УДК 519.72

І.О. Мартінкус, М.В. Ткачук, Р.О. Гамзаєв

*Національний технічний університет «Харківський політехнічний інститут», Харків*

# КОНСТРУЮВАННЯ ЛІНІЙОК ПРОГРАМНИХ ПРОДУКТІВ ІЗ ЗАСТОСУВАННЯМ ДОМЕННОГО МОДЕЛЮВАННЯ ТА МЕТРИК ПОВТОРНОГО ВИКОРИСТАННЯ КОДУ

*Розглянуто проблему застосування методів предметно-орієнтованого проектування (domain-driven design - DDD) в процесах створення програмного забезпечення (ПЗ), зокрема для побудови лінійок програмних продуктів (ЛПП) і звернено увагу на важливість забезпечення повторного використання коду (ПВК) в таких розробках. Проаналізовано взаємозв'язок показників якості та складності ПЗ, а також їх вплив на ступінь ПВК. Запропоновано нову концептуальну схему конструювання ЛПП із застосуванням методів DDD та метрик складності ПЗ, яка уможливлює отримання певного рівня ПВК.*

***Ключові слова:*** *предметно-орієнтоване проектування, доменна модель, лінійка програмних продуктів, повторне використання, мапа пам'яті, метрики коду.*

## Вступ

**Актуальність проблеми.** Використання сучасних методологій розробки програмного забезпечення (ПЗ) має на меті зменшення витрат на реалізацію відповідного проекту з урахуванням функціональних вимог та атрибутів якості до майбутньої програмної системи (ПС). Одним з найбільш ефективних шляхів вирішення цієї задачі є повторне використання (reuse) різних проектних активів (assets), а саме: доменних знань, специфікацій вимог, архітектурних рішень і, нарешті, програмного коду. Для досягнення цієї мети в сучасній інженерії ПЗ застосовується, зокрема, предметно-орієнтоване проектування (domain-driven design - DDD), в якому центральне місце займає поняття доменної моделі (domain model - DM) як засобу для концептуалізації знань щодо предметної області (ПрО) розробки ПС [1]. В першу чергу, такий підхід є ефективним для розробки нових ПС, але з урахуванням можливостей інструментальних засобів, що підтримують методи DDD, він також може бути застосованим для створення лінійок програмних продуктів (software product lines - SPL) [2]. При цьому під SPL розуміється сукупність компонентів ПЗ, які можуть бути певним чином налаштовані для багаторазового використання при розв'язанні різних задач у відповідній ПрО. У [2, 3] розглядається процес побудови SPL, але при цьому не беруться до уваги деякі важливі чинники, що мають істотний вплив на структуру та якість компонентів майбутньої SPL. Зокрема, одним з таких відомих, але недостатньо досліджених чинників, який має суттєвий позитивний вплив на ефективність процесу побудови SPL, є ступінь повторного використання програмного коду (code reusability extent - CRE) в окремих компонентах SPL.

Слід також зазначити, що при розробці SPL необхідністю враховувати такі суттєві чинники впливу як різні методи моделювання ПрО, механізми аналізу та відновлення (трасування) вимог до ПЗ та деякі інші [4].

**Постановка задачі.** Беручи до уваги вище зазначені аспекти сучасних процесів розробки ПЗ, в цій роботі пропонується новий підхід до розробки SPL із застосуванням методів та інструментальних засобів предметно-орієнтованого проектування DDD, побудови доменних моделей DM та оцінки ступеня повторного використання коду CRE у якості критерію ефективності процесу створення відповідної SPL.

