УДК: 004.052.42

С. А. ЯРЕМЧУК[1], В. С. ХАРЧЕНКО[2]

[1] ***Национальный университет «Одесская морская академия», Украина***
[2]***Национальный аэрокосмический университет им. Н. Е. Жуковского «ХАИ», Украина***

МЕТОД ОТБОРА ДЕФЕКТОСОДЕРЖАЩИХ КОМПОНЕНТОВ ПРОГРАММНЫХ СИСТЕМ

Предложенный метод основан на учете многомерной сложности компонентов программных систем посредством метрик. Исследована представительная выборка данных и получено экспериментальное подтверждение допущения метода. Входными данными метода являются показатели сложности исходного кода компонентов. Выходными данными метода является ранжированная выборка дефектосодержащих компонентов. Размер выборки определяется с учетом ограничений ресурсов верификации исходного кода. Метод способствует увеличению количества выявленных дефектов и повышению надежности программных систем в условиях ограниченных ресурсов.

Ключевые слова: *надежность, программные системы, дефекты, дефектосодержащие компоненты, метрики сложности.*

Введение

Мотивация. Надежность программных систем (ПС) в значительной мере обуславливается человеческим фактором, т.е. ошибками разработчиков. Растущая сложность ПС влечет за собой рост количества не выявленных на этапе разработки дефектов и снижение эксплуатационной надежности, что недопустимо для систем, управляющих энергетическими, транспортными, аэрокосмическими, оборонными объектами, и проблематично для некритических областей. Создано большое количество моделей и методов оценивания и повышения надежности ПС. Однако требования к количественным показателям надежности в проектах, как правило, не задаются. Практическое использование моделей носит фрагментарный характер. Новые модели надежности демонстрируют увеличение точности оценивания на частных примерах и не завершаются системным внедрением. Существующие методы не определяют в явном виде риски от не выявленных дефектов, и не дают экономически обоснованных рекомендаций в терминах "надежность-прибыль-убытки".

Решение этих задач возможно путем разработки технологически ориентированного инструментария оценивания и управления надежностью ПС. Его встраивание в бизнес-процессы софтверных компаний позволит рационально управлять процессом повышения надежности разрабатываемых ПС с учетом ограниченных ресурсов на проведение верификации. В этих условиях особую актуальность приобретает разработка методов и технологий, позволяющих выявить компоненты с наибольшим числом дефектов (т.н. дефектосодержащих компонентов (ДСК)) для оптимизации ресурсов верификации и достижения предельно возможной надежности ПС.

Анализ. Модели и методы оценивания и классификации ДСК далеко не новы и были начаты в 70-х годах прошлого столетия. Проанализируем результаты последних лет

В работе итальянских исследователей (Department of Computer and Systems Engineering, Federico II University of Naples) [1] с использованием метрик программного продукта (и регрессионных моделей на их основе) оценивается количество дефектов различных типов с целью выбора наиболее эффективной комбинации методов тестирования. Для ряда ПС методами статистического анализа авторы установили, что эффективность обнаружения дефектов при выполнении функционального, статистического, нагрузочного и стрессового тестирования не зависит от типа ПС и от начального количества дефектов. Единственным влияющим фактором является выбранный метод тестирования. Типизация дефектов и выбор соответствующих методов тестирования придает несомненную практическую ценность данному исследованию.

В работе [2] авторы предлагают подход к повышению надежности ПС комплексированием двух методов: эксплуатационного тестирования, основанного на профиле использования, при ожидаемой рабочей нагрузке на ПС, и отладочного тестирования при непредвиденных входных данных. Целесообразность такого комплексирования обоснована тем, что первый метод тестирования выявляет много

дефектов, мало влияющих на надежность ПС, поскольку частота их возникновения на этапе эксплуатации низкая. Другой метод обнаруживает немногочисленные дефекты с высокой частотой проявления при эксплуатации, что обеспечивает более высокую надежность ПС. При ПС, содержащая больше дефектов, чем другая, может быть более надежной, поскольку эти дефекты вызывают редкие отказы.

В [3] выполнен анализ влияния времени отладки на повышение надежности ПС. Многочисленные модели роста надежности ПС основаны на допущении быстрого и успешного устранения дефектов без внесения новых. Однако известно, что отладка – сложный и длительный процесс, зависящий от типов дефектов, их серьезности, приоритетности, сложности программного решения и квалификации разработчиков. Поэтому временем отладки нельзя пренебречь. На основе исследования экспериментальных данных одной масштабной ПС и семи известных моделей надежности, авторы делают вывод: чем длительнее процесс отладки, тем больше погрешность оцениваемых показателей надежности. Если при оценивании показателей надежности пренебрегают временем отладки, то прогнозная надежность ПС будет завышена на 15%, а целевая надежность будет достигнута на 35% позже времени, рассчитанного по одной из моделей.

