

MODELLING THE RELIABILITY OF HARDWARE, SOFTWARE AND HARDWARE/SOFTWARE SYSTEM INTERACTIONS

V. Gulyaev

Summary. Hardware-software co-design systems are widely used today in various application areas. The article considers the essence and approaches to modelling the reliability of hardware-software systems and interaction of their parts. The diagram of state transitions at failures in software-hardware interaction is formalised separately. The approach for modelling the reliability of software-hardware system is described, which is based on a unified model of system reliability using Markov process, including three main categories of failures: hardware failures, software failures and hardware-software interaction failures.

Keywords: hardware, software, reliability, model.

Гуляев Владислав Евгеньевич
Аспирант, Дальневосточный Федеральный Университет
Guliaev.ve@dvfu.ru

Аннотация. Системы совместного проектирования аппаратно-программных средств находят сегодня свое широкое применение в различных прикладных областях. В статье рассмотрена сущность и подходы к моделированию надежности программно-аппаратных систем и взаимодействию их частей. Отдельно формализована диаграмма переходов состояний при сбоях в программно-аппаратном взаимодействии. Описан подход для моделирования надежности программно-аппаратной системы, который базируется на унифицированной модели надежности системы с использованием марковского процесса, включающей три основные категории сбоев: сбои оборудования, сбои программного обеспечения и сбои взаимодействия оборудования и программного обеспечения.

Ключевые слова: оборудование, программа, надежность, модель.

Отказ критически важных систем управления процессами в реальном времени, таких как системы управления безопасностью атомных электростанций и воздушным движением, военные и медицинские системы, промышленные объекты может иметь катастрофические последствия. Поэтому важно определить их надежность, чтобы гарантировать, что риски для общества находятся в приемлемых пределах. Подобного рода системы представляют собой встраиваемые компьютерные комплексы, состоящие как из аппаратных, так и из программных компонентов [1].

Существующие работы по моделированию надежности систем совместного проектирования в основном предполагали, что подсистемы оборудования и программного обеспечения ведут себя независимо друг от друга. В результате в большинстве существующих публикаций и отчетов подразумевалась либо независимость между оборудованием (HW) и программным обеспечением (SW), либо фиксированная доля зарегистрированных отказов оборудования для представления контактов HW/SW. Однако, практика свидетельствует о том, что эти две подсистемы могут иметь различные взаимодействия в реальной жизни, в результате чего сформулированные предположения в ряде случаев оказались ошибочными. Состояние работоспособности, например деградация и отказ, аппаратных компонентов составляет один из критических факторов, влияющих на производительность программных подсистем [2]. Соответственно, к числу значимых причин, вызывающих отказ программного обеспечения, относится неисправность/ошибки аппаратной платформы, на которой расположено программное обеспечение. Помимо этого, немаловажным является тот факт, что традиционные модели надежности часто игнорируют зависимость сбоев между подсистемами и наличие замаскированных данных об отказе системы, поэтому они не могут точно отразить анализ моделирования надежности всего программно-аппаратного комплекса.

Таким образом, вопросы, связанные с анализом вероятностных взаимодействий HW/SW в сочетании с надежностью аппаратного и программного обеспечения, представляются критически важными для моделирования и оценки общей надежности программно-аппаратных систем, что и предопределило выбор темы данной статьи.

Над усовершенствованием архитектурно-ориентированного подхода, который использует дискретно-непрерывное марковское моделирование на этапе проектирования программного обеспечения для руководства выбором характеристик надежности, которые должны быть включены в его архитектуру, трудятся Беля-

ков Д.С., Калинин Е.О., Брагин Д.С., Ehlimana Cogo, Almir Karabegović, Zhixuan Wang, Wen Chen.

Анализу программных ошибок, как основы новой методологии оценки взаимодействия между аппаратными и программными подсистемами, которая определяет надежность системы в целом, посвятили свои публикации Старжинская Н.В., Чернова А.И., Левачков С.О., Ларкин Е.В., Привалов А.Н., K. Isaieva, Marc Fauvel, Nicolas Weber.

Наличие широкого спектра публикаций по данной тематике свидетельствует об активном внимании ученых к различным вопросам и проблемам. Однако ряд спорных и недостаточно проработанных моментов требует проведения более детального анализа. Так, в дальнейшем развитии и уточнении нуждается модель надежности программно-аппаратной системы (ПАС) с маскированными данными и зависимостью от отказов. Кроме того, более четкого обоснования требует методология анализа надежности многокомпонентных систем, подверженных широкому спектру зависимых конкурирующих процессов отказа.

Итак, цель статьи заключается в рассмотрении подходов к моделированию надежности системы оборудования, программного обеспечения и взаимодействия между ними.

