УДК 519.71:658.51:629.58

А. Н. ТРУНОВ

Черноморский государственный университет им. Петра Могилы, г. Николаев, Украина

ОЦЕНКА ЭФФЕКТИВНОСТИ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Теоретически обосновано и выведено с помощью унифицированного метода выражения показателя эффективности, которые одновременно учитывают четыре фактора: объем ПП, вероятность безотказной работы, стоимостные и временные затраты. Сформирована модель динамики дефектов при создании ПП как результат интерактивного взаимодействия разработчиков, тестеров, заказчиков, учитывающая такие параметры, как коэффициент функциональной полноты, степень стандартизации и унификации, степень логического покрытия тестами, вероятности ошибки алгоритма и кода, вероятности обнаружения и устранения дефектов. Продемонстрировано, что учет таких данных, как вероятность совершения и обнаружения ошибки в качестве статистических характеристик проектировщиков-разработчиков, тестировщиков, позволяет разбивать ПС на ряд елементов и приглашать для их создания команды с разной квалификацией и стоимостью работ за единицу времени при формировании стратегии программирования.

Ключевые слова: *показатель эффективности, программный продукт, модель динамики дефектов, показатели, характеризующие программный продукт, вероятности ошибки алгоритма и кода, вероятности обнаружения и устранения дефектов.*

Введение

Современные методы программирования характеризуются в последние десятилетия становлением и развитием объектно-ориентированного проектирования программных систем [1, 3]. Его совершенствование достигло такого уровня, что стало основой порождающего программирования, в котором сформулированы направления его развития и пути устранения ряда недостатков. Основываясь на современных методах программирования и моделирования, позволяет создавать качественные программные продукты с использованием средств автоматизации [1, 3]. В этой связи следует отметить, что процессы интеграции команд программистов при создании такого типа программного продукта (ПП) наблюдаются в последние три десятилетия практически во всех странах мира. Они и проявления экономического кризиса все острее ставят две задачи: разработка критериев оценки эффективности затрат на осуществление плана создания ПП; формирование методологии применения этих критериев анализа для эффективного выбора стратегии программирования. Следует заметить, что на сегодняшний день уже понятно: как для заказчика, так и для разработчиков одинаково важно сформировать, представить и выбрать на конкурсной основе качественную и экономически оправданную, конкурентоспособную альтернативу. Однако вместе с тем, несмотря на имеющиеся общие подходы к управлению технологическими процессами вообще, основывающиеся на роботах *Peter-a Ferdinand-a Drucker*-а, известные под названием «Система КПЭ», а также классическая система «Управление по целям», которая включает в себя множество управленческих концепций, дополняющих ее, и, наконец, на положениях стандарта ISO 9000:2008 [4-7], существует два подхода. Каждый из них предлагает принципиально отличные оценки активности и эффективности. В этой связи, создание универсального инструмента для формирования критериев оценки эффективности такого вида деятельности, как программирование, приобретает актуальную и практически важную задачу. На сегодняшний день уже в литературе описаны и получили апробацию методы финансового анализа ИТ-проектов, такие как TCO, ROI, TVO, VCO [8-13]. С их помощью рассчитывают стоимость проекта в целом, срок возврата инвестиций или производят более широкий анализ, учитывающий не только видимый экономический эффект, но и новые возможности для развития бизнеса. Позитивным при этом является то, что они позволяют количественно обосновать руководству компаний, что ключевой фактор увеличения прибыли обеспечивается увеличением оборота и доходов за счет применения ИТ-технологий, а не за счет уменьшения расходов их менеджеров. Перестало требовать у ИТ-менеджеров сокращения расходов, а фокусируется на увеличении прибыли компании за счет ИТ. По опросу журнала CIO Insight [8],

© А. Н. Трунов

84 % ИТ-директоров рассчитывают ROI при планировании проектов, 46 % проверяют свои расчеты после окончания проекта и 68 % уверены в том, что расчет ROI положительно сказывается на планировании ИТ-проектов. Однако несмотря на такие оценки, они не применяются широко. Одними из причин являются: необходимость учета рисков; повторный учет приложений; трудность учета непрямой прибыли [14-19]. Таким образом, выбор метода формирования показателей эффективности, при создании ПП в целом и его отдельных модулей в частности, является главной нерешенной задачей, требующей теоретического обоснования. Второй, органически связанной с первой, является задача построения модели процесса динамики дефектов ПП как результат интерактивного взаимодействия специалистов трех категорий: постановщиков; программистов; тестировщиков.