## Результати досліджень

**Дослідження взаємозв'язку між показниками якості, метриками складності ПЗ та ступенем повторного використання програмного коду.** Як вже було зазначено вище, саме DDD –підхід до розробки ПЗ передбачає повторне використання різноманітних проектних артефактів, що в кінцевому рахунку має на меті забезпечити достатньо високі показники якості для ПС, що розробляється. Різноманітність видів цих проектних активів та досить складний, а також слабо формалізований характер зв'язків між ними ускладнюють та практично унеможливлюють їх кількісний аналіз, що є само по собі досить складною та актуальною проблемою програмної інженерії (див., напр. в [5]). Саме тому для структурування артефактів повторного використання, з можливістю подальшого якісного аналізу певних взаємозв'язків між ними, у цій роботі пропонується застосувати мапи пам'яті (mind map), які на відміну від більш формалізованих нотацій (таких, як UML, IDEF0 та ін.) дозволяють для будь-якої ПрО представити певні концептуальні сутності та їх семантичні зв'язки довільної природи [6]. Така мапа пам'яті для загальної класифікації та аналізу основних чинників впливу на процес повторного використання ПЗ може

бути побудована на підставі узагальнення результатів досліджень в [7, 8], і вона наведена на рис. 1. Так, зокрема, ця мапа містить наступні сутності та їх взаємозв'язки, що якісно описують будь-який процес повторного використання (Reuse), а саме: Область розробки (Development scope) визначає звідки отримані повторно використовувані компоненти (з того самого проекту або з іншого); Підхід (Approach) визначає які саме технічні методи будуть застосовані для реалізації ПВ; Область домену (Domain scope) визначає де відбувається ПВ - у рамках одного сімейства програмних систем або між декількома сімействами; Управління (Management) визначає наскільки систематично проводиться процес ПВ; Повторно використана сутність (Reused entity) визначає тип об'єкту, що підлягає ПВ.

Наступним кроком у цьому дослідженні є якісний аналіз взаємозв'язків між здатністю ПЗ до повторного використання (Reusability), такими показниками якості ПЗ як супроводжуваність (Maintainability), адаптивність (Adaptability) та зрозумілість (Understandability), а також показниками його структурної складності. Відповідна мапа пам'яті для їх якісного аналізу представлена на рис. 2. З неї можна зробити однозначний висновок відносно того, що вищезазначені показники якості ПЗ, а таким чином, і його здатність до повторного використання, залежать від рівня структурної складності відповідної ПС. Як відомо, цей показник для ПЗ, що розробляється на основі об'єктно-орієнтованого підходу, визначається за допомогою добре відомих метрик [9], а саме див. рис. 2.

