Кібернетика та системний аналіз

УДК 681.518:658.512

М.В. Ткачук, І.О. Мартінкус, К.А. Нагорний, Р.О. Гамзаєв

Національний технічний університет «Харківський політехнічний інститут», Харків

ПРО ОДИН ПІДХІД ДО ОЦІНКИ ЕФЕКТИВНОСТІ ЗАСТОСУВАННЯ МЕТОДІВ ДОМЕННОГО МОДЕЛЮВАННЯ ПРИ РОЗРОБЦІ СІМЕЙСТВ ПРОГРАМНИХ СИСТЕМ

Розглянуті особливості застосування методів та інструментальних засобів побудови доменних моделей (ДМ) в процесах розробки сімейств програмних систем (СПС) і зокрема, проблему забезпечення повторного використання коду (ПВК) в таких проектах. Проаналізовано взаємозв'язок показників ступеня ПВК та складності ДМ, введені необхідні для цього формалізовані визначення та евристичні твердження. Запропонована алгоритмічна модель процесу визначення коефіцієнту ефективності використання різних методів та засобів побудови ДМ, яка дозволяє врахувати як структурно-функціональну складність ДМ так і ступінь ПВК, який отримано на її основі. В результаті проведених програмних експериментів,із застосуванням побудованих метрик, отримані кількісні оцінки ефективності для тестових ДМ, що підтвердило працездатність запропонованого підходу.

Ключові слова: *сімейство програмних систем, лінійка програмних продуктів, доменна модель, повторне використання, програмний код, складність, алгоритмічна модель, метрики, ефективність.*

Вступ

Актуальність проблеми визначення ефективності застосування методів доменного моделювання при розробці сімейств програмних систем. Переважна більшість сучасних методологій розробки програмного забезпечення (ПЗ) має на меті досягнення двох основних цілей: по-перше, це зменшення витрат на реалізацію відповідного проекту з урахуванням функціональних вимог та атрибутів якості до майбутньої програмної системи (ПС), а по-друге - це скорочення часу, потрібного для виходу на ринок користувачів такої ПС [1–3]. Одним з найбільш ефективних шляхів вирішення цієї задачі є повторне використання (reuse) різних проектних активів (assets), а саме: специфікацій вимог (requirements) до ПЗ, еталонних архітектур (reference architectures), проектних патернів (design patterns) і, нарешті, вихідного коду (source code) ПС [1]. Для досягнення цієї мети в сучасній інженерії ПЗ широко застосовується концепція предметно-орієнтованого проектування (domain-driven design), в якій центральне місце займає поняття доменної моделі (domain model) як засобу для концептуалізації знань щодо предметної області (ПрО) розробки ПС [2]. Такий підхід дозволяє перейти від розробки окремих ПС до створення сімейств програмних систем (software system families), або лінійок програмних продуктів (software product lines) [1]. Ці терміни досить часто використовуються як синоніми, у тому сенсі, що обидва вони позначають сукупність компонентів ПЗ, які мають як певні загальні, так і деякі специфічні функціональні властивості, за рахунок чого вони можуть бути налаштовані для багаторазового використання при розв'язанні різних задач у відповідній ПрО. Але, як зазначається у деяких роботах [1; 3], певна різниця між сімейством програмних систем (СПС) та лінійкою програмних продуктів (ЛПП) полягає в тому, що члени СПС, як правило, застосовуються разом для вирішення проблемних задач у відповідній ПрО, в той час коли окремі елементи ЛПП можуть бути використані автономно у певній проектній ситуації.

Створення як СПС так і ЛПП передбачає наявність доменної моделі (ДМ) для заданої ПрО, на основі якої, із використанням відповідних інструментальних засобів (CASE-tools), можлива генерація каркасу програмного коду (source code framework), який потім має бути основою для створення програмних компонентів повторного використання (КПВ). Таким чином, з методологічної точки зору, розробка як СПС так і ЛПП використовує загальну парадигму генерувального програмування (generative programming) [3], яка передбачає можливість комплексного застосування як різних методів побудови ДМ для заданої ПрО, так і відповідних технологічних середовищ для створення, накопичення та контролю різних версій КПВ.

Метою роботи є розробка підходу до визначення ефективності застосування різних методів побудови ДМ та відповідних інструментальних засобів предметно-орієнтованого проектування для створення СПС та ЛПП. При цьому цілком логічним

є припущення, що на це мають вплив такі чинники як: (i) ступінь повторного використання програмного коду, який може бути згенерований на основі певної ДМ; (ii) структурно-функціональна складність самої ДМ. Оскільки в [4–5] вже було запропоновано рішення для задачі визначення оцінки чинника (i), то в цій роботі основна увага приділяється розв'язанню задачі оцінки чинника (ii), а також експериментальній перевірці працездатності запропонованого підходу.