Д. А. Маевский в [4] впервые предложил математическую модель прогнозирования количества вторичных дефектов, внесенных в программное обеспечение динамических информационных систем. Использование модели позволяет выявить временные интервалы, на которых интенсивность внесения вторичных дефектов максимальна. Знание таких интервалов позволяет своевременно принять необходимые меры для сокращения количества вторичных дефектов и повышения надежности ПС. В данный момент это единственная модель прогнозирования вторичных дефектов.

В. С. Яковина в [5] предложил метод анализа надежности ПС с учетом ее сложности на разных этапах жизненного цикла. Это один из немногих методов надежности, использующих оценку сложности ПС, как наиболее влияющего фактора на количество дефектов. Метод использует модели надежности, базирующиеся на неоднородном пуассоновском процессе, и на марковском процессе высшего порядка с дискретным и непрерывным временем. Метод позволяет разработчикам выбрать соответствующую модель и оценить показатели надежности, исходя из особенностей процесса разработки и сложности разрабатываемой ПС.

Для решения задачи выбора адекватной модели надежности ПС в [6] предложен метод, предполагающий: (а) построение матрицы допущений известных моделей; (б) формирование вектора допущений для разрабатываемой ПС с учетом характеристик и особенностей процесса и продукта разработки; (в) выбор модели исходя из вектора допущений и расчет ее параметров.

Большинство вышеперечисленных моделей и методов являются апостериорными [2-6]. Для их практического использования необходимы экспериментальные данные о выявлении дефектов в процессе тестирования. Чем объемнее наборы данных, тем точнее оценки и прогноз показателей. Однако чем больше данных тестирования, тем ближе его завершение. Возникает вопрос: что делать, если ресурсы тестирования исчерпаны, требуемая надежность не достигнута, а завершение проекта уже обещано заказчику? Для этого необходимо управлять процессом достижения целевой надежности в условиях ограниченных ресурсов путем оценивания ее показателей на более ранних этапах разработки.

Тогда может быть предложено следующее: из всего множества разработанных компонентов и тестов для них отобрать такие, выполнение которых гарантированно обеспечит выявление максимального количества дефектов. Частично эту задачу решают априорные модели и методы надежности, однако их разработано значительно меньше. Описанные в работах [1, 7] модели на основе метрик сложности исходного кода оценивают общее количество дефектов в ПС. В работе [7] установлено, что эти модели неприменимы для оценивания дефектов в отдельных компонентах ПС ввиду низкой точности прогнозирования. При этом знание, какие компоненты содержат дефекты, для эффективного тестирования более важно, чем знание общего количества дефектов. Эти пробелы в знаниях обуславливают необходимость создания нового знания, и диктуют цель и задачи настоящего исследования.

Целью данной статьи является разработка метода отбора дефектосодержащих компонентов ПС. Для достижения намеченной цели необходимо выполнить следующие *задачи*: сформировать глоссарий исследования; обосновать приемлемые допущения метода; разработать процедуру отбора ДСК ПС.

1. Глоссарий исследования

Программная система – это совокупность программ и (или) подсистем, состоящих из компонентов (функций, подпрограмм, методов, классов), взаимодействующих между собою для выполнения требований пользователя. *Компонент* (модуль) ПС – это отдельная идентифицированная структурная единица программной системы, которая может быть протестирована. Для объектно-ориентированных ПС компонентом можно считать класс.

Надежность ПС – это способность ПС обеспечивать заданный уровень функционирования в заданных спецификацией условиях. *Ошибка* – некорректное действие разработчика, в результате которого в ПС вносится дефект. *Дефект* – это программная аномалия, некорректное определение операции, процесса или данных в ПС или ее спецификации, которые могут привести к отказу.

Верификация исходного кода – это процесс выявления дефектов в исходном коде, проверка соответствия ПС требованиям на различных этапах. *Методы верификации* – обзор, аудит, статический анализ, проверка на моделях, тестирование и др.

Сложность исходного кода ПС – это комплексное свойство, обусловленное использованием в коде различных типов данных, циклических и условных конструкций, конструкций переходов, вычислений, и разветвленной иерархии взаимосвязанных компонентов. Сложность исходного кода ПС оценивается с помощью метрик.

