УДК 004.052
DOI: 10.15827/2311-6749.19.1.1

ОБЗОР МЕТОДОВ ПРОГНОЗИРОВАНИЯ ДЕФЕКТОВ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Н.В. Юхименко, магистр, fn1-kf@mail.ru; Ю.С. Белов, к.ф.-м.н., доцент, fn1-kf@mail
Калужский филиал ФГБОУ ВО «Московский государственный технический университет им. Н.Э. Баумана (национальный исследовательский университет)», г. Калуга, 248000, Россия

В данной статье представлен обзор методов прогнозирования дефектов ПО. Дефект – это логическая ошибка в исходном коде программы, наличие которой при определенных условиях может привести к отказу этой программы. Основными причинами возникновения дефектов являются сложность реализации задачи, сжатые сроки разработки, несовершенство документации, изменение требований, недостаточная квалификация и опыт разработчиков, неправильная организация процесса разработки. Раннее обнаружение дефектов снижает затраты на разработку и повышает качество и надежность ПО. Методы прогнозирования дефектов могут дать ответ на следующие вопросы: каково количество необнаруженных дефектов в ПО и в каких программных компонентах они содержатся. Знание о компонентах, содержащих наибольшее число дефектов, позволяет распределить ресурсы тестирования так, чтобы в первую очередь наиболее тщательно проверялись компоненты с высокой вероятностью наличия дефектов. В статье методы прогнозирования дефектов ПО сгруппированы в зависимости от цели – прогнозирование количества дефектов или классифицирование дефектов. На основе приведенных методов рассматриваются следующие модели и алгоритмы: модель роста надежности, экспертное мнение, метод исторических аналогий, конструктивная модель качества, методы машинного обучения и регрессионные модели.

Ключевые слова: *дефект ПО, методы прогнозирования дефектов ПО, модель роста надежности, генетический алгоритм, методы машинного обучения, регрессионные модели.*

В настоящее время программные системы становятся все более сложными и многокомпонентными. Чем больше объем исходного кода, тем больше трудозатрат необходимо для поддержания качества программной системы на должном уровне. Своевременное выявление и исправление дефектов играют большую роль в жизненном цикле ПО. Прогнозирование того, содержит ли программный компонент дефекты проектирования, помогает улучшить качество разрабатываемых систем.

В общем случае дефектом считается любое поведение системы, которое отличается от ожидаемого. Согласно глоссарию ISTQB, дефект – это недостаток, который может привести к неспособности системы выполнять свои функции. Он проявляется в процессе выполнения программы и может вызвать отказ в ней [1]. Выделяют три вида дефектов: программные, технические, архитектурные. Программные ошибки возникают из-за несовершенства исходного кода конечного продукта. Технические дефекты сводятся к доступу тех или иных функций готового решения или его дизайна. Архитектурные ошибки – это ошибки, вызванные внешними факторами, которые заранее не были учтены при проектировании решения, вследствие чего приложение демонстрирует результат работы, отличный от ожидаемого. Основными причинами возникновения дефектов являются сложность реализации задачи, сжатые сроки разработки, несовершенство документации, изменение требований, недостаточная квалификация и опыт разработчиков, неправильная организация процесса разработки.

Классификация методов прогнозирования программных дефектов.

Метод прогнозирования представляет собой последовательность действий, которые нужно совершить для получения модели прогнозирования. Модели прогнозирования дефектов ПО используются либо для классифицирования компонентов, подверженных дефектам, либо для прогнозирования количества дефектов, ожидаемых в программном компоненте. На рисунке 1 приведены наиболее распространенные методы и модели прогнозирования дефектов ПО, сгруппированные в зависимости от цели – прогнозирование количества дефектов или классифицирование дефектов.

Прогнозирование количества дефектов. Методы прогнозирования, которые оперируют только количеством дефектов, обнаруженных во время разработки и тестирования без учета других атрибутов, связанных с внутренней структурой, дизайном или реализацией продукта, называют методами черного ящика. Методы прогнозирования, которые используют атрибуты, связанные с процессом и продуктом, например, размер, сложность, изменения, относятся к методам белого ящика. Рассмотрим наиболее распространенные модели, применяемые для прогнозирования количества дефектов ПО.