Виклад основного матеріалу

Алгоритмічна модель процесу визначення ефективності застосування методів доменного моделювання

Зважаючи на те, що для оцінки впливу чинників (i)–(ii) на ефективність використання методів доменного моделювання для створення СПС та ЛПП необхідно враховувати значну кількість складних та слабоформалізованих інформаційних процесів та об'єктів, було запропоновано розробити для цього відповідну алгоритмічну модель [6]. Методологічною основою для побудови цієї моделі слугує наступна система евристичних тверджень, які було сформульовано на підставі вивчення сучасних інформаційних джерел та узагальнення власного досвіду щодо розробки ПЗ із застосуванням методів доменного моделювання, а саме:

Твердження 1. Існує певна множина методів доменного моделювання (множина M):

$$(\mathrm{DMM})_i \in \mathrm{M}, \mathrm{i} = 1, 2, 3..., \quad (1)$$

де DMM (DomainModelingMethod) – це ідентифікатор, що позначає окремий метод, який може бути застосовано для побудови ДМ.

Також існує множина відповідних технологій T:

$$(\mathrm{DMT})_j \in \mathrm{T}, \mathrm{j} = 1, 2, 3..., \quad (2)$$

де DMT (DomainModelingTechnology) – це ідентифікатор, що визначає певну технологію реалізації відповідного DMM.

В результаті застосування певного DMM із відповідним DMT, для обраної ПрО можуть бути побудовані відповідні ДМ з множини D:

$$(\mathrm{DM})_{i,j} \in \mathrm{D}, \quad (3)$$

де $\mathrm{DM}_{i,j}$ (DomainModel) – це ДМ, яка була отримана в результаті застосування i-го методу доменного моделювання та j-ї технології його реалізації.

Твердження 2. Доменні моделі з множини D мають різний рівень складності, тобто існує таке відображення ρ:

$$\rho: \mathrm{D} \to \mathrm{DMC}, \quad (4)$$

де DMC (DomainModelComplexity) – це множина можливих значень кількісного рівня структурно-функціональної складності доменних моделей.

Твердження 3. На основі кожної ДМ з множини D може бути отриманий відповідний каркас програмного коду, тобто існує таке відображення ϕ:

$$\phi: \mathrm{D} \to \mathrm{GCF}, \quad (5)$$

де GCF (GeneratedCodeFramework) – це множина каркасів програмного коду, який може бути використаний для побудови СПС або ЛПП.

Твердження 4. Каркаси коду з множини GCF мають різний показник ступеню повторного використання програмного коду (code reusability extent – CRE), тобто існує відображення σ:

$$\sigma: \mathrm{GCF} \to \mathrm{CRE}, \quad (6)$$

де CRE це множина можливих значень ступеню повторного використання програмного коду.

Твердження 5. Коефіцієнт ефективності застосування певної ДМ при розробці або супроводі ПЗ може бути визначений як відношення ступеню повторного використання програмного коду, який отримано із застосуванням цієї ДМ, до рівня її структурної складності, тобто:

$$\mathrm{K}_{\mathrm{Eff}}\left[(\mathrm{DM})_{i,j}, (\mathrm{GCF})_{i,j}\right] = \frac{(\mathrm{CRE})_{i,j}}{(\mathrm{DMC})_{i,j}}, \quad (7)$$

де $\mathrm{K}_{\mathrm{Eff}}$ – це коефіцієнт ефективності, а змінні $(\mathrm{CRE})_{i,j}$ та $(\mathrm{DMC})_{i,j}$ визначаються за виразами (6) та (4) відповідно.

Функціональна залежність (7) не може бути отримана аналітично, тому для її дослідження необхідно розробити сукупність інформаційних моделей, методів та метрик. Відповідна алгоритмічна модель (AM) дозволяє в формалізованому вигляді представити всі ці компоненти, и вона бути подана у наступному вигляді:

$$\mathrm{AM} = \left\langle \mathrm{InfoBase.WorkFlow(Methods)}, \mathrm{Metrics} \right\rangle, \quad (8)$$

де InfoBase – інформаційний базис моделі; WorkFlow(Methods) – сукупність алгоритмів (WorkFlow) реалізації методів (Methods) оцінки ефективності застосування ДМ; Metrics – колекція метрик, що застосовуються в відповідних алгоритмах моделі AM.