Метрика сложности ПС – мера (метод и шкала) оценивания одного или нескольких аспектов сложности. Используем следующие метрики объектно-ориентированных ПС.

Метрика RFC (Response For a Class) определяет количество методов класса и количество методов других классов, вызываемых из данного класса. *Метрика WMC* (Weighted Methods Per Class) определяет количество и сложность методов класса. *Метрика LCOM* (Lack of Cohesion in Methods) определяет количество пар методов, не связанных по свойствам класса, без количества пар методов, имеющих такую связь. *Метрика LOC* (Lines Of Code) определяет количество строк исходного кода ПС. *Метрика NPM* (Number of Public Methods) определяет количество публичных методов класса. *Метрика CE* (Efferent Coupling) определяет количество классов внутри категории, которые зависят от классов вне этой категории. *Метрика CBO* (Coupling between object classes) определяет количество классов, с которыми связан данный класс.

2. Допущения метода

Предлагаемый метод основан на допущении, что наиболее сложные компоненты содержат наибольшее количество дефектов. Это допущение является логичным и традиционным, однако требует экспериментальной проверки. Для этого были использованы пять наборов экспериментальных данных, опубликованных на Интернет-ресурсе депозитария международной научной конференции PROMISE (PRedictOr Models In Software Engineering) [8]. Данные представляют собой показатели сложности по метрикам и количество дефектов, выявленных при верификации кода отдельных компонентов пяти открытых объектно-ориентированных ПС различных предметных областей. Характеристики ПС приведены в таблице 1. Общий объем исследуемых данных превышает миллион строк кода, содержит 4330 модулей и 4449 дефектов. Сравнительный анализ данных в таблице 1 показывает, что исследуемые ПС существенно отличаются между собой количеством модулей, дефектов, соотношением дефектных и бездефектных модулей, плотностью дефектов. В таблице 2 приведены меры сложности по метрикам в расчете на один модуль для исследуемых ПС. Выбор множества метрик

$$M = \{RFC, WMC, LCOM, LOC, NPM, CE, CBO\}$$

обоснован в исследовании [7], в котором установлены наивысшие точечные и интервальные оценки коэффициентов корреляции между показателями сложности (метриками) и количеством дефектов.

Таблица 1

Характеристики исследуемых ПС

Показатели ПС	Сокращенные названия и версии ПС				
	Luc 2.4	Xer 1.4	Ant 1.7	Xal 2.7	Cam 1.4
Количество модулей	340	588	746	910	1 746
Количество дефектов	632	1 596	338	1213	670
Часть модулей без дефектов	60%	74%	22%	99%	17%
Часть модулей с дефектами	40%	26%	78%	1%	83%
Количество дефектов в модуле	1.86	2.71	0.45	1.33	0.38
Плотность дефектов на 1 тыс.строк	6.14	11.30	1.62	2.83	3.42

Таблица 2

Меры сложности по метрикам в расчете на модуль

Меры сложности	Luc 2.4	Xer 1.4	Ant 1.7	Xal 2.7	Cam 1.4
Метрика RFC	25	19	34	29	21
Метрика WMC	10	10	11	11	9
Метрика LCOM	69	75	89	126	73
Метрика LOC	303	240	280	471	112
Метрика NPM	7	8	8	9	7
Метрика CE	5	3	6	7	6
Метрика CBO	11	6	11	12	11

Анализ данных в таблице 2 показывает, что сложность исследуемых ПС существенно отличается по ряду метрик. Значительный общий объем и указанные различия позволяют считать выборку представительной. Поскольку сложность каждого модуля измеряется рядом метрик, будем называть ее многомерной. Для установления соотношений между многомерной сложностью модуля и количеством дефектов выполнены следующие действия. Полное множество модулей SET_{mod} каждой ПС было сортировано *n* раз в порядке убывания показателя сложности по каждой из метрик

