УДК 681.3

К.М. Лавріщева

ГЕНЕРУВАЛЬНЕ ПРОГРАМУВАННЯ ПРОГРАМНИХ СИСТЕМ І ЇХ СІМЕЙСТВ

У роботі наведено аналіз основних положень і досягнень сучасного генерувального програмування, сформульовані нові напрями його розвитку щодо систематизації і удосконалення базових засобів подання програмних систем, сімейств систем предметно-орієнтованими мовами і компонентами повторного використання, а також нових інструментів їхній трансформації з генераційним змістом та адаптованих до умов генерації моделі якості та життєздатності систем сімейств.

Вступ

На сьогодні в галузі програмування постійно іде динамічний розвиток комп'ютерних систем (Internet, Web-сервіси, Web-застосування тощо), систем загального призначення (COM, CORBA, Java, .NET й інші), середовищ програмування (Eclipse, JBuilder, MS Visual Studio тощо), серверів застосувань (IIS, JBoss, Geronimo), систем керування базами даних (MySQL, MSSQL, Oracle) тощо.

Все це сприяє виробництву програмних продуктів для їхнього використання, а також породженню нових і різнорідних програмних об'єктів, структур даних, інформаційних ресурсів, а також мов їх опису, засобів та інструментів їх специфікації і оброблення у сучасних операційних середовищах. Але методи та інструментальні засоби цих середовищ у повної мірі не відповідають підходам до якісної реалізації багатьох задач деяких складних систем сімейств з предметних областей і доменів. Багато з вироблених програмних елементів членів сімейств не завжди доводять до їх повторного використання і тому на практиці досить часто їх розробляють як нові програмні компоненти, на що витрачають значні додаткові ресурси [1].

Наприклад, механізми створення і застосування компонентів повторного використання (КПВ) для заново будованих програмних систем (ПС) і сімейств систем у межах широко розповсюдженого об'єктно-орієнтованого підходу (ООП) практично відсутні. В цілому напрямок КПВ у програмної інженерії сам по собі досяг великого впровадження як у плані їхнього подання і зберігання, так і нових інженерних напрямів виготовлення з них складних систем:

– *інженерія* КПВ (reuse engineering) – систематична і цілеспрямована діяльність щодо визначення їхній можливості застосовуватися в різних ПС або об'єднуватися у складні сукупності;

– *прикладна інженерія* (application engineering) – процес виробництва нових ПС із КПВ як одиночних, самостійних систем або як складових деякої предметної області (ПрО), домену, що автоматизують;

– *інженерія домену* (domain engineering) – це сімейство процесів виробництва окремих членів сімейства ПС і сімейств програмних систем (СПС) у цілому, яке базується на використанні готових КПВ у межах деякої ПрО або їхніх сукупностей у сімействах систем. Ця інженерія базується на мовах опису окремих компонентів і специфіки членів сімейства в DSL (Domain Specific Language), моделі домену GDM (Generative Domain Model) та методу аналізу, основаного на характеристиках домену FODA (feature Oriented Domain Analysis) [2–5], функціональних вимогах (functional requirements) та вимогах якості (quality requirements) відповідно моделі якості, відображаючій аспекти генерації членів сімейства.

Ці інженерії виробництва є головними напрямами програмної інженерії і *генерувального програмування* (ГП) сімейств систем. Їх виробництво здійснюють засобами та інструментами різних типів програмування (компонентного, аспектного, агентного, сервісного тощо) при побудові відповідних КПВ, складних ПС та

членів сімейств систем. Серед типів програмування – аспектне програмування сприяє розв'язанню загальних системних проблем взаємодії, синхронізації та змінювання КПВ у системах сімейства. Головним цільовим об'єктом ГП є *домен* і механізми генерації будь-якого його елемента (члена) сімейства та КПВ, що використовують для розв'язання конкретних задач *домену* або ПрО.

Далі аналізуються основні положення ГП та концепцій розвитку теоретичних та інструментальних засобів його підтримки при виробництві СПС.

