УДК 338.24.01 DOI 10.21685/2307-4205-2018-4-2

Н. А. Северцев, А. В. Бецков, А. Н. Дарьина

АДАПТИВНАЯ МОДЕЛЬ ОЦЕНКИ БЕЗОПАСНОСТИ И НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

N. A. Severtsev, A. V. Beckov, A. N. Daryina

ADAPTIVE MODEL OF SAFETY AND RELIABILITY ESTIMATION OF SOFTWARE

Аннотация. При разработке программного обеспечения для безопасности и надежности динамической системы при ее функционировании необходимо учитывать специфику работы системы, ее управляемость, а также адаптивность. Рассмотрен новый метод отработки программного обеспечения (ПО) безопасности и надежности динамической системы при ее функционировании. В процессе разработки ПО предложен унифицированный критерий адаптации модели оценки надежности работы анализируемой системы в виде текущей навязки. Процесс отработки программного обеспечения считается случайным, что позволяет применять к его исследованиям методы статистической теории подобия и вероятностные методы. Задача оптимизации процесса отработки ПО представляется двухступенчатой иерархической задачей, состоящей из задач глобальной и локальной адаптации. Методический подход основан на том, что все положительное основывается на базе прогноза на один шаг (тест) вперед. Показано, что в данной задаче этот подход оказывается оптимальным. В статье предлагаются алгоритмы адаптации. Для наглядности изложения представлены блок-схемы разработанных алгоритмов.

Ключевые слова: программное обеспечение, надежность, безопасность, модель, оптимизация, стратегия, управление, адаптация, навязка, параметр, тест, отработка.

Abstract. The article describes a new method of software development of security and reliability of the dynamic system in its function. In the process of software development a unified criterion of adaptation of the model of reliability evaluation of the analyzed system in the form of the current imposition is proposed. When you develop software for security and reliability of a dynamical system when it is running, you must take into account the specifics of the system, its handling, as well as adaptability. This article describes a new method of refining the software safety and reliability of dynamic system when it is running. In the process of elaborating on proposed uniform criterion of reliability assessment model adaptation of the analyzed system in the form of the current navjazki. The process of refining the software is considered to be accidental, that allows you to apply his research methods, statistical theory of similarity and probabilistic methods. The task of optimizing the process of refining on it seems a two-tier hierarchical task consisting of global and local adaptation. A methodical approach is based on the fact that all positive based prediction based on one step forward (test). It is shown that in this case this approach is optimal.

Key words: software, reliability, security, model, optimization, strategy, control, adaptation, imposition, parameter, test, testing.

Рассмотрим адаптивную модель оценки безопасности и надежности программного обеспечения (ПО), общие требования к которой состоят в том, чтобы она была адаптивной к реальному процессу разработки ПО и позволяла учитывать следующие требования:

– специфику ПО – иначе, полноту и стратегию тестирования в штатных и нештатных ситуациях при верификации алгоритмов, оценку корректности и правильности перевода алгоритмов на машинный язык при программировании, дисциплину взаимодействия программных модулей и т.д.;

– управляемость процессом отработки, обработки и испытаний ПО в зависимости от текущих результатов отработки, числа выявленных ошибок, интенсивности их проявления, временных стоимостных затрат на отработку надежности;

– адаптацию к реальному процессу отработки по мере накопления статистических данных о выявлении ошибок и адекватному описанию реального процесса роста надежности ПО в ходе испытаний;

– оценку реального времени функционирования модели, т.е. определение оптимального момента остановки процесса отработки ПО;

– использование в своем составе ряда частных моделей с целью последующего обобщения результатов их работы для того, чтобы принять решение о величине надежности ПО.

Для того чтобы управлять процессом отработки программного обеспечения по ходу испытаний в зависимости от результатов, целесообразно считать его как управляемый случайный процесс. Это будет правильным, так как в процессе контроля величины надежности ПО ошибки, внесенные на этапе разработки и приводящие в процессе функционирования к невыполнению сложной технической системой (особенно такой, как вооружение и военная техника) целевого предназначения, проявляются в случайные моменты времени. Случайный характер проявления результатов функционирования ПО будет обусловлен следующими причинами:

– случайностью внесения ошибок в ПО в процессе его разработки;

– случайностью изменения окружающей среды в виде входных данных, поступающих на вход ПО;

– случайностью использования маршрута обработки данных в программе;

– недостаточностью информации о возможных состояниях отрабатываемого программного обеспечения.

Так как затраты времени на отработку ПО, т.е. поиск, выявление, устранение ошибок, растут достаточно быстро, то потери на отработку ПО должны суммироваться по ходу тестов. Тогда задача управления процессом отработки ПО будет сводиться к управлению процессом накопления этих потерь в зависимости от текущих результатов отработки ПО.

Требуется определить критерий, который носил бы обобщенный смысл оценки качества ПО, так как оценки безопасности и надежности получены с помощью разных моделей, которые могут существенно расходиться в силу особенностей структуры и ограничений каждой модели.

