УДК 519.8:004.414.23+004.94:004.413.4+004.412

В.О. МИЩЕНКО

Харьковский национальный университет имени В.Н. Каразина, Украина

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ХАРАКТЕРИСТИК СХЕМ ПРОГРАММНЫХ СИСТЕМ

Одним из подходов к оценке качества производства программной продукции является использование научных метрик Холстеда и их обобщений – энергетических мер. Некоторые вопросы интерпретации этих характеристик (определённых в терминах исходных текстов программ) оставались не вполне ясными, в особенности, – трактовка уровня языка программирования (УЯП). Предполагалось, что выход состоит в накоплении и обновлении подходящего объёма статистических данных. Мы на примере вопроса об УЯП показываем, что сама по себе статистика реальных проектов здесь не поможет. Нами разработан метод компьютерного моделирования на базе имитационной математической модели, параметры которой берём, исходя из известной статистики и правдоподобных гипотез. Такое моделирование помогло объяснить природу УЯП и его прогностические возможности.

Ключевые слова: *компьютерный эксперимент, математическая модель, научные метрики Холстеда.*

Научные, энергетические метрики, схемы программных систем

Модель качества программной продукции в зависимости от её масштаба и ответственности может опираться на полную систему или подсистему характеристик надёжности, закреплённых в своё время стандартом IEEE 982 (см. [1]). Для программного обеспечения (ПО) компьютерного моделирования физических процессов, создаваемого в рамках научных исследований, характерна тенденция к максимальному снижению затрат на менеджмент качества. Поэтому, принимая во внимание невысокую, как правило, в данной области критичность по отношению к отказам в реальном времени и отсутствие сложных программно-аппаратных решений, рекомендуем для таких приложений использовать 4-е из 9 характеристик. Это «Оставшееся количество дефектов» и «Сложность» для продукции и для процесса – «Риски; преимущества, стоимость», «Управление менеджментом». Причём последнее только в том отношении, что относительно любого программного проекта нужно представлять, в какой мере его разработчики в состоянии выполнять поставленные перед ними задачи его реализации.

Привлекательны в связи с этими ограничениями научные метрики Холстеда [1] для простых (состоящих из информационно крепких модулей) программ и энергетические меры (метрики) для сложных [2]. Действительно, они очень экономиичны, основываясь на показателях, определяемых (бычно автоматически) на основании анализа исходных текстов программ или – при прогнозировании – по проектной документации. В данной работе подразумевается система энергетических мер, которая обобщает холстедовскую, имея свои особенности в интерпретации метрик, более сложные алгоритмы их оценивания и модифицированную систему примитивов (из соображений адекватности современным реалиям разработки ПО).

Набор мини энергетических метрик (размерность всех, кроме первой, символ×бит) включает:

B – потенциал *привнесённых ошибок* кодированиия – вероятное среднее число ошибок, допускаемых реализаторами программ, подобных данной по объёму исходного текста (безразмерна);

E – *спецификационная энергия* – мера сложности, исходящая из структуры программы и подсчёта для каждого из её мелкомасштабных элементов («блоков») η_2^* – числа формальных параметров, определяемых по числу каналов взаимодействия с окружением при своём выполнении;

A – *работа программирования* – мера затратности кодирования, исходящая из объёмов исходных текстов и формальных параметров блоков;

λ – *уровень языка* программирования – мера, отражающая эффективность языка для кодирования модуля, программы или множества программ.

На множествах модулей уровень языка получается осреднением. Было принято использовать среднее арифметическое [3, 4], но мы это еще обсудим.

