МЕТРИКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

УДК 004.4 (075.8)

ЗАХАРЧЕНКО Раиса Николаевна
к.т.н., доцент кафедры информационных технологий Херсонского национального технического университета.
Научные интересы: новые информационные технологии.

КИРЮШАТОВА Татьяна Григорьевна
к.т.н., доцент кафедры информационных технологий Херсонского национального технического университета.
Научные интересы: новые информационные технологии.

КИРЮШАТОВА Екатерина Владимировна
ассистент кафедры информационных технологий Херсонского национального технического университета.
Научные интересы: новые информационные технологии.

ВВЕДЕНИЕ

Разработчики программного обеспечения (ПО) в настоящее время уделяют большое внимание не только написанию ПО для различных проектов, но и ставят задачу написания качественного и наименее сложного программного кода.

Среди особенностей программ, на которые обращают внимания разработчики и пользователи ПО можно выделить надежность ПО, эффективность ПО, модульность ПО и структурность ПО. В основном заказчиков или пользователей программного обеспечения интересуют функциональные свойства, характеризующие полезность и сложность ПО.

В современной индустрии разработки программ для оценки сложности ПО применяют метрики, позволяющие потенциальным менеджерам проектов и организаций изучить сложность проектов, которые предполагается разработать, оценить объем работ, стиль программы и другие параметры. Однако, метрики носят только рекомендательный характер, на них нельзя полностью опираться при определении сложности программных продуктов.

Различают количественные метрики, метрики сложности потока управления программами, метрики сложности потока управления данными, метрики сложности потока управления и данными программы, а также гибридные метрики [1]. В связи с развитием объектно-ориентированных языков программирования появился новый класс метрик, объектно-ориентированные метрики. В данной группе наиболее часто используемыми являются наборы метрик Мартина и набор метрик Чидамбера и Кемерера [1].

Многие из предложенных метрик имеют высокую погрешность, но без использования метрик оценки сложности программного кода невозможно оценить сложность тестирования и спрогнозировать главный критерий – время выполнения конкретного проекта [2].

Целью работы является изложение методики вычисления сложности программного обеспечения.

ПОСТАНОВКА ЗАДАЧИ

Внешние особенности, которые отображают точку зрения пользователя, обуславливают качество программного обеспечения, то есть являются его факторами (рис. 1). Для разработчиков представляют интерес не только внешние, но и внутренние, или конструктивные свойства, от которых зависит выполнение требований к программному обеспечению.

Характеристики качества отображают свойства, которые определяют качество ПО. Для определения количественной оценки характеристик качества ПО исполь-

зуют иерархические системы измерений, при этом учитывают факторы, критерии метрики и оценочные элементы (рис. 2). Факторы и критерии отображают функциональные характеристики ПО и составляют два верхних уровня иерархии измерений. Метрики и оценочные элементы отображают конструктивные характеристики, влияющие на качество ПО - это нижние уровни иерархии. Оценочные элементы являются самым нижним уровнем состава характеристик и обеспечивают измеряемость характеристик качества.

