# 3 АНАЛИЗ НАДЕЖНОСТИ ВЕБ-ПРИЛОЖЕНИЯ

## 3.1 Расчет надежности по модели сложности

Модель сложности программного обеспечения основывается на предположении, что уровень безошибочности продукта может быть предсказан с использованием метрик сложности. Это справедливо для непреднамеренных уязвимостей, так как, чем сложнее и больше программа, тем выше вероятность ошибок при её написании и модификации. Для расчета надежности программного средства по модели сложности используются метрики размера, сложности потока управления, сложности потока данных, а также объектно-ориентированные метрики.

К метрикам размера относят объём (*V*) и потенциальный объём (*V\**) программного средства.

Объём программного средства *V* рассчитывают по формуле (3.1):

|  |  |
| --- | --- |
|  | (3.1) |

где:

– n1 =15 (число уникальных операторов);

– n2 =30 (число уникальных операндов);

– N1 =170 (общее число операторов);

– N2 =85 (общее число операндов).

Потенциальный объем программы определяется по формуле 3.2:

(3.2)

,

где *n*\* – теоретический словарь программы (словарный запас, необходимый для написания программы с учетом того, что необходимая функция уже реализована в языке программирования). Теоретический словарь программы равен сумме количества операторов и операндов программы, при условии того, что нужная функциональность уже реализована в программе. Для разрабатываемого программного средства *n*\*= 136.

Для разрабатываемого программного средства значения объема и потенциального объема равны:

.

Для определения метрик сложности потока управления используются следующие характеристики:

– CL – абсолютная сложность программы, характеризующаяся количеством операторов условия;

– cl – относительная сложность программы, определяемая как отношение CL к общему числу операторов;

– CLI – максимальный уровень вложенности операторов условия.

Для разрабатываемого программного средства значения этих характеристик равны: CL =178, cl = 79, CLI = 57.

Метрика сложности потока данных, предложенная Чепиным, оценивает информационную прочность программного модуля на основе анализа использования переменных из списка ввода-вывода. Все переменные делятся на четыре группы:

– P (вводимые переменные для расчетов и обеспечения вывода);

– M (модифицируемые или создаваемые внутри программы переменные);

– C (переменные, участвующие в управлении работой программного модуля);

– T (не используемые в программе переменные).

Метрика Чепина рассчитывается по формуле 3.3:

(3.3)

Для разрабатываемого программного средства значение метрики Чепина равно:

Теоретическая длина программы N рассчитывается по формуле 3.4:

(3.4)

где n1 – словарь операторов, n2– словарь операндов.

Для разрабатываемого программного средства = 190,81.

Уровень качества программирования L определяется как отношение потенциального объема к фактическому по формуле 3.5:

(3.5)

Для разрабатываемого программного средства L = 0,69.

Аппроксимированный уровень качества программирования L^ рассчитывается по формуле 3.6:

(3.6)

.

Для разрабатываемого программного средства = 0,05.

Интеллектуальные усилия на разработку программы E рассчитываются по формуле 3.7:

Для разрабатываемого программного средства E = 2034,64

(3.7)

Для оценки объектно-ориентированных характеристик программного средства используются метрики Чидамбера и Кемерера:

– WMC – суммарная сложность всех методов класса;

– DIT – глубина дерева наследования;

– NOC – количество потомков;

– CBO – сцепление между классами;

– RFC – отклик для класса;

– LCOM – недостаток сцепления методов.

Для разрабатываемого программного средства значения этих метрик равны: WMC = 201, DIT = 5, NOC = 3, CBO = 13, RFC = 64, LCOM = 7.

Результаты расчетов представлены в таблице 3.1.

Таблица 3.1 – Результаты расчетов

|  |  |  |  |
| --- | --- | --- | --- |
| Метрики ПС | amin | amax | ai |
| V | 108 | 1364 | 1400,4 |
| V\* | 540 | 83362,41 | 963,9 |
| CL | 7 | 368 | 178 |
| cl | 25 | 1278 | 79 |
| CLI | 41 | 4213 | 57 |
| Q | 67 | 3589 | 170 |
| N^ | 6 | 186 | 190,81 |
| L | 0,0054 | 2 | 0,69 |
| L^ | 0 | 3 | 0,05 |
| E | 29 | 967 | 2034,64 |
| WMC | 13 | 299 | 201 |
| DIT | 1 | 8 | 5 |
| NOC | 1 | 32 | 3 |
| CBO | 1 | 27 | 13 |
| RFC | 1 | 163 | 64 |
| LCOM | –39 | 387 | 7 |

Исходя из полученных метрик рассчитаны нормативный и фактический уровни каждой метрики (формулы (3.8) и (3.9) соответственно):

(3.8)

(3.9)

По полученным результатам определен дискриминант каждой метрики, используя формулу (3.10):

(3.10)

Результаты вычислений по формулам (3.8 – 3.10) представлены в таблице 3.2.

