УДК 004.052.42

Е.В. Сидорова, О.А. Котельникова

ИССЛЕДОВАНИЕ МОДЕЛЕЙ НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В СРЕДЕ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ANY LOGIC

Нижегородский государственный технический университет им. Р.Е. Алексеева

Статья посвящена сравнительному анализу моделей надежности программного обеспечения на основе их имитационного моделирования в среде Any Logic. Представлены этапы имитационного моделирования и предложены рекомендации к выбору модели надежности программного обеспечения.

Ключевые слова: надежность программного обеспечения, модели надежности программного обеспечения, моделирование, имитационная среда Any Logic.

Чрезвычайно высокая структурная сложность программных систем, динамичность версий и технологий сильно влияют на задачу снижения числа уязвимостей в программных системах. На надежность программного обеспечения (ПО), в основном, влияют содержащиеся в нем ошибки и их последствия. Все действия разработчиков по повышению надежности направлены на то, чтобы минимизировать ошибки при разработке программ и постараться выявить и устранить их как можно быстрее после их написания. Можно утверждать, что число ошибок в программе характеризует в первую очередь ее создателей-программистов и используемый ими IT-инструментарий. Также следует отметить большие объемы современных программных проектов и их сложность, что повышает вероятность ошибок в них.

Одним из путей повышения уровня безопасности ПО является использование математических моделей, позволяющих получить гарантированные оценки показателей безопасности программного обеспечения и эффективности технологии его разработки.

Математические модели надежности программного обеспечения

Для количественной оценки показателей надежности ПО используют модели надежности, под которыми понимаются математические модели, построенные для оценки зависимости надежности от заранее известных или определенных в ходе выполнения заданий параметров.

В рамках данной статьи, для проведения исследования, выбраны четыре модели надежности: Шумана, Муса, Джелински-Моранды и Миллса [2-4]. В среде имитационного моделирования Any Logic [1] был проведен ряд вычислительных экспериментов, цель которых – определить, насколько эффективны выбранные модели для получения различных показателей надежности программ на отдельных стадиях жизненного цикла ПО.

Дадим краткую характеристику выбранным моделям.

Модель Шумана [2]: при корректировке новые ошибки не вносятся, интенсивность обнаружения ошибок пропорциональна числу оставшихся ошибок, нахождение первоначального количества ошибок – метод перебора, функция надежности:

$$R_i(t) = e^{-\lambda_i t}, \quad (1)$$

где λ_i – интенсивность отказов на i-м этапе; t – продолжительность этапа**.**

Модель Муса [3]: надежность программного обеспечения на этапе эксплуатации оценивается по результатам тестирования, первоначальное количество ошибок определяется только при помощи модели Шумана, функция надежности

$$R(t) = e^{\frac{T}{t}}, \quad (2)$$

где T – суммарное время тестирования; t – средняя наработка до отказа после тестирования.

Модель Джелински-Моранды [4]: время до следующего отказа распределено экспоненциально, интенсивность отказов пропорциональна количеству оставшихся в программе ошибок, нахождение первоначального количества ошибок – метод подбора, функция надежности

$$R(t_i) = e^{-\lambda t_i}, \tag{3}$$

где λ – интенсивность отказов; t_i – среднее время до следующего отказа.

Модель Миллса [3]: необходимо перед началом тестирования внести искусственные ошибки, все ошибки, как естественные, так и искусственные, имеют равную вероятность быть найденными в процессе тестирования, первоначальное количество ошибок

$$N = n\frac{S}{v}, \tag{4}$$

где n – обнаруженные собственные ошибки; S – количество искусственно внесенных ошибок; v – обнаруженные искусственные ошибки.

Вторая часть модели Миллса связана с проверкой гипотезы о первоначальном количестве ошибок.

Результаты моделирования в среде Any Logic

Модель Джелински-Моранды

Согласно данной модели, вероятность безотказной работы ПО как функция времени t, равна

$$P(t_i) = e^{-\lambda t_i}, \tag{5}$$

где λ – интенсивность отказов:

$$\lambda_i = C_D(N - (i-1)), \tag{6}$$

где N – первоначальное количество ошибок; C_D – коэффициент пропорциональности

$$C_D = \frac{\sum_{i=1}^{k-1} \frac{1}{N-i+1}}{\sum_{i=1}^{k-1} t_i}, \tag{7}$$

В формуле (5) отсчет времени начинается от момента последнего (i-1)-го отказа программы. По методу максимума правдоподобия на основании формулы (5), функция правдоподобия имеет вид

$$F = \prod_{i=1}^{k-1} C_D(N-i+1)e^{-C_D(N-i+1)t_i}, \tag{8}$$

где k – номер прогнозируемого отказа.

Логарифмическая функция правдоподобия имеет вид

$$L = \ln F = \sum_{i}^{k-1} \left[\ln(C_D(N-i+1)) - C_D(N-i+1)t_i\right]. \tag{9}$$

Отсюда условия для нахождения экстремума

$$\frac{\partial L}{\partial C_D} = \sum_{i=1}^{k-1}\left[\frac{1}{C_D} - (N-i+1)t_i\right] = 0, \tag{10}$$

$$\frac{\partial \mathrm{L}}{\partial \mathrm{N}} = \sum_{i=1}^{k-1}\left[\frac{1}{N-i+1} - C_D t_i\right] = 0. \qquad (11)$$

Из формулы (11) находится коэффициент пропорциональности C_D - формула (7).