Надежность ПО – это вероятность того, что в определенной среде и в определенное время оно будет работать должным образом. Модели роста надежности ПО основаны на использовании функции надеж-

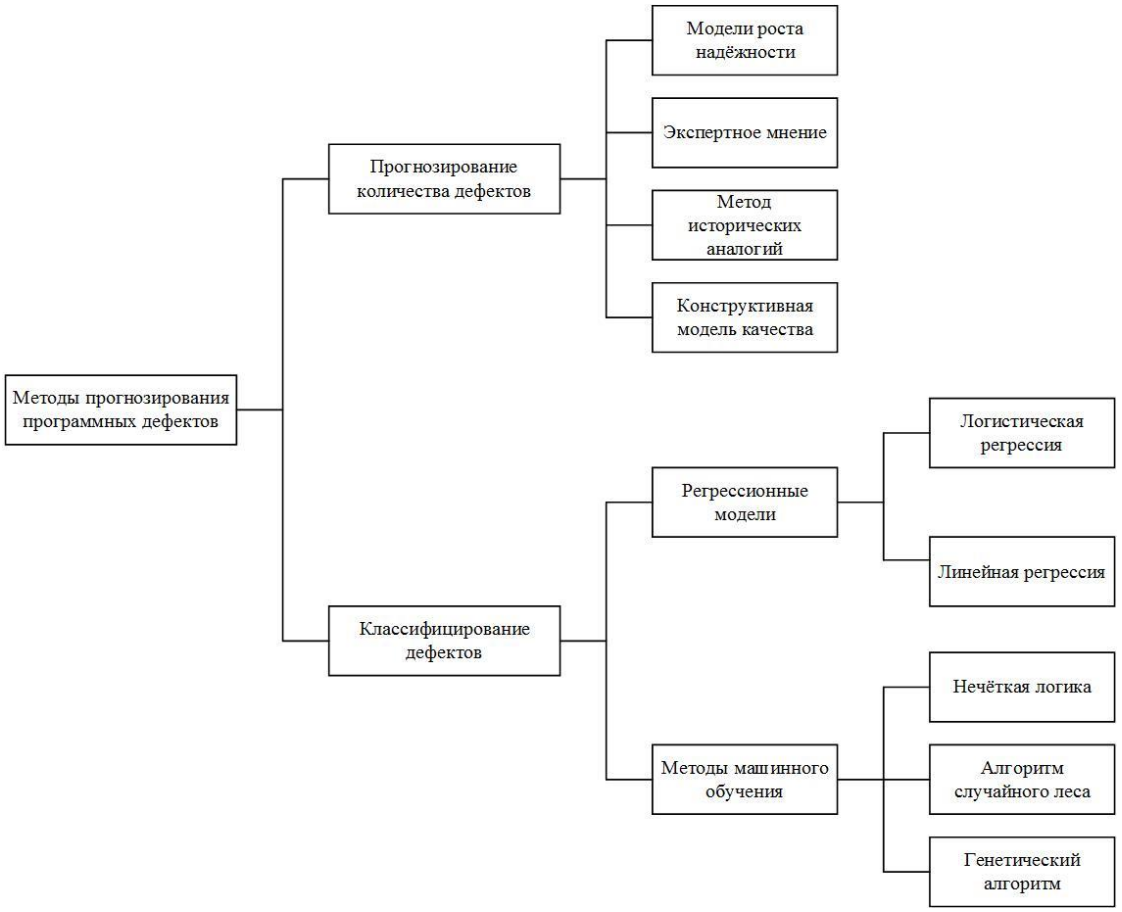

Рис. 1. Классификация методов и моделей прогнозирования программных дефектов

ности. Для получения этой функции чаще всего выбирается экспоненциальное распределение, и на основе предыдущего опыта и текущей информации об обнаруженных дефектах оцениваются его параметры [2].

Экспертное мнение. Метод прогнозирования дефектов на основе опыта экспертов является самым быстрым и простым методом. Сущность такого метода заключается в том, что в основу прогноза закладывается мнение специалиста или коллектива специалистов, основанное на профессиональном, научном и практическом опыте. Данный метод может быть полезен в тех случаях, когда прогнозирование дефектов должно выполняться на уровне всего проекта или для крупных компонентов. Недостатками данного метода являются его субъективный характер и невозможность корректного масштабирования на более низких уровнях детализации [3].

Метод исторических аналогий основан на сборе и сравнении различных метрик между прошлым и текущим проектами. На основе установления сходства метрик по некоторым признакам появляется возможность сделать вывод о сходстве в других признаках, то есть делать умозаключения по аналогии. Для прогнозирования дефектов ПО обычно используются размер, тип приложения, функциональная сложность и другие параметры. Анализ может быть выполнен на уровне проекта, подсистемы или компонента. Модели, основанные на методе исторических аналогий, особенно полезны, когда для предметной области трудно выявить конкретные правила.

Основные недостатки метода исторических аналогий:

- конечный результат сильно зависит от правильности подобранного объекта-аналога, поэтому к его выбору необходимо относиться очень внимательно;
- следует учитывать все специфические особенности объекта прогнозирования, а также действия внешних факторов.