Інформаційний базис алгоритмічної моделі AM може бути поданий у вигляді наступного кортежу:

$$\mathrm{InfoBase} = \left\langle \mathrm{M}, \mathrm{T}, \mathrm{D} \right\rangle, \quad (9)$$

де M – це множина методів побудови ДМ за виразом (1), T – множина технологій їх реалізації за виразом (2), D – множина ДМ за виразом (3).

До методів оцінки ефективності Methods з виразу (9) належать:

- метод визначення ступеня повторного використання програмного коду, який запропоновано в [4–5];
- метод визначення показника складності ДМ, який подано за виразом (4) і який більш детально розглянуто нижче.

І, нарешті, колекція метрик Metrics з виразу (9) складається з:

- метрик оцінки структурної складності програмного коду [7], які застосовуються для аналізу каркасів, отриманих шляхом генерації на основі відповідної ДМ за виразом (5);
- метрик оцінки ступеня повторного використання коду [8], який потім має бути використаним для побудови відповідних СПС або ЛПП;
- метрик оцінки складності ДМ [9–10] з множини D за виразом (4)
- метрики визначення коефіцієнту ефективності застосування певної ДМ при розробці або супроводі СПС та ЛПП, яка подана за виразом (7).

Наступним кроком в запропонованому підході є безпосередня розробка методу визначення складності ДМ.

Розробка методу визначення складності доменних моделей на основі об'єктно-орієнтованого аналізу

Деякі існуючи метрики та методи оцінки складності доменних моделей. Аналізуючи сучасні публікації з цієї проблеми, можна зробити висновок, що задача оцінки структурної складності ДМ є досить актуальною як для розробки окремих складних ПС, так і для проектування ЛПП та СПС.

Так, в [9] наведено оцінку складності ДМ, яка враховує усі основні структурні компоненти, а саме: типи об'єктів ДМ (#OT), зв'язки між цими об'єктами (#ED) та операції над ними (#DO). Загальна структурна складність CD такої ДМ розглядається як відношення наявних асоціацій до кількості типів об'єктів: CD= (#ED) / (#OT), але при цьому кількість операцій (#DO) не враховується при розрахунку CD, і є окремим показником складності моделі. Таким чином, цей підхід не дозволяє комплексно оцінити складність відповідної ДМ.

В [10] проблема оцінки складності ДМ розглядається вже більш детально. Базуючись на методології GOPPRR (Graph-Object-Property-Port-Role-Relationship) для оцінки окремих структурних одиниць моделі, автори пропонують їх наступні кількісні оцінки:

$$C_{\text{interface}} = \#\text{Relationships} + \#\text{Roles} + \#\text{Constrains};$$

$$C_{\text{element}} = \#\text{Objects} + \#\text{Ports};$$

$$C_{\text{property}} = \#\text{Properties},$$

де $C_{\text{interface}}$, C_{element}, C_{property} є відповідно кількісними показниками складності структурних елементів ДМ. Для кінцевої оцінки складності ДМ пропонується застосовувати наступний вираз:

$$C_{\text{Overall}} = C_{\text{interface}} + C_{\text{element}} + C_{\text{property}}.$$

Але при цьому цей метод також не враховує функціональну складність ДМ, а оцінює лише складність її структури.

Враховуючи вищезазначені недоліки деяких існуючих методів оцінки складності ДМ, необхідно запропонувати комплексний підхід до оцінки їх структурно-функціональної складності, що дозволить в подальшому отримати єдиний інтегральний показник для визначення ефективності застосування окремих методів та засобів доменного моделювання в процесах розробки СПС та ЛПП .

Підхід до визначення складності ДМ на основі об'єктно-орієнтованого аналізу. В роботі [11] запропоновано підхід для оцінки складності архітектурних моделей ПС, які розробляються із застосуванням пост об'єктно-орієнтованих технологій (ПООТ). Він передбачає можливість отримання аналітичних виразів для оцінки питомої складності застосування окремих ПООТ на підставі врахування їх специфічних компонентів та відповідних вагових коефіцієнтів, які можуть бути отримані експертним шляхом. Подібний підхід можна застосувати й для оцінки складності ДМ. Для цього необхідно ввести формалізовані визначення для всіх основних артефактів в контексті побудови відповідної ДМ, приклад фрагменту якої наведено на рис. 1.