Важнейшим условием расчёта указанных метрик является идентификация *схемы программной системы* (СПС). Основа СПС – это представление реальной системы, состоящей из файлов, программных модулей, их оформленных частей и т.п., в фор-

ме системы модулей верхнего уровня, внутренних блоков этих модулей (описанные в модулях функции, например) и внутренних групп модулей (если такие можно выделить в программе по формальным признакам, скажем, классы). При этом условии примитивами для расчёта указанных метрик являются:

длины N всех программных модулей, измеренные в *программных символах,* обычно соответствуют лексемам данного языка программирования и их легко идентифицируемым сочетаниям [2];

словари η всех модулей – мощности множеств использованных программных символов;

количества p *параметров* (всех видов), которыми снабжены блоки и модули;

количества j_1, j_2 *файлов* и *операций ввода-вывода*, использованные в телах блоков и модулей.

Определения, формулы данных метрик, были даны в [5], а подробности расчёта – в [2].

Другие стандартные метрики не образуют подобных групп, обслуживающих разнообразные характеристики на небольшой общей базе примитивов. Следовательно, самой экономичной минимальной моделью качества программной продукции является на сегодня система энергетических мер. Но, как обстоит дело с надёжностью такой модели качества, куда не включены дублирующие метрики, которые бы использовали другие примитивы?

1. Проблема статистического обоснования интерпретации

Предсказательный (с неоцениваемой вероятностью успеха) характер выводов, которые основываются на научных или энергетических метриках, заставляет искать подтверждения надёжности этих выводов в статистике программных проектов. Это было проделано М.Холстедом и его учениками [3] с успехом, к которому трудно добавить что-то существенное. В дальнейших исследованиях (например, [4, 6, 7]) в целом подтвердились следующие результаты. Первое – статистическая зависимость, известная как уравнение длины программы. В нашей системе примитивов оно получает форму линейной корреляции между N и $\eta \log \eta$. Второе – линейно корреляционная зависимость между A и E для модулей, если на больших выборках рассматривать все величины как случайные. При этом для конкретного модуля и, в особенности для программ в целом, соотношение между A и E не случайно, а отражает особенности разработки [8].

Из проблем, которые в науке о программах Холстеда оставляли вопросы, своей важностью выделяется интерпретация уровня языка. Для программного модуля

$$\lambda = \frac{\left((2+\eta_2^*)\log(2+\eta_2^*)\right)^2}{N \log \eta} = \frac{V^{*2}}{V}, \qquad (1)$$

где N, η и η_2^* определены выше, V и V^* называются соответственно объёмом и потенциальным объёмом модуля.

Современные Холстеду исследователи (70-е годы прошлого века) сочли статистические данные по распределению величин (1) «противоречивыми» [4], так как разброс значений при фиксированном языке велик, области значений для языков перекрываются. Тем не менее, средние (1) для языков программирования, которые определённо считаются имеющими более высокий уровень (в сравнении с другими) оказывались больше (чем у тех, других).

В энергетическом анализе величину (1) логично модифицировать так [2], чтобы она стала чувствительна к межмодульным связям в программах:

$$\lambda = \lambda_W = \frac{V^{*2}}{\sqrt{W^2 + W_B^2}}, \qquad (2)$$

где W – объём разработки модуля (его объём V с учётом определённого расширения словаря + объёмы интерфейсных модулей, от которых зависит его создание, как процесс во времени [2]);

W_B – объём разработки тела модуля, если он интерфейсный, а иначе полагается нулём.

Научное значение метрики уровня языка состоит в том, чтобы служить представительной характеристикой этого качества. А техническое значение – в предсказательной оценке работы программирования A. В силу выше упомянутой корреляции значений, эту величину для будущей программы можно по Холстеду приравнять её спецификационной энергии. Энергия E оценивается, если оценены η_2^* всех блоков и дан УЯП, который определяется по прежним оценкам величины (1).

Если в научной роли логичность замены (1) на (2) очевидна, то в отношении технической роли нет.

