УДК 004.052.42

А.В. Тырва

МЕТОДИКА ЗАДАНИЯ ИСХОДНЫХ ДАННЫХ ДЛЯ МОДЕЛЕЙ НАДЕЖНОСТИ ПРОГРАММНЫХ СРЕДСТВ ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА

Выполнен сравнительный анализ моделей надежности программных средств. Особое внимание уделено используемым в моделях исходным данным, предположениям и области применения моделей. Приведено описание предлагаемой модели надежности на основе нестационарной системы обслуживания. Описана методика задания исходных данных для моделирования.

надежность программных средств, моделирование, подготовка исходных данных

Введение

На железнодорожном транспорте используется широкий набор специализированных программных средств (ПС), участвующих в оперативном управлении технологическими процессами, работой станций, системы сетевого документооборота и делопроизводства, ПС для технико-экономических расчетов и многие другие.

Задача обеспечения надежности ПС, используемых на железнодорожном транспорте, является важной и актуальной.

С обеспечением надежности ПС связаны процессы измерения, оценивания и предсказания надежности. Для количественного оценивания и предсказания надежности ПС могут использоваться модели надежности. Они позволяют выразить показатели надежности (вероятность безотказной работы, наработка на отказ, вероятность отсутствия ошибок) через известные параметры (интенсивности отказов, метрики сложности и др.). При определении пригодности той или иной модели важным вопросом является соответствие решаемой задаче предположений, использованных в модели, а также пригодность модели на рассматриваемом этапе жизненного цикла. Другим важным вопросом, возникающим при практическом применении моделей надежности, является вопрос сбора и подготовки исходных данных.

Далее в статье приводится сравнительный анализ разработанных ранее и широко применяемых на практике моделей надежности ПС (систематическое описание моделей приведено в [1, 2]). Для каждой модели указаны используемые предположения и связанные с ними недостатки. Затем приводится описание разработанной модели надежности ПС, основанной на использовании цепи Маркова с дискретным

множеством состояний и непрерывным временем. Также приводится методика задания исходных данных для моделирования.

1 Обзор моделей надежности программных средств

Большинство существующих моделей использует предположение об экспоненциальном характере распределения интервалов времени между обнаружением программных ошибок и конечном числе ошибок. Большинство моделей не определяет вид времени функционирования (календарное или оперативное), хотя модель Мусы использует два вида времени.

В качестве исходных данных для моделирования используется последовательность интервалов времени между обнаружением ошибок либо последовательность числа обнаруженных ошибок за периоды времени. Эти данные могут быть собраны в процессе выполнения программных проектов и использованы для моделирования на этапе тестирования и отладки или для моделирования будущих аналогичных проектов на начальных этапах жизненного цикла – проектирования, реализации.

Исторически одной из первых была предложена модель Джелинского-Моранды. В модели используется предположение о том, что интервалы времени между моментами обнаружения ошибок распределены по экспоненциальному закону с интенсивностью $\lambda(t)$, пропорциональной числу не выявленных ошибок. То есть интенсивность обнаружения ошибок $\lambda(t)$ постоянна на интервале времени между обнаружением i-й и $(i+1)$-й ошибок и равна:

$$\lambda(t) = \phi(N - i + 1),$$

где ф – коэффициент пропорциональности;

N – общее число ошибок.

Входными данными для модели является последовательность интервалов времени между обнаружением программных ошибок $X_1, X_2, \ldots, X_n$ – независимые экспоненциально распределенные случайные величины со средним значением $1/\phi(N-i+1)$. Предполагается, что обнаруженная ошибка сразу исправляется. Ошибки обнаруживаются независимо друг от друга.

В геометрической модели Моранды используется предположение об экспоненциальном распределении интервалов времени между обнаружением последовательных программных ошибок с интенсивностью, зависимость которой от количества остающихся в программе ошибок имеет вид геометрической прогрессии:

$$\lambda(t) = D\phi^{i-1},$$

где D, ф – константы, параметры модели, $0<\phi<1$.

Другой вариант геометрической модели Моранды предполагает изменение интенсивностей обнаружения ошибок через равные промежутки времени, а не в момент обнаружения очередной ошибки.