Моделирование надежности ПАС во многом аналогично моделированию надежности только аппаратных систем. В процессе создания модели разрабатываются и используются блок-схемы надежности элементов системы, которые точно отображают взаимосвязь между аппаратными платформами и программным обеспечением, выполняемым на этих платформах, и применяются для оценки показателей надежности [3]. Для сложных структур создаются также диаграммы состояний, точно фиксирующие уникальные взаимосвязи моделируемой структуры. На рис. 1 представлена блок-схема моделирования надежности ПАС.

Для ПАС, критичных к безопасности, непрерывная готовность является важным требованием, а надежность программного обеспечения — одним из ключевых компонентов работоспособности всей системы. Надежность определяется как вероятность того, что программное или аппаратное обеспечение будет выполнять поставленную задачу в течение некоторого периода времени при заданных условиях [4]. Надежность всей ПАС определяется путем вычисления надежности каждого компонента. В ПАС ее компоненты предназначены для незави-

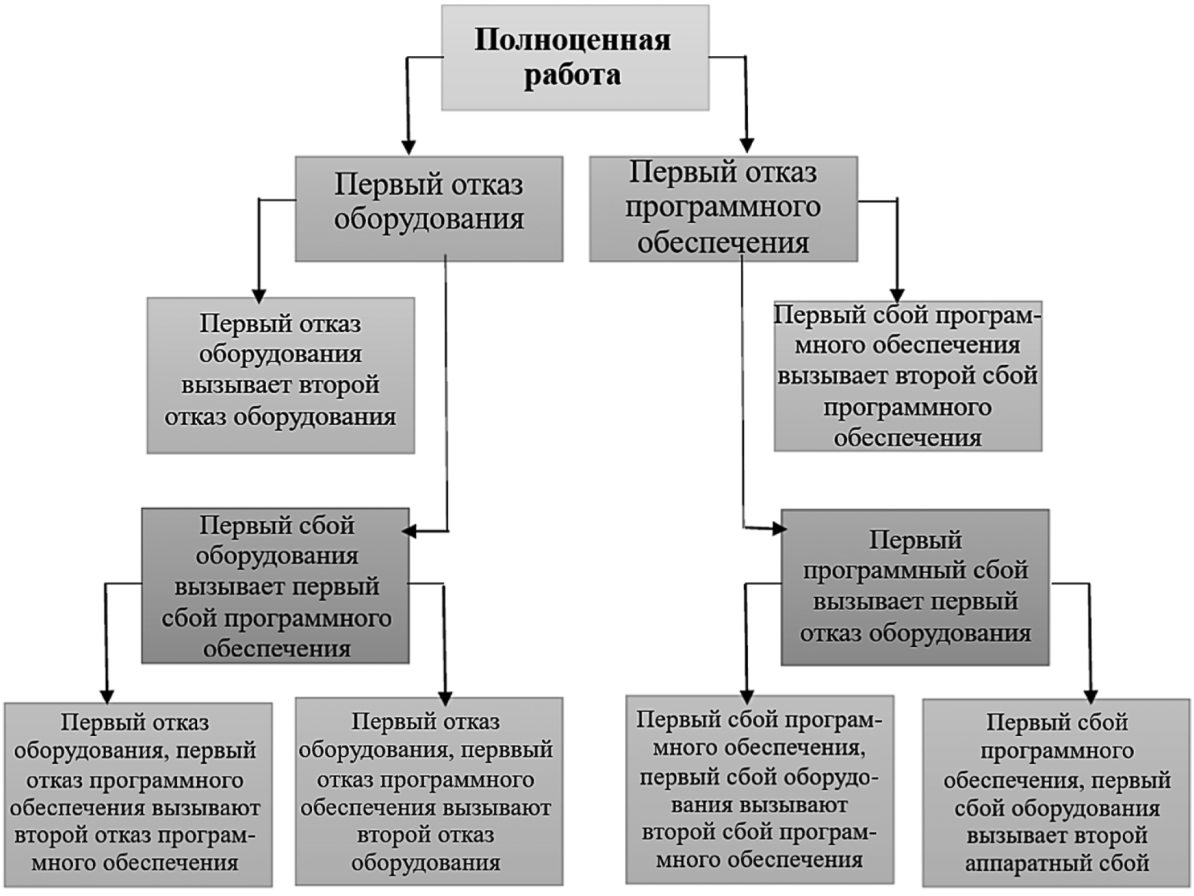

Рис. 1. Блок-схема моделирования надежности ПАС

симой работы. Однако для достижения общей цели все они должны работать совместно и правильно. Следовательно, отказ одного компонента может привести к отказу всей системы. Поэтому реальная ПАС имеет ряд конфигураций с точки зрения ее надежности. Надежность компонента ПАС вычисляется вероятностью успешного функционирования $P(X_i)$ этого компонента независимо. Надежность всей системы R_{sys} может быть пересечением вероятностей $P(X_i)$ каждого компонента системы, представленных в виде уравнения:

$$R_{sys} = P(X_1 \cap X_2 \cap X_3 \cap \ldots \cap X_3 = \\ = P(X_1)P(X_2|X_1)P(X_3|X_1X_2)\ldots P(X_n|X_1X_2\ldots X_{n-1}) \quad (1)$$