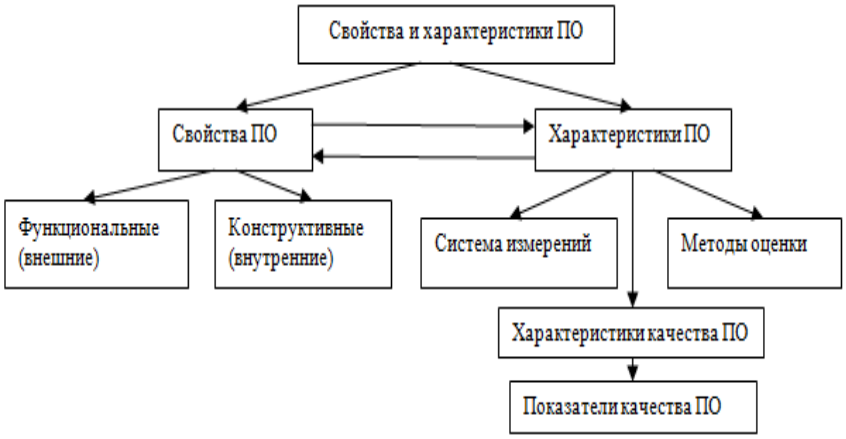

Рисунок 1 – Группы свойств и характеристики ПО

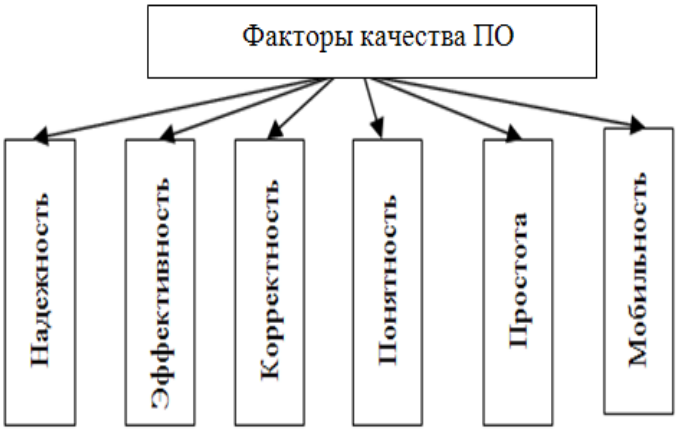

Рисунок 2 – Факторы качества, которые отображают полезность ПО

Фактором качества называют свойство, которое обусловливает качество ПО. Качество ПО обусловлено несколькими факторами. Численная оценка фактора качества использует один или несколько критериев качества.

ОСНОВНАЯ ЧАСТЬ

Для расчета количественной оценки фактора качества используют один или несколько критериев качества. В настоящее время имеется немало программных продуктов, позволяющих определять метрики, для оценки сложности разработанных программ. Например, метрики программ можно получить в программе IBM Rational Clear Case: модуль формирования отчетов с вычислением метрик размера и сложности программы [3]. Report Builder – инструмент формирования отчетности в этой программе. Он позволяет создавать различные предустановленные отчеты и позволяет экспортировать эти отчеты в различных форматах. В группе стандартных отчетов Report Builder отсутствует функция формирования отчета по группе разработчиков при необходимости.

Формирование отчета начинается с выбора нужного отчета из контекстного меню каталога ClearCase. При

этом производится запуск скрипта, который позволяет выполнять запрос входных данных. В разработанной программе предусмотрены следующие функции:

– сбор входной информации для отчета;
– формирование строки запроса cleartool find;
– анализ выбранных версий;
– подсчет числа измененных версий;
– расчет необходимых метрик;
– создание атрибутов со значениями полученных метрик на отработанных версиях;
– формирование конечного результата в виде удобочитаемого отчета.

На рис.3 представлена экранная форма окна программы Rationl Clear Case в режиме создания типов атрибутов.