Предположение о пропорциональной зависимости интенсивности обнаружения ошибок от количества остающихся не исправленными ошибок используется в модели Шумана. Данная модель также содержит зависимость интенсивности обнаружения ошибок от скорости исполнения команд и числа команд в программе. Масштабная переменная модели учитывает, что структура программы (например, циклы) может приводить к повторениям последовательностей команд и ошибки вне этой последовательности не проявляются.

В модели Шика-Волвертона используется предположение о пропорциональной зависимости интенсивности обнаружения i-й ошибки от количества не исправленных ошибок и длительности отладки t:

$$\lambda(t) = \phi(N - i + 1)t ,$$

где ф – коэффициент пропорциональности;

N – общее число ошибок.

Временные интервалы между обнаружением ошибок распределены по рэлеевскому закону.

Модель Липова является обобщением моделей Джелинского-Моранды и Шика-Волвертона. Модель допускает обнаружение в течение интервала t_i более одной ошибки. Интенсивность обнаружения ошибок пропорциональна числу не исправленных по истечению (i–1)-го интервала времени ошибок, суммарному времени тестирования и среднему времени поиска ошибок в текущем интервале времени:

$$\lambda(t)=\phi(N - i + 1)(T_{i-1}+t_i/2),$$

где $T_{i-1}=\sum_{j=1}^{i-1} t_j$ – суммарное время тестирования до истечения (i–1)-го интервала времени.

Модель Вейбулла определяет следующий вид зависимости интенсивности обнаружения ошибок от числа не исправленных ошибок:

$$\lambda(t)=m\lambda^m t^{m-1},$$

где m, λ – параметры модели, $0<m<1$.

Модель Мусы относится к классу экспоненциальных, то есть предполагает экспоненциальное распределение времени обнаружения отказов. При этом в модели используются два вида времени функционирования: суммарное время тестирования и отладки вплоть до контрольного момента, когда производится оценка надежности программы, и оперативное время исполнения программы, планируемое от контрольного момента. Число устраненных ошибок n за время отладки τ определяется следующим образом:

$$n = N(1 - e^{-\frac{C\tau}{M_0 T_0}}),$$

где N – общее число программных ошибок;

M_0 – общее число ошибок, которые могут проявиться при отладке;

T_0 – средняя наработка на отказ в начале отладки;

C – коэффициент сжатия тестов.

В модели Литтлвуда интервалы времени между последовательным обнаружением ошибок распределены по экспоненциальному закону. Интенсивность предполагается случайным процессом, описываемым гамма распределением. Одним из параметров модели является вид функции $\psi(i)$, отражающей квалификацию программистов и сложность задачи (чем быстрее растет данная функция, тем быстрее растет надежность ПС).

Кроме приведенных, широко известны и другие экспоненциальные модели, а также модели, использующие для описания длительностей интервалов времени между обнаружением ошибок более общие законы распределения: Вейбулла, гамма и другие (см. [1, 2]).

Описанные выше модели относятся к классу непрерывных динамических аналитических моделей. Также были разработаны модели, учитывающие архитектуру ПС. Например, в модели Чунга-Смагина [3] для анализа надежности ПС представляется в виде совокупности программных компонент (модулей). Для каждого модуля определяются значения показателей надежности, после чего рассчитывается надежность всего ПС. В этом случае каждый модуль ПС представляет собой аналог элемента расчета теории надежности. Данная модель позволяет решить и обратную задачу – планирование испытаний с целью повышения надежности.

В [4] предложены варианты использования данной модели для различных видов архитектуры: отказоустойчивых ПС с резервированием, с последовательным и параллельным вызовом модулей, с клиент-серверной архитектурой. А в [5] предложен вариант повышения эффективности планирования испытаний путем учета в качестве факторов влияния на надежность всего ПС не только показателей надежности каждого модуля, но и его среднего времени работы.

Помимо этих моделей в научной литературе предлагались и другие подходы к моделированию надежности ПС. В частности, разработан ряд статических и эмпирических моделей, позволяющих вычислять показатели надежности на основе программных метрик.

Миллс разработал модель посева ошибок, предполагающую внесение в исходные коды программ псевдоошибок. Затем на основе процентной доли обнаруженных известных ошибок модель позволяет делать вывод о количестве скрытых фактических программных ошибок. Недостатком этой модели является предположение о том, что внесенные и реальные ошибки имеют одинаковое распределение и вероятность нахождения, что не подтверждается на практике. Модель Миллса была развита моделями Базина и Липова, которые учитывали несовершенность процесса

обнаружения дефектов. Однако данные модели по-прежнему сложно использовать на практике, так как требуются дополнительные трудозатраты на внесение и поиск новых ошибок. При этом модели не позволяют получать динамические характеристики надежности.