Целью данной статьи является обоснование методики и вывод фактических выражений для расчета эффективности технологического процесса программирования. Детализация и апробация методики определения эффективности с учетом параметров оценки качества и вероятности успешной реализации процесса создания ПП, учитывая динамику дефектов и других характеристик программного продукта, является другой задачей данного исследования.

1. Постановка задачи выбора показателя эффективности

Предположим, как это показано в работах [8; 9], что для анализа эффективности необходимо учитывать n факторов, которые могут быть представлены n мерным вектором $\overline{X}$. Далее, основываясь на гипотезе о влиянии количества подпространств на выбор метрики, введем норму

$$\left\|\overline{X}\right\| = \left(\sum_{i=1}^{n}\left|\overline{X}\right|_i^n\right)^{1/n},$$

тогда для случая четырех равнозначных измерений определяющих факторов будем иметь в соответствии с геометрическим неравенством после простых алгебраических преобразований:

$$\overline{E} = \frac{1}{4}\left(\left|\overline{G}\right|^4 + \left|\frac{1}{\overline{C}}\right|^4 + \left|\frac{1}{\overline{T}}\right|^4 + |P|^4\right) \geq \left|\overline{G}\right|\left|\frac{1}{\overline{C}}\right|\left|\frac{1}{\overline{T}}\right||P|,$$

где обозначено относительные величины: количественно измеряемый результат технологии $\overline{G}$, полученный за промежуток времени $\overline{T}$, суммарные затраты на создание технологии программирования оцениваются величиной $\overline{C}$, при этом предположим, что вероятность правильной реализации ПП – P. Масштабом каждой из величин являются их наибольшие значения. Для случая, когда влияния различных факторов не равнозначны, а определяются коэффициентами веса, то наименьшее значение показателя эффективности будет вычисляться на основании геометрического неравенства

$$\left(\left|\overline{G}\right|^4 + \left|\frac{1}{\overline{C}}\right|^4 + \left|\frac{1}{\overline{T}}\right|^4 + |P|^4\right) \geq \left(\frac{1}{k_1}\right)^{k_1}\left(\frac{1}{k_2}\right)^{k_2} \times$$

$$\times\left(\frac{1}{k_3}\right)^{k_3}\left(\frac{1}{k_4}\right)^{k_4}\left|\overline{G}\right|^{4k_1}\left|\frac{1}{\overline{C}}\right|^{4k_2}\left|\frac{1}{\overline{T}}\right|^{4k_3}|P|^{4k_4}.$$

Таким образом, в силу свойств степенной функции и коэффициентов веса, полученное выражение позволяет оценить верхнюю и нижнюю границу показателя эффективности

$$\left|\overline{G}\right|\left|\frac{1}{\overline{C}}\right|\left|\frac{1}{\overline{T}}\right||P| \leq \overline{E} \leq 4\left|\overline{G}\right|\left|\frac{1}{\overline{C}}\right|\left|\frac{1}{\overline{T}}\right||P|.$$

В связи со значительно выросшим в последнее время числом задач, решаемых программным продуктом, и все большей сложностью каждой из них, а также их вычислительным объемом, проекты создания программных систем характеризуются большими величинами временных и стоимостных затрат, а для их реализации все чаще используется принцип написания ПО командой в виде отдельных функциональных модулей, структур, компонентов и т. д. [16-19]. Этот подход имеет свои преимущества, поскольку за счет одновременного написания программ отдельных иерархических структур, структурных элементов, в том числе и модулей как логически законченных частей программы, значительно ускоряется процесс программирования в целом и уменьшается время полного завершения проекта. Сокращение периода разработки и тестирования проекта существенно, даже несмотря на то, что за счет реализации технологии построения синтезированного программного обеспечения, наблюдается значительное увеличение общего числа ошибок, а для их нахождения и переписывания программы неизбежны дополнительные затраты времени.

