УДК 004.05 DOI:10.31673/2412-4338.2021.040411

Лихвар А.В., Негоденко О.В., Золотухіна О.А., Шевченко С.М., Олімпієва Ю.І.
Державний університет телекомунікацій, м. Київ

МЕТОД ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ОЦІНКИ СУПРОВОДЖЕНОСТІ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Анотація: *В роботі розглядаються питання підвищення якості та рівня супроводженості об'єктно-орієнтованого програмного забезпечення. Аналіз існуючих моделей якості програмного забезпечення продемонстрував актуальність підтримки належного рівня супроводженості програмного забезпечення. При цьому, існуючі підходи здебільшого відрізняються термінологічною неоднозначністю, різноманітністю формулювань та способів вимірювання та інтерпретації результатів. Проблему також становить складність самого процесу супроводження, яка виникає із-за великої кількості внутрішніх та зовнішніх факторів, що робить коригування моделі супроводженості довготривалим та складним процесом. Пропонується модифікація моделі дельта супроводженості (DMM) за рахунок розширення вимірюваних властивостей вихідного коду, визначення їх взаємозв'язку з підхарактеристиками супроводженості, що визначені стандартом ISO/IEC 25010:2016. Побудована матриця зв'язків підхарактеристик якості ISO 25010 з характеристиками властивостей вихідного коду. В роботі використано метрики WMC, NOC, NOM, MLOC, PAR, VG, які враховують особливості об'єктно-орієнтованої парадигми програмування. Розроблена математична модель для розрахунку показників супроводженості. Для перевірки адекватності моделі проведено експериментальне дослідження на прикладі 6 програмних продуктів з відкритим кодом з імплементацією об'єктно-орієнтованої частини мовою програмування Python, що мають різну функціональність. Доведено, що показники вимірювання моделі DMMS+ разом із DMMS відображають взаємозв'язок із змінами вихідного коду, що впливають на супроводженість. Кореляція показників DMMS maDMMS+ згідно аналізу 1000 внесених змін в репозиторії становить від 0.77 до 0.86. Прогнозні припущення щодо розвитку об'єкта включають супроводженння об'єктно-орієнтованого програмного забезпечення шляхом модифікація моделі дельта супроводженості (DMM) шляхом розширення вимірюваних властивостей вихідного коду.*

Ключові слова: *супроводженість, об'єктно-орієнтована парадигма, якість програмного забезпечення, метрики якості.*

Lykhvar A.V., Negodenko O.V., Zolotukhina O.A., Shevchenko S.M., Olimpiyeva Yu.I.
State University of Telecommunications, Kyiv

THE METHOD OF IMPROVING THE EFFICIENCY OF ASSESSMENT OF SUPPORT OF OBJECT-ORIENTED SOFTWARE

Abstract*: The work examines issues of improving the quality and level of support of object-oriented software. Analysis of existing software quality models demonstrated the relevance of maintaining an appropriate level of software support. At the same time, the existing approaches are mostly distinguished by terminological ambiguity, variety of formulations and methods of measurement and interpretation of results. The problem is also the complexity of the support process itself, which arises from a large number of internal and external factors, which makes the adjustment of the support model a long-term and complex process. A modification of the delta maintainability model (DMM) is proposed by expanding the measurable properties of the source code, determining their relationship with the maintainability sub-characteristics defined by the ISO/IEC 25010:2016 standard. A matrix of relations of sub-characteristics of ISO 25010 quality with the characteristics of the source code properties is constructed. The work uses WMC, NOC, NOM, MLOC, PAR, VG metrics, which take into account the features of the object-oriented programming paradigm. A mathematical model has been developed for calculating compliance indicators. To check the adequacy of the model, an experimental study was conducted on the example of 6 open source software products with the*

© Лихвар А.В., Негоденко О.В., Золотухіна О.А., Шевченко С.М., Олімпієва Ю.І. 2021

implementation of the object-oriented part in the Python programming language, which have different functionality. The metrics of the DMMS+ model together with DMMS have been proven to reflect the relationship between source code changes that affect maintainability. The correlation of DMMS and DMMS+ indicators according to the analysis of 1000 changes made in the repository is from 0.77 to 0.86. Predictive assumptions about object development include maintaining object-oriented software by modifying the delta maintainability model (DMM) by extending the measurable properties of the source code.