Анализ существующих подходов к моделированию надежности показывает, что общим является предположение о зависимости интенсивности отказов (обнаружения ошибок) от количества не исправленных программных ошибок (или номера текущей обнаруженной ошибки). Различные модели предлагают разные виды такой зависимости: пропорциональную, геометрическую, в виде условного гамма-распределения и другие. Это представляет сложность при практическом использовании моделей, так как требуется проводить расчеты для нахождения параметров нескольких моделей, а затем выбирать ту, которая ближе к исходным данным. Таким образом, существует задача разработки обобщенной модели, снимающей ограничения на вид зависимости интенсивности обнаружения и устранения ошибок от номера текущей ошибки.

Другим недостатком множества существующих моделей является ряд используемых упрощений, которые ограничивают их применение для моделирования процесса отладки программ. Например, модель Джелинского-Моранды и множество подобных моделей не учитывают время, которое затрачивается на устранение найденных программных ошибок.

Предположение об экспоненциальном (или другом) законе распределения временных интервалов обнаружения ошибок также представляется ограничением указанных моделей, так как не позволяют использовать их для оценки и прогноза надежности программных проектов с более общими распределениями потока обнаружения (и устранения) ошибок.

2 Модель надежности программных средств на основе нестационарной системы обслуживания

Для преодоления указанных недостатков рассмотренных моделей надежности ПС предложена модель надежности на основе модели нестационарных систем обслуживания [6]. В предлагаемой модели используется допущение об экспоненциальном распределении длительностей интервалов между обнаружением и исправлением последовательных ошибок. Однако, в отличие от названных выше моделей, не накладывается ограничений на вид зависимости интенсивностей обнаружения и устранения ошибок от номера ошибки (количества еще не исправленных ошибок в программе). Кроме того, как показано в [6], на основе данной модели могут быть построены более

общие модели, не ограниченные предположением об экспоненциальном виде распределения времени обнаружение и устранения ошибок.

Модель имеет следующие параметры:

N – общее число ошибок в программе;

λ_k – интенсивность обнаружения k-й ошибки, $k = 1, \ldots, N$;

μ_k – интенсивность исправления k-й ошибки, $k = 1, \ldots, N$.

Происходит последовательное обнаружение N программных ошибок, распределение длительностей интервалов между моментами обнаружения ошибок – экспоненциальное с интенсивностями $\{\lambda_1, \lambda_2, \ldots, \lambda_N\}$, зависящими от номера ошибки. Распределение времени исправления ошибок – экспоненциальное с интенсивностями $\{\mu_1, \mu_2, \ldots, \mu_N\}$, также зависящими от номера ошибки.

Такая система может быть представлена цепью Маркова с дискретным множеством состояний и непрерывным временем. Состояния системы в каждый момент времени характеризуется парой (i, j), где i – число обнаруженных, но еще не исправленных ошибок (i=0, …, N), а j – число уже исправленных ошибок (j=0, …, $N–i$). Вероятности пребывания системы в этих состояниях обозначим через $P_{i,j}(t)$. В таком представлении система будет иметь конечное число состояний. Обозначим ее как систему обслуживания типа $M(i)/M(j)/N$.

Переход из состояния (i, j) в состояние $(i+1, j)$ означает, что при тестировании была обнаружена $(i+j+1)$-я ошибка. Переход из состояния (i, j) в состояние $(i–1, j+1)$ означает, что была исправлена $(j+1)$-я ошибка.

Описанная модель описывается системой линейных дифференциальных уравнений (см. [6]). Задав начальные условия к системе в виде

$$P_{i,j}(0) = \begin{cases} 0, i + j \neq 0 \\ 1, i + j = 0 \end{cases}$$

можно найти численное решение соответствующей задачи Коши для произвольного значения t. Используя полученные решения, можно рассчитать характеристики надежности ПС: вероятность безотказной работы, вероятность отсутствия ошибок и др.

