**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Гжельский государственный университет»** (ГГУ)

Колледж ГГУ

Специальность 09.02.07. Информационные системы и программирование.

**Реферат**

по предмету «Управление проектами»

на тему «Метрики сложности программного кода»

ВЫПОЛНИЛ:

Студент группы ИСП-О-17

Скрябин С.И.

ПРОВЕРИЛА:

Прокуронова А. Ю.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

п. Электроизолятор

2019 г.

**Ме́трика програ́ммного обеспе́чения** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *software metric*) — мера, позволяющая получить численное значение некоторого свойства [программного обеспечения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) или его [спецификаций](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D1%86%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F).

Поскольку количественные методы хорошо зарекомендовали себя в других областях, многие теоретики и практики информатики пытались перенести данный подход и в [разработку программного обеспечения](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F). Как сказал [Том ДеМарко](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D0%BC_%D0%94%D0%B5%D0%9C%D0%B0%D1%80%D0%BA%D0%BE), «вы не можете контролировать то, что не можете измерить».

Метрики

Набор используемых метрик включает:

* порядок роста (имеется в виду [анализ алгоритмов](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%BE%D0%B2) в терминах [асимптотического анализа](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9#%D0%90%D1%81%D0%B8%D0%BC%D0%BF%D1%82%D0%BE%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C) и [O-нотации](https://ru.wikipedia.org/wiki/%C2%ABO%C2%BB_%D0%B1%D0%BE%D0%BB%D1%8C%D1%88%D0%BE%D0%B5_%D0%B8_%C2%ABo%C2%BB_%D0%BC%D0%B0%D0%BB%D0%BE%D0%B5)),
* [количество строк кода](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D1%82%D0%B2%D0%BE_%D1%81%D1%82%D1%80%D0%BE%D0%BA_%D0%BA%D0%BE%D0%B4%D0%B0),
* [цикломатическая сложность](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D0%BA%D0%BB%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C),
* [анализ функциональных точек](https://ru.wikipedia.org/w/index.php?title=%D0%90%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D1%85_%D1%82%D0%BE%D1%87%D0%B5%D0%BA&action=edit&redlink=1),
* количество [ошибок](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3) на 1000 строк кода,
* степень [покрытия кода](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BA%D1%80%D1%8B%D1%82%D0%B8%D0%B5_%D0%BA%D0%BE%D0%B4%D0%B0) тестированием,
* [покрытие требований](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BA%D1%80%D1%8B%D1%82%D0%B8%D0%B5_%D1%82%D1%80%D0%B5%D0%B1%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9),
* количество [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) и [интерфейсов](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_(%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)),
* [метрики программного пакета](https://ru.wikipedia.org/w/index.php?title=%D0%9C%D0%B5%D1%82%D1%80%D0%B8%D0%BA%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BF%D0%B0%D0%BA%D0%B5%D1%82%D0%B0&action=edit&redlink=1) от Роберта Сесиль Мартина,
* [связность](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%8F%D0%B7%D0%BD%D0%BE%D1%81%D1%82%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)).

**Критика**

Потенциальные недостатки подхода, на которые нацелена критика:

* Неэтичность: Утверждается, что неэтично судить о производительности программиста по метрикам, введенным для оценки эффективности программного кода. Такие известные метрики, как количество строк кода и цикломатическая сложность, часто дают поверхностное представление об "удачности" выбора того или иного подхода при решении поставленных задач, однако нередко они рассматриваются как инструмент оценки качества работы разработчика. Такой подход достаточно часто приводит к обратному эффекту: в коде появляются более длинные конструкции и избыточные необязательные методы.
* Замещение «управления людьми» на «управление цифрами», которое не учитывает опыт сотрудников и их другие качества.
* Искажение: Процесс измерения может быть искажён за счёт того, что сотрудники знают об измеряемых показателях и стремятся оптимизировать эти показатели, а не свою работу. Например, если количество строк исходного кода является важным показателем, то программисты будут стремиться писать как можно больше строк и не будут использовать способы упрощения кода, сокращающие количество строк.
* Неточность: Нет метрик, которые были бы одновременно и значимы, и достаточно точны. Количество строк кода — это просто количество строк, этот показатель не даёт представления о сложности решаемой проблемы. Анализ функциональных точек был разработан с целью лучшего измерения сложности кода и спецификации, но он использует личные оценки измеряющего, поэтому разные люди получат разные результаты.

##### **Основные типы метрик**

##### В общем случае, для проекта я считаю вполне разумным начинать определять следующие типы метрик: Показатель текучести кадров

* Показатель утилизации ресурсов
* Показатели, связанные со сроками и бюджетом проекта
* Показатели, позволяющие оценить качество разрабатываемого продукта
* Интегральные показатели прогресса проекта

В целом, можно использовать следующий подход к выбору метрик для проекта:

1. Метрики этапов ЖЦ и календарного плана: Следить за графиком работ по этапам ЖЦ и сравнивать фактические и запланированные значения.
2. Метрики расходов по проектам / добавленной стоимости: Следить за значениями кумулятивной величины расходов в сравнении с бюджетом, а также общей стоимости проекта, постоянно обновляя данные по мере реализации проекта.
3. Метрики отслеживания изменений в требованиях: Число изменений в требованиях в масштабах проекта.  
   Метрики процесса разработки: Следить за числом реализованных в модели требований в сравнении с общим числом требований в проекте.
4. Метрики типов отказов: Отслеживать причины отказов ПО.
5. Остальные метрики по дефектам: Графическое представление числа отказов в месяц по месяцам на протяжении всего времени выполнения проекта.
6. Обзор метрики эффективности: Отслеживать плотность ошибок по фазам и использовать диаграммы для определения «пиков» и «провалов» на кривой, а также превышений предельно допустимых значений.