Конструктивная модель качества использует экспертно-детерминированные подмодели внедрения и устранения дефектов для построения качественной модели. В рамках этой модели с помощью подмодели «внедрение дефектов» (defect introduction, DI) сначала оценивается количество нетривиальных требований, вводимых дефектов проектирования и кодирования. Эта подмодель использует оценки размера ПО и прочие атрибуты, связанные с проектом и процессом (платформа и т.д.). Выходные данные DI

подмодели являются входными данными для подмодели устранения дефектов (defect removal, DR). Результат подмодели DR – оценка количества оставшихся дефектов на единицу размера.

Классифицирование дефектов. Методы классифицирования позволяют определить подверженные дефектам программные модули и обычно применяются на более низких уровнях детализации, например, на уровне файлов и классов.

Регрессионные модели прогнозирования используются для решения задач, требующих изучения отношения между двумя и более переменными. В основе любой регрессионной модели лежит использование регрессии какого-либо вида. Параметры регрессии оцениваются на основе метрик для имеющегося исходного кода, а затем полученная регрессия используется для прогнозирования дефектов. Наиболее распространены логистическая и линейная регрессии.

Логистическая регрессия – это тип регрессионного анализа, используемый для прогнозирования результата категориальной переменной (то есть зависимой переменной, которая может принимать ограниченное число значений, величины которых не являются значимыми, но порядок этих величин может быть или не быть значимым) [4]. С помощью логистической регрессии можно спрогнозировать вероятность наступления некоторого события в зависимости от множества признаков:

$$P(Y) = \frac{1}{1 + e^{-(\Theta_1 x_1 + \ldots + \Theta_n x_n)}}, \quad (1)$$

где Θ_i – коэффициенты регрессии; x_i – независимые переменные.

Линейная регрессионная модель является самым простым вариантом регрессионной модели. В ее основу положено предположение, что существует дискретный внешний фактор $X(t)$, оказывающий влияние на исследуемый процесс $Z(t)$, при этом связь между процессом и внешним фактором линейна. Модель описывается уравнением

$$Z(t) = \alpha_0 + \alpha_1 X(t) + \varepsilon_t, \quad (2)$$

где α_0 и α_1 – коэффициенты регрессии; ε_t – ошибка модели.

Для получения прогнозных значений $Z(t)$ в момент времени t необходимо иметь значение $X(t)$ в тот же момент времени t, что редко выполнимо на практике. Главным недостатком линейной регрессии является требование равномерного распределения данных.

Следующей группой методов, применяемых для классифицирования дефектов ПО, являются методы машинного обучения – динамические алгоритмы обучения; их производительность, как правило, улучшается по мере поступления дополнительных данных.

На рисунке 2 показан общий процесс прогнозирования дефектов ПО на основе моделей машинного обучения.

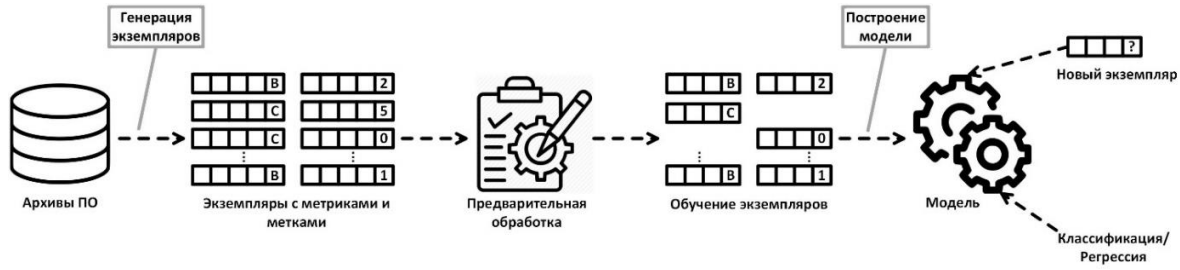

Рис. 2. Процесс прогнозирования дефектов ПО на основе методов машинного обучения

Первым шагом для построения модели прогнозирования является создание экземпляров из архивов ПО (системы контроля версий, системы отслеживания ошибок, архивы электронной почты и т.д.). Каждый экземпляр может представлять собой программный компонент, файл исходного кода, класс или функцию в зависимости от выбранной степени детализации прогнозирования. Каждый извлеченный экземпляр имеет метрики и метки, которые указывают, склонен ли данный экземпляр к дефектам (если да, то дополнительно указывается их количество). Например, на рисунке 2 экземпляры, содержащие дефекты, помечены как «B» с указанием количества ошибок, а экземпляры без ошибок – как «C». Далее производится предварительная обработка экземпляров, включающая нормализацию данных и шумоподавление. Данный этап не является обязательным. После получения окончательного набора обучающих элементов можно переходить к обучению модели прогнозирования. Модель прогнозирования может предсказать наличие ошибки в новом экземпляре. Прогнозирование того, содержит ли данный экземпляр дефекты, представляет собой классификацию, а прогнозирование количества ошибок в экземпляре – регрессию.