3 Определение исходных данных для расчета надежности программных средств

При выборе модели надежности необходимо учитывать следующие критерии. Во-первых, этап жизненного цикла ПС, на котором выполняется моделирование. Например, многие модели можно применять на этапе тестирования, используя уже собранные сведения о длительностях интервалов времени между отказами и продолжительности отладки. Однако при использовании исторических данных, те же модели можно использовать и на более ранних этапах.

Необходимо располагать требуемыми для моделирования исходными данными. Так, для моделей, не учитывающих структуру ПС (модели «черного ящика») требуется, например, вид распределения интервалов времени между отказами и интенсивности отказов. Для моделей на основе архитектуры ПС требуется также информация о количестве модулей, среднем времени их исполнения, вероятностях переходов между ними.

Для выбора наилучшей среди тех моделей, которые соответствуют этапу жизненного цикла и требуемым исходным данным, можно использовать критерий Акайке. Данный критерий является эвристической попыткой свести в один показатель два требования: уменьшение числа параметров модели и близость модели к исходным данным. Согласно этому критерию из двух моделей следует выбрать модель с наименьшим значением информационного критерия Акаике (см. [7]).

При использовании моделей надежности на этапе тестирования в качестве исходных данных могут использоваться те сведения об ошибках, которые получены за время испытаний до момента моделирования. Результатом моделирования с использованием моделей, описанных в разделе 1, в этом случае будет оценка общего числа программных ошибок и времени до нахождения очередной ошибки.

При использовании моделей надежности на более ранних этапах, когда статистика обнаружения и устранения ошибок недоступна, возникает проблема отсутствия исходных данных для моделирования. В этом случае могут использоваться данные, которые были собраны при разработке завершенных ранее проектов. Такая статистика должна быть скорректирована с учетом различия между проектами. Например, в зависимости от квалификации программистов и тестировщиков, возможностей используемых средств разработки и тестирования, сложности проекта, собранные временные характеристики процесса поиска и устранения ошибок могут быть скорректированы в большую или меньшую сторону. В любом случае оценка, полученная на этапе проектирования, ввиду отсутствия объективных данных о проекте хоть будет и не точной, однако достаточно важной, так как будет получена в самом начале жизненного цикла ПС, когда цена внесения изменений в проект не столь высока, как на последующих этапах жизненного цикла.

Далее предложена методика определения исходных данных для построения модели, описанной в разделе 2. Пусть при разработке уже завершенного проекта (1-ый проект) была собрана статистика временных характеристик процессов поиска и устранения программных ошибок, которая используется для моделирования надежности другого проекта на ранних этапах жизненного цикла (2-го проекта).

Пусть в процессе разработки 1-го проекта была собрана следующая статистика о процессах обнаружения и устранения программных ошибок: общее количество ошибок $N^{(1)}$, длительности интервалов времени

обнаружения ошибок $t_i^{(1)}$, i=1, … , $N^{(1)}$ и длительности интервалов времени устранения ошибок $f_i^{(1)}$, i=1, … , $N^{(1)}$. Известны данные о структуре ПС 1-го проекта: оно состоит из $l^{(1)}$ классов, и для каждого класса собраны значения объектно-ориентированных метрик сложности набора Чидамбера (длина пути иерархии наследования, количество классов-наследников, количество классов, использующих данный класс – в качестве поля, параметров методов, локальных переменных – и другие метрики, подробное описание которых см. в [8–10]). Для каждого класса на основе значений объектно-ориентированных метрик сложности рассчитана вероятность наличия в нем ошибки $\pi_i^{(1)}$, i=1, … , $N^{(1)}$ с использованием подхода, предложенного в статьях [8, 10]:

$$\pi_i^{(pr)} = \frac{1}{1+e^{-(\beta_0+\sum \beta_j x_j^i)}}, \quad (1)$$

где x_j^i – значение j-й объектно-ориентированной метрики сложности набора Чидамбера, j=1, … , S для i-го класса, i=1, …, $l^{(pr)}$;

S – количество используемых объектно-ориентированных метрик сложности набора Чидамбера;

β_j – параметры логистической регрессии, которые определяются по методу максимального правдоподобия исходя из статистики влияния сложности классов на их надежность (см. [8, 10]);

pr – номер проекта.

