1.1 Модель сложности.

В ходе расчета модели сложности рассчитываются следующие метрики:

* метрики размера: объем программы *V*; потенциальный объем программы ;
* метрики сложности потока управления (по Джиблу): абсолютная сложность программы *CL*; относительная сложность программы *cl*; максимальный уровень вложенности оператора *CLI[1]*;
* метрики сложности потока данных: метрика Чепина *Q*;
* метрики стилистики и понятности программ (по Холстеду): теоретическая длина программы ; метрика корректности программы *L*; метрика корректности реальной программы ; число элементарных решений, принятых при написании программы *E*;
* объектно-ориентированные метрики: суммарная сложность всех методов класса *WMC*; глубина дерева наследований *DIT*; количество потомков *NOC*; сцепление между классами *CBO*; мощность множества классов *RFC*; недостаток сцепления методов *LCOM*.

Метрики размера программного средства (ПС). В данные метрики включают следующее[7]:

* объем программы *V*;
* потенциальный объем программы *V\**.

Объем V определяется по формуле:

|  |  |
| --- | --- |
| , | (1.1) |

где (*n*1 +*n*2) – словарь ПС; *n*1 – число уникальных операторов программы, включая символы-разделители, имена процедур и знаки операций (словарь операторов); *n*2 – число уникальных операндов программы (словарь операндов); (*N*1 + *N*2) – длина ПС *N*; *N*1 – общее число операторов в программе; *N*2 – общее число операндов в программе.

Подставив значения в формулу (1.1) получим выражение

*V* = (5463+7812) ∙ *log*2(4658 + 6647) = 13275 ∙ 13,465 = 178747,875

Потенциальный объем *V*\* рассчитывается по формуле:

|  |  |
| --- | --- |
|  | (1.2) |

где *n*\* – теоретический словарь ПС.

Подставив значения в формулу (1.2) получим выражение

|  |  |
| --- | --- |
|  |  |

Метрики сложности потока управления. Используются следующие метрики предложенные Джиблом:

* абсолютная сложность управления *CL*, характеризующаяся количеством операторов условия;
* относительная сложность *cl*;
* насыщенность ПС операторами условия определяется отношением *CL* к общему числу операторов;
* максимальный уровень вложенности оператора условия *CLI*.

Суть метода состоит в оценке информационной прочности отдельно взятого программного модуля с помощью анализа характера использования переменных из списка ввода-вывода.

Для этого все переменные, составляющие список ввода-вывода, разбивается на четыре функциональные группы:

1. *Р* – вводимые переменные для расчетов и обеспечения вывода.
2. *М* – модифицируемые, или создаваемые внутри программы переменные.
3. *С* – переменные, участвующие в управлении работой программного модуля (управляющие переменные).
4. *Т* – не используемые в программе («паразитные») переменные.

Поскольку каждая переменная может выполнять одновременно несколько функций, необходимо учитывать ее в каждой соответствующей функциональной группе.

Сложность потока данных *Q* определяется выражением:

|  |  |
| --- | --- |
|  | (1.3) |

Подставив значения в формулу (1.3) получим следующее значение:

|  |  |
| --- | --- |
|  |  |

Метрики стилистики и понятности ПС. В данной группе используются три метрики:

* отклонение реальной *N* от теоретической *N*^ длины ПС ∆*N* (в %);
* уровень качества программирования *L* (уровень ПС);
* оценка интеллектуальных усилий на разработку ПС *E*.

Отклонение *N* от *N^* определяется выражением:

|  |  |
| --- | --- |
| , | (1.4) |

где *N*^ определяется выражением:

|  |  |
| --- | --- |
|  | (1.5) |

где *n*1 – словарь операторов; *n*2–словарь операндов ПС.

Таким образом, теоретическая длина *N*^ информационной системы передачи информации составит:

|  |  |
| --- | --- |
| 141161.336 |  |

Отклонение *N* от *N*^ определим по формуле (1.4):

Уровень качества программирования *L* расчитывается по формуле:

|  |  |
| --- | --- |
|  | (1.6) |

Подставив численные значения получим:

|  |
| --- |
|  |

Исходным для введения этой характеристики является предположение о том, что при снижении стилистического качества программирования уменьшается содержательная нагрузка на каждый компонент программы и, как следствие, расширяется объем реализации исходного алгоритма.

Интеллектуальные усилия на разработку ПС *Е* рассчитывается по формуле:

|  |  |
| --- | --- |
| , | (1.7) |

где *V* – объем программы; *V*\* – потенциальный объем программы.