Для нахождения первоначального количества ошибок используют итеративную процедуру, в результате которой находят значение, обеспечивающее наименьшую ошибку (метод перебора) при решении уравнения

$$(k-1)\frac{\sum_{i=1}^{k-1} t_i}{\sum_{i=1}^{k-1}\frac{1}{N-i+1}} = \sum_{i=1}^{k-1}(N-i+1)t_i\,, \qquad (12)$$

где t_i – интервалы между отказами.

Рис. 1. Метод перебора

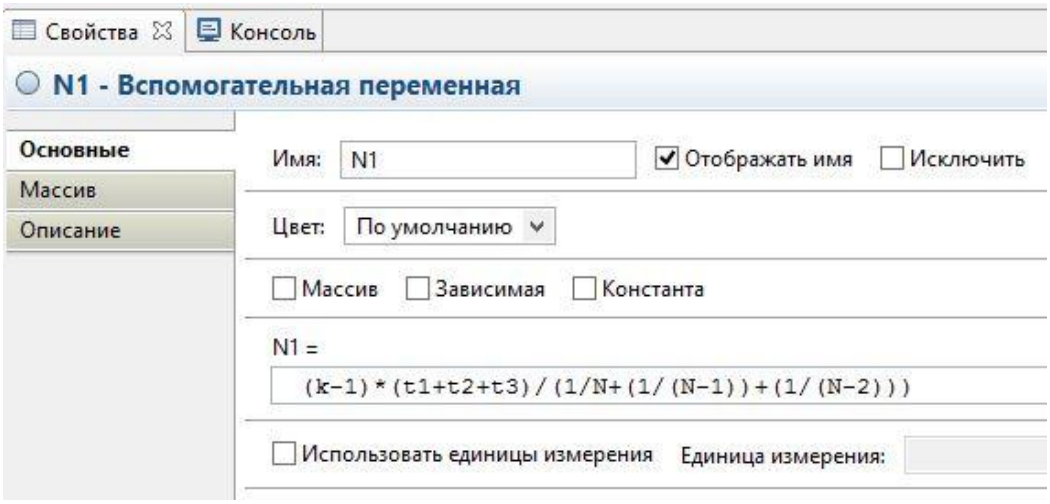

Рис. 2. Переменная *N*1

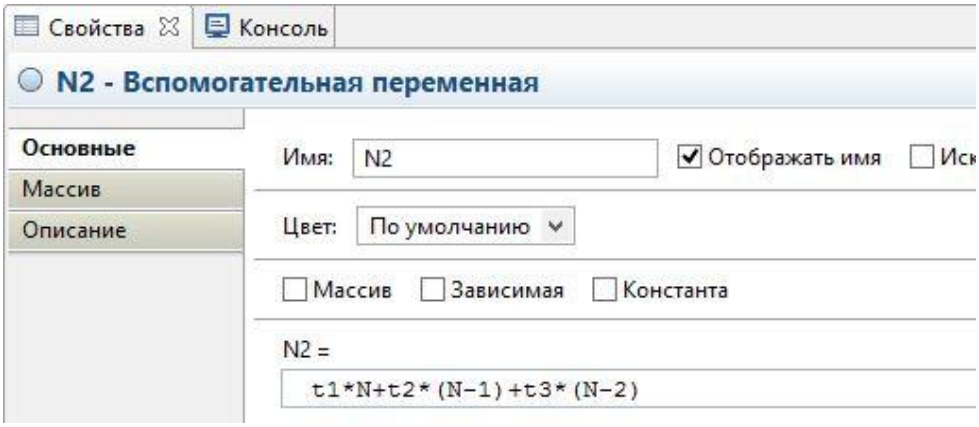

Рис. 3. Переменная *N*2

После нахождения значений параметров модели C_D и N можно найти интенсивность отказов λ (формула (6)), а также время от последнего до следующего отказа t_{k+1} и вероятность безотказной работы через время t_{k+1} после последнего отказа (формула (5)).

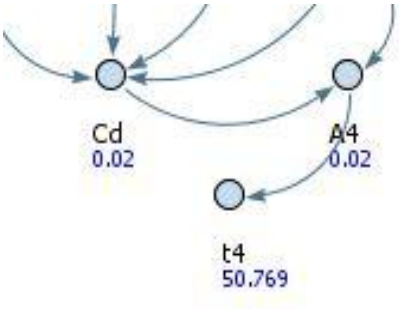

Рис. 4. Связь между коэффициентом пропорциональности C_D, интенсивностью отказов λ и временем от последнего отказа до следующего t_4