Данный критерий должен отражать прогнозирующие возможные модели определения безопасности и надежности, как ее способность предсказывать поведение контролируемого параметра; позволять сравнивать модели между собой по отношению к любому комплексу ПО независимо от числа и содержания параметров, входящих в модель; иметь простую, легко понятную интерпретацию. Таким унифицированным (общим) критерием адаптации модели оценки надежности может быть текущая *невязка-обновление* [1]. Под термином «текущая невязка» понимается отклонение прогнозируемого значения наблюдаемого параметра от его фактического значения, полученного по результатам тестирования ПО, в виде

$$v_t = \tilde{\xi} - \xi_t,$$

где v_t – текущая невязка прогноза; $\tilde{\xi}$ – прогнозное значение наблюдаемого параметра. Так как величина текущей невязки представляет собой функцию от случайной величины, то процесс ее изменения следует рассматривать как случайный процесс. Для каждой модели оценки надежности невязка может быть вычислена в любой момент времени процесса отработки ПО. В результате получается обновляющая стохастическая последовательность $\{v_t\}$. Если данная модель оценки надежности и безопасности хорошо разработана к процессу ПО, то ее обновляющая последовательность будет последовательностью независимых случайных величин со стандартным нормальным распределением [2].

Если же модель плохо отражает ход процесса создания ПО, то параметры распределения стохастической обновляющей последовательности изменяются так, что не будет обеспечиваться сходимость по вероятности параметров распределения, полученной по результатам статистики о ходе отработки ПО к указанным значениям. В простейшем случае адекватность модели надежности и безопасности реальному процессу создания поможет определяться по минимуму значения невязки.

Практика использования статистических методов контроля сложных технических систем, позволяющих организовать стохастическое оптимальное управление, приводит к выводу, что эффективность методов контроля прямо пропорциональна значению функций распределения вероятностей тех случайных величин, которые и делают процесс контроля случайным. В данном случае это функции распределения результатов тестирования ПО $F^i, i = 1, ..., m.$ Опыт разработки различных комплексов ПО для значительного числа используемых моделей оценки безопасности и надежности

позволяет составить определенное суждение о семействе тех функций распределения, которыми описываются показатели, характеризующие безопасность и надежность ПО. Тем не менее конкретные значения параметров этих распределений для конкретного ПО априори являются неизвестными, тогда задача оценки неизвестного распределения является задачей оценки параметров этого распределения.

Обозначим значение неизвестного наблюдаемого параметра функции распределения F^i как θ. До тех пор пока не получена ошибка для неизвестного параметра θ данного распределения, нельзя статистически оптимизировать процесс отработки ПО. С одной стороны, предпочтительно получить такую ошибку как можно быстрее, а с другой – с минимальным числом остаточных ошибок в ПО с максимальной точностью.

Однако преждевременное ускорение процесса отработки не даст существенного выигрыша, поскольку потери от проявления оставшегося числа ошибок в процессе эксплуатации ПО и низкой достоверности полученных оценок могут оказаться большими. Вместе с тем увеличение процесса отработки может привести также к большим потерям на верификации ПО. Данные противоречивые требования и создают основу для введения такого критерия оптимизации процесса отработки программного обеспечения, который бы учитывал как нарастание потерь на отработку и верификацию ПО, так и уменьшение потерь, связанных с числом ошибок, оставшихся в ПО. В связи с этим целесообразно применять критерии оптимизации стратегии отработки ПО в виде

$$X\left(\delta^*\right) = \min\left\{E\left(\sum_{t=1}^{\tau} C\left(v_t, u_t, t\right)\right) + g_\tau\left(v_t, t\right)\right\},$$

где E – математическое ожидание; $X\left(\delta^*\right)$ – потери на отработку ПО; C – функция траекторных потерь; g – функция терминальных потерь; τ – момент остановки процесса отработки ПО; t – время функционирования программного обеспечения, связанное с дискретными шагами процесса отработки ПО; v_t – невязка показателя надежности ПО; u_t – управление в виде конкретного теста для проверки ПО в том или ином режиме; δ – стратегия управления процессом отработки ПО:

$$\delta = (l, \mu, f, d),$$

где l – стратегия управления локальной адаптацией параметров, $l = \left(l(i)\right)$; $i = 1, \ldots, n$ – общее число используемых моделей; μ – стратегия управления выбором, т.е. продолжить отработку ПО или закончить (представить) по требованию заказчику; f – стратегия управления выбором следующего теста ПО, если решено продолжить отработку; d – стратегия управления выбором оценки безопасности и надежности ПО; δ^* – оптимальная стратегия управления процессом отработки ПО; δ – множество стратегий управления процессом создания ПО.

Будем понимать под траекторными потерями сумму всех математических и вероятностных затрат, связанных с верификацией программного обеспечения. Сюда могут входить затраты на тестирование, накопление и обработку статистической информации, стоимость используемой аппаратуры, оплата труда разработчиков, время разработки программного обеспечения и др.