***Keywords**: maintainability, object-oriented paradigm, software quality, quality metrics.*

Лыхварь А.В., Негоденко Е.В., Золотухина О.А., Шевченко С.М., Олимпиева Ю.И.
Государственный университет телекоммуникаций, г. Киев

МЕТОД ПОВЫШЕНИЯ ЭФФЕКТИВНОСТИ ОЦЕНКИ СОПРОВОЖДАЕМОСТИ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

***Аннотация:** В работе рассматриваются вопросы повышения качества и уровня сопровождения объектно-ориентированного программного обеспечения. Анализ существующих моделей качества программного обеспечения продемонстрировал актуальность поддержания должного уровня сопровождения программного обеспечения. При этом существующие подходы в большинстве своем отличаются терминологической неоднозначностью, разнообразием формулировок и способов измерения и интерпретации результатов. Проблема также составляет сложность самого процесса сопровождения, возникающая из-за большого количества внутренних и внешних факторов, что делает корректировку модели сопровождения длительным и сложным процессом. Предлагается модификация модели дельта сопровождаемости (DMM) за счет расширения измеряемых свойств исходного кода, определение их взаимосвязи с подхарактеристиками сопровождения, определенными стандартом ISO/IEC 25010:2016. Построена матрица связей подхарактеристик качества ISO 25010 с характеристиками свойств исходного кода. В работе использованы метрики WMC, NOC, NOM, MLOC, PAR, VG, учитывающие особенности объектно-ориентированной парадигмы программирования. Разработана математическая модель для расчета показателей сопровождения. Для проверки адекватности модели проведено экспериментальное исследование на примере 6 программных продуктов с открытым кодом с имплементацией объектно-ориентированной части на языке программирования Python, имеющих разную функциональность. Доказано, что показатели измерения модели DMMS+ вместе с DMMS отражают взаимосвязь с изменениями исходного кода, влияющими на сопровождение. Корреляция показателей DMMS и DMMS+ согласно анализу 1000 внесенных изменений в репозитории составляет от 0,77 до 0,86. Прогнозные предположения о развитии объекта включают в себя сопровождение объектно-ориентированного программного обеспечения путем модификация модели дельта сопровождаемости (DMM) путем расширения измеряемых свойств исходного кода.*

***Ключевые слова:** сопровождаемость, объективно-ориентированная парадигма, качество программного обеспечения, метрики качества.*

1. Вступ

Сучасні системи програмного забезпечення все глибше інтегруються в життєво важливі сфери, від керування критичної інфраструктури до пілотування транспортними засобами. Саме тому одним із найважливіших пріоритетів є зменшення можливих дефектів програмних засобів**.** Швидкість розвитку суспільних процесів та технологій обумовлює необхідність адаптації, що в свою чергу вимагає внесення коригувань програмного забезпечення. Дослідження C. Jones демонструє стійкий зріст залучення інженерів до робот із підтримки ПЗ, з 9.09 % відсотків у 1950 році до 72.73% у 2000 році та прогнозованою долею залучення 77.27% у 2025 році [1]. Вимоги до підвищення рівня якості та супроводженості є причиною багатьох досліджень та постійного пошуку методологій вимірювання та оцінки програмного забезпечення. Супроводженість програмного забезпечення є добре дослідженою темою. Більшість досліджень обґрунтовується важливістю супроводженості програмного

забезпечення, підкреслюючи взаємозв'язок з адаптивністю та зменшення кількості дефектів. Якість програмного забезпечення, зокрема, супроводженість відрізняються від інших властивостей складністю факторів, що їх обумовлюють, тому їх коригування майже завжди є складним та довготривалим.

2. Аналіз останніх досліджень і публікацій.

Моделі якості програмного забезпечення призначені для визначення критеріїв оцінки якості, та визначення термінології. Не зважаючи на відмінність, існуючі моделі визначають супроводженість як один із головних атрибутів якості програмного забезпечення. Таким чином більшість методів вимірювання супроводженості, враховують не тільки супроводженість, а є частиною моделей якості. До наукових досліджень, присвячених проблемам якості програмного забезпечення, можна віднести публікації авторів: C. Jones [1], R. Plösch, H. Gruber, C. Korner, M. Saft [2], A. Madi, O.K. Zein [3].

Моделі якості Модель Дженерал Електрікс (модель МакКола) [38] визначає наступні критерії якості, що пов'язані із внутрішніми факторами якості:

– використання продукту, визначає аспекти якості продукту під час його використання;

– перенесення продукту, визначається як здатність продукту до зберігання робочого стану під час змін оточення;

– модифікування продукту, стосується всіх аспектів виправлення помилок та адаптації системи.