Рассчитанные вероятности $\pi_i^{(1)}$, i=1, … , $l^{(1)}$ используются для вычисления ожидаемого количества ошибок $\Pi^{(1)}$ как числа классов с оценкой вероятности наличия ошибки, превышающей некоторое пороговое значение h (например, h=0.5):

$$\Pi^{(pr)} = \sum_{i=1}^{l^{(pr)}} \delta(\pi_i^{(pr)} - h), \quad (2)$$

где $\delta(k) = \begin{cases} 1, k > 0 \\ 0, k \le 0 \end{cases}$,

pr – номер проекта.

Если доступны только средние вероятности наличия ошибок в классе π, ожидаемое количество ошибок $\Pi^{(1)}$ определяется как:

$$\Pi^{(pr)} = \pi l^{(pr)}. \quad (3)$$

Описанные выше данные о первом проекте используются для моделирования надежности 2-го проекта, для которого по результатам этапа проектирования известны общее число классов $l^{(2)}$ и рассчитаны предполагаемые вероятности наличия ошибок в классах $\pi_i^{(2)}$, i=1, … , $N^{(2)}$. По формулам (1–2) на основе значений $\pi_i^{(2)}$, i=1, … , $N^{(2)}$ рассчитывается величина $\Pi^{(2)}$. Предполагается, что 1-й и 2-й проекты похожи, поэтому должно соблюдаться соотношение:

$$\frac{\Pi^{(1)}}{N^{(1)}} = \frac{\Pi^{(2)}}{N^{(2)}},$$

откуда получается предполагаемое количество ошибок в ПС 2-го проекта:

$$N^{(2)} = \frac{N^{(1)}\Pi^{(2)}}{\Pi^{(1)}}. \quad (4)$$

Если $N^{(2)} \geq N^{(1)}$, то для упрощения моделирования имеет смысл представить ПС 1-го проекта в виде набора модулей, оценить надежность каждого модуля, а затем рассчитать показатели надежности всего ПС согласно модели Чунга-Смагина [3].

Если $N^{(2)} < N^{(1)}$, то для построения модели 2-го проекта могут быть использованы следующие значения: общее количество ошибок $N^{(2)}$ и длительности интервалов времени устранения ошибок $f_i^{(2)} = f_i^{(1)}$, $i=1, \ldots, N^{(2)}$. Длительности интервалов времени обнаружения ошибок $t_i^{(2)}$ должны быть экспертно скорректированы относительно $t_i^{(1)}$ с учетом соотношения количества ошибок $N^{(pr)}$ и объема ПС $S^{(pr)}$ (выраженного, например, в количестве строк кода). В простейшем случае, когда «плотность» ошибок в коде ПС обоих проектов равна, то есть $S^{(1)}/N^{(1)} = S^{(2)}/N^{(2)}$, для моделирования 2-го проекта можно использовать $t_i^{(2)} = t_i^{(1)}$, $i=1, \ldots, N^{(2)}$.

На основании этих данных находятся вероятности состояний модели надежности, описанной в разделе 2 (см. [6]) и рассчитываются показатели надежности, выполняется планирование и выбор стратегии испытаний ПС.

В качестве примера рассмотрим следующие данные и выполним моделирование надежности ПС. В табл. 1 приведены следующие данные по двум проектам: средние значения метрик сложности набора Чидамбера *NAI* – количество атрибутов классов, *DIT* – длина пути иерархии наследования, *OCMEC* – количество классов с параметрами методов типа данного класса, общее количество классов $l^{(1)}$ и $l^{(2)}$. Также в табл. 1 приведены средние вероятности наличия ошибки в классе $\pi^{(pr)}$, рассчитанные по формуле (1), и ожидаемые значения количества ошибок в программах $\Pi^{(pr)}$, рассчитанные по формулам (2–3).

