УДК 004.052

МЕТОДЫ ОЦЕНКИ НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Штарик А.В.
Научный руководитель - к.т.н., доцент Царев Р.Ю.

Сибирский федеральный университет

В работе рассмотрены основные направления построения оценки надёжности программного обеспечения, а также приведена классификация моделей оценки надёжности программного обеспечения.

Для количественной оценки показателей надежности программного обеспечения используют модели надежности, под которыми понимаются математические модели, построенные для оценки зависимости надежности от заранее известных или определенных в ходе выполнения задания параметров. Эти модели можно разделить на две основные группы: эмпирические и аналитические.

Эмпирические модели основаны на анализе накопленной информации о функционировании ранее разработанных программ.

Наиболее простая эмпирическая модель связывает число ошибок в программном обеспечении с его объемом. Опытные данные свидетельствуют, что к началу системного тестирования в программном обеспечении на каждые 1000 операторов приходится примерно 10 ошибок. Уровень надежности программного обеспечения считается приемлемым для начала эксплуатации, если тому же объему операторов будет соответствовать одна ошибка.

Аналитические модели разделяются на статические и динамические. Среди динамических, также можно выделить непрерывные и дискретные.

При использовании непрерывной динамической модели предполагается, что функционирование программного обеспечения описывается набором последовательных состояний, переход между которыми происходит в случае возникновения отказа, за которым также следует восстановление. К непрерывной динамической модели относятся модель Джелински – Моранды и модель переходных вероятностей Маркова.

В дискретных моделях предполагается, что сначала проводится тестирование программного обеспечения (возможно, в несколько этапов). В случае появления отказов ищутся и исправляются все ошибки, из-за которых произошли отказы. После этого начинается период эксплуатации программного обеспечения. Примерами дискретных моделей могут послужить модель Шумана и модель Мусса.

Статические модели отличаются от динамических прежде всего тем, что в них не учитывается время появления ошибок. К статические модели относятся модель Миллса, модель Нельсона и модель Коркорэна.

Так как между надежностью и сложностью программного обеспечения существует тесная связь, то проблем надежности касается еще одна группа моделей – модели, предназначенные для оценки сложности программного обеспечения. Эти модели оценивают множество характеристик программного обеспечения, таких как длина программы, информационное содержание, число подсистем, число операторов, сложность интерфейса и т.п. Все существующие модели сложности и метрики показателей определяют только отдельные, частные характеристики сложных программ. Общим понятием для всех видов сложных программ является их структура.

При анализе структурной сложности программного обеспечения с целью определения его надежности необходимо производить многошаговую процедуру снижения его сложности. Распространение сложных программных систем и комплексов требует новых подходов к оценке их надежности. Сложность решения этой задачи обусловлена отсутствием универсальных методов и моделей.

Для удобства анализа показателей надежности сложных программных комплексов целесообразно представить их в виде совокупности менее сложных составляющих, программных модулей (ПМ). Такими модулями могут быть программные комплексы, отдельные программы, блоки или операторы. Количество модулей в программном комплексе может быть слишком большим для обработки, поэтому чаще программные модули группируют по типам. Каждый тип содержит программные модули, близкие по свойствам, в том числе по надежности. По заданной структуре программного комплекса, состоящего из некоторой совокупности программных модулей, имеющих известные показатели надежности, существует возможность найти показатель надежности программного комплекса. Для этого используются так называемые графовые модели программы (ГМП).

В качестве графовой модели программы рассмотрим ориентированный граф G (V, Г), где V = $\{v_i\}$ – множество вершин, Г = $\{g_{ij}\}$ – множество дуг. Граф системы G(V, Г) определяется структурой программной системы. Множество V вершин графа составляет программные модули (типы модулей), а множество дуг Г отражает связь между модулями, то есть, если из i-го модуля есть переход в j-ый модуль, то в графе G имеется дуга g_{ij}, ведущая из i-ой вершины в j-ую. Введем модельные ограничения. Предположим наличие в графовой модели программы одной начальной вершины v_0 (вход) и одной конечной v_k (выход). Допустим также, что из каждой вершины исходит не более двух дуг, число входящих в вершину дуг не ограничивается. Будем считать, что графовая модель программы не содержит циклов, а отображаемая ею программа относится к категории несамоизменяющихся.

При моделировании вычислительного процесса на графовой модели программы и исследовании свойств программного обеспечения предусматривается сообщение каждому элементу модели некоторого веса. Допустим, что каждая вершина v_i характеризуется аддитивным элементарным показателем d_i, связанным с исследуемым свойством программы. Введенные показатели образуют на графе множество D = $\{d_i\}$. Модель G(V, Г, D) может использоваться для статистического исследования различных маршрутов графовой модели программы. Выбор пути прогона на графе обусловливается совокупностью реализаций передач управления в вершинах, которые связаны со случайным процессом поступления на вход программы различных векторов входных данных, что приводит к случайному характеру выбора маршрутов в графе. Таким образом, исследуемое программное обеспечение можно представить сложной системой со случайной структурой, динамику функционирования которой целесообразно описать статистически с помощью вероятностей перехода от i-ой к j-ой вершине графовой модели программы.

Для завершающей стадии жизненного цикла программного комплекса следует использовать модель определения надежности с системно-независимым аргументом (количество прогонов ПО), например, модель Нельсона.

Однако практическое использование этой модели вызывает трудности, особенно для относительно больших программных комплексов массового применения, так как связывает оценку надежности программного обеспечения с количеством возможных программных маршрутов реализации вычислений и не рассматривает характеристики этих маршрутов. Для устранения этих недостатков модели Нельсона – единственной,

определяющей надежность программного обеспечения в период эксплуатации, – используют структурные графовые модели исследуемого программного обеспечения.

Есть и другие модели, которые в том или ином виде дают оценки надёжности. Модели эти часто используют какую-либо дополнительную информацию о процессе разработки, например данные о покрытии кода тестами или некоторую информацию об организации процесса разработки. Как следствие, возможность применения таких моделей часто ограничена, из-за чего они находят мало применения на практике