2. Формирование модели процесса динамики дефектов ПП

Полученный результат, является особенно важным для информационных технологий, поскольку современный подход к созданию ПП базируется на принципе обязательного написания тестов для каждой логически связанной группы кода, некоторые из которых могут объединяться в отдельный смысловой элемент. Как показывает опыт применения инструментов Test – Driven Development, Continues Integration или других – эффективными, с точки зрения возможностей общения представителей

различных групп постановщиков, программистов, тестировщиков, являются инструменты Continues Integration. Однако степень полноты логического тестирования как один из параметров, который при этом применяется и позволяет проводить количественную сравнительную оценку ПП, является только косвенным показателем качества [19] и, тем более, эффективности. В этой связи, с учетом обоснованного выше понятия эффективности и предложенного метода оценки его нижней и верхней границы, использование четырех факторов позволит построить выражение критерия эффективности. Рассмотрим процесс создания информационной технологии, в результате применения которой получен программный код объемом функций V_d за промежуток времени T, а суммарные затраты на создание технологии программирования оцениваются величиной C_s, при этом предположим, что вероятность правильной реализации ПП – P_d – определяется вероятностью возможных отказов или наличия дефектов R_d, приводящих к ложным результатам, известны, тогда минимальная оценка (нижняя граница) эффективности технологического процесса создания ПП определится

$$E \geq \frac{V_d}{C_s T} P_d = \frac{V_d}{C_s T}\left(1 - R_d\right).$$

Применение такого подхода к оценке эффективности действующей информационной технологии требует оценки результата ее действия. Предположим, что результат оценивается измеряемым объемом данных V_d, с доверительной вероятностью достоверности содержащейся в нем информации P_d для заданного доверительного интервала, с приведенными затратами C_s, т. е. учитывающими затраты на создание, эксплуатацию и амортизацию, а также времени достижения и визуального представления в необходимой форме данного результата T.

Таким образом, оценка эффективности информационной технологии определяется качеством алгоритма, надежностью написанного и стоимостью лицензионного программного обеспечения (ПО), типом выбранного аппаратного обеспечения. В последнее время в связи с большими по величине объемами операций и их сложностью, и большими величинами временных затрат на написание программ все больше используется принцип командного написания ПО в виде отдельных функциональных модулей [16-19]. Такой подход имеет свои преимущества, что обусловлено существенным ускорением и уменьшением общего времени на выполнение проекта в целом. Однако увеличенное количество дефектов, возникающих, как правило, вследствие реализации такой технологии построения синтезированного ПО, обычно требует больше времени для их нахождения и на переписывание программы. Динамика процесса устранения дефектов и влияние на нее различных факторов исследовалась в работах [17-18]. Последнее позволяет исследовать динамику оценки эффективности во времени для информационных технологий с синтезированным ПО, определить вероятности отказов, а, следовательно, и эффективность. Введем понятие степени функциональной полноты модуля программы как отношение количества задач, решаемых при работе модуля к общему количеству задач, сформулированных в соответствии с техническим заданием, которое сформулировано в виде требования для данного ПО

$$f_i = \frac{m_i}{M};\ \ M = \sum_{i=1}^{I} M_i;\ \ i = \overline{1,I};\ f_i \in [0,1],$$

где обозначено f_i – степень функциональной полноты модуля программы, m_i – количество решаемых модулем задач, M_i – общее количество задач, которые решаются модулем в соответствии с техническим заданием; M – общее количество задач, которые решаются ПО в целом, I – количество модулей. В соответствии с данным определением, степень функциональной полноты изменяется от нуля до единицы. Нулевое значение показателя соответствует не работающему в программе модулю, а единица – модулю, который полностью дублирует все функции модулей программы.