Таблица 4.2 – Результаты вычисления

|  |  |  |  |
| --- | --- | --- | --- |
| Метрики ПС | xmin | xфi | di |
| V | 0,07917888563 | 1,026686217 | -0.00223638 |
| V\* | 0,006477739787 | 0,01156276552 | 0.557359718 |
| CL | 0,01902173913 | 0,05434782609 | 0.020697812 |
| Cl | 0,01956181534 | 0,02034428795 | 0.302817542 |
| CLI | 0,009731782578 | 0,009969143128 | 0.716539671 |
| Q | 0,01866815269 | 0,04736695458 | 0.382591776 |
| N^ | 0,03225806452 | 1,102150538 | -0.00084029 |
| L | 0,0027 | 0,345 | 0.005159476 |
| L^ | 0 | 0,01666666667 | 0 |
| E | 0,02998965874 | 2,105305067 | -0.01622308 |
| WMC | 0,04347826087 | 0,6722408027 | 0.022161918 |
| DIT | 0,125 | 0,25 | 0.085714286 |
| NOC | 0,03125 | 0,09375 | 0.311827957 |
| CBO | 0,03703703704 | 0,2592592593 | 0.041420118 |
| RFC | 0,006134969325 | 0,1226993865 | 0.009548611 |
| LCOM | -0,1007751938 | 0,05167958656 | -4.96981891 |

Риск снижения надежности рассчитывается по формуле 3.11:

(3.11)

Где λi – весовые коэффициенты для каждой метрики. Для упрощения расчетов можно считать, что все метрики вносят равный вклад в надежность, т.е. λi = 1/16.

Для разрабатываемого программного средства вероятность безотказной работы равна 3.12:

(3.12)

Таким образом, вероятность безотказной работы программного средства по модели сложности составляет приблизительно 0,907.

## 3**.2 Расчет надежности по модели Джелинского – Моранды**

Модель Джелинского – Моранды используется для оценки надежности программного обеспечения на основе данных о времени между обнаружением ошибок Такая модель строится на основе следующих допущений:

1 Интенсивность обнаружения ошибок λ(t) пропорциональна текущему числу ошибок в программном средстве, т. е. числу оставшихся ошибок.

2 Все ошибки одинаково вероятны, и их появления независимы.

3 Каждая ошибка имеет один и тот же порядок серьезности.

4 Время до следующего отказа (ошибки) распределено экспоненциально.

5 Программное средство функционирует в среде, близкой к реальной.

6 Ошибки постоянно корректируются без внесения в программное средство новых.

7 λ(t) = const в интервале между двумя соседними ошибками.

В данном разделе проведем расчет надежности программного средства на основе модели Джелинского – Моранды. Общее время тестирования программы составило 17 дней. За это время было обнаружено 16 ошибок. Время между обнаружением ошибок (Xi) приведено в таблице 3.3.

Таблица 3.3 – Время между ошибками (Xi) в часах:

|  |  |
| --- | --- |
| Номер ошибки (i) | Время между ошибками (Xi) |
| 1 | 5 |
| 2 | 8 |
| 3 | 10 |
| 4 | 12 |
| 5 | 14 |
| 6 | 7 |
| 7 | 9 |
| 8 | 16 |
| 9 | 17 |
| 10 | 18 |
| 11 | 19 |
| 12 | 20 |
| 13 | 21 |
| 14 | 22 |
| 15 | 23 |
| 16 | 25 |

Метод оценивает вероятность безотказной работы веб-приложения. Метод использует формулы 3.13 и 3.14 для вычисления этого значения:

|  |  |
| --- | --- |
|  | (3.13) |

|  |  |
| --- | --- |
|  | (3.14) |

Для расчёта коэффициента B необходимо найти решения уравнения 3.15:

|  |  |
| --- | --- |
|  | (3.15) |

где A – константа, зависящая от данных и равная значению, которое подходит под условие 3.16:

|  |  |
| --- | --- |
|  | (3.16) |

Для выполнения расчета примем A = 9.