ТАБЛИЦА 1. Исходные данные для моделирования

Характеристика	1-ый проект	2-ой проект
NAI	6	4
DIT	0.4	0.2
OCMEC	4	3
$l^{(pr)}$	12	9
$\pi^{(pr)}$	0.93	0.70
$\Pi^{(pr)}$	11.16	7

Кроме того, известно, что в процессе испытаний 1-го проекта было найдено и исправлено $N^{(1)}=10$ ошибок. Интервалы времени между обнаружением последовательных ошибок равны $t_1^{(1)}=1.25$ час., $t_2^{(1)}=1.32$ час., $t_3^{(1)}=1.39$ час., $t_4^{(1)}=1.47$ час., $t_5^{(1)}=1.56$ час., $t_6^{(1)}=1.67$ час., $t_7^{(1)}=1.79$ час., $t_8^{(1)}=1.92$ час., $t_9^{(1)}=2.08$ час., $t_{10}^{(1)}=2.27$. Интервалы времени между исправлением последовательных ошибок равны $f_1^{(1)}=2.50$ час., $f_2^{(1)}=2.63$

час., $f_3^{(1)}$=2.78 час., $f_4^{(1)}$=2.94 час., $f_5^{(1)}$=3.14 час., $f_6^{(1)}$=3.33 час., $f_7^{(1)}$=3.57 час., $f_8^{(1)}$=3.85 час., $f_9^{(1)}$=4.17 час., $f_{10}^{(1)}$=4.55 час.

На основе статистики, собранной при разработке 1-го проекта будет выполнено моделирование надежности 2-го проекта. Данные выбраны умозрительно и приведены с целью иллюстрации предлагаемой методики и модели надежности.

По формуле (4) получаем оценку числа ошибок во 2-ом проекте:

$$N^{(2)} = \frac{10 \cdot 7}{11.16} = 6.27 \approx 6.$$

Будем считать, что оба проекты выполнены одной группой разработчиков с использованием одинаковых технологий и средств разработки и можно ожидать, что «плотность» ошибок в коде ПС обоих проектов приблизительно равна. Тогда для моделирования надежности 2-го проекта будут использованы следующие данные: $N^{(2)}$=6, интервалы времени между обнаружением ошибок $t_i^{(2)}=t_i^{(1)}$, i=1, … , $N^{(2)}$ и интервалы времени между исправлением последовательных ошибок $f_i^{(2)}=f_i^{(1)}$, i=1, … , $N^{(2)}$.

Эти данные используются для расчета показателей надежности ПС. Для примера, на рис. 1 показан график зависимости вероятности безотказной работы в течение 10 час. от времени испытаний ПС 2-го проекта.