$$M = \{RFC, WMC, LCOM, LOC, NPM, CE, CBO\}.$$

Из каждого множества была взята некоторая его часть (1/2, 1/5, 1/10) наиболее сложных модулей с наивысшими показателями сложности по каждой из метрик. В результате были получены семь базовых подмножеств $SUBSET_{mod}^{RFC}$, $SUBSET_{mod}^{WMC}$, $SUBSET_{mod}^{LCOM}$, $SUBSET_{mod}^{LOC}$, $SUBSET_{mod}^{NPM}$, $SUBSET_{mod}^{CE}$, $SUBSET_{mod}^{CBO}$. Пересечение базовых подмножеств по семи метрикам образовало новое подмножество модулей $SUBSET_{mod}^{7}$, по шести метрикам $SUBSET_{mod}^{6}$, по пяти метрикам $SUBSET_{mod}^{5}$, по четырем метрикам $SUBSET_{mod}^{4}$, по трем метрикам $SUBSET_{mod}^{3}$, по двум метрикам $SUBSET_{mod}^{2}$, по одной метрике $SUBSET_{mod}^{1}$. Для каждого их этих подмножеств было подсчитано среднее фактическое количество дефектов на один модуль. Результаты приведены в таблице 3.

Из таблицы 3 следует, что наибольшее количество дефектов содержится в наиболее сложных модулях. По мере уменьшения сложности модулей среднее количество дефектов уменьшается, что подтверждает приемлемость допущений метода.

3. Метод отбора дефектосодержащих компонентов ПС

Предлагаемый метод выполняется после написания исходного кода ПС до начала его верификации. Входными данными являются показатели сложности отдельных модулей ПС по метрикам. Многие среды разработки автоматически подсчитывают метрики при компиляции исходного кода. Это говорит об устойчивом использовании метрик в ежедневной практике программной инженерии. Если разработчики не располагают корпоративным и проверенным на практике набором метрик, в качестве стартовых можно использовать известные и наиболее информативные для прогноза метрики $M = \{RFC, WMC, LCOM, LOC, NPM, CE, CBO\}$, исследованные в [7].

Таблица 3

Среднее фактическое количество дефектов в одном модуле для исследуемых ПС

Сложность модулей	Сокращенные названия и версии ПС				
	Luc 2.4	Xer 1.4	Ant 1.7	Xal 2.7	Cam 1.4
$SUBSET_{mod}^{7}$	13,5	20,9	3,1	3,0	4,6
$SUBSET_{mod}^{6}$	5,4	13,2	1,9	2,3	2,0
$SUBSET_{mod}^{5}$	3,1	14,2	1,1	2,0	1,1
$SUBSET_{mod}^{4}$	2,1	5,8	1,4	1,7	1,2
$SUBSET_{mod}^{3}$	2,0	4,3	1,0	1,7	1,9
$SUBSET_{mod}^{2}$	2,3	3,8	1,0	1,5	1,6
$SUBSET_{mod}^{1}$	1,5	2,8	0,4	1,7	0,4

Обозначим множество используемых метрик $M = \{M_1, M_2, ... M_n\}$. Метод предполагает выполнение следующих семи шагов.

Шаг 1. При помощи стандартных или корпоративных программных средств рассчитать показатели сложности по метрикам $M = \{M_1, M_2, ... M_n\}$ для отдельных компонентов (модулей, классов) ПС.

Шаг 2. Полное множество модулей SET_{mod} отсортировать *n* раз в порядке убывания показателя сложности по каждой метрике.

Шаг 3. Из каждого сортированного множества взять некоторую часть наиболее сложных модулей по отдельной метрике. Размер части определяется ограничениями ресурсов для проведения верификации. Чем меньше ресурсов, тем меньше часть взятых модулей. В результате для *n* метрик было получено *n* базовых подмножеств $SUBSET_{mod}^{M_1}$, $SUBSET_{mod}^{M_2}$, …, $SUBSET_{mod}^{M_n}$.

Шаг 4. Пересечением базовых подмножеств $SUBSET_{mod}^{M_1}$, $SUBSET_{mod}^{M_2}$, …, $SUBSET_{mod}^{M_n}$ образовать промежуточные подмножества с наивысшими показателями сложности по всем метрикам $SUBSET_{mod}^{n} = SUBSET_{mod}^{M_1} \cap SUBSET_{mod}^{M_2} \cap \ldots \cap \cap SUBSET_{mod}^{M_n}$. Всем модулям этого подмножества присвоить ранг n. По любым n-1 метрикам $SUBSET_{mod}^{n-1} = SUBSET_{mod}^{M_1} \cap SUBSET_{mod}^{M_2} \cap \ldots \cap$

$\cap\, \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{M_n}}$. Всем модулям этого подмножества присвоить ранг n-1, и т.д. по любой одной метрике $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{l}} = \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{M_1}} \cap \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{M_2}} \cap \ldots \cap \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{M_n}}$. Всем модулям этого подмножества присвоить ранг 1. Количество рангов равно количеству используемых метрик.