1. Основи проблематики генерувального програмування

Генерувальне (generative programming) програмування – нова сучасна парадигма програмної інженерії, орієнтована на інженерію СПС для певних ПрО і генерацію окремих ПС як членів сімейства СПС. Тобто, в межах цього програмування склалося достатньо засобів для виробництва КПВ, одиничного використання ПС та сімейств систем згенерованих з накопичених різнорідних програмних елементів, що задовольняють вимогам замовника [3, 4]. Головний метод розробки і генерації GSD (generative software development) програмних продуктів домену базується на різних підходах з специфікації цього домену, опису окремих членів СПС різними мовами програмування (МП), виходячи з вимог до членів сімейства, та механізмів автоматизованої генерації адаптованих до завдань окремих членів сімейства і СПС у складі доменів.

1.1. Визначення домену

Під *доменом* або проблемною областю розуміють множину систем, яка має набір понять, що однаково трактуються всіма системами та особами, які мають справи з цими поняттями.

У межах кожного домену розв'язують задачі, які реалізують спільні й варіантні аспекти його проблем, а також правила маніпулювання ними. Завдання домену подають у моделі домену або на перехресті декількох областей, що входять до складу спільної мови між розробниками, користувачами і замовником системи.

Тобто, кожний домен має притаманну йому систему знань, а також характерні властивості (атрибути), відношення та правила поведінки. Головним об'єктом кожного члена СПС або усього сімейства є КПВ. Посередником між фахівцями ПрО і розробниками різних членів систем сімейства є специфікована *модель* ПрО за її поняттями, комунікацією між окремими системами сімейства в складі КПВ та *процесів* генерування кожного члена *сімейства* шляхом трансформації опису специфіки ПрО в предметно-орієнтованої мові DSL в опис проміжних мов типу HTML, XML або в опис сучасних МП з подальшою генерацією вихідного коду для кожного члена сімейства ПрО.

Розроблення домену виконують у просторі проблеми (speace problem) і просторі рішень (speace solution) в поняттях, які розуміють однаково всі системи та їхні користувачі.

Простори проблем і рішень. У просторі проблем (speace problem) виконують аналіз домену, виявлення понять ПрО, визначення компонентів моделі характеристик та завдань домену (рис. 1). Простір рішень (speace solution) – це засоби розв'язання завдань ПрО та механізми їхній взаємодії та адаптації. Результатом розгляду домену в цих просторах є конфігураційна база знань, яка зберігає зв'язки характеристик членів сімейства та їхні залежності, операції конструювання та об'єднання за GDM-моделлю [4].

Простір проблеми включає предметні поняття ПрО для майбутньої ПС, нові компоненти та КПВ за відповідними об'єктами і їх характеристиками. Як результат аналізу простору проблем є модель характеристик з компонентів і об'єктів, змінювані параметри різних членів сімейства та проектні рішення, зв'язані з операціями взаємодії членів сімейства між собою та з середовищем.

У базисі конфігурації ГП відображені знання загального і спеціального призначення, зв'язки між характеристиками, технологічні знання з виробництва

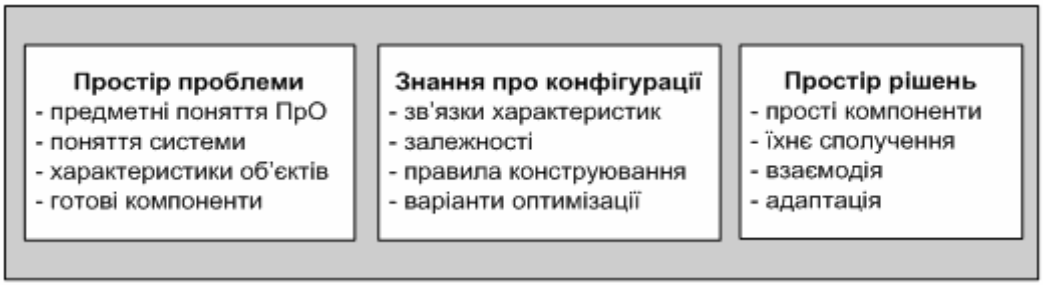

Рис. 1. Структура генерувальної моделі GDM

нових компонентів, засоби їх експертизи, тестування, вимірювання та оцінювання. В ньому зберігають завдання ПрО та їх опис у відповідній предметно-орієнтованій мові.