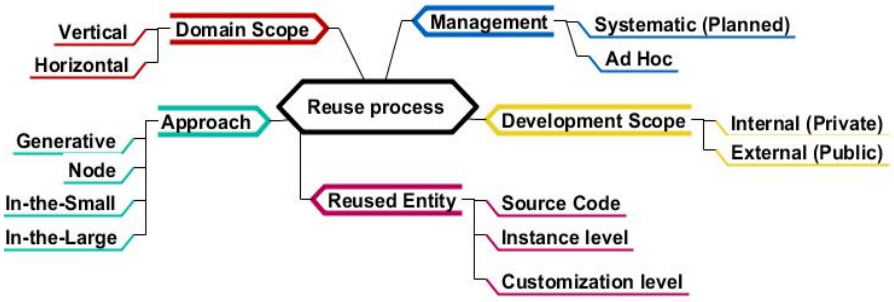


Рис. 1. Артефакти повторного викростання

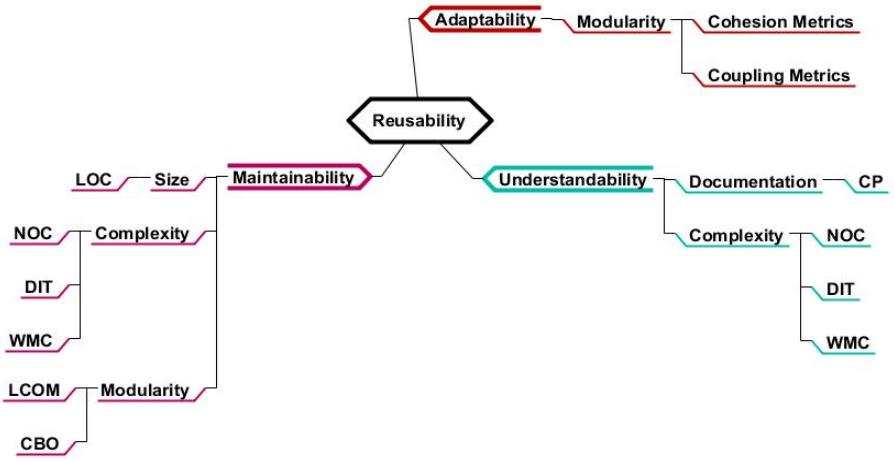


Рис. 2. Атрибути якості, метрики коду та їх вплив на повторне використання ПЗ

Також у [10] визначається кореляція між ступенем повторного використання та такими метриками як DIT, RFC, NOC, CBO та WMC. Розглянемо ці метрики більш детально. Глибина дерева успадкування (Depth of Inheritance Tree - DIT) визначається як найдовший шлях по ієрархії класів до даного класу від батьківського класу.

Відповідь для класу (Reponses for a Class - RFC) визначається як кількість різних методів, що можуть бути викликані, коли застосовується об'єкт класу. Кількість нащадків (Number of children - NOC) показує кількість безпосередніх нащадків класу. Зчеплення між об'єктами (Coupling between Object Classes - CBO) показує взаємодію об'єктів класу та визначає кількість сторонніх класів, з якими зв'язаний даний клас крім успадкованих класів. Зважена насиченість класу (Weighted Methods per Class) - метрика визначається сумою складності всіх методів класу, де кожен метод оцінюється підрахунком його цикломатичного числа. При цьому слід окремо зазначити [11], що при розробці ПЗ із застосуванням підходу на основі DDD, на теперішній час ще недостатньо проаналізовано вплив окремих методів доменного моделювання на складність відповідного програмного коду, який генерується на основі відповідної доменної моделі (DM). Тому є досить важливим завданням визначити

цю кореляцію, щоб зменшити витрати на реалізацію DDD -орієнтованих програмних проектів, і зокрема, на створення SPL.

Таким чином, на основі проведеного аналізу взаємозв'язку між показниками якості, метриками складності ПЗ та ступенем повторного використання програмного коду можливо запропонувати підхід до розробки SPL, який буде використовувати методи за інструментальні засоби підтримки доменного моделювання та в якості критерію ефективності враховувати ступінь повторного використання програмного коду CRE.

**Концептуальна схема розробки лінійок програмних продуктів із використанням методів доменного моделювання.** Запропонована схема підходу наведена на рис. 3, яка представляє процес конструювання SPL як систему управління із зворотним зв'язком. Основні її функціональні блоки взаємодіють у наступний спосіб:

– первинний опис певної ПрО тобто бізнес-вимоги користувачів (User stories) до функціональності майбутньої ПС слугує інформаційним базисом для побудови доменної моделі (DM) на концептуальному рівні;

– методи доменного моделювання (domain modeling method – DMM), напр.: FODA, JODA, ODM [12], та CASE-засоби їх інструментальної підтримки (domain modeling tool - DMT) [2], такі як FeatureIDE, Actifsource, та ін., за допомогою яких відбувається програмна реалізація доменної моделі (domain model realization – DMR),

– каркас програмного коду (code framework - CF), що може бути отриманий шляхом його генерації на основі DMR, і який потім, після певних доробок (напр., із застосуванням відповідних патернів кодування), може бути використано для побудови компонентів цільової лінійки програмних продуктів SPL;

– метрики оцінки рівня повторного використання коду (code reusability metrics – CRM), які дозволяють аналізувати отриманий CF, та в кінцевому рахунку робити мотивний висновок щодо можливості його ефективного застосування для створення потрібної SPL.