Если отказ одного из компонентов системы влияет на интенсивность отказов остальных компонентов (т.е. при отказе одного компонента изменяются характеристики распределения ресурса других компонентов), необходимо учитывать условные вероятности в уравнении (1). Однако в случае независимых компонентов, уравнение (1) может быть переписано следующим образом:

$$R_{sys} = P(X_1)P(X_2)P(X_3)\ldots P(X_n) \quad (2)$$

Учитывая, что интенсивность отказов для каждого компонента системы не зависит и постоянна от их использования во времени, уравнение (2) имеет следующий вид (3):

$$R_{sys} = \prod_{i=1}^{n} R_i \quad (3)$$

где R_i — надежность каждого независимого компонента i, а n — количество компонентов на одну операцию в системе.

Вероятность того, что ПАС в момент времени $t_0 \geq 0$ находится в состоянии s_i, определяется как $h_i(t), i = 0,1,2,3,\ldots,10$. В свою очередь $h(t) = \left[h_1(t), h_2(t), \ldots, h_{10}(t)\right]$ в момент времени t представляет собой вектор вероятностей ряда. Тогда начальным условием является:

$$h_i(0) = \begin{cases} 1, i = 0 \\ 0, i = 1,2,3,\ldots,10 \end{cases} \quad (4)$$

Дифференциально-разностные уравнения, полученные из конфигурации системы, имеют следующий вид:

$$\frac{d}{dt}h_0(t) = -(\delta_1 + \delta_2)h_0(t) + \eta_1 h_1(t) + \eta_2 h_2(t)$$

$$\frac{d}{dt}h_1(t) = \\ = -(\eta_1 + \delta_1 + \delta_2)h_1(t) + \delta_1 h_0(t) + \eta_2 h_3(t) + \eta_1 h_5(t)$$

$$\frac{d}{dt}h_2(t) = \\ = -(\eta_2 + \delta_1 + \delta_2)h_2(t) + \delta_2 h_0(t) + \eta_1 h_4(t) + \eta_2 h_6(t) \quad (5)$$

$$\frac{d}{dt}h_3(t) = \\ = -(\eta_2 + \delta_1 + \delta_2)h_3(t) + \delta_2 h_1(t) + \eta_2 h_7(t) + \eta_1 h_9(t)$$

$$\frac{d}{dt}h_4(t) = \\ = -(\eta_1 + \delta_1 + \delta_2)h_4(t) + \delta_1 h_2(t) + \eta_2 h_8(t) + \eta_1 h_{10}(t)$$

Далее используем процесс Маркова для представления перехода состояний при взаимодействии аппаратного и программного обеспечения, как показано