Терминальные потери учитывают возможность остаточных ошибок в ПО после его сдачи потребителю и определяются достоверностью получаемых оценок параметров функции распределения измеряемых величин. Очевидно, что чем точнее определены параметры оцениваемой функции распределения, тем меньше будут потери за счет точной оценки оставшегося числа ошибок в ПО. В целом же, как траекторные, так и терминальные потери в зависимости от решаемых задач могут выражаться либо стоимостными, либо временными характеристиками.

Временные характеристики влияют на сроки отработки ПО, а стоимостные – на финансово-экономические затраты. Слагаемые критерии должны быть взяты с определенными весовыми коэффициентами, которые определяют не только степень влияния каждого слагаемого на выбранный критерий, но и приводят их к безразмерной величине или к единой размерности. Предоставим характеристику предложенного критерия оптимизации отработки ПО.

Отметим то, что он является статистическим, поскольку предполагает достижение минимума или инфинума, в общем случае математическое ожидание некоего функционала, определенного на траекториях процесса создания ПО; цена отработки имеет технико-экономический смысл для всех комплексных программ; данный критерий является гибким, а гибкость его обусловлена тем, что конкретное значение цены отработки зависит от технологии самого процесса отработки, цена же очень гибко перестраивается на основе изменения различных стратегий управления. Кроме того, критерий обладает свойством большой общности поскольку обобщает множество известных частных критериев. Тем не менее, несмотря на общность, критерий можно формализовать и определить его значения. Оптимизация процесса отработки ПО заключается в принятии таких решений (в соответствии со стратегией δ), которые обеспечивают минимальное значение потерь X. Исходной информацией для решения задачи оптимизации является последовательность наблюдаемых измерений результатов испытаний программного обеспечения. Наиболее верный метод в оценке параметров функций распределения этих измерений – это метод Байеса, обладающий рядом преимуществ:

– он может применяться к любым вариантам и частным случаям задачи оценки безопасности и надежности;

– при нем не возникает сложного вопроса о выборе необходимых оценок неизвестного параметра и доверительных интервалов, как в классической выборочной теории оценок;

– он позволяет достаточно просто использовать системный анализ результатов отработки ПО;

– в рамках байесовского подхода хорошо описываются обновляющие процессы, что позволяет по поведению обобщенного параметра адаптации модели оценки безопасности и надежности проверить степень адекватности выбранной вероятностной модели к реальному процессу создания ПО;

– байесовский метод снижает объем стохастической достоверности получаемых оценок.

Безусловно, можно построить процесс отработки ПО так, что оценка известного параметра будет осуществляться после проведения большого фиксированного числа тестов. Очевидно, достоверность такой оценки будет прямо пропорциональна объему выборки, а накопленные потери будут большими. Методы оценки, основанные на использовании обучающей выборки фиксированного объема, не могут быть эффективными в используемой задаче из-за явной неэкономичности. Если рассматривать адаптивную модель оценки надежности как частный случай системы управления сложным процессом и считать при этом, что управление осуществляется в соответствии с некоторой стратегией, то данная неформальная постановка задачи приводит к блок-схеме адаптивной модели оценки надежности ПО (рис. 1).

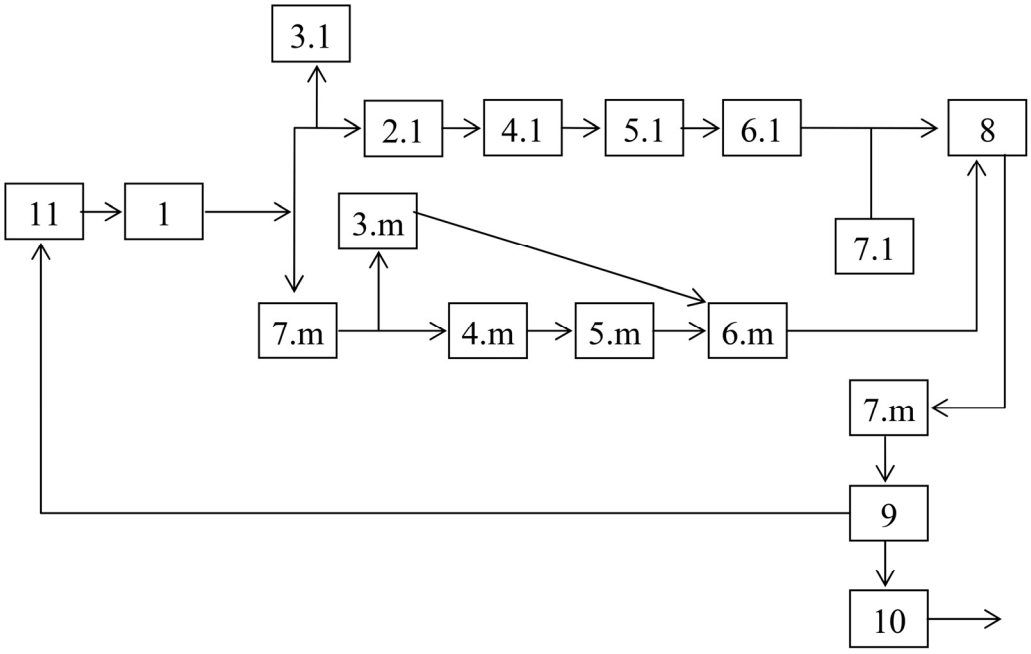

Рис. 1. Блок-схема адаптивной модели оценки надежности программного обеспечения

На рис. 1 приняты следующие обозначения: 1 – процесс отработки ПО; (2.1 – 2m) – блок формирования измерений, отражающих ход отработки различных моделей оценки надежности.