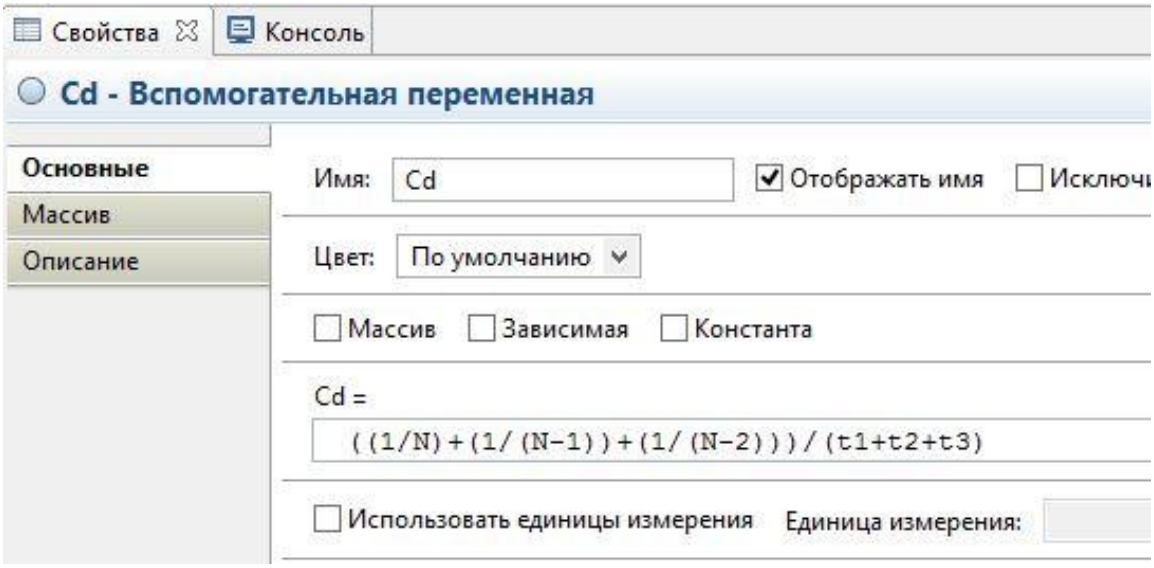

Рис. 5. Коэффициент пропорциональности C_D

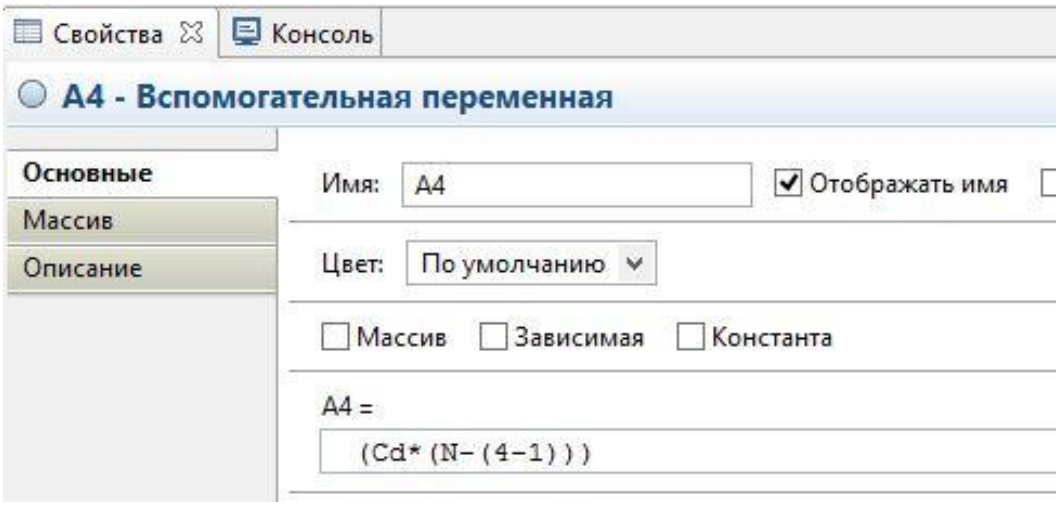

Рис. 6. Интенсивность отказов λ

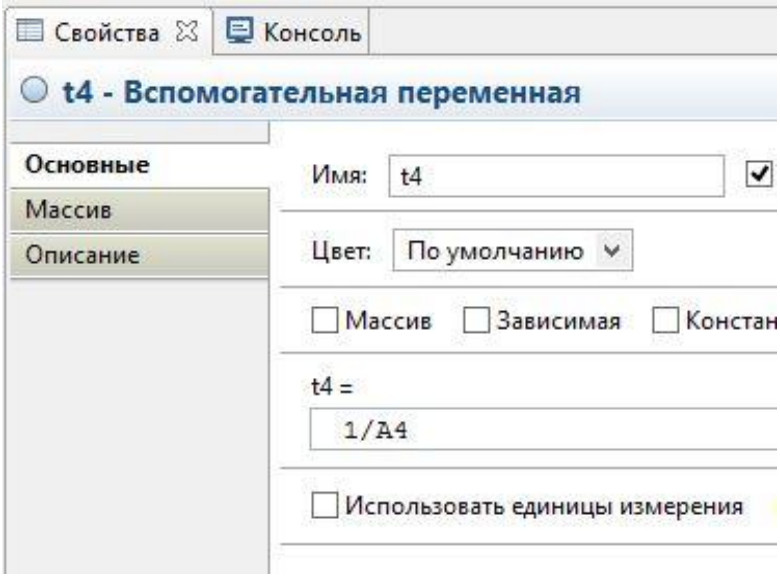

Рис. 7. Время от последнего отказа до следующего t_4

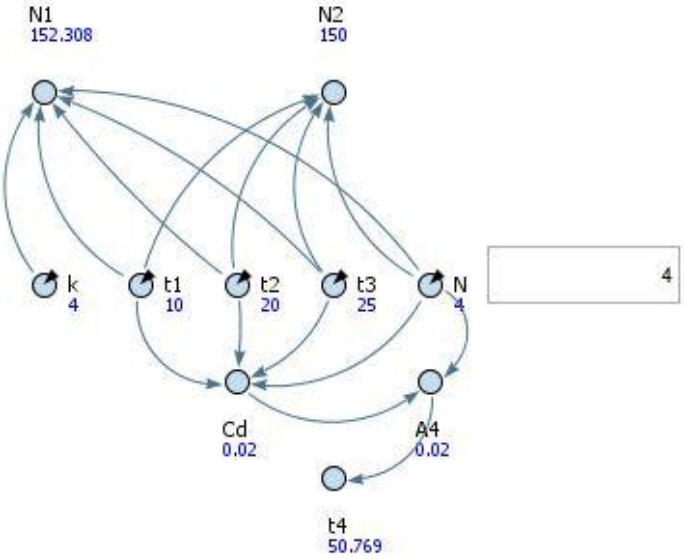

Рис. 8. Работа модели Джелински-Моранды

Основным преимуществом модели является простота расчетов. Недостаток состоит в том, что при неточном определении величины N интенсивность отказов программы может стать отрицательной, что приводит к бессмысленному результату. Кроме того, предполагается, что при исправлении обнаруженных ошибок не вносятся новые ошибки, что тоже не всегда выполняется.

Модель Маркова

Рассмотрим систему, начинающую работу в момент времени $t = 0$. Система работает до появления ошибки в соответствии с предпочтенным критерием. Результаты эксперимента собираются в отрезки времени, за которые могут произойти отказы в работе. Тогда переменная t' времени случайного сбоя может быть определена как:

$$t'(\xi) = \xi, \tag{13}$$

где $\xi \geq 0$ – местоположение точек на дискретной временной оси эксперимента.

Предположим, что случайная переменная t' имеет функцию распределения

$$F(t) = P\{\xi : t(\xi) \le t\}, \tag{14}$$

и, если она существует, то плотность функции распределения будет

$$f(t) = \frac{dF(t)}{dt}. \tag{15}$$