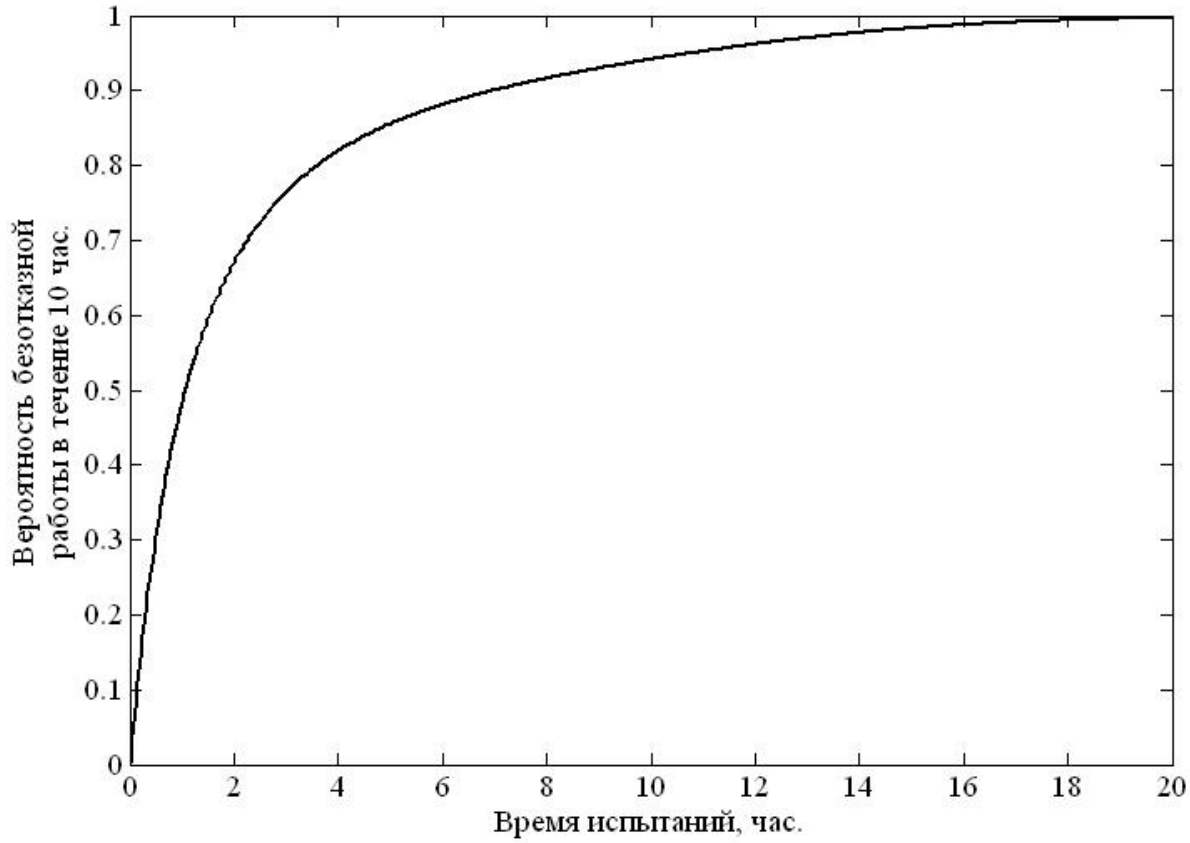

Рис. 1. Вероятность безотказной работы

Заключение

Модели надежности программных средств являются одним из средств оценки и повышения надежности ПС. Было предложено большое количество вероятностных моделей, наиболее известны из которых модели Джелинского-Моранды, Мусы, Липова, Чунга-Смагина и др. Возможности этих моделей могут быть расширены предложенной моделью надежности на основе нестационарной системы обслуживания.

Модели надежности ПС могут использоваться для расчета показателей надежности (вероятность безотказной работы, вероятность исправления всех ошибок, средняя наработка на отказ), планирования испытаний ПС, выбора стратегии отладки и сроков завершения проекта.

Важным вопросом практического применения моделей является подготовка исходных данных. Эта задача особенно актуальна при моделировании на ранних этапах жизненного цикла. Исходные данные (общее количество ошибок и временные характеристики процессов их поиска и устранения) могут быть получены с использованием объектно-ориентированных метрик сложности и статистики, собранной при разработке похожих программных проектов.

Библиографический список

1. **Handbook of Software Reliability Engineering**. / Lyu M.R. – McGraw Hill, 1996. – 819 p. – ISBN 0-0703-9400-8.
2. **Надежность аппаратно-программных комплексов. Учебное пособие**. / Черкесов Г.Н. – СПб.: Питер, 2005. – 479 с. – ISBN 5-469-00102-4.
3. **Прямая и обратная задачи надежности сложных программных комплексов**. / Кузнецов В.В., Смагин В.А. // Надежность и контроль качества. – 1997. – № 10. – с. 56–62.
4. **An Architecture-Based Software Reliability Model**. / Wang W.-L., Wu Y., Chen M.-H. // Proceedings Of Pacific Rim International Symposium on Dependable Computing. – 1999. – p. 143–150.
5. **Метод планирования тестирования сложных программных комплексов на этапах проектирования и разработки**. / Тырва А.В., Хомоненко А.Д. // Научно-технические ведомости СПбГПУ. – 2009. – № 4 (82) . – с. 125–131.
6. **Разработка динамических моделей нестационарных систем обслуживания**. / Бубнов В.П., Сафонов В.И. – СПб.: Издательство “Лань”, 1999. – 64 с. – ISBN 5-8114-0194-9.
7. **Software Reliability Model Selection: A Case Study /** Khoshgoftaar T.M., Woodcock T.G. // Proceedings Of International Symposium on Software Reliability Engineering. – 1991. – p. 183–191.
8. **A validation of object-oriented design metrics as quality indicators**. / Basili V.R., Briand L.C., Melo W.L. // IEEE Transactions on Software Engineering. – 1996. – Volume 22, Issue 10. – p. 751–761.
9. **A metrics suite for object oriented design**. / Chidamber S.R., Kemerer C.F. // IEEE Transactions on Software Engineering. – 1994. – Volume 20, Issue 6. – p. 476–493.
10. **The prediction of faulty classes using object-oriented design metrics**. / El-Emam K., Melo W., Machado J.C. // Journal of Systems and Software. – 2001. – Volume 56, Number 1. – p. 63–75.