Для временных моделей это связано с регистрацией интервалов между ошибками; для считывающих – с количеством ошибок в интервале; (3.1 – 3m) – блоки задержек на один шаг; (4.1 – 4m) – блоки различных моделей оценки надежности. На их выводе определяется функция надежности ПО в соответствии с конкретной моделью оценки надежности и с уточненными значениями параметров этих функций; (5.1 – 5m) – блоки прогноза показателя надежности ПО (среднего времени до следующей ошибки или среднего числа ошибок в предстоящем временном интервале); (6.1 – 6m) – блоки определения невязок показателя надежности; (7.1 – 7m) – блоки управления локальной адаптацией отдельных моделей; они позволяют настраивать параметры моделей адекватного описания текущего изменения надежности отрабатываемого комплекса ПО; 8 – блок выбора метода определения надежности; он осуществляет на каждом шаге процесса отработки, связанного с проведением очередного теста, выбор наиболее адекватной модели в соответствии с унифицированным параметром адаптации; 9 – блок выбора решения о продолжении процесса отработки или его окончании и передачи ПО потребителю; 10 – блок окончания оценки надежности (функции надежности, критерия надежности); она работает в некоторый момент τ, если в блоке 9 принято решение об остановке процесса; 11 – блок выбора очередного теста.

Фактически задача оптимизации процесса отработки ПО представляется двухступенчатой иерархической задачей, состоящей из задач глобальной и локальной адаптации. Верхний уровень – это глобальная адаптация, включающая в себя задачи выбора очередного теста оптимальной остановки процесса отработки ПО и его окончательной оценки.

Нижний уровень соответствует задаче локальной адаптации каждой из моделей оценки надежности к реальному процессу отработки программного обеспечения. В работе [3] из всех составляющих стратегии управления процессом отработки ПО задача окончательной оценки надежности по стратегии достаточно освещена. Задачу контроля надежности ПО к последовательной параметрической оценке функции распределения наблюдаемых измерений следует формализовать как задачу с дискретным временем, связывая каждый временной шаг процесса отработки с проведением очередного теста.

Рассмотрим задачу оценки надежности программного обеспечения в формализованной постановке. Пусть параметр θ_t, $t = 1, 2, \ldots$ характеризует надежность ПО, т.е. в процессе отработки ПО надежность контролируется с помощью одной из множеств моделей оценки надежности (МОН). Структурная схема представлена на рис. 2.

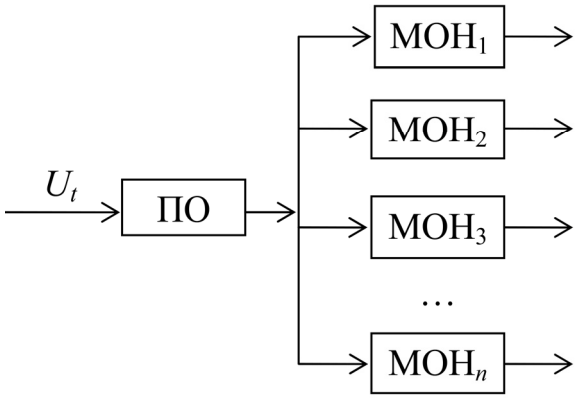

Рис. 2. Структурная схема контроля надежности программного обеспечения

Обозначим через $\hat{\theta}_t$ – оценку параметра, характеризующую состояние отработки ПО для достигнутой надежности; ξ_t – наблюдаемые изменения результатов тестов по выявлению ошибок ПО; U_t – управление в виде конкретного теста для проверки ПО в определенном режиме.

Выбор конкретного содержания параметров θ_t и ξ_t обусловлен выбором критерия оптимизации процесса оценки надежности. Физический выбор θ_t обусловливается оцениваемым показателем надежности ПО. Рассмотрим один основной показатель – вероятность ошибки программного обеспечения за заданное время функционирования сложной технической системы. Как правило, в модель оценки надежности входит функция распределения времени до появления следующей ошибки.

При этом θ_t и будет параметром этой функции распределения – физически он зависит от числа ошибок, оставшихся в ПО на момент *t*. Предположим, что $\theta_t \in (\theta, \Gamma)$ при всех *t*, где (θ, Γ) – измеримое пространство с σ-алгеброй Г. Значения параметра θ_t являются независимыми и наблюдаемыми. В процессе отработки ПО можно наблюдать лишь $\xi_t, t = \overline{1, \tau}$.