# **Метрики Холстеда**

*Качество ПО* - это совокупность свойств, определяющих полезность изделия (программы) для пользователей в соответствии с функциональным назначением и предъявленными требованиями.

*Характеристика качества программы* - понятие, отражающее отдельные факторы, влияющие на качество программ и поддающиеся измерению.

*Критерий качества* - численный показатель, характеризующий степень, в которой программе присуще оцениваемое свойство.

Критерии качества включают следующие характеристики: экономичность, документированность, гибкость, модульность, надёжность, обоснованность, тестируемость, ясность, точность, модифицируемость, эффективность, легкость сопровождения и т.д.

Критерий должен:

\* численно характеризовать основную целевую функцию программы;

\* обеспечивать возможность определения затрат, необходимых для достижения требуемого уровня качества, а также степени влияния на показатель качества различных внешних факторов;

\* быть по возможности простым, хорошо измеримым и иметь малую дисперсию.

Для измерения характеристик и критериев качества используют метрики.

**Метрики Холстеда**

Метрики Холстеда включают следующие характеристики.

1) Длина программы:

N'= n1 x log2(n1) + n2 x log2(n2)

Объем программы:

V = N \* log2(n) (бит)

2) Потенциальный (минимальный) объем:

V\* = (2+ n2\*) \* log2(2+ n2\*)

3) Граничный объем:

V\*\* = (2+ (n2\*)\* log2(n2\*)) \* log2(2+ n2\*)

5) Соотношения между операциями и операндами (зависимость числа операндов n2 от числа операций n1):

A = n2\* /(n2\* +2) \* log2(n2\*/2)

B = n2\* - 2 \* A

n2 = A \* n1 + B

6) Уровень программы:

L = V\*/V

где V\* - потенциальный объем, V - объем программы.

Альтернативное определение уровня L:

L' = (n1\*) \* n2/(n1\*N2)

где n1\*=2

Альтернативным методом определение уровня программы получили значение на 0.001 превышающее значение полученное в первом методе.

7) Интеллектуальное содержание:

I = L' \* V

8) Работа по программированию (общее число элементарных мысленных различий, требуемых для порождения программы):

E = V 2/ V\*

9) Приближенное время программирования:

T' = E/S

где S = 18 моментов (различий)/сек - постоянная Страуда.

T' =6882(сек)

Время программирования указанное в техническом задании – 1 час

10) Уровень языка:

A = L \* L\* V

Уровень языка определяет его производительность.

11) Уравнение ошибок:

Число переданных ошибок в программе: B = V/Е0

где E0 = V\* x V\* x V\*/(A x A) - среднее число элементарных различений между возможными ошибками в программировании.

 Исходя из полученных данных, можно предположить, что качество реализации алгоритма в большей степени определяется следующими параметрами:

\* интеллектуальное содержание

\* работа по программированию

\* приближенное время программирования

\* уровень языка

\* уравнение ошибок

*Описание метрик Холстеда*

Метрики Холстеда предлагают разумный подход к решению следующих задач:

- предсказание условий, необходимых для программирования по предложенным проектам;

- определение норм первоначальных ошибок;

- количественная оценка языков программирования и эффекта модульности;

- обоснование метода измерения различий между программами, написанными специалистами разного уровня.

 В основе вычисления метрик Холстеда лежит концепция, согласно которой алгоритм состоит только из операторов и операндов (проверяется рассмотрением простых вычислительных машин с форматом команд, содержащим две части: код операции и адрес операнда). Операнды - переменные или константы, используемые в данной реализации алгоритма. Операторы - комбинации символов, влияющие на значение или порядок операндов.

 В основе вычисляемых свойств алгоритма лежат следующие характеристики:

- n1 - число различных операторов данной реализации;

- n2 - число различных операндов данной реализации;

- N1 - общее число всех операторов;

- N2 - общее число всех операндов;

- n2\*- число различных входных и выходных операндов

На основании приведенных выше характеристик вычисляются:

словарь n = n1 + n2;

длина реализации N = N1 + N2.

объем программы V=N\*log2(n) (бит).

Под битом подразумевается логическая единица информации - символ, оператор, операнд.

Далее М. Холстед вводит n\* - теоретический словарь программы, т.е. словарный запас, необходимый для написания программы, с учетом того, что необходимая функция уже реализована в данном языке и, следовательно, программа сводится к вызову этой функции. Например, согласно М. Холстеду, возможное осуществление процедуры выделения простого числа могло бы выглядеть так:

- CALL SIMPLE (X,Y),

- где Y - массив численных значений, содержащий искомое число X.

- Теоретический словарь в этом случае будет состоять из

- n1\* : {CALL, SIMPLE (...)},

- n1\*=2; n2\* : {X, Y},

- n2\*=2,

- а его длина, определяемая как

- n\* = n1\* + n2\*,

- будет равняться 4.

- Используя n\*, Холстед вводит оценку V\*:

- V\* = n\* \* log2 n\*, (3)

с помощью которой описывается потенциальный объем программы, соответствующий максимально компактному тексту программы, реализующей данный алгоритм.