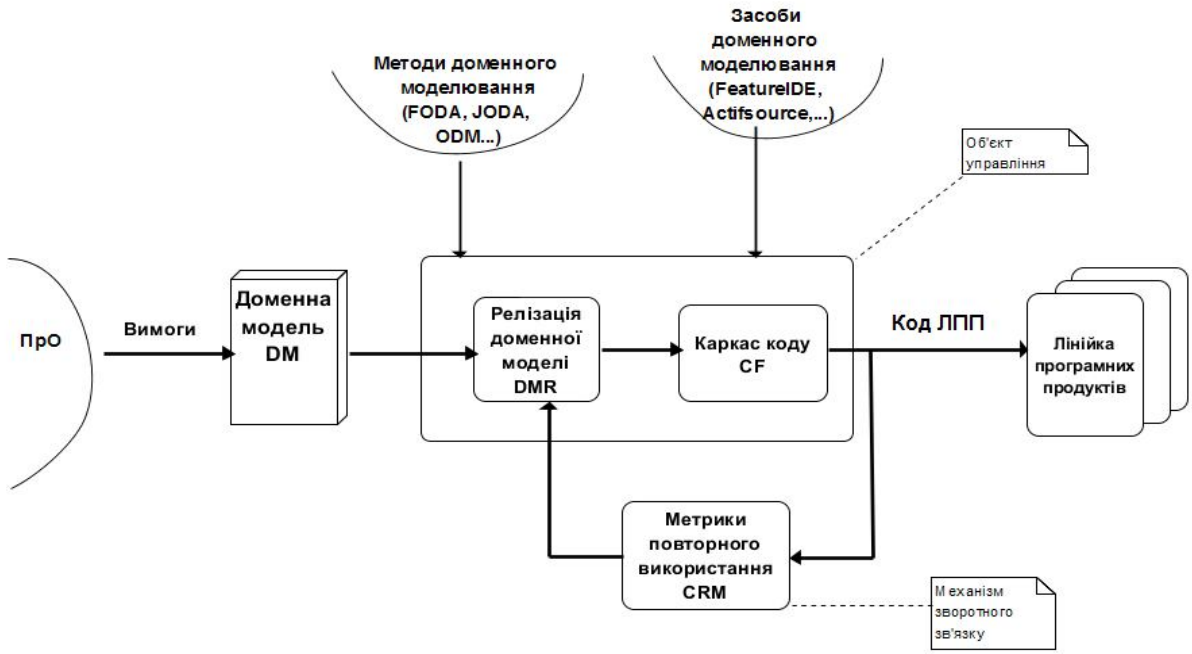


Рис. 3. Концептуальна схема розробки лінійок програмних продуктів із використанням методів доменного моделювання

Таким чином, схема на рис. 3 реалізує керований процес побудови SPL, в якому у якості вхідних даних використовуються первинний опис ПрО та побудована на його основі доменна модель DM, вихідним результатом є програмний код для побудови SPL, а зворотний зв'язок імплементують метрики оцінки CRM, застосування яких забезпечує можливість отримання потрібного ПВК. Слід зазначити, що концептуальна схема на рис. 3 не виключає можливості застосування інших метрик якості програмного коду - не тільки CRM, - для аналізу ефективності різних варіантів побудови SPL, і в цьому сенсі вона може розглядатися як удосконалення вже існуючих підходів до вирішення загальної проблеми варіабельності (variability) при розробці SPL.

**Розробка показника ступеня повторного використання коду на основі метрик структурної складності.** Аналіз сучасних публікацій, присвячених дослідженню проблеми підвищення ступеня CR при розробці ПС показує, що досить складно знайти зв'язок між певним рівнем CR та різними факторами впливу в розробці ПЗ. Так, зокрема, в [10] представлені результати емпіричного дослідження рівня CR в ПС з відкритим кодом і наведено набір метрик