Супроводженість, як внутрішній фактор якості пов'язується із модифікуванням продукту, і визначається як обсяг зусиль необхідних для визначення та виправлення дефекту програми в операційному середовищі. Супроводженість як і всі внутрішні фактори якості вимірюється опосередковано, через пов'язані властивості програмного забезпечення.

Вимірювання властивостей пропонується здійснювати шляхом ранжування від 1 (ціль не досягнуто) до 10 (відмінна реалізація), без зазначення конкретних метрик чи способів вимірювання.

Модель Боема [4] є схожою із моделлю МакКола, оскільки також має ієрархічну структуру, що складається з 3 сегментів характеристик. Боем визначає «загальну корисність», як основу якості програмного забезпечення, що має найвищий пріоритет, і підпорядковує усі інші характеристики. Загальна якість системи визначається сумарним значенням всіх характеристик. Наступний рівень, який визначає «загальну корисність» містить утилітарність, супроводженість та портативність, при цьому перші дві характеристики мають додатковий рівень підхарактеристик. На останньому рівні ієрархії модель визначає первинні характеристики, які саме і призначені для вимірювання. При цьому модель не передбачає будь яких визначень або формулювань таких вимірювань або показників.

Модель якості ISO/IEC 9126 визначає супроводженість як високорівневу характеристику якості продукту, що пов'язаний із 4 підхарактеристиками. Властивостями вихідного коду програми, однак не містить формулювання чи визначення метрик. В подальшому, стандарт був переглянутий та замінений стандартомISO/IEC 25010:2011 [5].

Модель якості об'єктно-орієнтованого дизайну (QMOOD) [6]складається з 4 рівневої ієрархічної структури. Модель також включає набір метрик призначених для вимірювання атрибутів якості. В основі визначення атрибутів якості, з деякими модифікаціями є модель якості програмного забезпечення ISO 9126 [5]. Також модель визначає атрибути дизайну та відповідні їм метрики. Атрибути дизайну в свою чергу, пов'язуються із атрибутами якості.

Стандарт ISO/IEC 25010:2016 "Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Моделі якості системи та програмних засобів" [7, 8] є заміною стандарту ISO 9126 [5]. Стандарт розширює модель якості двома новими високорівневими характеристиками: сумісність (є новою характеристикою),та захищеність (в попередньому стандарті є підхарактеристикою

функційної придатності). Також стандартом внесені зміни в структуру супроводженості. Введені нові підхарактеристики: «модульність» та ”повторна використовність». Підхарактеристики «змінність» та «стабільність» були замінені новою характеристикою «модифіковність». Стандарт дає наступне визначення супроводженості: супроводженість визначається як ступінь результативності та ефективності, в якому продукт або система можуть бути модифіковані призначеними фахівцями з обслуговування При цьому супроводженість можна інтерпретувати як притаманну продукту або системі здатність сприяти полегшенню робіт з технічного обслуговування або як якість під час застосування.

3. Метою роботи є підвищення якості та рівня супроводженості об’єктно-орієнтованого програмного забезпечення за рахунок зменшення кількості дефектів.

4. Основна частина

Супроводженість є складною концепцією, неодноразово розглянутою дослідженнями, що пропонують різні формулювання та підходи до вимірювання. Однак, більшість досліджених підходів, в тій чи іншій мірі мають:

- неоднозначність термінології та визначень критеріїв якості;
- абстрактність, відсутність визначень формулювань та способів вимірювання;
- складність або неможливість інтерпретації;
- проведення причинно-наслідкового аналізу результатів вимірювання.

Модель супроводженості (SIG-MM) не має означених недоліків, однак базується на вимірюванні всього вихідного коду програмного продукту, що обумовлює *слабка репрезентативність результатів вимірювання при незначних змінах вихідного коду*. Прикладом цього є запис #402331 в системі реєстрації дефектів програмного продукту Mozilla Rhino (https://bugzilla.mozilla.org/showbug.cgi?id=402331). Зазначений дефект було виправлено комітом #262602 (https://github.com/mozilla/rhino/commit/262602), який в діапазоні вимірювань від -5 до 5. Модель супроводженості (SIG-MM) має рейтинг − 0.007. Зазначений результат не демонструє будь-яких істотних змін в супроводженості, що не відповідає дійсності, оскільки внесені змінами 200 рядків коду істотно негативно впливають на супроводженість. Однак відносно розміру всіх строк коду Mozilla Rhino, в порівнянні до якого зміни були валідовані, показник отримав не репрезентативний результат [9]. Модель дельта супроводженості (DMM) не містить наведених недоліків, однак не враховує специфіки об’єктно-орієнтованої парадигми програмування.