Изучение и сравнение стабильности измеренных величин (1) и (2) по исходным текстам модулей реальных программных систем на разных языках показали [8, 9], что их значения могут варьироваться от 0.1 (и меньше) до 10 (и больше). Это согласуется с данными [3, 4] четверть вековой давности по тогдашним языкам высокого уровня, только масштаб вариаций возрос еще больше. Рассмотрение причин, по которым УЯП на конкретных модулях отклоняется в десятки раз от средних значений, привело к такому выводу. Характер программы (вычисления, WEB приложения и т.п.), функции модуля в программе (ввод-вывод, преобразования массивов и т.п.), способ взаимодействия с другими модулями

и, впрочем, отчасти, язык и стиль разработчика определяют характерный статистический портрет этого модуля (много ли внутри подпрограмм, операторов тела и т.п.). Этот портрет непосредственно (и предсказуемо) определяет рамки, в которых будут заключены необходимые примитивы для расчёта метрик (1), (2), а, значит, и числовые значения этих мер.

Парадокс в том, что, не смотря на высокую актуальность определения ориентировочных значений метрик уровня языка, шансов на это тем меньше, чем больший объём статистических данных привлекается путем механического добавления результатов оценок новых систем! Действительно, ни о какой статистической однородности таких совокупностей не может быть речи, даже, как обнаруживается, если изучаемые системы принадлежат к одной и той же прикладной области в общепринятом смысле.

2. Постановка задачи имитационного моделирования

Нами реализуется та идея, что, имея в виду количественное измерение для научных или технических целей таких характеристик, как УЯП, можно задать профиль – характерный статистический портрет программной системы и провести на компьютере имитацию состава такой системы. Точнее, – рандомизованную генерацию её СПС.

Целью представляемой здесь работы являлась разработка метода и инструмента компьютерного моделирования оценок научных и энергетических метрик виртуальных анасамблей СПС для решения задач подбора нормативных значений или предсказания качественных характеристик проектов.

Для этого требовалось построить математическую модель генерации СПС, реализующую предполагаемые тенденции разработки, реализовать эту модель в форме программы для компьютерного моделирования, испытать результаты такого моделирования в процессе решения актуальной задачи.

3. Основные решения в математической модели

Построенная математическая модель определяется набором случайных величин, имеющих свою интерпретацию в отношении программных систем. Это такие, как количество модулей в СПС, количества интерфейсных модулей и тел таких модулей в СПС, количества самостоятельных реализующих модулей, групп реализующих блоков в форме субмодулей и многие другие. Модель также имеет средние значения для числа параметров настройки шаблонов, числа формальных параметров у процедур и функций и т.п. Упомянутые средние служат в качестве параметров для соответствующих случайных величин числа параметров настройки или числа формальных параметров процедур и функций конкретных модулей и т.п., но сами получаются предварительным розыгрышем значений случайных величин. При этом существенно следующее:

– случайные величины модели независимы и имеют разные функции распределения, учитывающие закономерности, на которые указывают статистические исследования реальных программ (как, например, [10, 11]), используется параметризация;

– соблюдаются (вплоть до поправок или нормировки уже разыгранных величин) все необходимые соотношения для реальных СПС. Например, число тел интерфейсных модулей не может превосходить числа таких модулей, дочерний модуль не может иметь в своём списке импорта собственный родительский модуль, длины модулей, блоков не могут быть слишком малы при наличии у них большого числа параметров или операций ввода-вывода;

– параметры модели, которые могут существенно сказываться на оценках виртуальных (сгенерированных на основе случайных значений) СПС, сделаны управляющими.

Построенная модель не обошлась без эмпирических упрощающих соотношений там, где для этого имелись надёжные соображения. Даже при этом её полное описание перекрыло бы объём статьи. Ограничимся пояснением указанных положений.

Использованы 3 типа распределений:

– равномерное дискретное (на заданном диапазоне целых) и равномерное непрерывное (на отрезке);

– дискретное унимодальное распределение $F = F_{M,R}$ на диапазоне целых от 0 до заданного максимума R при заданной моде M;

– дискретное на заданном диапазоне целых значений распределение G, параметр $Q \in [0;1]$ которого определяет L-образное распределение с заострением пика, когда Q приближается к 0.