Надежность системы $R(t)$ определяется вероятностью отсутствия сбоя на интервале времени $[0; t]$:

$$R(t) = P\{t' \ge t\}. \tag{16}$$

Под готовностью системы к моменту времени t понимается вероятность того, что система находится в рабочем состоянии во время t, и определяется как результат простого сложения всех вероятностей состояний занятости:

$$A(t) = \sum_{k=0}^{\infty} P_{n-k}(t). \tag{17}$$

Предположим, что в начальный период ($t = 0$) система содержит неизвестное число (n) ошибок. В качестве начала отсчета времени работы системы выбирается начало фазы тестирования. Принимаем также, что процессы обнаружения и исправления ошибок реализуются попеременно и последовательно и зависят от текущего состояния системы.

Предположим, что к моменту t система только что вошла в состояние ($n - k$), т. е. ошибка k только что устранена. Тогда в интервале времени $(0, t_k+1)$, где $t = t_{k+1}$ может проявиться ошибка ($k + 1$) при принятой постоянной интенсивности проявления ошибок λ_k.

На основании формулы функции надежности, порождающей вероятность отсутствия сбоев в интервале времени от 0 до t, получим выражение для надежности:

$$R_k(\tau) = e^{-\lambda_k t}, \tag{18}$$

где $0 \le \tau \le t_{k+1}$; k=0, 1, 2, …

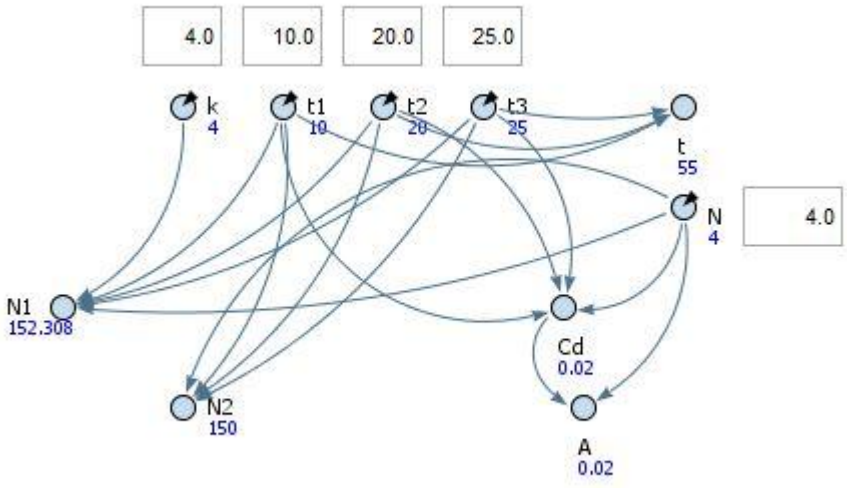

Рис. 9. Работа модели Маркова

Предполагается, что модель в начальный период будет использоваться со значением λ, которое было получено на базе накопления прошлого опыта.