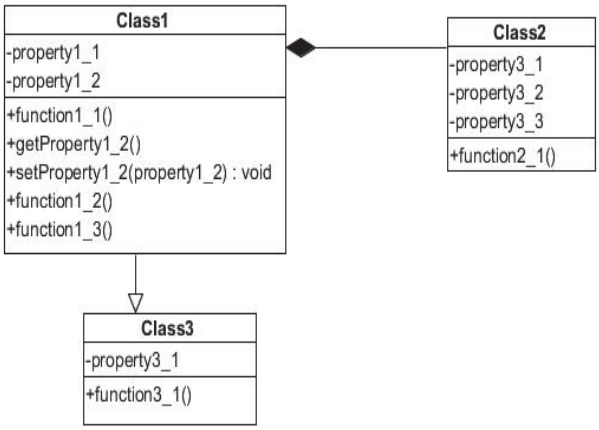

Рис. 1. Приклад фрагменту опису ДМ

Визначення 1: Доменна модель (Domain Model – DM) – це кортеж:

$$\text{DM} = \langle \underline{C}, \underline{R} \rangle, \qquad (10)$$

до складу якого входить множина класів (елементів) моделі $\underline{C}$ та множина зв'язків між ними $\underline{R}$.

Визначення 2: Клас (class – C) – це кортеж:

$$C = \langle \underline{P}, \underline{F} \rangle, \quad (11)$$

де $\underline{P} = \{p_i\}, i = \overline{1,n}$ – множина властивостей класу, яка дозволяє зберігати стан сутності домену; $\underline{F} = \{f_j\}, j = \overline{1,m}$ – множина функцій (або методів) класу, яка представляє його функціональне наповнення.

Визначення 3: Множина зв'язків між елементами моделі:

$$\underline{R} = \{r_{i,j}\}, \quad (12)$$

де $r_{i,j}$ – це зв'язок між i-м та j-м елементом ДМ. В той же час кожен з елементів множини R відноситься до одного з елементів множини RT.

Визначення 4: Множина типів зв'язків між елементами:

$$\underline{RT} = \{As, Ag, Cm, In\}, \quad (13)$$

де множина RT включає 4 основних типа зав'язків в доменній моделі, а саме [12]: асоціація (Association – As), агрегація (Aggregation – Ag), композиція (Composition – Cm) та успадкування (Inheritance – In).

В [13] проведено кількісний аналіз впливу кожного з цих типів зв'язків на поведінку об'єктів (класів) у відповідній ДМ. Аналітично цю залежність можна виразити наступним шляхом:

$$H_{As} < H_{Ag} < H_{Cm} < H_{In}, \quad (14)$$

де параметри $H_{As}, H_{Ag}, H_{Cm}, H_{In}$ є коефіцієнтами ступеню впливу відповідного типу зв'язку на загальну складність ДМ.

Метод розрахунку структурно-функціональної складності ДМ. Метод визначення структурно-функціональної складності ДМ реалізує відображення ρ (вираз 4), що передбачає послідовне виконання наступних етапів: 1) розрахунок складності окремого класу ДМ, 2) розрахунок складності зв'язків між класами та 3) безпосередньо визначення загальної складності моделі. При цьому вагові коефіцієнти для кожного типу компонентів ДМ визначаються експертним шляхом, наприклад, із застосуванням методу аналізу ієрархій [14].

Крок 1. Розрахунок складності окремого класу.

Враховуючи те, що вплив функціональних властивостей класу є більш вагомим на загальну складність ДМ, складність кожного класу ДМ можна отримати за допомогою наступного виразу:

$$CC = 0,3 \times PC + 0,7 \times FPC, \quad (15)$$

де CC (ClassComplexity) – це параметр, що кількісно визначає загальну структурно-функціональну складність відповідного класу; PC (PropertyComplexity) – параметр, що визначає складність властивостей класу (складність його структури); FPC (FuncPropertyComlexity) – параметр, що визначає складність функціональних властивостей класу (тобто складність поведінки класу).

Аналогічно, кількісно вплив на складність класу його «простих» (тобто атомарних) атрибутів та їх колекцій може бути представлена наступним виразом:

$$PC = 0,4 \times \#STP + 0,6 \times \#CTP, \quad (16)$$

де #STP (SipmleTypeProp) – це кількість властивостей будь-яких «простих» типів (не колекцій) #CTP (CollectionTypePrperty) – кількість властивостей будь-яких типів колекцій

Згідно з тим, що визначення ДМ передбачає лише сигнатуру її функцій, то для визначення їх впливу на загальну складність окремого класу пропонується наступний вираз:

$$FPC = \#FP, \quad (17)$$

де #FP **(**FunctionalProperty) – це кількість функціональних властивостей (операцій) відповідного класу.

Крок 2 Розрахунок складності зав'язків.