Враховуючи вищевикладене, пропонується модифікація моделі дельта супроводженості (DMM) шляхом розширення вимірюваних властивостей вихідного коду, визначення їх взаємозв’язку з підхарактеристиками супроводженості, що визначені ISO/IEC 25010:2016 [7, 8], визначення способів вимірювання та порогових значень. В подальшому базова модель позначається DMM, результуючий показник DMMS. Посилання на запропоновану модель позначаються як DMM+, показник відповідно DMMS+.

В таблиці 1 наведена матриця зв’язків підхарактеристик якості ISO 25010 [7] з характеристиками властивостей вихідного коду.

Обчислення відбувається з урахуванням в порядку встановленому для моделі дельта супроводженості (DMM) [10] з наступними змінами визначень:

RC = {низький, високий}
CP = {Пов'язаність класу, Складність класу, Складність методу,
Кількість методів, Розмір методу, Кількість параметрів,
Дублювання, Залежність модуля}.

В якості метрик використовуються такі, що враховують особливості об’єктно-орієнтованої парадигми програмування:

– Зважені методи на клас (WMC) - сума показників складності всіх методів класу. Класи з високим показником складніше підтримувати та мають значний вплив на підкласи. Обрахунок показнику складності не визначений, для забезпечення найбільш широкого застосування цієї метрики.

– Кількість спадкоємців класу (NOC) - кількість прямих спадкоємців(підкласів) класу. Враховуючи, що наслідування є формою повторного застосування, наявність великої кількості спадкоємців класу може свідчити про високі показникиповторного застосування. Однак, висока кількість спадкоємців, може спровокувати багато змін при змінах в спадкодавці (суперкласі), що, в свою чергу, негативно впливає на можливості підтримуваності.

– Кількість методів (NOM) - кількість методів класу. Складність класу підвищується із збільшенням кількості методів класу.

– MLOC – розмір методу в рядках коду (SLOC).

– PAR – кількість формальних параметрів інтерфейсу метода.

– VG – цикломатична складність методу.

Таблиця 1

Матриця зв’язків підхарактеристик якості ISO 25010 з характеристиками властивостей вихідного коду

Підхарактеристики супроводженості ISO 25010						
Властивості вихідного коду		Модульність	Повторна використовність	Аналізовність	Модифіковність	Тестовність
	Пов'язаність класу	x	x		x	x
	Складність класу			x		
	Складність методу			x	x	x
	Кількість методів		x			x
	Розмір методу			x		
	Кількість параметрів		x			x
	Дублювання	x		x	x	
	Залежність модуля		x		x	

Дослідженням 111 систем програмного забезпечення проведеним Філо, Тарсіо Г.С. та М. Бігонья [11] запропоновано визначення порогових значень метрик об’єктно-орієнтованого програмного забезпечення (табл. 2).

Таблиця 2

Порогові значення для метрик програмного забезпечення Філо, Тарсіо Г.С. та М. Бігонья [11]

	Рівень		
Метрика	Кращій	Середній	Поганий
WMC	$m \leq 11$	$11 < m \leq 34$	$m > 34$
NOC	$m \leq 11$	$11 < m \leq 28$	$m > 28$
NOM	$m \leq 6$	$6 < m \leq 14$	$m > 14$
MLOC	$m \leq 10$	$10 < m \leq 30$	$m > 30$
PAR	$m \leq 2$	$2 < m \leq 4$	$m > 4$
VG	$m \leq 2$	$2 < m \leq 4$	$m > 4$

Опис властивостей вихідного коду та їх зв'язок із пороговими значеннями призначеними для визначення ризику вихідного коду наведено в таблиці 3.

Таблиця 3

Властивості вхідного коду

Властивість вихідного коду	**Коротка назва**	**Опис**	**Критерій низького ризику**
Пов'язаність класу	LCOM	Відсутність пов'язаності в методах LCOM4	LCOM = 1
Складність класу	WMC	Зважені методи на клас WMC, з обрахування цикломатичної складності CC	WMC ≤ 11
Складність методу	VG	Цикломатична складність методу	VG ≤ 2
Кількість методів	NOM	Кількість методів одного класу	NOM ≤ 6
Розмір методу	MLOC	Розмір методу в рядках коду (SLOC)	MLOC ≤ 10
Кількість параметрів	PAR	Кількість формальних параметрів інтерфейсу метода	PAR ≤ 2
Дублювання	DCL	Повторенням більше ніж 6 строк вихідного коду на текстовому рівні, із урахуванням пробілів, без урахування строк без символів.	DCL = 0
Залежність модуля	MC	Кількість вхідних залежностей модуля. Модулем є групуючий елемент одиниць програми, файл.	MC ≤ 10

Математична модель дельта супроводженості об'єктно-орієнтованого програмного забезпечення (DMM) описується таблицею 4.