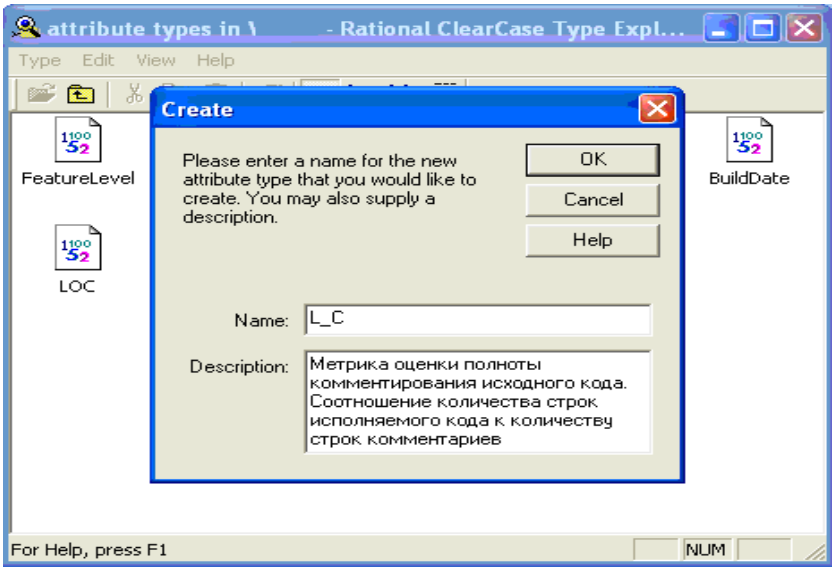

Рисунок 3 - Форма окна типов атрибутов в программе

В 2014 году Verifysoft (Германия) выпустила Verybench, который является графическим дополнением для определения сложности кода для Testwell CMT++ версии 2.1. Verybench позволяет менеджерам, разработчикам и тестировщикам быстро оценить исходные качества кода, ремонтопригодность и сложность [4]. Измерения сложности может быть выполнено в различных областях. Результаты этих измерений (метрические значения), представляются в ясной и агрегированной форме. Для представления результатов можно использовать графический интерфейс. Кроме того, Verybench поддерживает анализ кода путем растворения сложности и представления основных аспектов, которые являются точными и актуальными для улучшения качества источника кода.

С Verybench возможно:

– быстрее планировать и выполнять анализ кода и рефакторинга, просто глядя на графическое представление метрических значений;
– быстрее и точнее оценить ремонтопригодность исходного кода с помощью встроенной в Snapshot системы Verybench;
– лучше документировать и улучшить качество исходного кода с помощью рейтинга качества системы;
– легче и более универсально формировать выходные отчеты.

Все полученные результаты преобразуются в графическую оценку. Система Verybench имеет интерактивный и интуитивно понятный пользовательский интерфейс.

На занятиях студентам по направлению «Программная инженерия» была поставлена задача ознакомления с метриками программного обеспечения и с существующим ПО для их определения. Для будущих разработчиков ПО большой интерес представляют конструктивные свойства, от которых зависит выполнение требований к ПО и выполнение проектов в определенные сроки.

Студентам для приобретения практических навыков необходимо не только ознакомление с метриками ПО, но и обучение написанию программ для их определения.

Основной метрикой сложности является метрика Маккейба, которая предлагает считать цикломатическую сложность графа программы, или, как ее еще называют, цикломатическое число Маккейба, характеризующее трудоемкость тестирования программы [1].

В основу применения метода Маккейба положена теорема о том, что изначально строится ориентированный граф, у которого только один вход и только один выход. Вершины графа соотносятся с теми участками хода программы, у которых имеются только последовательные вычисления и нет операторов ветвления и циклов, а дуги соотносятся с переходами от блока к блоку и ветвями выполнения программы. При построении данного графа должно соблюдаться следующее условие: каждая вершина досягаема из начальной, а конечная вершина досягаема из любой другой вершины [1].

Цикломатическое число Маккейба основано на анализе полученного графа и вычисляется по формуле [2]:

$$Z(G)=e-v+2p, \quad (1)$$

где:

e – число дуг ориентированного графа G;

v – число вершин;

p – число компонентов связности графа.

Число компонентов связности графа – это количество дуг, добавляемых для преобразования графа в сильно связный, то есть такой, у которого любые две вершины взаимно достижимые. Для корректных программ графы не имеют недостижимых от точки входа участков и «висячих» точек входа и выхода. Поэтому, сильно связный граф для таких программ получается путем замыкания дугой вершины, обозначающей конец программы, на вершину, обозначающую точку входа в эту программу [2].

При помощи Z(G) можно определить число линейно независимых контуров в сильно связном графе, то есть, определяя цикломатическое число Маккейба, определяют требуемое количество проходов для покрытия всех контуров сильно связанного графа, или количество тестовых прогонов программы, для проверки правильности работы каждой ветви программы.