Відповідно до виразу (14) можна визначити наступні вагові коефіцієнти для кожного типу зв'язків між класами ДМ:

$$W_{As} = 0.1, W_{Ag} = 0.2, W_{Cm} = 0.3, W_{In} = 0.4. \quad (18)$$

З урахуванням коефіцієнтів з виразу (19) загальну складність зв'язків у певній ДМ пропонується отримати за допомогою наступного виразу:

$$RC = W_{As} * (\#As) + W_{Ag} * (\#Ag) + \\ + W_{Cm} * (\#Cm) + W_{In} * (\#In), \quad (19)$$

де RC (RelationalComplexity) – це параметр, який визначає загальну складність зв'язків у ДМ; #As (AssociationRelation) – це кількість зв'язків типу асоціації; #Ag (AggregationRelation) – кількість зв'язків типу агрегації, #Cm (CompoistionRelation) – кількість зв'язків типу композиції; і нарешті #IR (InharitanceRelation) – це кількість зв'язків типу наслідування.

Крок 3. Розрахунок загальної структурно-функціональної складності ДМ.

Для фінальної оцінки структурно-функціональної складності ДМ пропонується наступний вираз:

$$DMC = 0,7 \times \#Class \times CC + 0,3 \times RC, \quad (20)$$

де DMC (DomainModelComplexity) – це параметр, що визначає загальну структурно-функціональну складність ДМ, #Class – кількість класів у цій ДМ.

Таким чином, сукупність аналітичних виразів (15–20) визначає алгоритм реалізації методу визначення структурно-функціональної складності відповідної ДМ, який є необхідною складовою частиною алгоритмічної моделі АМ, яка подана за виразом (9).

Процедура визначення ефективності застосування методів доменного моделювання

На основі алгоритмічної моделі, поданої за виразом (8), з урахуванням складу її компонентів, які визначаються за формулами (1–6) та (9), і з застосуванням алгоритму методу оцінки структурно-функціональної складності ДМ, який представлено формулами (15–20), можливо запропонувати процедуру визначення коефіцієнту ефективності застосування методів доменного моделювання за формулою (7). Ця процедура у вигляді UML-діаграми дій [10] представлена на рис. 2.

Рис. 2. Процедура визначення коефіцієнту ефективності застосування методів доменного моделювання

Згідно виразу (7), для визначення K_{Eff} необхідно визначити дві змінні: CRE та DMC. Етап щодо визначення складності доменної моделі DMC детально розглянуто у попередньому розділі. Етап «Отримання каркасу програмного коду GCF» реалізує відображення φ (вираз 5), що можливо завдяки застосуванню відповідних інструментальних засобів доменного проектування. Етап «Визначення ступеня повторного використання коду CRE» (вираз 6) передбачає реалізацію відображення σ, застосовуючи відповідні метрики. Ці етапи більш детально розглянуто у [4–5].

Експериментальна перевірка працездатності запропонованого підходу та аналіз отриманих результатів

Тестові доменні моделі. Для тестування запропонованого підходу було розроблено 2 тестові ДМ для ПрО «Обробка персональної інформації учнів в системі автоматизації навчального закладу» із застосування методів доменного моделювання ODM (Organizational Domain Modeling) та JODA (Joint integrated avionics Object oriented Domain Analysis), які програмно можуть бути реалізовані за допомогою таких інструментальних CASE-засобів як EMF (Eclipse Modeling Framework) та Actifsource (більш детально методика розробки та імплементації цих ДМ представлена в [4]). Приклад побудови ДМ для варіанту застосування засобів JODA / Actifsource представлено на рис. 4, а згенерований на її основі Java-код – на рис. 3. Код отриманий у засобі Actifsource містить 2 пакета класів: *student* (містить додаткові ресурсні класи) та *student.javamodel* (містить реалізацію та інтерфейси).

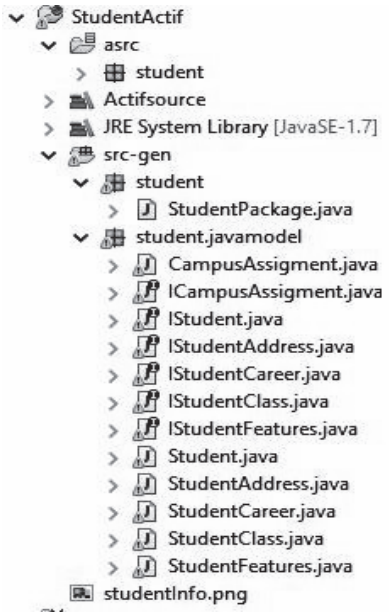

Рис. 3. Каркас коду GCF за технологію JODA/Actifsource

Результати обчислювальних експериментів та їх аналіз. Згідно запропонованої процедури для визначення коефіцієнту ефективності застосування методів доменного моделювання (схема на рис. 2) необхідно визначити такі характеристики ДМ як CRE та DMC. Для визначення структурно-функціональної складності DMC, на основі аналізу структури класів у тестових ДМ та із застосуванням *Визначень 1 – 4* були отримані дані для їх реалізацій за допомогою засобів JODA / Actifsource (табл. 1–2) та ODM / EMF (табл. 3–4) відповідно.