Таблиця 4

Формули для розрахунку показників супроводжуваності

Показник	**Порядок обчислення**
Дельта профіль ризику (RDP)	RPD(f^1, f, cp, r) = $LOC(f, cp, r)$ - $LOC(f^1, cp, r)$ де LOC - кількість рядків коду, r - категорія ризику, cp - властивість коду, f - файл версії із змінами, f^1 - файл попередньої версії
Дельта профілю низького ризику (LRPD)	LRPD(ср) = CRPDI(ср, низький) + $\sum_{h \in \text{(середній, високий, дуже високий)}} CRDPD(cp, h)$ де CRPDI(ср, r) = $\sum_{\{f^1, f\} \in D}$ RPDI (f^1, f, cp, r) RPDI(f^1, f, cp, r) = max(0, RPD(f^1, f, cp, r))
Дельта профіля високого ризику (HRPD)	HRPD(ср) = CRPDD(ср, низький) + $\sum_{h \in \text{(середній, високий, дуже високий)}} CRDPI(cp, h)$ де CRPDD(ср, r) = $\sum_{\{f^1, f\} \in D}$ RPDD (f^1, f, cp, r) RPDD(f^1, f, cp, r) = | min(0, RPD(f^1, f, cp, r))
Дельта показник властивості коду (DS)	DS(cp) =$LRDP(cp)$ / ($LRDP(cp)$+$HRPD(cp)$)
Показник дельта супроводженості (DDMS)	CP = {Дублювання, Розмір одиниці програми, Складність одиниці програми, Взаємодія одиниці програми, Залежність модуля} DDMS = $\sum_{cp \epsilon CP} DS(cp)$ / $\|CP\|$

Практичну валідацію запропонованої моделі проведено шляхом аналізу програмних продуктів з відкритим вихідним кодом, імплементованих із застосуванням об'єктно-орієнтованої парадигми програмування, процесом розробки із застосуванням систем контролю версій та значною кількістю учасників процесу розробки та тривалою історією змін коду. Оскільки аналіз вихідного коду програмних продуктів потребує дослідження абстрактного синтаксичного дерева (АСД), тому з метою спрощення імплементації застосунку, призначеного для аналізу, всі програмні продукти обрані з вимого імплементації об'єктно-орієнтованої частини спільною мовою програмування. Для порівняльного аналізу було обрано 6 програмних продуктів різною функціональної призначенності, з відкритим вихідним кодом, з імплементацією об'єктно-орієнтованої частини мовою програмування Python:

– Sentry є сервісом призначеним для моніторингу дефектів програмногозабезпечення в режимі реального часу;

– Zulip є застосунком для синхронного та асинхронного спілкуваннякоманд, що розробляється розподіленою командою розробників;

– Saleor є комерційною платформою з відкритим вихідним кодом,призначеним для роботи із великою кількістю даних та навантаженнями;

– Django є фреймворком призначеним для створення сервернихзастосунків та високорівневим доступом до баз данних;

– Tensorflow є комплексною платформою для машинного навчання, зрозгалуженою екосистемою застосунків, і великою спільногото дослідників;

– Odoo є комплектом застосунків для автоматизації всіх основнихаспектів комерційної діяльності.

Область аналізу охоплює лише зміни в файлах (модулях), що містять інструкції на мові програмування Python, та мають розширення "*.py", при цьому не враховуються зміни до файлів із інструкціями для модульного тестування. З метою проведення дослідження та вимірювань, було імплементовано програму, з підтримкою інтерфейсу командної строки та забезпечення одночасного вимірювання показників моделі DMM та DMM+.

Відповідно до результатів проведеного аналізу значний показник коефіцієнта кореляції Пірсона в значенні від 0.77 до 0.86 демонструє сильну позитивну кореляцію між значеннями DMMS та DMMS+. Таким чином, показники вимірювання DMMS+ разом із DMMS відображають взаємозв'язок із змінами вихідного коду, що впливають на супроводженість. Значення показників DMMS та валідація були підтверджені [9] в ході емпіричних досліджень. Кореляція показників DMMS таDMMS+ згідно аналізу 1000 внесених змін в репозиторії становить:

1) Tensorflow - r = 0.86,
2) Sentry - r = 0.8,
3) Django - r = 0.82,
4) Odoo - r = 0.84,
5) Saleor - r = 0.77,
6) Zulip - r = 0.77.