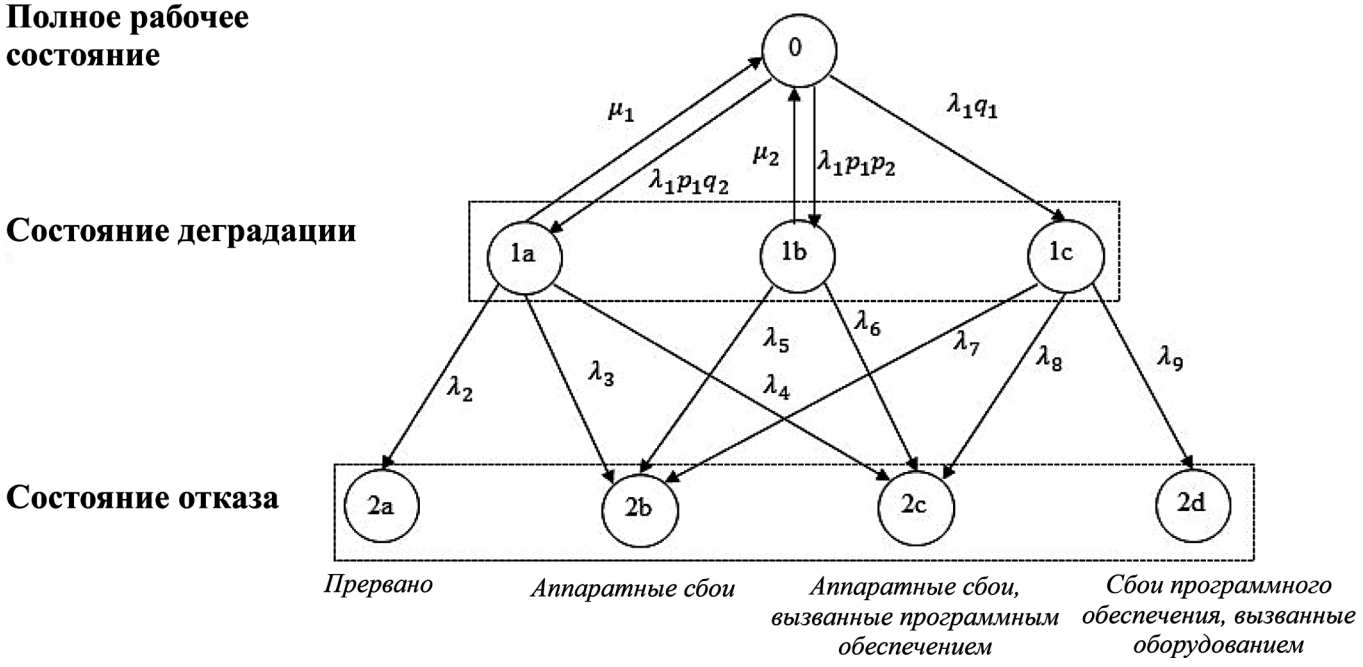

Рис. 2. Диаграмма переходов состояний при сбоях в программно-аппаратном взаимодействии

на рисунке 2. Как указано в предположениях модели, для программно-аппаратных взаимодействий определены три основных состояния: состояние полной работоспособности, состояние деградации и состояние отказа, а также восемь подсостояний {0, 1a, 1b, 1c, 2a, 2b, 2c, 2d}. Состояние (0) представляет собой полностью рабочее состояние, которое означает, что система находится в идеальном рабочем положении. Состояние деградации (1a) означает, что обнаружен частичный отказ оборудования, но он не может быть восстановлен программными средствами. Состояние деградации (1b) означает, что выявлен частичный отказ оборудования, который можно устранить программными средствами. Состояние деградации (1c) свидетельствует о том, что частичный отказ оборудования не установлен. Состояние отказа (2a) означает прерывание выполнения. Состояние отказа (2b) указывает на наличие аппаратных сбоев. Состояние отказа (2c) означает программно-индуцированные аппаратные сбои. Состояние отказа (2d) указывает на программные сбои, которые вызваны аппаратным обеспечением.

Процесс обнаружения и устранения программных ошибок рассматривается как неоднородный пуассоновский процесс [5]. Также предположим, что время, которое тестировщик программного обеспечения тратит на устранение обнаруженных программных ошибок, пренебрежимо мало. В частности, скорость обнаружения программных сбоев и общее количество программных сбоев в программе рассматриваются в данном исследовании как константы. Таким образом, для оценки ожидаемого количества обнаруженных программных сбоев до момента времени t может быть использована модель G-O. Модель G-O имеет следующий вид:

$$m(t) = a\left(1 - e^{-bt}\right)$$

где $m(t)$ обозначает ожидаемое количество программных сбоев до момента времени t. Константы a и b обозначают скорость обнаружения программных сбоев и их общее количество в программе, соответственно.

Таким образом, подводя итоги проведенного исследования, можно сделать следующие выводы. В статье описан подход для моделирования надежности программно-аппаратной системы и взаимодействия ее частей. Разработанный подход базируется на унифицированной модели надежности системы на основе процесса Маркова, включающей три основные категории сбоев: сбои оборудования, сбои программного обеспечения и сбои взаимодействия оборудования и программного обеспечения.

ЛИТЕРАТУРА

1. Лукин В.Н. Проблемы сопровождения аппаратно-программных комплексов // Труды МАИ. 2022. № 123.
2. Скачков С.А. Дуализм аппаратных угроз функциональной надежности управляющих вычислительных систем // Успехи современной радиоэлектроники. 2022. Т. 76. № 11. С. 40–51.
3. Kizito Salako Demonstrating software reliability using possibly correlated tests: Insights from a conservative Bayesian approach // Quality and Reliability Engineering International. 2023. Volume 40, Issue 3. P. 19–29.
4. Xiao-Jian Yi. A new reliability analysis method for software-intensive systems with degradation accumulation effect based on goal-oriented methodology // Quality and Reliability Engineering International. 2023. Volume 40, Issue 1. P. 24–31.
5. Anum Shafiq An updated software reliability model using the shanker model and failure data // Quality and Reliability Engineering International. 2024. Volume 40, Issue 4. P. 90–95.

Журнал «Современная наука: актуальные проблемы теории и практики»