Введем также определение степени функциональной избыточности программы как отношение разности суммарного количества, решаемых задач и общего количества задач, что подлежат решению к самому же их общему количеству, т. е.

$$a = \frac{\sum_{i=1}^{I}\left(m_i - M_i\right)}{\sum_{i=1}^{I} M_i};\ i = \overline{1,I};\ a \in [0, I-1].$$

Этот показатель изменяется от нуля до величины на единицу меньше чем количество модулей для вариантов проекта, удовлетворяющего техническим требованиям. Ноль соответствует факту отсутствия дублирования функций других модулей, а второе крайнее значение – факту полного дублирования для всех модулей одновременно. Отрицательные значения показателя соответствуют модулю, в котором не удовлетворены требования технического задания или в котором они неправильно сформулированы.

Введем также определение степени стандартизации S_k в классе данного ПО как отношение количества стандартных подпрограмм к общему количеству подпрограмм в данном ПП

$$S_k = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J} SP_{ij}^{s}}{\sum_{i=1}^{I}\sum_{j=1}^{J} SP_{ij}};\quad i=\overline{1,I};\quad j=\overline{1,J},$$

где SP_{ij} – количество подпрограмм в j-той задаче i-того модуля, а SP_{ijk}^{s} – количество стандартных подпрограмм в j–той задаче i–того модуля.

Введем также определение степени унификации подпрограммы в классе данного ПО как отношение количества раз использования k-той подпрограммы при решении j-той задачи и при решении других задач в данном модуле и данном ПО к общему количеству подпрограмм

$$U_k = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} SP_{ijk}^{u}}{\sum_{i=1}^{I}\sum_{j=1}^{J} SP_{ij}};\ i=\overline{1,I};\ j=\overline{1,J};\ k=\overline{1,K},$$

где SP_{ijk}^{u} – количество раз использования k-той подпрограммы в j-той задаче i-того модуля. Оба эти показателя изменяются от нуля до единицы, при этом ноль определяет полное отсутствие стандартных программ, а единица – полное отсутствие новых написанных подпрограмм. Показатели стандартизации и унификации характеризуют уровень развития и опыт работы программиста, естественно и предопределяют в большей мере вероятность безотказной работы, а, следовательно, и качество ПП. Введем величину L – степень логического покрытия тестами как отношение суммарного количества тестов к общему числу логически связанных структурных групп кода подпрограмм

$$L = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} Q_{ijk}}{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} O_{ijk}SU_{ijk}};\ i=\overline{1,I};\ j=\overline{1,J};\ k=\overline{1,K},$$

где SU_{ijk} – количество логически связанных структурных групп кода, O_{ijk} – количество состояний выходной переменной, а Q_{ijk} – количество тестов, для проверки той же k-той подпрограммы при решении j-той задачи, решаемых i-тым модулем.

Таким образом, в соответствии с введенными понятиями становится возможным формулировка задач проектирования модулей программы как задачи максимизации степени унификации и стандартизации с заданными показателями функциональной полноты и избыточности.

3. Моделирование и выбор стратегии программирования

Обозначим вероятности: R_{aijk} – ошибки формирования алгоритма решения задачи; R_{cijk} – ошибки, возникающие при написании кода; R_{ijk}^{u} – вероятность ошибки в унифицированной подпрограмме; R_{apijk} – обращения к данной подпрограмме. Тогда вероятность успешной реализации программы

$$P_d = \begin{cases} 1, & \text{if} \quad c \ge 1, \\ c, & \text{if} \quad c < 1, \end{cases}$$