Физика этих значений зависит от конкретной модели оценки надежности. Существующие модели состоят условно из двух групп: временных и счетных. Для временной модели ξ_t будет означать время, прошедшее от момента обнаружения $(t-1)$-й ошибки до момента обнаружения *t*-й ошибки. Для счетной модели ξ_t будет означать число ошибок, обнаруженных за *t*-й мерный интервал времени. В обоих случаях эти значения должны отвечать соответствующей функции распределения, принятой в данной модели.

Сама функция распределения имеет параметр θ, так как в вероятностном смысле между временной и счетной моделями разницы нет, то за основу можно принимать временную модель оценки надежности и дальнейший анализ следует привязывать к ней [4].

Примем, что $\xi_t \in (\Xi, S)$, $u_t \in (V, U)$, $t = \overline{1, \tau}$, где (V, U) – измеряемое пространство тестов. Возможны ограничения вида:

$$u_t \subset (V_t, U_t), V_t \subset V, \qquad U_t \subset U.$$

Поскольку последовательность θ_t наблюдаемая, вся информация, на которой можно строить выбор следующего теста или решать задачу об остановке процесса отработки, заключена в последовательности $\xi_1^t = \{\xi_1, \xi_2, ..., \xi_t\} \in H_t$.

Вводя модель оценки надежности, введем функцию распределения $F\{\xi_{t+1} | h_t, u_{t+1}\}$ для значения ξ_{t+1}. Опыт отработки не является непрерывно растущей функцией и оперировать с ним достаточно громоздко, поэтому необходимо переходить к минимальной достаточной статистике [5]. Именно поэтому задача оценки надежности и безопасности ПО должна сводиться к выбору момента остановки τ и выбору оценки $\hat{\theta}_t$ для θ_t, тогда минимальной достаточной статистикой будет служить апостериорная вероятность Π_t, на τ – алгебре Г. Таким образом, от h_t надо переходить к Π_t, $\Pi_t \in (\Pi, P), t = 1, 2, ...$ Обозначим Π_0, при $t = 0$, априорную вероятность на значениях неизвестного параметра θ_0.

Теперь вместо $F\{\xi_{t+1} | h_t, u_{t+1}\}$ будем иметь $F\{\xi_{t+1} | \Pi_t, u_{t+1}\}$. Проиллюстрируем эту схему рис. 3, где пунктирная стрелка от u_t к ξ_t означает, что ξ_t получается случайно в соответствии с функцией распределения $F\{\xi_t | \Pi_t, u_t\}$. Имея Π_t, можно получить байесовскую оценку $\hat{\theta}_t$.

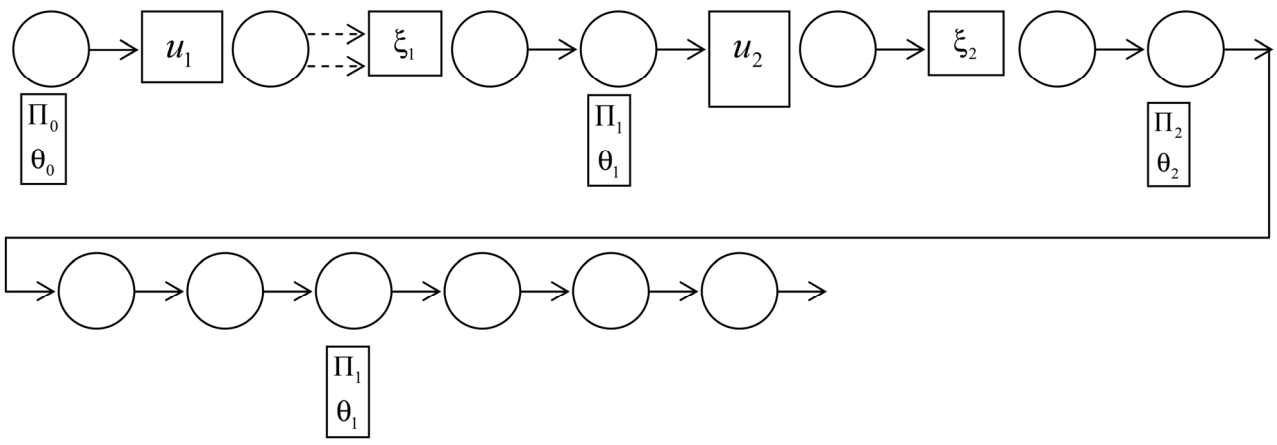

Рис. 3. Наблюдаемый опыт (история отработки программного обеспечения)