щодо його оцінки. В роботах [8, 13] розглядаються різні підходи щодо оцінки CR. Ряд із них розглядають CR як відношення повторно використаних компонентів (коду) до загального обсягу компонентів (коду) та їх модифікацій (наприклад Reuse Percentage, Reuse level, Reuse size та ін). Але їх недолік полягає у тому, що вони можуть бути застосовані лише етапі супроводу (коли вже певні артефакти були повторно використані), і не має можливості застосувати їх на етапі проектування. Низка інших авторів [14, 15] розглядає оцінку CR із застосуванням ООП-метрик (та їх модифікацій). Особливо впливовими для оцінки ступеня CR визнаються метрики таких груп як зв'язність, зчеплення та глибина дерева успадкування. В даному дослідженні для визначення рівня повторного використання застосовано саме ООП-метрики. Їх основна перевага полягає у тому, шо їх можливо застосовувати на ранній стадії проектування ЛПП.

У роботі [11] запропоновано підхід для оцінки рівня CR у ПС, код якої отримано з використанням методів та засобів доменного моделювання. Запропонований підхід має три основні фази, а саме: 1) створення DM на основі опису ПрО у вигляді бізнес-вимог користувачів ПЗ (user story) та генерація відповідного програмного коду; 2) оцінка складності отриманого коду на основі вищезазначених ООП-метрик; 3) визначення експертної оцінки ступеня CR із застосуванням методу аналізу ієрархій (MAI). Як вже було зазначено, в роботі [10] розглядається кореляція між ООП-метриками та рівнем CR, а також неведені статистичні данні щодо цієї кореляції, отримані на основі роботи із проектами різного обсягу. На їх основі було визначено значення щодо ступеня впливу кожної метрики на рівень CR. В результаті було отримано наступний аналітичний вираз для визначення інтегрованого показника ступеня повторного використання коду $CR_{extent}$:

$$CR_{extent} = 0,12 * WMC + 0,04 * RFC + 0,27 * DIT + 0,36 * NOC + 0,21 * CBO, \quad (1)$$

де WMC, RFC, DIT, NOC, CBO є значеннями відповідних ООП-метрик складності для ПС, що розробляється. Таким чином, стає можливим вже на етапі проектування нової ПС порівнювати будь-які методи DM з точки зору їх впливу на ступінь CR у вихідному програмному коді цієї системи.

Вагові коефіцієнти у виразі (1) були отримані із застосуванням методу MAI, але слід зазначити, що для цього також можуть бути застосовані інші експертні підходи до розв'язання подібних багатокритеріальних задач, досить детальний огляд яких наведено, наприклад, в роботі [16].

З метою експериментального дослідження працездатності запропонованого підходу було розглянуто ПрО «Обробка персональної інформації учнів в системі автоматизації навчального закладу» та розроблено відповідну DM у двох реалізаціях: методами JODA та ODM (більш детально цей приклад розглянуто у [11]). На основі цих моделей та із використанням інструментальних засобів EMF та Actifsource було згенеровано програмний код та розраховані відповідні ООП-метрики (рис. 3 та 4):

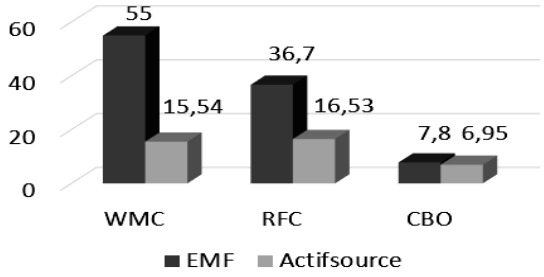


Рис. 3. Порівняння результатів для метрик WMC, RFC та CBO

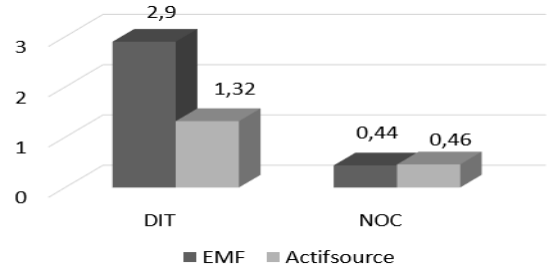


Рис. 4. Порівняння результатів для метрик DIT та NOC

Тоді на підставі виразу (1) були отримані наступні значення параметру $CR_{extent}$ для кожної із реалізацій, а саме:

$CR_{extent}(EMF) = 10,58$; $CR_{extent}(Actifsource) = 4,45$.