Дослідження змін внесених комітом 2798c937deb6625a4e6a36e70d4d60ce5faac954, демонструє реакцію показника DMMS+ на зміни в аспектах об'єктно-орієнтованого дизайну, що знижують супроводженість, але не були відзначені при вимірюваннях методами DMM. На відміну від DMM, що дорівнює 0.41, DMMS+ дорівнює 0.21 оскільки внесеними змінами було збільшено розмір класу, що за цією ознакою має низьку супроводженість, додано рядки в клас, але клас при цьому перевищує порогову складність і не втратив її, внаслідок змін. Також змінами внесено додатковий метод до класу, що перевищує порогове значення кількості класів, також внесені, направлення на збільшення зміни, в метод, що не втратив внаслідок змін, ризикованості за кількістю параметрів. Незважаючи на експерементальність підходу,

запропоновані зміни демонструють стабільність та ефективність вимірювання змін об'єктно-орієнтованого програмного забезпечення шляхом порівняльного аналізу внесених змін вихідного коду, що дає можливість проведення вимірювань супроводженості в процесах з методологічними підходами безперервної доставки та неперервної інтеграції при цьому інтерпретація результатів оцінювання дає можливість проведення причинно-наслідкового зв'язку та усунення недоліків.

5. Висновки.

Модель дельта супроводженості об'єктно-орієнтованого програмного забезпечення модифіковано шляхом розширення вимірюваних властивостей вихідного коду, визначення їх взаємозв'язку з підхарактеристиками супроводженості. Визначено способи вимірювання та порогові значення. Проведено практичну валідацію запропонованої моделі шляхом аналізу програмних продуктів з відкритим вихідним кодом, імплементованихі з застосуванням об'єктно-орієнтованої парадигми програмування, процесом розробки із застосуванням систем контролю версій та значною кількістю учасників процесу розробки та тривалою історією змін коду. Продемонстровано стабільність та ефективність вимірювання змін об'єктно-орієнтованого програмного забезпечення шляхом порівняльного аналізу внесених змін вихідного коду, що дає можливість проведення вимірювань супроводженості в процесах з методологічними підходами безперервної доставки та неперервної інтеграції. При цьому інтерпретація результатів оцінювання дає можливість проведення причинно-наслідкового зв'язку та усунення недоліків.

References

1. C. Jones, "The Economics of Software Maintenance in the Twenty FirstCentury," 2006, p.4

2. Plösch, R., Gruber, H., Korner, C., & Saft, M. (2010). A Method for Continuous Code Quality Management Using Static Analysis. In 7th International Conference on the Quality of Information and Communications Technology(QUATIC) (pp. 370–375).

3. A. Madi, O.K. Zein, and Seifedine Kadry. On the improvement of cyclomatic complexity metric. International Journal of Software Engineering and its Applications, 7:67–82, 01 2013.

4. Barry W. Boehm, John R. Brown, and Mlity Lipow. "Quantitative evaluation of software quality". In: Proceedings of the 2nd international conference on Software engineering. IEEE Computer Society Press, 1976, pp. 592–605. (Visited on 2017-01-31)

5. ISO/IEC 9126-2001. Software engineering - Product quality- Part 1: Quality model

6. Basili, V. R., Briand, L. C., & Melo, W. L. 1996. A validation of object-oriented design metrics as quality indicators. IEEE Transactions on software engineering, 22(10), 751–761.

7. ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.

8. ДСТУ ISO/IEC 25010:2016 Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Моделі якості системи та програмних засобів.

9. M. di Biase, A. Rastogi, M. Bruntink and A. van Deursen, "The Delta Maintainability Model: Measuring Maintainability of Fine-Grained CodeChanges," 2019 IEEE/ACM International Conference on Technical Debt(TechDebt), 2019, pp. 113-122, doi: 10.1109/TechDebt.2019.00030.

10. M. di Biase, A. Rastogi, M. Bruntink, and A. van Deursen. The Delta Maintainability Model: Measuring Maintainability of Fine-Grained Code ChangesTechnical Report, 2019.

11. Filó, Tarcísio G. S. and Mariza Bigonha. "A Catalogue of Thresholds for Object-Oriented Software Metrics." (2015).