Модель Шумана

В модели Шумана программное обеспечение на i-м этапе тестирования имеет функцию надежности

$$R_i(t) = e^{-\lambda_i t}, \tag{19}$$

где λ_i – интенсивность обнаружения ошибок:

$$\lambda_i = C(N - n_{i-1}), \tag{20}$$

где N – первоначальное количество ошибок; $(N - n_{i-1})$ – количество ошибок, оставшихся к началу i-го этапа; n – общее число обнаруженных и исправленных при тестировании ошибок:

$$n = m_i + \ldots + m_k, \tag{21}$$

n_i – число ошибок, исправленных к началу (i+1)-го этапа тестирования:

$$n_i = m_1 + \ldots + m_i, \tag{22}$$

C – коэффициент пропорциональности, равный:

$$C = \frac{\sum_{i=1}^{k} \frac{m_i}{N - n_{i-1}}}{\sum_{i=1}^{k} t_i}, \tag{23}$$

где t_i – длительность этапов тестирования; m_i – число отказов на i-м этапе; k – общее число этапов тестирования.

Для нахождения первоначального количества ошибок в программном обеспечении используется уравнение

$$\sum_{i=1}^{k} m_i \frac{\sum_{i=1}^{k} t_i}{\sum_{i=1}^{k} \frac{m_i}{N - n_i}} = \sum_{i=1}^{k} (N - n_i) t_i, \tag{24}$$

Уравнение представляет собой метод перебора, с помощью которого находят значение, обеспечивающее наименьшую ошибку при решении данного уравнения.

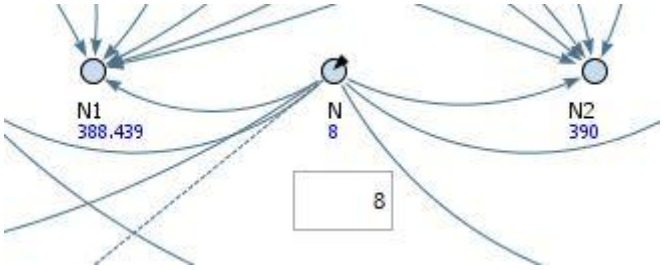

Рис. 10. Метод перебора по уравнению (24)

При известных значениях k (общее число этапов тестирования), t_i (длительность этапов тестирования), m_i (число отказов на i-м этапе) можно найти значения параметров модели C и N по формулам (23) и (24).