где

$$c = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} P_{ajk}P_{kjk}P_{apjk}SP_{ijk}(1-S_{ijk})\left\{1+U_{ijk}\frac{P_{ijk}^{u}}{P_{apjk}}\right\},$$

$$P_{aijk} = 1-R_{aijk},\ P_{cijk} = 1-R_{cijk},$$

$$P_{apijk} = 1-R_{apijk},\ \ P_{ijk}^{u} = 1-R_{ijk}^{u}.$$

Последнее соотношение демонстрирует, что в процессе создания программы решающую роль играет вероятность отсутствия или наличия ошибочных концепций, алгоритмов, информационных посылов баз данных и, конечно, ошибки при написании кода. Оценка доверительного интервала или доверительной вероятности при наличии данных о наблюдении дефектов при обработке ограниченного числа опытов сводится к известной задаче статистики. В этой связи формирование модели динамики дефектов является актуальной и самостоятельной задачей статистики, неразрывно связанной с задачей определения эффективности ПП.

Для ее постановки и решения обозначим количество дефектов, что содержится в k-той подпрограмме j-той задачи i-того модуля. Как показано в работе [18], приращение дефектов в процессе интерактивного взаимодействия разработчика алгоритма, программиста, тестировщика может быть представлено в виде модели:

$$\begin{cases} dN_{ijk} = \alpha_{ijk}N_{ijk}\psi_{ijk}(t)\theta_{ijk}(t)dt + \\ \quad + \sum_{q=1,q\ne k}^{K} \alpha_{ijq}N_{ijq}\psi_{ijq}(t)\theta_{ijq}(t)dt\ ; \\ t=0,\ \ N_{ijk} = N_{0ijk}; \\ i=\overline{1,I};\ \ j=\overline{1,J};\ \ k=\overline{1,K};\ \ q=\overline{1,K}. \end{cases}$$

В данной модели использованы следующие обозначения: t – текущее время; N_{ijk} – количество деффектов; α_{ijk} - коэффициент пропорциональности физический смысл, которого относительная скорость устранения дефектов; $\psi_{ijk}(t)$ – зависимость

вероятности обнаружения дефектов от времени; $\theta_{ijk}(t)$ – зависимость вероятности устранения дефектов от времени. Естественно необходимо отметить, что указанные величины записаны, соответственно, для k-той подпрограммы j-той задачи i-того модуля. Также необходимо указать, что индекс q пробегает значения, указывающие номер подпрограммы j-той задачи i-того модуля. Для k-того уравнения не должен принимать значение $q = k$. Также для стандартных программ $\alpha_{ijk} = 0$. Выразим начальное число дефектов N_{0ijk} в начальный момент времени через общее число записей – структурных элементов кода N^{w}_{ijk} для k-той подпрограммы j-той задачи i-того модуля и вероятность ошибки алгоритма и написания кода

$$N_{0ijk} = N^{w}_{ijk}\left(1 - P_{ajk}\right)\left(1 - P_{kjk}\right),$$

причем следует заметить, исходя из физического смысла величин, должно выполнятся условие нормировки

$$\frac{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} N^{w}_{ijk}}{V_d} = 1; \quad i = \overline{1,I}; \quad j = \overline{1,J}; \quad k = \overline{1,K}.$$

Таким образом, решение системы дифференциальных уравнений опишет динамическое поведение дефектов в программе.

Стоимостная и временная оценка затрат на создание ПП с учетом затрат на создание набора тестов для логической, качественной и количественной апробации представится:

$$\tilde{N}_s = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left\{C_{ijk}\left[C^{wp}_{ijk} + LC^{wt}_{ijk} + LC^{pt}_{ijk}\right]\right\};$$

$$i = \overline{1,I}; \quad j = \overline{1,J}; \quad k = \overline{1,K};$$

$$G_{ijk} = \left(1 - s_{ijk}\right)\left(1 - a_{ijk}\right)\left(1 - u_{ijk}\right);$$

$$T = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left\{C_{ijk}\left[T^{wp}_{ijk} + LT^{wt}_{ijk} + LT^{pt}_{ijk}\right]\right\};$$