Предположим, что при обнаружении *t*-й ошибки она быстро исправляется, так как интервал восстановления не влияет на оценку надежности ПО в процессе отработки, что дает возможность считать, что ξ_{t+1} в вероятностном смысле не зависит от ξ_t и от ξ_1^t вообще [6]. Иначе, функция распределения ξ_{t+1} объективно зависит лишь от реального θ_t и выбранного теста u_{t+1}. Однако вся наблюдаемая информация о θ_t сосредоточена в Π_t, поэтому в итоге имеем $F\{\xi_t|\Pi_t, u_{t+1}\}$. Соображения о независимости ξ_t дают основание сделать вывод о марковском изменении θ_t и, следовательно, о марковском изменении Π_t, т.е. $\Pi_{t+1} = Y(\Pi_t, U_t, \xi_{t+1})$, где *Y* соответствует формуле Байеса. Таким образом, статистика Π_t является транзитивной (равенству, эквивалентности, сравнимости по некоторому модулю). Нет необходимости помнить всю историю (опыт) h_t для перехода к Π_{t+1}, а достаточно лишь знать аргументы Π_t, U_t, ξ_{t+1}. В этом случае необходимо останавливать процесс Π_t. Критерий оптимизации процесса теперь можно определить на процессе Π_t в виде

$$X_t = \sum_{i=1}^{t} C(\Pi_i, U_i) + g(\Pi_i).$$

Математическое ожидание берется по вероятности P вероятностного пространства (Ω, F, P), на котором задана случайная последовательность $(\theta_0, \xi_1, \theta_1, \xi_1, ...)$. Это вероятностная мера *P* будет зависеть от стратегии отработки программного обеспечения [7]:

$$\delta = Y(f, \mu, l, d),$$

где *f* – стратегия выбора следующего теста; μ – стратегия остановки процесса отработки; l – стратегия адаптации параметров модели; d – стратегия принятия окончательной оценки надежности и безопасности ПО (параметра θ_τ). Так как стратегия δ определяет в среднем и значение X_τ, тогда можно представить выражение:

$$X_\tau^{\delta^*} = \min\left(X_\tau^{\delta}, \delta\right) \in \Delta.$$

Задача сводится к поиску оптимальной стратегии δ^* отработки ПО в классе Δ. Поскольку ξ_t независимы, а θ_τ и Π_t подчинены марковскому процессу, то Δ можно задать как множество марковских стационарных стратегий. Из этого следует следующее равенство [8]:

$$U_t = f(\Pi_{t-1})\big|_{\tau=t} = \mu\left(\Pi_{t-1}, X_{t-1}, \hat{\theta}_\tau\right) = d(\Pi_t).$$

Так как вся схема является байесовской, то возможно упростить задачу, положив стратегию d в виде байесовского выбора $\hat{\theta}_\tau$. Схема оптимизации процесса обработки дана в рамках применения одной типовой модели в виде функции распределения:

$$F^{(i)}\{\xi_{t+1}|\Pi_t, u_{t+1}\}.$$

При изменении i меняются:

а) вид функции $F^{(i)}$;

б) число и содержание параметров функции $F^{(i)}$;

в) соответствующая ей наблюдаемая величина ξ_t.

Поэтому более точно можно представить

$$F^{(i)}\left\{\xi_t^{(i)}\middle|\Pi_{t-1}^{(i)}, u_t\right\},$$

где $\Pi_{t-1}^{(i)}$ – апостериорная вероятность на значениях параметра $\theta_{t-1}^{(i)}$ для $F^{(i)}$.

Таким образом, следует ввести обобщение задачи на векторный случай использования сразу m моделей. Тогда для каждой *i*-й модели будем иметь свой опыт и свою историю в виде массива данных $t = 1, 2, \ldots$, $i = \overline{1, m}$. Теперь критерий оптимизации приобрел векторный характер, что не является благоприятным фактором. К тому же он субъективно связан со степенью адаптации к реальности каждой отдельной модели. В то же время в данном случае этот недостаток только кажущийся. Если $X_t^{(i)}$ показывает степень близости модели к объекту (системе), то его можно использовать и для оптимизации процесса, а именно:

– адаптации параметров модели оценки надежности и безопасности;
– выбора следующего теста;
– определения момента остановки;
– окончательной оценки надежности и безопасности.

Зная значения теста u_{t+1}, можно вычислить математическое ожидание $\xi^{(i)}\left\{\Pi_t^{(i)}, u_{t+1}\right\}$, а имея эту оценку, можно получить для $\hat{\Pi}_t^{(i)}$, что в свою очередь дает возможность, зная $X_t^{(i)}$, оценить $\hat{X}_t^{(i)}$. Представим следующую разность, как

$$V_t^i = X_{t+1}^i - \hat{X}_{t+1}^i,$$

тогда процесс V_t^i является текущей невязкой прогноза X_{t+1}^i на один шаг вперед. Невязка показывает точность прогноза накопленных потерь по *i*-й модели. Очевидно, что прогноз $\hat{X}_{t+1}^i$ зависит от выбора теста u_{t+1}. Перед выбором каждого шага появляются такие вопросы:

– продолжать тестирование или остановить процесс обработки *j*-е.

Если принято решение продолжать тестирование, то:

– какой тест выбрать и какую модель оценки надежности и безопасности использовать для выбора теста.