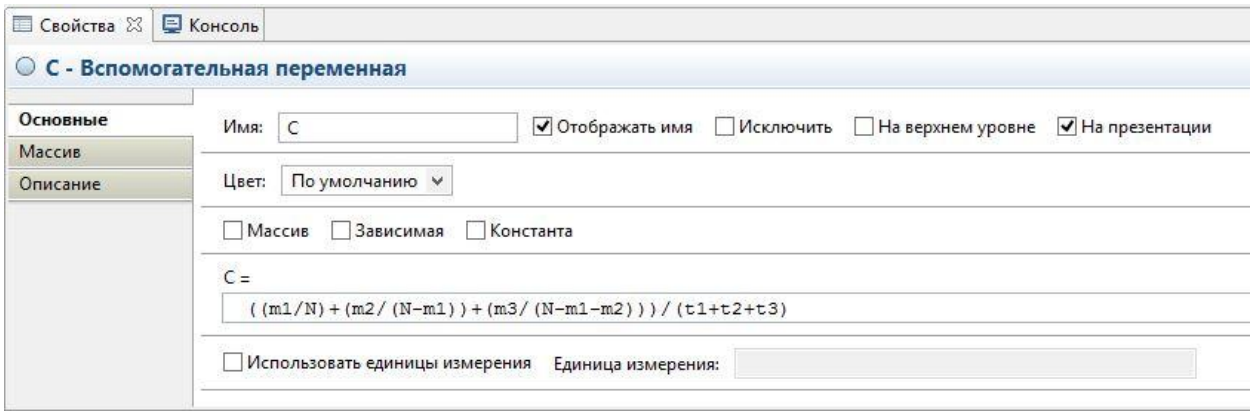

Рис. 11. Коэффициент пропорциональности *C*

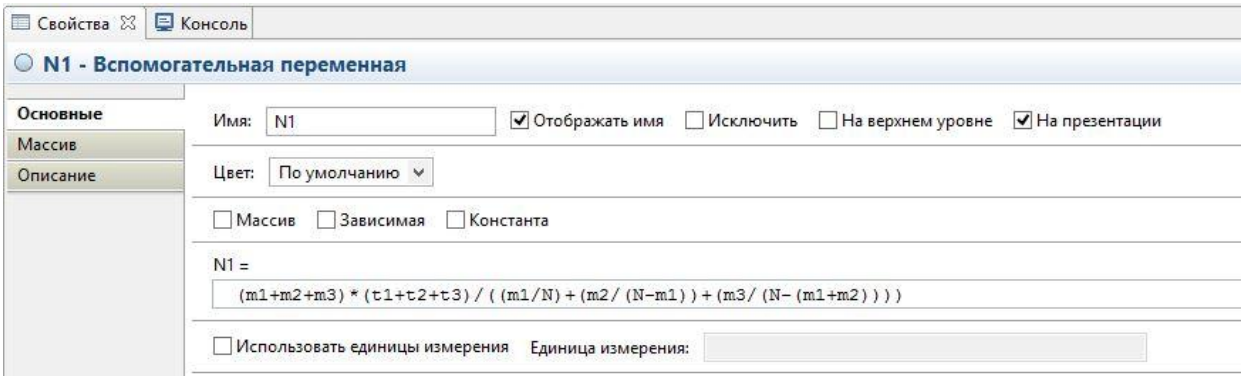

Рис. 12. Метод перебора, параметр *N*1

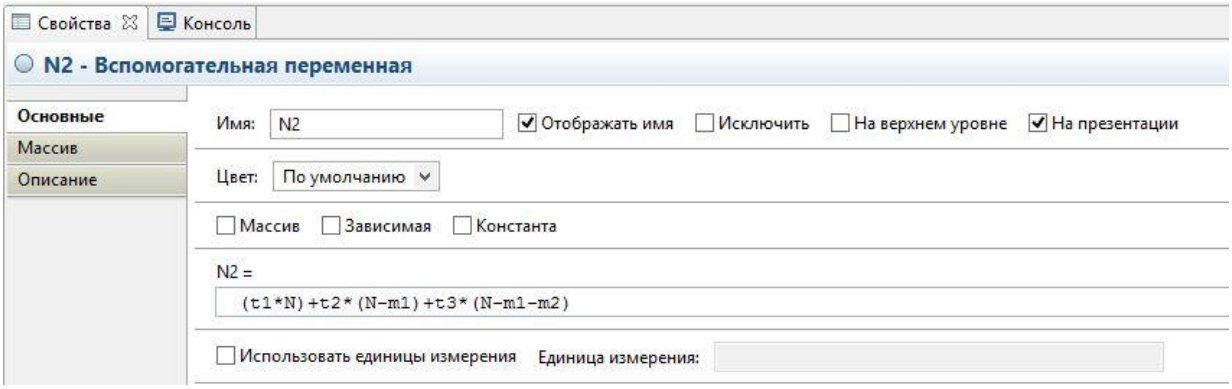

Рис. 13. Метод перебора, параметр *N*2

После чего можно определить показатели:

- число оставшихся ошибок в программном обеспечении:

$$N_T = N - n; \tag{25}$$

- функцию надежности программного обеспечения по завершении тестирования:

$$R(t) = e^{-\lambda t}; \tag{26}$$

- интенсивность обнаружения отказов:

$$\lambda = C(N - n). \tag{27}$$

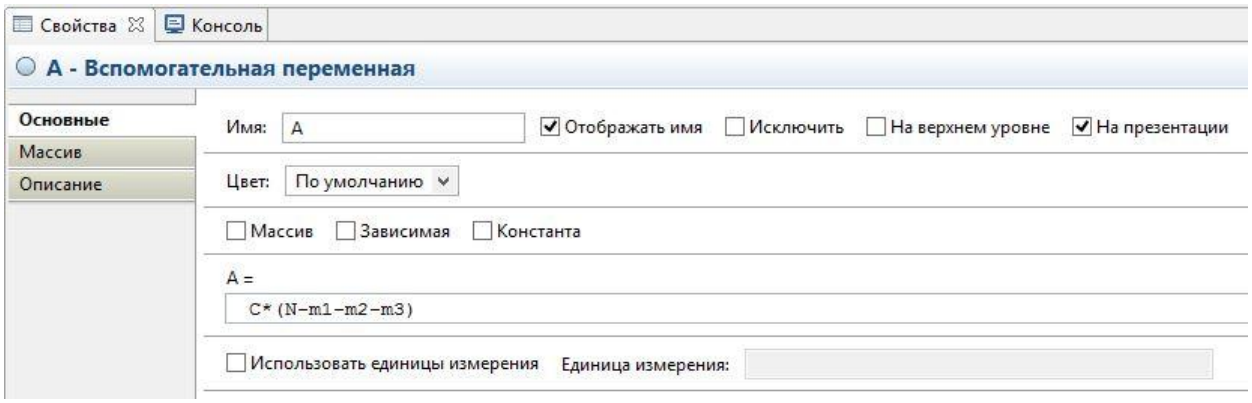

Рис. 14. Интенсивность обнаружения отказов λ

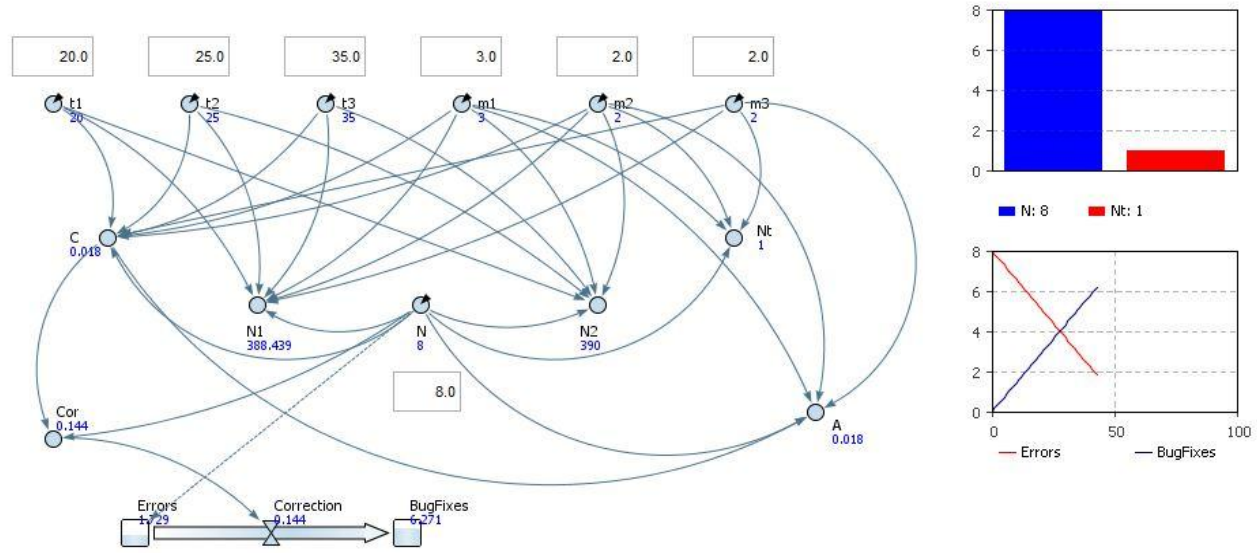

Рис. 15. Работа модели Шумана

Представить наглядно первоначальное количество ошибок и количество оставшихся в программе ошибок позволяют графические инструменты в среде Any Logic. Это столбиковая диаграмма и временной график.

Столбиковая диаграмма отображает несколько элементов данных в виде столбцов, «растущих» в заданном направлении от базовой линии. Размеры столбцов пропорциональны значениям соответствующих элементов данных.

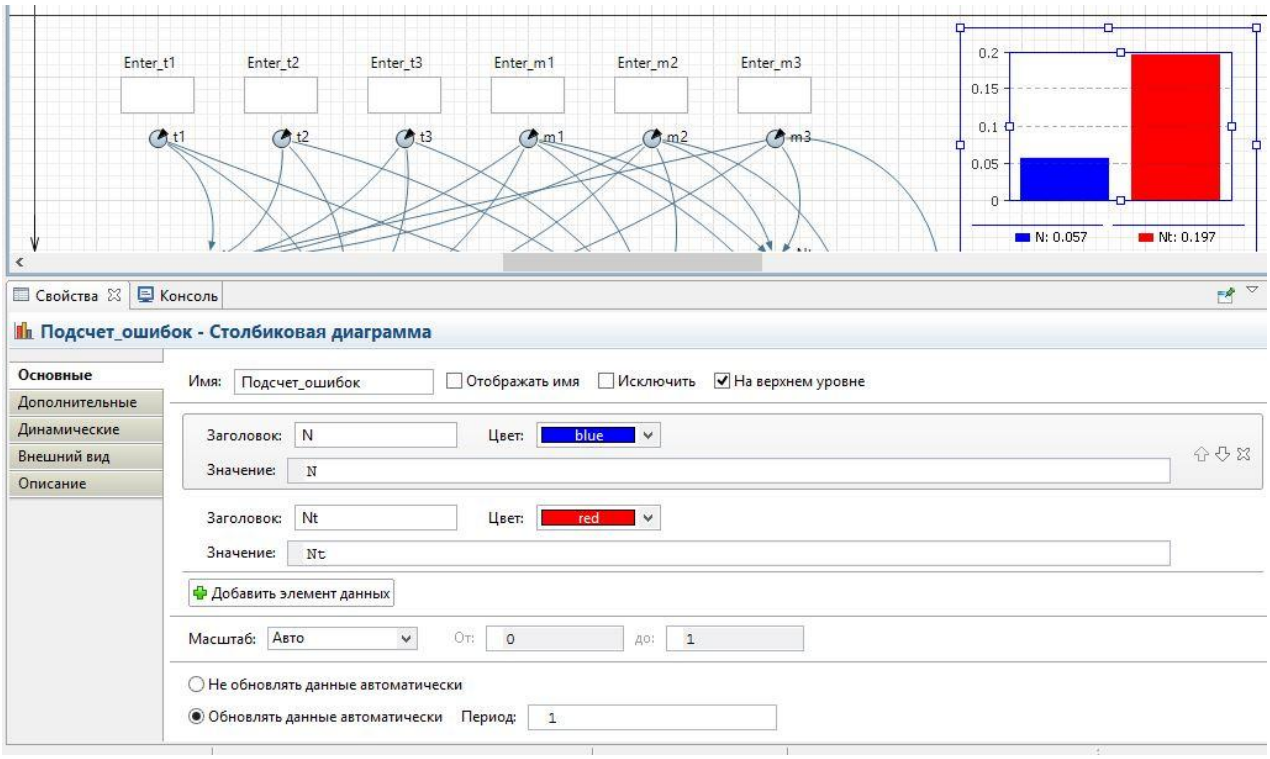

Рис. 16. Столбиковая диаграмма для подсчета ошибок

При разных значениях N, k, t_i и m_i диаграмма подсчитывает ошибки и выдает результат на экран.