Шаг 5. Объединением подмножеств $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{n}}, \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{n-1}}, \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{n-2}}, \ldots,$ $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{l}}$ образовать результатную ранжированную выборку модулей. $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{M}} = \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{n}} \cup \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{n-1}} \cup \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{n-2}} \cup \ldots \cup \mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{l}}$. Результатная выборка включает модули с наивысшими показателями сложности по *n* метрикам, по *n - 1* метрикам, …, по одной метрике. Ранжирование модулей в результатной выборке необходимо для установления порядка их верификации. Модули с высшими рангами наиболее сложные, содержат наибольшее количество дефектов и подлежат первоочередной верификации. Полученную ранжированную выборку модулей ПС $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{M}}$ необходимо верифицировать различными методами (экспертиза, контроль, инспекции, аудит, статистический анализ, проверка формальных моделей, тестирование).

Заключение

На основе анализа научных публикаций последних лет в области разработки моделей и методов оценивания надежности ПС установлено, что отбор и ранжирование ДСК является одним из важных направлений оптимизации верификационных усилий в условиях ограниченных ресурсов. Для этого предложен метод отбора дефектосодержащих компонентов ПС, основанный на учете многомерной сложности компонентов. Исследована представительная выборка ПС и получено экспериментальное подтверждение допущения метода о том, что наиболее сложные компоненты содержат наибольшее количество дефектов. Входными данными метода являются показатели сложности отдельных компонентов ПС по метрикам. Выходными данными метода является ранжированная выборка дефектосодержащих компонентов ПС. Размер выборки определяется с учетом ограничений ресурсов верификации. Актуальным направлением дальнейшей работы представляется апробация метода на представительной выборке экспериментальных данных.

Литература

1. Cotroneo, D. Testing techniques selection based on ODC fault types and software metrics [Text] / D. Cotroneo, R. Pietrantuono, S. Russo // The Journal of Systems and Software. – 2013. – №. 86. – С. 1613–1637.

2. Cotroneo, D. Combining Operational and Debug Testing for Improving Reliability [Text] / D. Cotroneo, R. Pietrantuono, S. Russo // The Journal of Systems and Software. – 2013. – Т. 62, №. 2. – С. 408–423.

3. On the Impact of Debugging on Software Reliability Growth Analysis: A Case Study [Text] / M. Cinque, C. Gaiani, D. D. Stradis, A. Pecchia, R. Pietrantuono, S. Russo // Computational Science and Its Applications – ICCSA. – 2014. – Т. 8583, of the series Lecture Notes in Computer Science. – С. 461–475.

4. Маєвський, Д. А. Теоретичні та прикладні основи забезпечення якості динамічних інформаційних систем [Текст] : дис. ... докт. техн. наук : 05.13.06 / Д. А. Маєвський ; Одеськ. нац. політехн. ун–т. – Одеса, 2013. – 440 с.

5. Яковина, В.С. Метод аналізу надійності програмних засобів з урахуванням їх складності [Text] / В. С. Яковина // Радіоелектронні і комп'ютерні системи. – 2015. – № 2 (72). – С. 127–133.

6. Kharchenko, V. S. The Method of Software Reliability Growth Models Choice Using Assumptions Matrix / V. S. Kharchenko, O. M. Tarasyuk, V.V. Sklyar // Proc. of 26th Annual Int. Computer Software and Applications Conference (COMPSAC). – Oxford, England, 2002. – P. 541–546.

7. Яремчук, С. О. Моделі, методи і технологія апріорного оцінювання показників надійності обліково-аналітичних інформаційних систем [Text]: дис. .... канд. техн. наук : 05.13.06 / С. О. Яремчук ; Одеськ. нац. політехн. ун-т. – Одеса, 2015. – 210 с.

8. The PROMISE Repository of empirical software engineering data [Электронный ресурс] / Режим доступа: http://openscience.us/repo/defect/ck/. – 01.03.2016.

References

1. Cotroneo, D., Pietrantuono, R., Russo, S. Testing techniques selection based on ODC fault types and software metrics. *The Journal of Systems and Software,* 2013, no. 86, pp. 1613–1637.

2. Cotroneo, D., Pietrantuono, R., Russo, S. Combining Operational and Debug Testing for Improving Reliability. *The Journal of Systems and Software*, 2013, vol. 62, no. 2, pp. 408–423.