Нечеткая логика. С помощью модели нечетной логики можно формализовать величины, имеющие качественную основу, выявить причинно-следственные связи между регулируемыми параметрами и влияющими на них величинами и сформулировать прогноз в условиях неопределенности параметров прогнозирования [5].

Данная методология предполагает следующие шаги.

1. Определение входов и выходов создаваемой системы.
2. Определение функции принадлежности для каждой метрики.
3. Разработка нечетких правил и получение экспертного мнения для реализуемой нечеткой системы.
4. Агрегирование всех отдельных нечетких множеств для различных правил.
5. Поиск четкого значения путем дефаззификации агрегированного нечеткого множества.

Достоинством данной модели является то, что она использует как количественные, так и качественные показатели метрик для прогнозирования дефектов, недостатком – увеличение сложности вычислений в связи с увеличением входных показателей.

Алгоритм случайного леса (Random forest) основан на множестве деревьев решений и использует следующую стратегию.

– Корневой узел каждого дерева содержит начальную выборку данных. Для каждого отдельного дерева эта выборка уникальна.

– На каждом узле выборка случайным образом разбивается на два подмножества для получения двух следующих узлов с меньшими выборками.

– Каждое дерево растет до максимально возможного размера, пока на каждой ветви не будет исчерпана вся выборка.

– Когда все деревья построены, осуществляется классификация объектов путем голосования. Каждое дерево «голосует» либо за то, что компонент содержит дефекты, либо за то, что дефектов нет. В итоге компонент признается дефектным либо недефектным в зависимости от того, за какой класс решения отдано больше голосов.

Алгоритм показывает высокую точность классификации на больших наборах данных.

Генетический алгоритм – это подход к машинному обучению, подобный человеческому гену и дарвиновской теории естественного отбора. Относится к эволюционным алгоритмам, генерирующим решения на основе таких понятий, как мутация, отбор, кроссовер и т.д.

Реализация генетического алгоритма начинается с большой популяции, в которой каждая особь представляет собой возможное решение задачи. Затем особи кодируются в двоичную строку, называемую хромосомой. Группа индивидуумов будет соревноваться друг другом за возможность размножения и формирования следующего поколения. Существует функция, называемая функцией приспособленности, которая определяет, какая из конкурирующих особей получит право на воспроизведение. Данная функция гарантирует, что только лучшие особи популяции смогут переносить свое потомство в следующее поколение. Следующее поколение формируется такими видами деятельности, как размножение и мутация. Процесс размножения происходит при обмене частью кода между двумя хромосомами. Мутации представляют собой внесение изменений в следующее поколение, которое препятствует достижению локальных минимумов. Процесс повторяется до тех пор, пока не будет найден наилучший набор решений (оптимальный уровень приспособленности) [6].

Вывод. В данной статье был сделан обзор методов прогнозирования дефектов ПО, приведены их достоинства и недостатки. Несмотря на разнообразие существующих методов прогнозирования дефектов, некоторые серьезные проблемы остаются нерешенными. Эти проблемы ограничивают сферу применения существующих методов прогнозирования и делают процесс сложным и недостаточно автоматизированным.

Литература

1. International Software Testing Qualifications Board, Certified Tester Foundation Level Syllabus. URL: https://www.istqb.org/downloads/send/51-ctfl2018/208-ctfl-2018-syllabus.html (дата обращения: 24.12.2018).

2. Кирносенко С.И., Лукьянов В.С. Прогнозирование обнаружения дефектов в программном обеспечении // Программные продукты и системы. 2011. № 3. С. 68.

3. Sabnis P., Kadam A. Software Reliability Growth Model with Bug Cycle and Duplicate Detection Techniques. Bharati Vidyapeeth Deemed Univ. College of Eng., Pune, India, 2013, pp. 345–349.

4. Dulal C.S. Software Defect Prediction Based on Classification Rule Mining. Department of comp. sci. and Eng. National Institute of Technology Rourkela, 2013, pp. 19–65.

5. Шадрина В.В. Применение методов прогнозирования в технических системах // Изв. ЮФУ: Технич. науки. 2011. № 2. С. 141–145.

6. Rosli M., Hasimah N., Yusop M. Fault Prediction Model for Web Application Using Genetic Algorithm. Proc. Intern. Conf. on Comp. and Sof. Modeling IPCSIT, 2011, vol. 14, pp. 71–77.