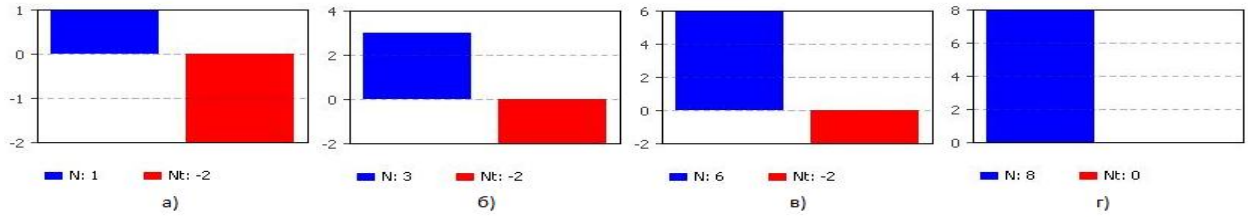

Рис. 17. Подсчет оставшихся ошибок:
а – при *N*=1; *б* – при *N*=3; *в* – при *N*=6; *г* – при *N*=8, когда все ошибки исправлены

Сравнительный анализ имитационных моделей

Для оценки показателей надежности программного обеспечения и выбора эффективной математической модели рекомендуем использовать табл. 1, пользуясь критерием выбора модели

$$K = \max \sum_{i=1}^{n} \alpha_i \beta_i , \qquad (28)$$

где α_i – весовой коэффициент *i*-го свойства рассматриваемой модели; β_i – коэффициент значимости *i*-го свойства рассматриваемой модели: β_i=0, если свойством модель не обладает, β_i=1, если этим свойством модель обладает.

Таблица 1

Свойства моделей

Свойства моделей	Модели					
	Миллса	Шумана	Джелински-Моранды	Муса	Маркова	Нельсона-Коркорэна
Машино-независимость	1	0	0	0	0	1
Возможность априорной оценки	0	0	0	0	0	0
Точность оценки	0	1	1	1	1	1
Возможность прогнозирования параметра	0	1	1	1	1	1
Устойчивость к порядку следования тестов	1	0	0	0	1	1
Учет категорий ошибок	0	0	0	0	0	0
Инвариантность к потоку проявления ошибок	1	0	0	0	0	1
Инвариантность к потоку исправления ошибок	0	0	0	0	0	0
Инвариантность к уровню подготовки программиста	0	1	1	1	1	0
Независимость от предыдущего состояния	1	0	0	0	0	1

Рассмотренные модели позволяют оценивать различные свойства ПО.

Выводы

В среде Any Logic были смоделированы основные принципы работы отдельных моделей надежности ПО. Проведен ряд вычислительных экспериментов, результаты которых показали, что на разных стадиях жизненного цикла программы эффективны различные модели надежности, следовательно, выбранные модели не являются универсальными в применении. Сформирована сравнительная таблица, служащая базой для выбора конкретной модели надежности, исходя из различных критериев.

Библиографический список

1. **Боев, В.Д.** Компьютерное моделирование: пособие для практических занятий, курсового и дипломного проектирования в AnyLogic7 / В.Д. Боев. – СПб.: ВАС, 2014. – 432 с.
2. **Фатуев, В.П.** Надежность автоматизированных информационных систем / В.П. Фатуев, В.И. Высоцкий, В.И. Бушинский. – Т.: ТГУ, 1998. – 104 с.
3. **Усенко, О.А.** Модели и методы оценки программного обеспечения информационных систем: учеб. пособие / О.А. Усенко. – Т.: ТТИ ЮФУ, 2008. – 40 с.
4. **Шураков, В.В.** Надежность программного обеспечения систем обработки данных / В.В. Шураков. – М.: Статистика, 1981. – 216 с.

Дата поступления
в редакцию 22.10.2015

E. V. Sidorova, O.A. Kotelnikova

INVESTIGATION OF MODELS OF SOFTWARE RELIABILITY IN SIMULATION ENVIRONMENT ANY LOGIC

Nizhny Novgorod state technical university n.a. R.E. Alexeev

Purpose: Conduct a study of models of software reliability in a simulation tool Any Logic.
Design/methodology/approach: As objects of study chosen theoretical models Dzhelinski reliability, Markov, Schumann, Musa. For selected models conducted a simulation using the Any Logic tool with the following analysis of results of the study.
Finding: As a result, a comparative table is composed of models based on various criteria software.
Research limitation/implication: This study provides the basis for the selection method of calculating the theoretical model of software reliability in various stages of the life cycle.

Key words: software reliability, model software reliability, modeling, simulation environment Any Logic.