В результате решения поставленной задачи студентами разработаны программы для определения метрик сложности ПО. Например, на рис. 4 представлена экранная форма программы для определения цикломатической сложности программы по методу Маккейба.

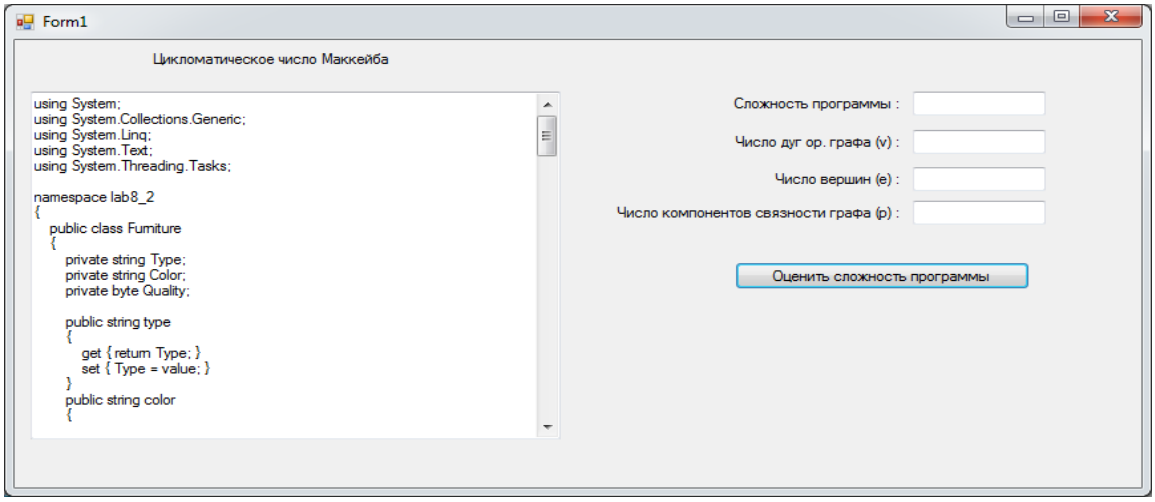

Рисунок 4 – Окно программы для определения цикломатической сложности программы по методу Маккейба

ВЫВОДЫ

В настоящее время процессы, происходящие на производстве определяют требуемый уровень качества ПО. Одним из параметров управления качеством ПО являются метрики, позволяющие оценивать информационный процесс воздействия на программы и доку-

ментацию с целью обеспечения нужного качества при различных внешних и внутренних условиях. Базовые значения показателя качества определяются требованиями к уровню качества по каждому фактору. Все стадии жизненного цикла предполагают измерение, оценку и контроль уровня сложности программ. Для оценки сложности ПО студентами специальности по направлению «Программная инженерия» изучаются метрики и выполняется программная реализация их определения.

ЛИТЕРАТУРА:

1. Tabunshhik G.V., Kudermetov R.K., Bragina T.I. T-12 Inzhenerija jakosti programnogo zabezpechennja: navchal'nij posibnik. – Zaporizhzhja: ZNTU, 2013. – 176 s.
2. Varenica V.V. Problemy vychislenija metrik slozhnosti programnogo obespechenija pri provedenii audita bezopasnosti koda metodom ruchnogo recenzirovanija. – Vestnik MGTU im. N.Je. Baumana. Ser. «Priborostroenie», 2011.
3. Metriki koda i ih prakticheskaja realizacija v IBM Retional Clear Case. Jelektronnyj istochnik. URL: http://www.ibm.com/developerworks/ru/edu/0108novich/section2html.
4. Verifysoft Technology GmbH Testwell STA ++ Testwell STS ++ Offenburg (Germanija), 2015. Jelektronnyj istochnik. URL:http://www.verifysoft.com/en_cmtx_news.html.

Рецензент: *д.т.н., доц. Шерстюк В.Г.,*
Херсонский национальный технический университет.