где C^{wp}_{ijk}, C^{wt}_{ijk}, C^{pt}_{ijk} – обозначено, соответственно, затраты в денежном выражении на написание кода, тестов и их реализация в период тестирования, и отладки для k-той подпрограммы j-той задачи i-того модуля, а T^{wp}_{ijk}, T^{wt}_{ijk}, T^{pt}_{ijk} – обозначено аналогичные затраты во временном выражении, a_{ijk}, s_{ijk}, u_{ijk} – степень избыточности, стандартизации и унификации для k-той подпрограммы j-той задачи i-того модуля. Не менее важным является сформировать затраты стоимостных и временных ресурсов в период эксплуатации ПП, последние формируются с учетом стоимостных характеристик оборудования лицензионного ПО и коэффициентов их амортизации.

Выводы

1. Теоретически обоснованно и выведено на основании унифицированного метода выражение показателя эффективности, учитывающее одновременно четыре фактора: объем, вероятность безотказной работы, стоимостные и временные затраты.

2. Сформирована модель динамики дефектов при создании ПП как результат интерактивного взаимодействия, учитывающая такие параметры, как коэффициент функциональной полноты, степень стандартизации и унификации, степень логического покрытия тестами, вероятности ошибки алгоритма и кода, вероятности обнаружения и устранения дефектов.

Литература

1. Буч, Г. Объектно-ориентированный анализ [Текст] / Г. Буч. – М. : Бином, 1998. – 560 с.

2. Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. –СПб. : Питер, 2001. – 368 с.

3. Чернецки, К. Порождающее программирование. Методы, инструменты, применение [Текст] / К. Чернецки, У. Айзенекер. – М. Спб. : Изд. дом «Питер», 2005. – 730 с.

4. Панов, М. М. Оценка деятельности и система управления компанией на основе KPI [Текст] / М. М. Панов. – М. : Инфра-М, 2012. – 255 с.

5. Клочков, А. К. KPI и мотивация персонала. Полный сборник практических инструментов [Текст] / А. К. Клочков. – М. : Эксмо, 2010. – 160 с.

6. Parmenter, David. Key Performance Indicators: Developing, Implementing and Using Winning KPI`s [Text] / D. Parmenter. – New Jersey, USA : John Wiley & Sons, inc., 2007. – 233 p.

7. Федоров, Ю. Н. Справочник инженера по АСУТП: проектирование и разработка [Текст] / Ю. Н. Федоров. – М. : Инфа-Инженерия, 2008. – 928 с.

8. «ROI: How Well Do You Work with the Business» [Electronic source] / By Editors of CIO Insight, CIO Insight, Research, April 1, 2004. – 20.03.2014.

9. «ROI Case Study: Microsoft Business Solutions Matrix Packaging Machinery» [Electronic source] / Nucleus Research, RESEARCH NOTE D79. – 12.02.2014.

10. «The Real ROI from SAP» [Electronic source] / Nucleus Research, RESEARCH NOTE E36, April 2004. – Mode of access: http://www.nucleusresearch.com/research/d23.pdf. – 20.05.2014.

11. «What`s Wrong With Application Software? Businesses Really Are Unique – One Size Can Never Fit All» [Electronic source] / Olin Thompson, February 10, 2003. – Mode of access: http://TechnologyEvaluation.Com. – 11.02.2014.

12. «A Guide to Maximizing the ROI of Web Applications» [Electronic source] / M7 Corporation, July 2003. – 20.03.2014.

13. Terry Theisen «Proven Business Value of the Borland Application Lifecycle Management Solution» [Electronic source] / A Consynity white paper Principal. – 2.02.2012.

14. Трунов, О. М. Розвиток методів оцінки ефективності систем управління роботизованими комплексами у глибоководних технологіях [Текст] / О. М. Трунов // Вестник ХНТУ «ХПИ». – 2013. – Вып. 1(46). – С. 328 – 337.

15. Trunov, A. N. The formation of unified method of technological process effectiveness evolution [Text] / A.N. Trunov // Проблемы информационных технологий. – 2013. – № 2 (14). – С. 104 – 108.