Рассмотрим второй вопрос. Представим, что совершен *i*-й тест, после чего следует *i*+1. Допустим, что на $(t + 1)$-м шаге выбрали тест u_{t+1}. После совершения теста u_{t+1} вычисляется невязка прогноза $V_{t+1}^i = X_{t+1}^i(u_{t+1})$ и проводится уточнение (адаптация) параметров каждой *i*-й модели оценки надежности и безопасности по результатам полученной статистики ε_1^{l+1} тестирования ξ в соответствии со стратегией l. Она будет характеризовать точность и степень адекватности $V_t^i = X_{t+1}^i$-й модели к реальности. В силу этого на шаге $(t - Z)$ целесообразно использовать эту модель оценки надежности и безопасности функционирования исследуемой системы (объекта), для которого разрабатывается ПО и для которой достигается минимум:

$$V_{t+1} = \min_t \left| V_{t+1}^i \right|.$$

Эту модель обозначим как i_{t+1}.

В итоге, при выборе теста u_{t+2} используется прогноз на модели i_{t+1}. Получаем $X_{t+2}^{i_{t+1}}(u_{t+2})$. Теперь представим минимум по u_{t+2}, т.е. $X_{t+2}^{i_{t+1}}(u_{t+2}) = \min\{U_{t+2}, u_{t+2}\} \in U_{t+2}$. Выбранный тест $u_{t+2}^{i_{t+1}}$ и рекомендуется для следующего проведения тестирования. После получения его результата вычисляется $V_{t+2} = \min_t \left| V_{t+2}^i \right|$ и проводится адаптация параметров каждой *i*-й модели по результатам статистики ξ_1^{t+2} и устанавливается модель i_1^{t+2}.

По ней выбирается $\hat{X}_{t+3}^{i_{t+2}}$ и $\hat{u}_{t+3}^{i_{t+2}}$ и т.д. Таким образом, на каждом шаге проводится адаптация каждой модели оценки надежности и безопасности, а лучшую модель выбирают, ориентируясь на минимум невязки V_i, и уже по этой модели выбирается лучший тест, ориентируясь на $\hat{X}_{t+1}^{i_1}$ и $\hat{u}_{t+1}^{i_1}$.

Теперь необходимо решить вопрос об остановке процесса отработки ПО. Введем в действие $\overline{u}$, эквивалентное остановке процесса отработки и принятию решения об оценке надежности и без-

опасности ПО с последующей сдачей его заказчику. На каждом шаге проводится выбор $\min\left\{\hat{X}_{t+1}^{i_1}, \hat{X}_{t+1}^{i_1}(\overline{u})\right\}$. Если $\hat{X}_{t+1}^{i_1}(\overline{u})$ окажется меньше, то целесообразно процесс отработки остановить.

Изложенный методический подход основан на том, что все положительное основывается на базе прогноза на один шаг (тест) вперед. В данной задаче этот подход оказывается оптимальным, а подробности этого следующие.

Все оценки $\hat{X}_{t+1}^{i_1}$ основаны на Π_t^i и $\hat{\Pi}_t^i$, где $\hat{\Pi}_t^i$ – апостериорная вероятность на наблюдаемых значениях параметра θ_t^i. Этот параметр характеризует интенсивность потока ошибок, которые выявляются при отработке ПО. Формально он выступает в роли параметра функции распределения для результатов следующего теста. Но фактически он связан с числом оставшихся на момент t ошибок ПО. По мере уменьшения этого числа θ_t^i будет изменяться так, что функция распределения F_i будет все больше деформироваться и сдвигаться на носитель положительных результатов теста.

В связи с этим Π_t^i будет асимптотически сходиться, все более формируясь вокруг соответствующего θ^i, при котором F_i сдвинута на положительные результаты теста. Если это одна точка, то F_i выводится в ступеньку в этой точке. Таким образом, сходимость гарантирована. Поэтому процесс X^i_{t+1} будет супермаргиналом, т.е.

$$X\left(\hat{X}^{i_t}_{t+1}\middle|X^{i_{t-1}}_t\right) \le X^{i_{t-1}}.$$

Следовательно, при нарушении супермаргинальности необходимо сразу остановить процесс, ибо при монотонной сходимости и возврат супермаргинальности невозможен, таким образом, оптимальная стратегия δ^*, отнесенная ко всему набору *m* моделей, для решения задачи об остановке процесса отработки ПО или выборе следующего теста будет основываться на вышеизложенном методе (правиле) прогноза «на один шаг вперед».