Равномерные распределения вменяются случайным величинам тогда, когда соответствующим параметрам СПС нет оснований приписывать какую-либо тенденцию, кроме ограничений на диапазон значений. Распределение F сконструировано для отражения тенденций, типичных для таких параметров, как число модулей в СПС из определённой предметной области. Его мода и медиана равны или близки между собой, нулевое значение допустимо, но вероятность малых значений при $M \neq 0$ сходит на ноль, как и значения вблизи максимума R, но при этом можно смоделировать длинный хвост гис-

тограммы. В варианте $M = 0$ максимум гистограммы распределения смещён на левый конец, но размыт.

Распределение G тоже ориентировано на характерные для ПО закономерности. Из обширных статистических данных [10] для таких элементов программ, как параметры, следует, что их число в случайно выбранной конструкции с большой вероятностью окажется скорее на левой стороне дискретного диапазона целых значений, чем на правой. Схематично алгоритм генерации значений таков. Пусть g – случайная величина, для которой с вероятностью $1 - Q$ возможен исход «левый» и с вероятностью Q «правый». Разыграв значение g, выбираем левый или правый поддиапазон значений G, и по отношению к нему снова применяем описанню процедуру выбора левого или правого поддиапазонов, и т.д., пока не останется одно значение.

Наиболее существенных управляющих параметров модели около десятка. По убыванию степени определённости, которую следует ждать для данного параметра при моделировании, и предполагаемой степени силы воздействия на результат, выделены параметры первой, второй и третьей очереди. Для имитации ансамбля СПС (виртуальной выборки проектов из некой предметной области) параметры первой очереди задаются, а для остальных – умалчиваемые значения. При этом автор ориентировался на ПО вычислительного и учебного назначения, разработанное на языке Ада. Профиль генерируемых СПС можно существенно изменить, в том числе, имитируя особенности программ, характерные для других языков, если явно задавать параметры второй, а, тем более, – третьей очереди.

4. Разработанный инструмент компьютерного моделирования

Средство компьютерного моделирования разработано в форме программы EA-SPS-Simulator на языке Ада, и состоит в первом издании из 12 компилирумых модулей. Они реализуют интегрированные в библиотеке EA подсистемы генерации СПС (EA.SPS, 4 модуля, включая главную процедуру), оценки энергетических метрик (EA.Evaluate, 2 модуля), подробного статистического анализа (EA.Statistic, 2 модуля), и спецбиблиотеку случайных величин (EA_Random, 3 модуля, включая демонстратор). Назначение – моделировать на компьютере оценивание многих СПС одного профиля.

Профиль можно задать в 4-х режимах. Первый – default (выбранный в диалоге, представленном на рис. 1) – позволяет только варьировать размер ансамбля СПС, имитирующих вычислительные и учебные приложения языка Ада (программы, состоящие из десятков компилируемых модулей, в которые могут входить десятки процедур, функций и прочих важных для энергетического анализа конструкций). Второй режим – custom – также ориентирован на быстрое проведение большого числа компьютерных экспериментов, но позволяет варьировать число модулей в генерируемых СПС, процент интерфейсных модулей и порядок числа внутренних блоков модулей. Режимы freely, my_old позволяют задать (с точностью до применяемой рандомизации) подробности структуры модулей и, частично, – блоков (т.е., подпрограмм и проч.).

```
Run: ea-sps-simulator
   *** **   Energy Analysis Simulator ** ***
               Issue 1: 2010-02-19
** Set the profile of software that will be tes
   Deside the mode:
   "defalt", "custom", "freely" or "my_old" -
defalt
   How many SPS do you plan to evaluate? -
5
   Will you put result to a new file ('Y','y')
   or not (others)? - y
** Simulation of 5 SPS :
   #  1  from 5 was generated and evaluated
   #  2  from 5 was generated and evaluated
   #  3  from 5 was generated and evaluated
   #  4  from 5 was generated and evaluated
   #  5  from 5 was generated and evaluated
** Computed results are in the file,
   which name is ea_simula_proj_defalt_009.ear
** Do you want to see statistics now?
 ('Y','y' -"yes", others - "no") - n
   *** **   Energy Analysis Simulator ** ***
               Put a Key to finish
```