16. Трунов, О. М. Особливості застосування критеріїв оцінки синтезованого програмного забезпечення систем гіперспектрального аналізу визначення складу речовин [Текст] / О. М. Трунов, С. О. Волкова // Збірник наукових праць НУК. – Миколаїв : НУК, 2007. – № 5(416). – С. 121 – 130.

17. Трунов, О. М. Аналіз методів і засобів підвищення якості та надійності систем медичної діагностики [Текст] / О. М. Трунов, С. О. Волкова // Математичні машини і системи. – 2008. – № 2. – С. 158–164.

18. Трунов, О. М. Моделювання надійності структурованого програмного забезпечення [Текст] / О. М. Трунов, С. О. Волкова // Математичне та комп'ютерне моделювання. Сер.: Технічні науки : зб. наук. праць. – 2008. – Вип. 1. – С. 156 – 165.

19. Трунов, О. М. Удосконалення ПЗ моделювання динаміки підводних апаратів за умов регулярних поверхневих хвиль [Текст] / О. М. Трунов, О. О. Новосадовский // Іновації в суднобудуванні та океанотехніці : матеріали 4-ї Міжнар. наук.-тех. конф. – Миколаїв : НУК ім. С.О. Макарова, 2013. – С. 412 – 413.

Поступила в редакцию 20.05.2014, рассмотрена на редколлегии 17.06.2014

Рецензент: д-р техн. наук, проф. Ю. П. Кондратенко, Черноморский государственный университет им. Петра Могилы, Николаев.

ОЦІНКА ЕФЕКТИВНОСТІ ТЕХНОЛОГІЇ ПРОГРАМУВАННЯ

О. М. Трунов

Теоретично обґрунтовано та отримано за допомогою уніфікованого методу вирази показника ефективності, який одночасно враховує чотири фактори: обсяги ПП, ймовірність безвідмовної роботи, витрати у вартісному та часовому вимірі. Сформовано модель динаміки дефектів у процесі створення ПП як результат інтерактивної взаємодії розробників, тестерів, замовників, яка враховує такі параметри, як коефіцієнт функціональної повноти, ступеня логічного покриття тестами, стандартизації та уніфікації, ймовірності помилки алгоритму та коду, ймовірності знаходження та усунення дефектів. Продемонстровано, що облік таких даних, як ймовірність здійснення та виявлення помилки в якості статистичних характеристик проектувальників-розробників, тестувальників, дозволяє розбивати ПС на ряд елементів і запрошувати для їх створення команди різної кваліфікації і різної вартості робіт за одиницю часу під час формуванні стратегії програмування.

Ключові слова: показник ефективності, програмний продукт, модель динаміки дефектів, характеристики програмного продукту, ймовірність помилки алгоритму та коду, ймовірності виявлення та усунення дефектів.

EVALUATING THE EFFECTIVENESS TECHNOLOGY PROGRAM

A. N. Trunov

Theoretically grounded and obtained by uniform expression method performance indicator, which also takes into account four factors: the volume of PP, probability, costs in cost and time dimension. The current model of the defects in the process of establishing PE as a result of interactivity developers, testers, customers, taking into account parameters such as the ratio of functional completeness, degree of logical coverage tests, standardization and unification, error probability algorithm and code, the probability of finding and eliminating defects. Demonstrated that the inclusion of such data as the probability of error and probability of detection as statistical characteristics of teams of designers and testers, respectively, allows the aircraft to break up into a number of elements and invite them to create a team of different skills and different cost per unit time in the formation of strategy programming.

Keywords: efficiency, the software model of the defect characteristics of the software error probability algorithm and code, the probability of finding and eliminating defects.

Трунов Александр Николаевич – канд. техн. наук, доцент кафедры медицинских приборов и систем, первый проректор, Черноморский государственный университет им. П. Могилы, Николаев, Украина, e-mail: ant@kma.mk.ua.