Подставив численные значения в формулу (1.7) получим:

|  |  |
| --- | --- |
|  |  |

Объектно-ориентированные метрики.Рассчитаем метрики Мартина:

* центростремительное сцепление *Ca* – количество классов вне этой категории, которые зависят от классов внутри этой категории;
* центробежное сцепление *Ce* – количество классов внутри этой категории, которые зависят от классов вне этой категории;
* нестабильность *I*.

Нестабильность определяется по формуле:

|  |  |
| --- | --- |
|  | (1.8) |

Подставив численные значения получим:

Далее рассчитаем нормативный *xmin* и фактический *xфi* уровни для каждой из метрик по формулам:

|  |  |
| --- | --- |
| *,* | (1.9) |
|  | (1.10) | |

где *ai* – значение метрики, рассчитанное для конкретного ПС; *amin* – минимально возможное значение этой метрики для данного типа ПС; *amax* – максимально возможное значение этой метрики для данного ПС.

Также рассчитываем дискриминант *di* для каждой метрики по формуле:

|  |  |
| --- | --- |
| , | (1.11) |

Результаты вычислений приведены в таблице 1.1.

Таблица 1.1 – нормативные пределы метрик, результаты вычислений нормативных и фактических уровней метрик, дискриминантов метрик

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Метрики | Значение *ai* | *amin* | *amax* | *xmin* | *xϕi* | *di* |
| *V* | 178747,875 | 31 000 | 240 000 | 0.129167 | 0.744 | 0.0508 |
| *V\** | 49637,716 | 508 | 69 000 | 0.007362 | 0.71 | 0.0028 |
| *Q* | 1070 | 57 | 3425 | 0.016642 | 0.312 | 0.0372 | |
| *ΔN* | 9,4 | 4 | 35 | 0.114286 | 0.2685 | 0.3514 | |
| *L* | 0,277 | 0,003 | 3,1 | 0.000968 | 0.089 | 0.00987 | |
| *E* | 643679,95 | 23000 | 991000 | 0.023209 | 0.6495 | 0.0128 | |
| *Ca* | 23 | 11 | 141 | 0.078014 | 0.1631 | 0.434 | |
| *Ce* | 25 | 11 | 210 | 0.052381 | 0.1190 | 0.409 | |
| *I* | 0,52 | 0 | 2 | 0 | 0.26 | 0 | |

Рассчитываем риск снижения надежности работы программного средства *R* по формуле:

|  |  |
| --- | --- |
|  | (1.12) |

где *λi* – весовые коэффициенты для конкретных метрик. Отражает на сколько та или иная метрика имеет больший вес для надежности ПС и должна удовлетворять условию:

|  |  |
| --- | --- |
|  | (1.13) |

где *N* – количество метрик, используемое при расчете риска снижения надежности.

Для упрощения задачи примем равнозначным вклад каждой метрики в результат расчета риска снижения надежности.

Таким образом, по формуле (1.12) для данного программного продукта риск снижения надежности равен:

Следовательно, надежность равна:

10,056 = 0,9432

## Модель Джелинского – Моранды

Модель Джелинского – Моранды строится на основе следующих допущений[6]:

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 соответствии с этими допущениями интенсивность возникновения ошибок в программном средстве можно представить в виде

|  |  |
| --- | --- |
|  | (1.22) |

где *t* – произвольное время между обнаружением (*i*-1) и *i*-й ошибок; *K* – неизвестный коэффициент; *B* – неизвестное общее число ошибок в программном средстве.

А вероятность безотказной работы программного средства рассчитывается согласно выражению

|  |  |
| --- | --- |
|  | (1.23) |

Для нахождения *B* используют выражения

|  |  |
| --- | --- |
|  | (1.24) |

|  |  |
| --- | --- |
|  | (1.25) |

|  |  |
| --- | --- |
|  | (1.26) |

|  |  |
| --- | --- |
|  | (1.27) |

|  |  |
| --- | --- |
|  | (1.28) |

где *m* ≥ *n*+1 – число прогнозируемых ошибок; *n –* количество выявленных ошибок.

Неизвестный коэффициент *K* рассчитывают по формуле

|  |  |
| --- | --- |
|  | (1.29) |

На этапе отладки программного средства за 179 день было выявлено 10 ошибок. Исходные данные сведены в таблицу 1.2 в виде интервалов времени *Хi* (дни) между соседними ошибками (*i* – номер ошибки). Необходимо найти вероятность безотказной работы программного средства.

Таблица 1.2 – Исходные данные для расчетов

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *i* | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| *Xi,* день | 2 | 9 | 11 | 14 | 16 | 18 | 22 | 27 | 33 | 39 |

Подставив численные значения в формулу (1.27) получим *A=*7,1712

Осуществим расчеты по формулам (1.25), (1.26) и сведем их в таблицу 1.3.