Рис. 1. Пример диалога с программой, в котором генерируются и оцениваются 5 СПС

Пользователь не может влиять на эмпирические алгоритмы розыгрыша чисел формальных параметров блоков (т.е., подпрограммы), характер использования ввода-вывода в модулях, количеств программных символов в блоках СПС. Дело в том, что здесь, при всём произволе, требуется учитывать такие ограничения как «не более 4 – 7 элементов» в программных конструкциях [10], приближенное уравнение длины программы Холстеда (в упрощенной форме, см. [2]), не говоря о чисто «материальных» соотношениях, которые упоминались в разделе 3. Особую роль играет статистическая связь между спецификационной энергией E и работой программирования A. Она не задаётся прямо, а должна получиться при правильной настройке генерации. A и E на самом деле коррелируют между собой. М. Холстед указывал [3] множества простых программ и комплексов, на которых эта корреляция имеет форму приближенного равенство (и потому считал обе величины оценками одной и той же характеристики). Автор на примерах слож-

ных программ показал [5], что обсуждаемая корреляция может иметь форму линейной связи с другим наклоном прямой регрессии (и потому ввёл отдельную характеристику – спецификационную энергию). Для генераций в режиме defalt нами обеспечено, чтобы знаки разности $E - A$ были для СПС примерно равновероятны («закон Холстеда» в ослабленной форме). Всё это требует тонкой настройки за счёт согласованного сдвига нескольких технических параметров генерации СПС в допустимых для них по смыслу пределах. Пользователь, который не разбирался в этом специально, запустив программу, нужные назначения таких параметров не сделает. Поэтому, как было подчёркнуто, данная возможность для него закрыта.

Одновременно с генерацией СПС программа компьютерного моделирования определяет научные и энергетические метрики для модулей и СПС и помещает их в текстовый файл результат (рис. 2). На этой базе может быть проведена дальнейшая статистическая обработка в интересах содержательного анализа.

Например, сделав предположения о статистическом портрете будущей программной системы, можно оценить вероятность значений метрических характеристик (или соотношений между ними) для будущей программы.

5. Исследование УЯП на основе компьютерных экспериментов

Рассмотрим упоминавшуюся проблему УЯП.

Поведение мер λ и λ_w аналогично, но значения метрики λ (1) больше (в данном примере где-то в 2.5 раза). Это зависит от интенсивности межмодульных связей и процента тел интерфейсных модулей среди всех модулей. Уточним ситуацию на примере λ_w. В табл. 1 даны частости значений, отнесённые к единице длины оси при $n = 160$.

```
Energy analisys of SPS # 9 of  1 modules
  eta      N         V          W          V*         El          A          lam         lam_W
 MR 1
 159   1262   9.229E+03   9.229E+03   1.477E+01   8.688E+03   5.765E+06   2.365E-02   2.365E-02
=-----------------------
E = 8.688E+03  A = 5.765E+06
Lam = 2.365E-02
Lam_W = 2.365E-02
Energy analisys of SPS # 10 of  1 modules
  eta      N         V          W          V*         El          A          lam         lam_W
 MR 1
  14    104   3.960E+02   3.960E+02   5.076E+01   5.231E+05   3.089E+03   6.507E+00   6.507E+00
=-----------------------
E = 5.231E+05  A = 3.089E+03
Lam = 6.507E+00
Lam_W = 6.507E+00
========================
General Lam = 6.14E+00
General Lam_w = 3.56E+00
```

Рис. 2. Пример конечного фрагмента выходного файла (ea_simula_proj_C01–01_099.ear)

Таблица 1

Распределение значений УЯП в совокупности СПС

Интервал:		Частота значений	Плотность распределения
от	до		
0	0.5	30	0.3750
0.5	1	18	0.2250
1	1.5	25	0.3125
1.5	2	22	0.2750
2	2.5	9	0.1125
2.5	3	8	0.1000
3	3.5	7	0.0875
3.5	4.5	8	0.0500
4.5	6.5	12	0.0375
6.5	30.5	21	0.0098

Судя по плотности эмпирической функции распределения (табл. 1), у предельного ($n \to \infty$) распределения λ_w может не быть первого момента, котрый бы приближался средними арифметическим значениями табл. 2.