Библиографический список

1. *Иванов, А. И.* Вопросы теории испытаний структурно-сложных систем / А. И. Иванов, А. С. Иванющенко, А. М. Московский. – М. : ТЕИС, 2006.
2. *Северцев, Н. А.* Введение в теорию безопасности / Н. А. Северцев, А. В. Бецков. – М. : ВЦ им. А. А. Дородницына РАН, 2008. – 176 с.
3. *Северцев, Н. А.* Системный анализ теории безопасности / Н. А. Северцев, А. В. Бецков. – М. : Изд-во МГУ, 2009. – 452 с.
4. *Северцев, Н. А.* Статистическая теория подобия в задачах безопасности динамических систем / Н. А. Северцев. – М. : Радиотехника, 2010.
5. *Северцев, Н. А.* Моделирование безопасности функционирования динамических систем / Н. А. Северцев, А. В. Бецков. – М. : ТЕИС, 2015. – 328 с.
6. *Дедков, В. К.* Функциональная надежность малосерийных технических устройств / В. К. Дедков, Н. А. Северцев, С. Куюнджич // Труды Международного симпозиума Надежность и качество. – 2000. – С. 23–24.
7. *Дедков, В. К.* Метод оптимального управления техническим состоянием бортовых систем космического аппарата на основе решения обратных задач / В. К. Дедков, В. Д. Бабишин, М. А. Дорошенко // Труды Международного симпозиума Надежность и качество. – 2011. – Т. 1. – С. 292–294.
8. *Дивеев, А. И.* Эффективный алгоритм для синтеза структуры системы автоматического управления / А. И. Дивеев, Н. А. Северцев, Е. А. Софонова // Труды Международного симпозиума Надежность и качество. – 2007. – Т. 1. – С. 20–23.

References

1. Ivanov A. I., Ivanyushchenko A. S., Moskovskiy A. M. *Voprosy teorii ispytaniy strukturno-slozhnykh sistem* [Questions of the theory of testing of structurally complex systems]. Moscow: TEIS, 2006.
2. Severtsev N. A., Betskov A. V. *Vvedenie v teoriyu bezopasnosti* [Introduction to the theory of security]. Moscow: VTs im. A. A. Dorod-nitsyna RAN, 2008, 176 p.
3. Severtsev N. A., Betskov A. V. *Sistemnyy analiz teorii bezopasnosti* [A systematic analysis of the theory of security]. Moscow: Izd-vo MGU, 2009, 452 p.

4. Severtsev N. A. *Statisticheskaya teoriya podobiya v zadachakh bezopasnosti dinamicheskikh sistem* [Statistical similarity theory in dynamic system security problems]. Moscow: Radiotekhnika, 2010.
5. Severtsev N. A., Betskov A. V. *Modelirovanie bezopasnosti funktsionirovaniya dinamicheskikh sistem* [Modeling of safety of functioning of dynamic systems]. Moscow: TEIS, 2015, 328 p.
6. Dedkov V. K., Severtsev N. A., Kuyundzhich S. *Trudy Mezhdunarodnogo simpoziuma Nadezhnost' i kachestvo* [Proceedings of the International Symposium Reliability and Quality]. 2000, pp. 23–24.
7. Dedkov V. K., Babishin V. D., Doroshenko M. A. *Trudy Mezhdunarodnogo simpoziuma Nadezhnost' i kachestvo* [Proceedings of the International Symposium Reliability and Quality]. 2011, vol. 1, pp. 292–294.
8. Diveev A. I., Severtsev N. A., Sofonova E. A. *Trudy Mezhdunarodnogo simpoziuma Nadezhnost' i kachestvo* [Proceedings of the International Symposium Reliability and Quality]. 2007, vol. 1, pp. 20–23.

Северцев Николай Алексеевич
доктор технических наук, профессор,
главный научный сотрудник,
Федеральный исследовательский центр
«Информатика и Управление»
Российской Академии Наук
(Вычислительный центр
им. А. А. Дородницына РАН)
(119333, Россия, г. Москва, ул. Вавилова, 40)
E-mail: safsyst@mail.ru

Бецков Александр Викторович
доктор технических наук, доцент,
заместитель начальника,
Академия управления МВД России
(125171, Россия, г. Москва,
ул. Зои и Александра Космодемьянских, 8)
E-mail: abckov@mail.ru

Дарьина Анна Николаевна
кандидат физико-математических наук, доцент,
ведущий научный сотрудник,
Федеральный исследовательский центр
«Информатика и Управление»
Российской Академии Наук
(Вычислительный центр
им. А. А. Дородницына РАН)
(119333, Россия, г. Москва, ул. Вавилова, 40)
E-mail: daryina@ccas.ru

Severtsev Nikolay Alekseevich
doctor of technical sciences, professor,
chief researcher,
Federal research center
«Computer science and control» of RAS
(Dorodnitsyn computer center
of the Russian Academy of Sciences)
(119333, 40 Vavilova street, Moscow, Russia)

Betskov Aleksandr Viktorovich
doctor of technical sciences,
associate professor, deputy chief,
Russian Academy of the interior Ministry
(125171, 8 Zoi i Aleksandra Kosmodem'yanskikh
street, Moscow, Russia)

Darina Anna Nikolaevna
candidate of physical and mathematical sciences,
associate professor, leading researcher,
Federal research center
«Computer science and control» of RAS
(Dorodnitsyn computer center
of the Russian Academy of Sciences)
(119333, 40 Vavilova street, Moscow, Russia)

УДК 338.24.01
Северцев, Н. А.
Адаптивная модель оценки безопасности и надежности программного обеспечения / Н. А. Северцев, А. В. Бецков, А. Н. Дарьина // Надежность и качество сложных систем. – 2018. – № 4 (24). – С. 19–28. – DOI 10.21685/2307-4205-2018-4-2.