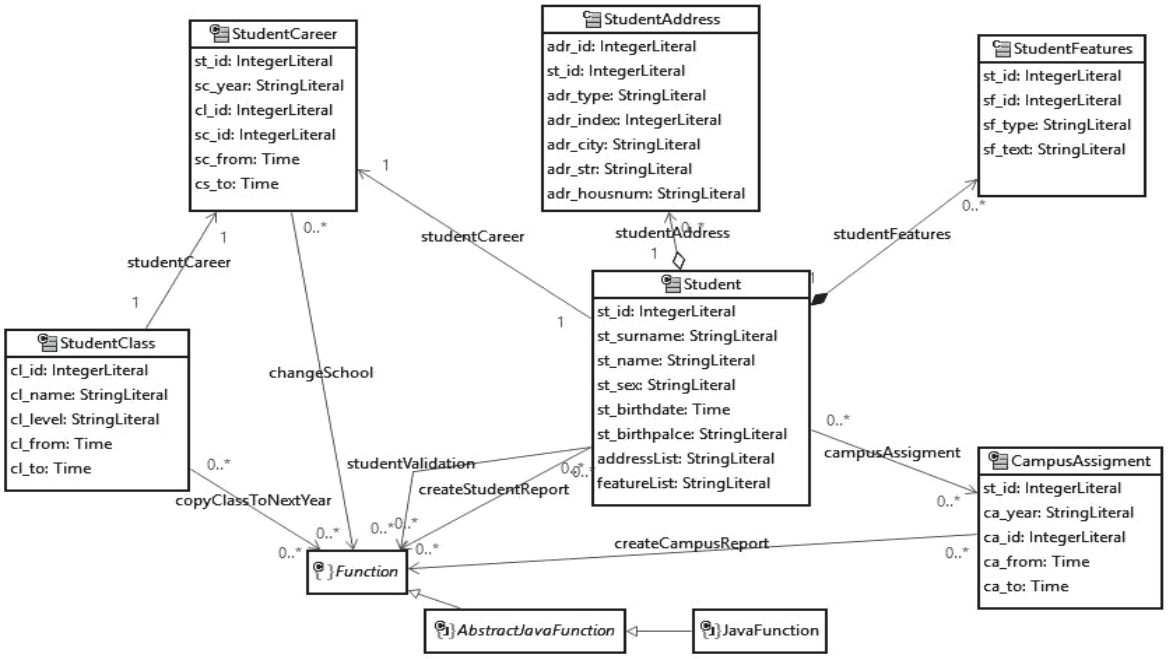

Рис. 4. Приклад доменної моделі за технологією JODA / Actifsource

Таблиця 1

Оцінка складності класів ДМ
для реалізації JODA /Actifsource

	Class	#STP	#CTP	#FP
1	Student	6	2	2
2	StudentAddress	7	0	0
3	SturdentFeature	4	0	0
4	StudentCareer	6	0	1
5	StudentClass	5	0	1
6	CampusAsignement	5	0	1
7	Function	0	0	0
8	AbstractFunction	0	0	0
9	JavaFunction	0	0	0
	∑	**33**	**2**	**5**

Таблиця 2

Оцінка складності відношень ДМ
для реалізації JODA /Actifsource

Association	8
Aggregation	1
Composition	1
Inheritance	2

Таблиця 3

Оцінка складності класів ДМ
для реалізації ODM / EMF

	Class	#STP	#CTP	#FP
1	Student	6	2	2
2	StudentAddress	7	0	0
3	SturdentFeature	4	0	0
4	StudentCareer	6	0	1
5	StudentClass	5	0	1
6	CampusAsignement	5	0	1
	∑	33	2	5

Таблиця 4

Оцінка складності відношень ДМ
для реалізації ODM / EMF

Association	3
Aggregation	0
Composition	2
Inheritance	0

Враховуючи дані з табл. 1–4 та із застосуванням формул (16–21) були отримані наступні параметри відповідних ДМ:

для реалізації JODA /Actifsource

*PC(JODA / Actifsource)=0.4*33+0.6*2 = 14.4*

FPC(JODA/Actifsource)=5

*CC(JODA/Actifsource)=0.3*14.4+0.7*5=7.82*

*RC(JODA /Actifsource)= 0,1*8+0.2*1 +*
*+0,3*1+0,4*2=2,1*

*DMC(JODA / Actifsource)=0.7*9*7.82+*
*+ 0.3*2.1=49,9*

для реалізації ODM / EMF

*PC(ODM / EMF)=0.4*33+0.6*2=13.2+1.2=14.4*

FPC(ODM / EMF)=5

*CC(ODM/EMF)=0.3*14.4+0.7*5=4.32+3.5=7.82*

*RC(ODM/EMF)= 0,1*3+0.2*0+0,3*2+0,4*0=0,9*

*DMC(ODM/EMF)=0.7*6*7.82+0.3*0.9=33.11*

В той же час крім DMC, згідно процедури (рис. 2) для визначення коефіцієнту ефективності застосування методів доменного моделювання необхідно визначити ступень повторного використання коду CRE, який може бути згенерований на основі відповідних ДМ (формули (5–6)), що реалізують відображення φ та σ. На основі методу визначення CRE, який детально розглянуто в [4–5] для тестових

ДМ було отримано наступні значення цього показника:

$$\mathrm{CRE}(\mathrm{JODA}/\mathrm{Actifsource}) = 7{,}78;$$

$$\mathrm{CRE}(\mathrm{ODM}/\mathrm{EMF}) = 16{,}67.$$

Використовуючи формулу (7) можливо отримати кінцеві значення для коефіцієнтів ефективності застосування відповідних технологій доменного моделювання, які представлені в табл. 5.

Таблиця 5

Результати обчислення ефективності застосування технологій доменного проектування

	CRE	DMC	K_{Eff}
JODA / Actifsource	7,78	49,90	0,156
ODM / EMF	16,67	33,11	0,503

У графічному вигляді ці результати представлені на рис. 5.

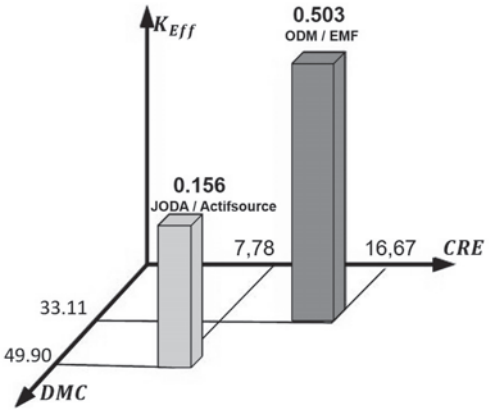

Рис. 5. Графічна інтерпретація отриманих результатів

За отриманими результатами можна зробити висновок, що реалізація ДМ із використанням технологій ODM / EMF забезпечує більш високий коефіцієнт ефективності к порівнянні із технологією JODA / Actifsource. Таким чином, для даної ПрО ці показники визначають доцільність використання саме цієї технології для даної ПрО.

Висновки

В роботі було розглянуто особливості застосування методів та інструментальних засобів побудови доменних моделей (ДМ) в процесах розробки сімейств програмних систем (СПС), лінійок програмних продуктів (ЛПП), і зокрема, проблему забезпечення повторного використання коду (ПВК) в таких проектах. Було проаналізовано взаємозв'язок показників ступеня ПВК та складності ДМ, введено необхідні формалізовані визначення та сформульовані відповідні евристичні твердження. Запропонована алгоритмічна модель процесу визначення коефіцієнту ефективності використання різних методів та засобів побудови ДМ, яка дозволяє врахувати як структурно-функціональну складність самої ДМ так і ступінь ПВК, який отримано на її основі. В результаті проведених програмних експериментів, із застосуванням побудованих метрик, отримані кількісні оцінки ефективності для тестових ДМ, що підтвердило, в цілому, працездатність запропонованого підходу.

Список літератури

1. Reinhartz-Berger I. Domain Engineering: Product Lines, Languages, and Conceptual Models / I. Reinhartz-Berger. – Heidelberg, Springer, 2013.

2. Эванс Э. Предметно-ориентированное проектирование (Domain-Driven Development – DDD): Структуризация сложных программных систем: пер. с англ. / Э. Эванс., 2011. – 448 с.

3. Нові теоретичні засади технології виробництва сімейств програмних систем у контексті генерувального програмування : монографія / К.М. Лавріщева та ін. – Ін-т програм. систем. НАН України. – К., 2011. – 277 с.

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

5. Мартінкус І.О. Конструювання лінійок програмних продуктів із застосуванням доменного моделювання та метрик повторного використання коду / І.О. Мартінкус, М.В. Ткачук, Р.О. Гамзаєв // Системи управління, навігації та зв'язку: зб. наук. пр. ЦНДІ НУ. – К., 2017. – Вип. 3(43). – С. 93-97.