Таблица 2

Колебания оценок уровня языка

n =	5	10	20	40	80	160
λ	6.08	5.99	6.23	6.59	8.69	8.59
λ_w	1.61	2.60	3.51	2.78	2.97	3.21
n =	320	640	1280	2560	5040	20160
λ	9.61	9.29	9.44	9.95	10.1	9.68
λ_w	3.58	3.42	3.56	3.90	3.86	3.73

Аналогичное поведение наблюдалось при выборе профилей СПС, отличных от обозначенного у

нас, как default. Аналогично распределяются и значения УЯП по модулям СПС, в том числе в реальных проектах [9].

Поэтому использование метрик УЯП невозможно на основе арифметических средних, как в [3], а только на основе интервальных оценок. Так, по табл. 1, если проектируемая СПС предположительно имеет профиль defalt, то для неё частость

$$P(A \le 2.5E) = 0.65, \qquad (3)$$

тогда как частость повышенной трудоёмкости

$$P(A \ge 6.5) = 0.13. \qquad (5)$$

К «точечным» прогнозам следует относиться с крайней осторожностью.

Возьмём, скажем, $n = 320$ и допустим, что будущая система с профилем defalt имеет характеристики, совпадающие с одной из СПС виртуального ансамбля.

Каким должно быть значение λ_w, чтобы вероятность не менее 10-кратного просчёта оценки работы A по энергии E была заметно меньше 0,5? Наш инструмент моделирования позволяет прямыми пробами определить, что только при $0.7 \le \lambda_w \le 0.95$ эта вероятность не более 0,42 .

При этом менее чем 4-кратная ошибка наиболее вероятна на левом конце $\lambda_w = 0.7$, составляя 0,45. Ясно, что, доверившись УЯП для простых программ (среднее λ_w не менее 6) и даже общепринятым средним для λ_w (табл. 2), мы гарантировали бы катастрофическую ошибку.

Мы сопроводили наши выводы данными экспериментов имитации СПС одного профиля, но имитировались, например, и приложения Фортрана-77). Компьютерное моделирование ради сравнения уровней разных языков требует для достоверности более скрупулезного обращения со статистикой реальных проектов: для разных языков характерная структура программ сложным образом согласуется с их синтаксисом.

Заключение

Дана новая постановка и решение задачи об адекватности интерпретации метрик процесса разработки программ, которые развивают известный подход М. Холстеда.

Установлено, что при интерпретации и использовании метрики уровня языка для сложных программ целесообразна модификация формулы и отказ от средних в пользу интервальных оценок на основе компьютерного моделирования.

Создан инструмент компьютерного моделирования ансамблей программ заданного профиля для исследований статистики научных и энергетических метрик в теоретических и практических целях.

В будущем на этой базе можно изучать особенности разных языков и методов программирования.

Литература

1. Оценка качества и экспертиза програмного обеспечения. Лекционный материал / Под. ред. В.С. Харченко – НАУ ім.. Жуковського Н.Є. „ХАІ", 2008. – 204 с.

2. Мищенко В.О. Энергетический анализ программного обеспечения с примерами реализации для Ада–программ / В.О. Мищенко. – Х.: ХНУ им. В.Н. Каразина, 2007. – 119 с.

3. Холстед М.Х. Начала науки о программах / М.Х. Холстед. – М.: Финансы и статистика. – 1981. – 128 с.

4. Christensen K. A Perspective on Software Science / K. Christensen, G.P. Fitsos, C.P. Smith IBM Systems Journal. – 1981. – 20, N 4. – P. 372-387.