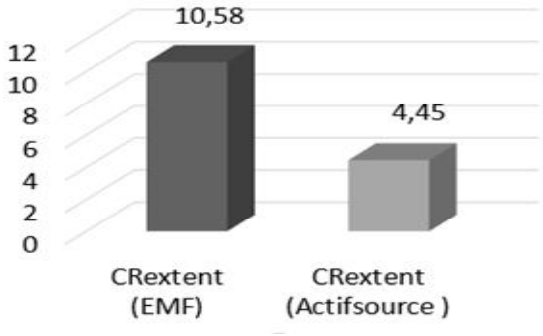


Рис. 5. Результати розрахунку параметра $CR_{extent}$

За отриманими результатами можна зробити висновок, що реалізація DM із ODM / EMF забезпечує більш високий рівень повторного використання у порівнянні із JODA /Actifsource реалізацією.

Слід зазначити, що для остаточного висновку щодо ефективності застосування того чи іншого методу доменного моделювання DMM з метою підвищення рівня повторного використання $CR_{extent}$ необхідно також оцінити витрати, які пов'язані з побудовою відповідних доменних моделей DM у процесі розробки ЛПП.

## Висновки

В роботі запропоновано новий підхід до розробки лінійок програмних продуктів (ЛПП) із застосуванням методів та інструментальних засобів предметно-орієнтованого проектування, побудови доменних моделей та кількісних метрик оцінки ступеня повторного використання коду у якості критерію ефективності процесу створення відповідної . Перевагами цього підходу є можливість аналітичного визначення інтегрованого показника ступеню повторного використання вихідного коду для більш ефективної генерації окремих програмних компонентів ЛПП.

В подальшому заплановано розвинути цей підхід шляхом порівняння результатів застосування інших методів розв'язання багатокритеріальних задач для визначення інтегрованого показника повторного використання вихідного коду, а також розробки кількісного критерію для оцінки ефективності застосування методів доменного моделювання.

## Список літератури


*1. Ткачук М.В., Гамзаев Р.О., Мартінкус І.О. Підхід до розробки лінійок програмних продуктів на основі успадкованих програмних систем із використанням методів доменного моделювання // Теоретичні та прикладні аспекти побудови програмних систем: матер. міжн. наук. конф., м. Київ, 5-9 грудня 2016р. / редкол.. М.С. Нікітченко та ін.. – Кіровоград : ЦОП «Авангард», 2016. - С. 236-241.*

*2. Reinhartz-Berger I. Domain Engineering: Product Lines, Languages, and Conceptual Models. Heidelberg, Springer, 2013.*

*3. Bosch J., Introducing agile customer-centered development in a legacy software product line / J. Bosch, P. M. Bosch-Sijtsema // Software: Practice and Experience, pp. 871-882, 2011.*

*4. Tkachuk M.V., Gamzayev R.O., Mayr H.C., Bolshutkin V.O.: Models and Tools for Effectiveness Increasing of Requirements Traceability in Agile Software Development // Проблемы программирования (Problems in Programming). – К.: НАН України. - 2012. - No 2-3 (спец. выпуск). – с.160-167.*

*5. Sommerville, I.: Software Engineering. Addison Wesley, 2011.*

*6. Guerrero JM, Ramos P. Mind Mapping for Reading and Understanding Scientific Literature. International Journal of Current Advanced Research 4(11), pp 485-487, 2015.*

*7. Frakes W., Tech V., Terrys C.: Software Reuse: Metrics and Models. INCODE Corporation. 1995.*

*8. Dubey A., Kaur H.: Reusability Types and Reuse Metrics: A Survey. International Journal of Computer Applications (0975 – 8887) Volume 131 – No.2, December 2015.*