6. Сергиенко А.М. Алгоритмические модели обработки потоков данных / А.М. Сергиенко, В.П. Симоненко // Электронное моделирование. – 2008. – Т. 30, № 6. – С. 49-60.

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

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

9. Preschern C. Evaluation of Domain Modeling Decisions for two identical Domain Specific Languages / C. Preschern, N. Kajtazovic, C. Kreiner // Lecture Notes on Software Engineering. – February 2014. – Vol. 2. – No. 1. – Pp. 37-41.

10. Poels G. A quantitative assessment of the complexity of static conceptual schemata for reference types of front-office / G. Poels, G. Dedene, S. Viane // In Proceedings of the Fifth International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE2001). – Hungary: Budapest.

11. Литвинчук М.М. Архітектурні моделі оцінки складності пост об'єктна-орієнтованих технологій розробки програмних систем / М.М. Литвинчук, К.А. Нагорний, М.В. Ткачук // Вісник ХНУ імені В.Н. Каразіна, Серія «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління». – 2012. – № 1000. – С. 225-235.

12. Unified Modeling Language (UML) Офіційний вєбресурс уніфікованої мови моделювання UML, що розробляється консорціумом OMG – Object Management Group [Електронний ресурс] – Режим доступу: http://www.uml.org.

13. Kang D. A Complexity Measure for Ontology based on UML / Dazhou Kang, Baowen Xu, Jianjiang Lu // Distributed Computing Systems, 2004. FTDCS 2004. Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems. – Pp. 222-228.

14. Саати Т.Л. Принятие решений. Метод анализа иерархий / Т.Л. Саати; пер. с англ. – М.: Радио и связь, 1993. – 278 с.

Надійшла до редколегії 11.07.2017

Рецензент: д-р техн. наук проф. Г.М. Жолткевич, Харківський національний університет ім. В.Н. Каразіна, Харків.

ОБ ОДНОМ ПОДХОДЕ К ОЦЕНКЕ ЭФФЕКТИВНОСТИ ПРИМЕНЕНИЯ МЕТОДОВ ДОМЕННОГО МОДЕЛИРОВАНИЯ ПРИ РАЗРАБОТКЕ СЕМЕЙСТВ ПРОГРАММНЫХ СИСТЕМ

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

Рассмотрены особенности применения методов и инструментальных средств построения доменных моделей (ДМ) в процессах разработки семейств программных систем (СПС) и, в частности, проблема обеспечения повторного использования кода (ПИК) в таких проектах. Проанализирована взаимосвязь показателей степени ПИК и сложности ДМ, введены необходимые для этого формализованные определения и эвристические утверждения. Предложена алгоритмическая модель процесса определения коэффициента эффективности использования различных методов и средств построения ДМ, которая позволяет учесть как структурно-функциональную сложность ДМ так и степень ПИК, полученный на ее основе. В результате проведенных программных экспериментов с применением построенных метрик, получены количественные оценки эффективности для тестовых ДМ, которые подтвердили работоспособность предложенного подхода.

Ключевые слова: *семейство программных систем, линейка программных продуктов, доменная модель, повторное использование, программный код, сложность, алгоритмическая модель, метрики, эффективность.*

TOWARDS THE EFFECTIVENESS ASSESSMENT APPROACH OF DOMAIN MODELING METHODS APPLICATION IN SOFTWARE PRODUCT FAMILY DEVELOPMENT

M. Tkachuk, I. Martinkus, K. Nagornyi, R. Gamzayev

Some essential features of methods and tools used for domain models (DM) construction in software system families development are considered, and a special attention is paid to the problem of code reusing (CR) in such projects. The interrelation between CR extend (CRE) and degree of DM complexity has been analyzed, and the appropriate formal definitions and heuristic rules related for this issue were introduced. The algorithmic model to estimate a usage efficiency coefficient of various methods and tools for DM constructing is proposed, which consists of an information base, a set of DM methods (algorithms), and a collection of several numerical metrics. This model allows to take into account both a structure-functional complexity of any DM, and CRE of a code framework which can be generated basing of this DM.

The proposed approach was tested experimentally to compare the efficiency coefficient usage of 2 different combinations of domain modelling methods and tools, namely: Organizational Domain Modelling (ODM) with Eclipse Modelling Framework (EMF) and Joint integrated avionics Object oriented Domain Analysis (JODA) with Actifsource tool. The conducted experiments have shown the following values for a target coefficient: 0,506 for ODM / EMF and 0,156 for JODA / Actifsource respectively. In general, the elaborated approach can be used to estimate an efficiency usage of any DM methods and tools within the development framework of software families and software product lines.

Keywords: *software system families, software product lines, domain model, reuse, source code, complexity, algorithmic model, metrics, effectiveness.*