5. Мищенко В.О. Математическая модель стиля Software Science для метрического анализа сложных наукоёмких программ / В.О. Мищенко // Вісник Харк. нац. ун-ту. – 2004. – № 629, Серія «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління», вип. 3. – С. 70-85.

6. Felician L., Zalateu G. Validating Halstead's Theory for Pascal Programs / L. Felician, G. Zalateu // IEEE Transactions on Software Engineering. – 1989. – № 12. – P. 1630-1632.

7. Gonzales Michael G., Seshadri Paravastu Correction of the Software Science Length Estimator Skewness for 'C' Language Programs / Michael G. Gonzales, Paravastu Seshadri // Empirical Software Engineering. – 2000. – 5, Issue 2. – P. 155-159.

8. Мищенко В.О. Применение математического моделирования в системном анализе проекта программного обеспечения методом дискретных особенностей / В.О. Мищенко // Труды VII Международного симпозиума «Методы дискретных особенностей в задачах математической физики». – Феодосия. – 1997. – С. 117-120.

9. Mishchenko V.O. One Experiment in Using Energy Metrics Proposed for Software Process Assessment / V.O. Mishchenko // Радіоелектронні і комп'ютерні системи. – 2007. – № 8 (27). – С. 121-124.

10. Коган Б.И. Экспериментальные исследования программ / Б.И. Коган – М.: Наука, 1988. – 184 с.

11. Capers J. Applied Software Measurement: Assuring Productivity and Quality / J. Capers – New York: McGraw Hill, 1991. – 493 p.

Поступила в редакцию 25.02.2010

Рецензент: д-р техн. наук, проф., зав. кафедры ТПИ ММФ Г.Н. Жолткевич, Харьковский национальный университет имени В.Н. Каразина, Харьков.

КОМП'ЮТЕРНЕ МОДЕЛЮВАННЯ ХАРАКТЕРИСТИК СХЕМ ПРОГРАМНИХ СИСТЕМ

В.О. Міщенко

Одним з підходів до оцінки якості виробництва програмної продукції є використання наукових метрик Холстеда та їх узагальнень – енергетичних мір. Деякі питання інтерпретації цих характеристик (що визначені в термінах вихідних текстів програм) залишалися не зовсім зрозумілими, особливо – трактовка рівня язику програмування (РЯП). Передбачалося, що вихід міститься у накопленні та оновленні необхідного об'єму статистичних даних. Ми на прикладі питання про РЯП показуємо, що сама по собі статистика реальних проектів тут не допоможе. Нами розроблено метод комп'ютерного моделювання на базі імітаційної математичної моделі, параметри якої беремо виходячи з відомої статистики и правдоподібних гіпотез. Таке моделювання допомогло пояснити природу РЯП та його прогностичні можливості.

Ключові слова: комп'ютерний експеримент, математична модель, наукові метрики Холстеда.

COMPUTER MODELING OF SOFTWARE SYSTEM SCHEMES CHARACTERISTICS

V.O. Mishchenko

One approach to quality analysis of the software production is the use Halstead's scientific metrics and their generalizations – the energy measures. But some interpretation problems of these characteristics were unclear, in particular, the treatment of programming language level (PLL). It was assumed that the solution is in collecting and updating a suitable amount of statistical data. Using the PLL as an example, we illustrate that the statistics of actual projects will not help by itself. We have developed the computer modeling method based on the mathematical simulation model, whose parameters take on the basis of certain statistics and plausible hypotheses. Such modeling helps us to explain the nature of PLL and its predictive capabilities.

Key words: computer experiment, mathematical model, Halstead's scientific metrics.

Мищенко Виктор Олегович – канд. физ.-мат. наук, доцент, доцент кафедры моделирования систем и технологий, Харьковский национальный университет им. В.Н. Каразина», Харьков, Украина, e-mail: Victor.O.Mischenko@univer.kharkov.ua.