Для нахождения B выполним замену (заменим B + 1 на m) и найдем минимально возможную разность между и . Также на m накладывается условие, что оно должно быть больше, чем n + 1. Такое число отражает число прогнозируемых (пока не обнаруженных) ошибок.

Функции f(m) и g(m,A) определяются следующим образом по формулам 3.17 и 3.18:

|  |  |
| --- | --- |
|  | (3.17) |

|  |  |
| --- | --- |
|  | (3.18) |

В ходе расчетов для различных значений m (диапазон значений был выбран от 17 до 80) получаем, что минимальная разность достигается при m=51, что соответствует общему числу ошибок B=50 (так как B=m−1).

Неизвестный коэффициент *K* рассчитывается по формуле 3.19:

|  |  |
| --- | --- |
|  | (3.19) |

Подставляя найденное значение в формулу, получаем:

|  |  |
| --- | --- |
|  | (3.20) |

Интенсивность отказов λ(t)после обнаружения 16 ошибок рассчитывается по формуле 3.13 и равно:

|  |  |
| --- | --- |
|  | (3.21) |

Вероятность безотказной работы *P*(*t*) на интервале t=1*t*=1 час определяется по формуле 3.14 и равно:

|  |  |
| --- | --- |
|  | (3.22) |

Среднее время до появления следующей ошибки (i + 1) рассчитывается по формуле 3.23:

|  |  |
| --- | --- |
|  | (3.23) |

Таким образом получаем, что время до появления следующей ошибки равно приблизительно 18 часов.

Время до окончания тестирования оценивается как сумма времени до обнаружения оставшихся ошибок (формула 3.24):

|  |  |
| --- | --- |
|  | (3.24) |

Предположив, что ошибки будут находиться через среднее значение времени, получим = 108 дней.

Таким образом, модель Джелинского – Моранды позволяет оценить надежность программного средства и спрогнозировать время, необходимое для обнаружения и исправления всех ошибок.

## 3.3 Расчет надежности по модели Муса

Проведем расчет надежности по модели Муса. В этой модели надежность программного средства на этапе эксплуатации оценивается по результатам тестирования. Пусть Т – суммарное время тестирования, М – число отказов, произошедших за время тестирования. Тогда по модели Муса средняя наработка до отказа после тестирования определяется по формуле (3.25).

|  |  |
| --- | --- |
|  | (3.25) |

В этой формуле τ0 – средняя наработка до отказа до начала тестирования, С – коэффициент, учитывающий уплотнение тестового времени по сравнению с временем реальной эксплуатации. Неизвестный параметр τ0 можно оценить из соотношения (3.26).

|  |  |
| --- | --- |
|  | (3.26) |

В этой формуле N – первоначальное число ошибок в программной средстве, которое можно оценить с помощью другой модели, позволяющей определить N на основе статистических данных, полученных при тестировании. К – коэффициент проявления ошибок, который определяется эмпирическим путем по однотипным программам, обычно это значение изменяется от 1,5·10–7 до 4·10–7. f – средняя скорость исполнения программы (A), деленная на число команд (операторов (B)).

Средняя скорость исполнения одного оператора программного средства определяется по формуле (3.27).

|  |  |
| --- | --- |
|  | (3.27) |

Длительности этапов тестирования составляют t1 = 50 ч, t2 = 42 ч, t3 = 36 ч. Число отказов на первом этапе m1 =8, на втором – m2 = 5, на третьем – m3 = 3. Средняя скорость исполнения программного средства = 107 операторов/ч, количество операторов в программном средстве = 968. Определим надежность системы для периода эксплуатации t = 128 ч.

Найдем среднюю скорость выполнения одного оператора по формуле (3.28):

(3.28)

Первоначальное количество ошибок в программном средстве N равно 16. Коэффициент проявления ошибок K примем равным 3,02·10-7. Найдем формуле (3.29).

(3.29)

Примем значение коэффициента С = 12. Тогда средняя наработка до отказа после тестирования на этапе эксплуатации программного обеспечения рассчитывается по формуле (3.25).

Найдем надежность программного средства для периода эксплуатации t, равному 128 часам, по формуле (3.23).

|  |  |
| --- | --- |
|  | (3.23) |

Надежность равна:

Таким образов в этом разделе были посчитаны метрики надежности веб-приложения для усовершенствованной визуализации мониторинга качества воды с использованием цветных гексагонов на карте города Минска. Метрики надежности веб-приложения были рассчитаны по модели сложности, по модели Джелинского-Моранда, по модели Муса и составили 0,907, 0,95, 0,87 соответственно.