*9. Paliwal N., Shrivastava V., Tiwari K.: An Approach to Find Reusability of Software Using Objet Oriented Metrics // International Journal of Innovative Research in Science, Engineering and Technology Vol. 3, Issue 3, March 2014.*

*10. Nandakumar A.N.: Constructing Relationship between Software Metrics and Code Reusability in Object Oriented Design, International Journal of Advanced Computer Science and Applications, Vol. 7, No. 2, 2016.*

*11. Tkachuk, M., Martinkus, I., Gamzayev, R., Tkachuk A.: An Integrated Approach to Evaluation of Domain Modeling Methods and Tools for Improvement of Code Reusability in Software Development // Heinrich C. Mayr, Martin Pinzger (Eds.): INFORMATIK 2016, Lecture Notes in Informatics (LNI), Vol. P-259: Kollen Druck+Verlag GmbH, Bonn, 2016. – pp. 143-156.*

*12. Ferré, X.: An Evaluation of Domain Analysis Methods.,In 4th CAiSE.IFIP8.1 International Workshop in Evaluation of Modeling Methods in Systems Analysis and Design, P.1-13, 1999.*

*13. Suri P. K., Garg N: Software Reuse Metrics: Measuring Component Independence and its applicability in Software Reuse. IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.5, May 2009.*

*14. Parul G., Kumar B.P.: Reusability Metrics for Object-Oriented System: An Alternative Approach International Journal of Software Engineering (IJSE), Malaysia, 1, 4, 62--73. 2010.*

*15. Gui Gui, Paul. D. Scott: Measuring Software Component Reusability by Coupling and Cohesion Metrics. Journal of Computers, vol. 4, no. 9, September 2009.*

*16. Лаврищева Е.М., Слабоспицкая О.А. Подход к экспертному оцениванию в программной инженерии // Кибернетика и системный анализ. – 2009. – № 4. – С. 151–168.*


*Надійшла до редколегії 2.03.2017*

**Рецензент:** д-р техн. наук, проф. С.І. Шматков, Харківський національний університет імені В.Н. Каразіна, Харків.

### КОНСТРУИРОВАНИЕ ЛИНЕЕК ПРОГРАММНЫХ ПРОДУКТОВ С ИСПОЛЬЗОВАНИЕМ ДОМЕННОГО МОДЕЛИРОВАНИЯ И МЕТРИК ПОВТОРНОГО ИСПОЛЬЗОВАТЬНИЯ КОДА

И.О. Мартинкус, Н.В. Ткачук, Р.А. Гамзаев

*Рассмотрена проблема применения методов предметно-ориентированного проектирования (domain-driven design - DDD) в процессах создания программного обеспечения (ПО), в частности для построения линеек программных продуктов (ЛППП) и обращено внимание на важность обеспечения повторного использования кода (ПИК) в таких разработках. Проанализирована взаимосвязь показателей качества и сложности ПО, а также их влияние на степень ПИК. Предложена новая концептуальная схема конструирования ЛППП с применением методов DDD и метрик сложности ПО, которая делает возможным получение определенного уровня ПИК.*

***Ключевые слова:*** *предметно-ориентированное проектирование, доменная модель, линейка программных продуктов, повторное использование, карта памяти, метрики кода.*

### SOFTWARE PRODUCT LINE CONSTRUCTION WITH DOMAIN MODELING AND SOFTWARE REUSE METRICS APPLICATION

I.O. Martinkus, M.V. Tkachuk, R.A. Gamzayev

*The problem of domain-driven design (DDD) methods' usage in software development is considered, especially for building for software product lines (SPL), and the attention is paid to an importance to provide the code reuse (CR) in such processes. The relationship between quality attributes and software complexity, and their impact on the level of CR is analyzed. The new conceptual scheme for SPL-design using DDD methods and software complexity metrics is proposed, which enables to obtain a certain CR-extent level.*

***Keywords:*** *domain driven design, domain model, software product line, reusing, mind map, code metrics.*