Таблица 1.3 – Результаты вычислений

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *i* | *Xi* | *i∙Xi* | *m* | *gn*(*m, A*) | *fn*(*m*) | *|fn*(*m*) – *gn*(*m, A*)| |
| 1 | 2 | 2 | 11 | 2.553191 | 2.928968254 | 0.375777 |
| 2 | 9 | 18 | 12 | 2.033898 | 2.019877345 | 0.01402 |
| 3 | 11 | 33 | 13 | 1.690141 | 1.603210678 | 0.08693 |
| 4 | 14 | 56 | 14 | 1.445783 | 1.346800422 | 0.09898 |
| 5 | 16 | 80 | 15 | 1.263158 | 1.168228993 | 0.09493 |
| 6 | 19 | 114 | 16 | 1.121495 | 1.03489566 | 0.0866 |
| 7 | 22 | 154 | 17 | 1.008403 | 0.930728993 | 0.07767 |
| 8 | 27 | 216 | 18 | 0.916031 | 0.84669538 | 0.06934 |
| 9 | 33 | 297 | 19 | 0.839161 | 0.777250935 | 0.06191 |
| 10 | 39 | 390 | 20 | 0.774194 | 0.718771403 | 0.05542 |

Из таблицы 1.3 видно, что наилучшим решением для уравнения (1.24) является *m=*12, в таком случае согласно выражению (1.28) *B* = 14.

Подставив в формулу (1.29) рассчитанные значения получим

Далее согласно с формулой (1.22) рассчитаем интенсивность возникновения ошибок после того, как обнаружена ( *i –* 1) ошибка.

По формуле (1.23) вероятность безотказной работы программного средства *P*(*t*) = 0,94.

## Модель Муса

В модели Муса надежность ПС на этапе эксплуатации оценивается по результатам тестирования [7].

Средняя наработка до отказа после тестирования определяется по формуле:

|  |  |
| --- | --- |
| , | (1.37) |

где 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 следующего соотношения:

|  |  |
| --- | --- |
|  | (1.38) |

Средняя скорость исполнения одного оператора ПС расчитывается по формуле:

|  |  |
| --- | --- |
|  | (1.39) |

Общее время тестирования *Т + ΔТ* должно удовлетворять соотношению:

|  |  |
| --- | --- |
|  | (1.40) |

Надежность ПС для периода эксплуатации *t* определяются по формуле:

|  |  |
| --- | --- |
|  | (1.41) |
|  |  |

Для данного программного модуля длительности этапов тестирования составляют *t1* = 19 часов, *t2* = 23 часов, *t3* = 28 часов. Число отказов на первом этапе *m*1 = 3, на втором – *m*2 = 5, на третьем – *m*3 = 1. Средняя скорость исполнения ПС *A* = 104 операторов/час, количество операторов в ПС *B* = 740. Период эксплуатации *t* = 189 часов.

Найдем среднюю частоту выполнения одного оператора с помощью формулы (1.39):

|  |  |
| --- | --- |
|  |  |

Первоначальное количество ошибок в программном средстве (из модели Шумана) *N* = 4. Коэффициент проявления ошибок *K* для данного программного средства 1,5 ∙ 10-6, в таком случае τ0 можно рассчитать по формуле (1.38)

Один час тестирования соответствует 6 часам работы, следовательно значение коэффициента *С* = 5. Тогда средняя наработка до отказа после тестирования на этапе эксплуатации ПО согласно с формулой (3.17) равна:

Найдем надежность программного средства для периода эксплуатации *t* = 189 часов по формуле (1.41):

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Пилиневич, Л. П. Эргатические системы: учебно-метод. пособие / Л. П. Пилиневич, Н. В.Щербина, К. Д. Яшин. – Минск: БГУИР, 2015. – 96 с.

[2] habr.com [Электронный ресурс] – Электронные данные. – Режим доступа: https://habr.com/ru/post/346396/.

[3] Элементы теории вероятностей и математической статистики. Учебное пособие / Пронькин Ю. С., Лесничевская И. А. – ТГТУ 2005. – 50

[4] Доманов, А.Т., Сорока, Н.И, Стандарт предприятия / А.Т. Доманов, Н.И. Сорока. – Минск: БГУИР, 2014 – 174 с.

[5] Надежность программного обеспечения: методическое пособие / В.В. Шедько. – Витебск: ВГУ имени П.М. Машерова, 2017 – 26 с.

[6] Осипович, В. С. Проектирование информационных систем промышленной безопасности. Курсовое проектирование : пособие / В. С. Осипович, В. В. Егоров. – Минск: БГУИР, 2020. – 84 с.

[7] Казарин, О. В. Надежность и безопасность программного обеспечения: учеб.пособие для бакалавриата и магистратуры / О. В. Казарин, И. Б. Шубинский. — М. : Издательство Юрайт, 2018. — 342 с.