3. Cinque, M., Gaiani, C., Stradis, D., Pecchia, A., Pietrantuono R., Russo, S. On the Impact of Debugging on Software Reliability Growth Analysis: A Case Study. *Computational Science and Its Applications*, *ICCSA,* 2014, vol. 8583 of the series Lecture Notes in Computer Science, pp 461–475.

4. Mayevs'kyy, D. A. *Teoretychni ta prykladni osnovy zabezpechennya yakosti dynamichnykh informatsiynykh system.* Diss. …. dokt. tekhn. nauk : 05.13.06 [Theoretical and practical bases of quality assurance dynamic information systems. Diss. … dokt. tech. sci.]. Odesa, Odes'k. nats. politekhn. un-t. Publ., 2013. 440 p.

5. Yakovyna, V. S. Method of the analysis of software reliability taking into account their complexity. *Radioelectronic and computer systems*, 2015, no. 2 (72), pp. 127–133.

6. Kharchenko, V. S., Tarasyuk, O. M., Sklyar, V. V. The Method of Software Reliability Growth Models Choice Using Assumptions Matrix. *Proceedings of 26-th Annual Int. Computer Software and Applications Conference, COMPSAC*, 2002, Oxford, England, pp. 541–546.

7. Yaremchuk, S. O. *Modeli, metody i tekhnolohiya apriornoho otsinyuvannya pokaznykiv nadiynosti oblikovo-analitychnykh informatsiynykh system.* Diss. …. kand. tekhn. nauk : 05.13.06 [Models, methods and technology of a priori estimation reliability indices of accounting and analytical information systems. Diss. … cand. tech. sci.]. Odesa, Odes'k. nats. politekhn. un–t. Publ., 2015. 210 p.

8. *The PROMISE Repository of empirical software engineering data*. Available at: http://openscience.us/repo/defect/ck/ (accessed 01.03.2016).

Поступила в редакцию 11.03.2016, рассмотрена на редколлегии 14.04.2016

МЕТОД ВИБОРУ ДЕФЕКТОМІСТКИХ КОМПОНЕНТІВ ПРОГРАМНИХ СИСТЕМ

С. О. Яремчук, В. С. Харченко

Запропонований метод базується на обліку багатовимірної складності компонентів програмних систем за допомогою метрик. Досліджена репрезентативна вибірка даних і отримано експериментальне підтвердження припущення методу. Вхідними даними методу являються показники складності вихідного коду компонентів. Вихідними даними методу являється вибірка дефектомістких компонентів. Розмір вибірки визначається з урахуванням обмежень ресурсів верифікації вихідного коду. Метод сприяє збільшенню кількості виявлених дефектів і підвищенню надійності програмних систем в умовах обмежених ресурсів.

Ключеві слова: надійність, програмні системи, дефекти, дефектомісткі компоненти, метрики складності.

METHOD OF SELECTION OF THE PROGRAM SYSTEM COMPONENTS CONTAINING DEFECTS

S. A. Yaremchuk, V. S. Kharchenko

The offered method is based on the accounting of multidimensional complexity of program systems components by means of metrics. Representative data selection is investigated. Experimental confirmation of method assumption is received. Entrance data of the method are indicators of source code complexity. The output data of the method is the ranged selection of components with defects. The amount of selection is defined with restrictions of verification resources of a source code. The method promotes increase of the revealed defects amount. The method provides increase of program systems reliability in the limited resources conditions.

Key words: reliability, program systems, defects, containing defects components, complexity metrics.

Яремчук Светлана Александровна – канд. техн. наук, старший преподаватель кафедры общенаучных дисциплин, Национальный университет «Одесская морская академия», Дунайский институт, Измаил, Украина, e-mail: svetlana397@yandex.ru.

Харченко Вячеслав Сергеевич – д-р техн. наук, профессор, заведующий кафедрой компьютерных сетей и систем, Национальный аэрокосмический университет им. Н. Е. Жуковского «Харьковский авиационный институт», Харьков, Украина, e-mail: v_s_kharchenko@ukr.net.

Yaremchuk Svetlana Aleksandrovna - Candidate of Technical Science, researcher, senior teacher of department of general scientific disciplines, National University "Odessa Maritime Academy", Danube institute, Izmail, Ukraine, e-mail: svetlana397@yandex.ru.

Kharchenko Vyacheslav Sergeevich – Doctor of Technical Science, Prof., Head of Dep. of Computer Systems and Networks, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine, e-mail: v_s_kharchenko@ukr.net.