Простір рішень – це реалізація області-залежних абстракції з простору проблем, які подають компонентами, КПВ або каркасами архітектури домену, а також операціями їхнього з'єднання, вбудовування у відповідний член сімейства. Елементи цього простору, а саме компоненти, розв'язують задачі цієї ПрО. Каркас оснащено механізмом зміни параметрів, які вимагають фрагментацію множини дрібних методів і класів. Він забезпечує побудову різних типів ПС сімейства із застосуванням аспектів синхронізації, взаємодії або захисту даних, враховуючи умови операційних середовищ (JBuilder, Eclipse, .Net і т.п.) [6].

Елементи простору проблем перетворюють до простору рішень задач за допомогою GDM моделі та шаблонів опису членів сімейства предметно-орієнтованою мовою DSL, найбільш близькою до їхньої сутності. Абстракції простору впливають на спосіб трансформації поданих DSL-описів членів сімейства у МП простору, де ці завдання розв'язують. Вони можуть бути перетворені, якщо є залежними від специфіки DSL-опису домену або від змінюваних його характеристик та особливостей.

1.2. Предметно-орієнтовані мови опису специфіки і задач домену

Задачі простору проблем предметної області або окремих членів сімейства, як правило, визначають різними предмет но-орієнтованими мовами. В даному випадку термін "мова" використовується в загальному розумінні. Така мова може бути подана як засіб опису специфічних понять ПрО, різних аспектів функціонування задач через операції взаємодії членів сімейств або їх складових тощо. Поняття ПрО можуть бути подані функціями, методами як в ООП. Вони, як відомо, зберігаються в бібліотеках або вбудовуються в універсальну МП (наприклад, у C++, C# тощо). Коли в таку мову додають різні типи абстракцій опису різних задач ПрО, то її називають модульною предметно-орієнтованою мовою [3–5].

Задачі домену можуть бути функціонального (наприклад, бухгалтерські, кадрові тощо) та системного (наприклад, захист даних, безпека, взаємодія тощо) типів. Специфікація задач домену може виконуватися декількома предметно-орієнтованими мовами і кожній з них притаманна своя специфічна особливість для його опису.

До засобів предметно-орієнтованих мов належать такі:

– опис залежних від домену абстракцій, близьких до його простору проблем та відповідного члена сімейства;
– DSL як опис специфіки домену та характеристик його членів;
– специфікація функціональних задач домену;
– опис аспектів взаємодії, синхронізації компонентів ПС у середовищі;
– подання завдань захисту даних та безпеки виконання членів сімейства систем;
– SQL як специфікація програм доступу до БД;
– Опис інтерфейсів членів сімейства

та КПВ (PDL, IDL тощо).

Кожний член сімейства може бути поданий в одної із наведених мов.

Предметно-орієнтована мова – DSL. Її відносять до класу мов опису специфіки ПрО або домену, притаманної саме цьому домену. Опис кожного члена сімейства виконують мовою, специфічної до змісту задач домену і вміщують у генерувальну модель GDM, до складу якої входять моделі членів сімейства, також специфіковані засобами предметно-орієнтованих мов [5, 7, 8].

Мова DSL не є новим винаходом, тому що загальні абстракції програмування ПрО були раніше визначені й убудовані в МП загального предметного призначення. І хоча мову DSL створюють особисто для кожного нового домену, систематичне її виконують при специфікації особливостей понять доменів і визначенні їх загальних рис. Ця мова – більш спеціалізована, чим МП (Паскаль, Java, C#), які створені з метою і орієнтацією на конкретні області застосування. Порівняно з ними, мова DSL близька спеціалізованим мовам, таким, як HTML, XML і має такі специфічні особливості:

– абстракції DSL для визначення концепцій і абстрактних понять ПрО;

– синтаксис мови DSL призначений для опису специфіки, запобігає синтаксичної непогодженості поняттям домену аналогічно використаним МП;

– перевірку описів в DSL виконує аналізатор як механізм знаходження помилок і повідомлень про їхню наявність мовою, більш зрозумілої експертам цього домену;

– оптимізація коду з опису в DSL базується на поданих знаннях про домен, що не доступно компілятору з МП;

– інструменти підтримки мови DSL вимагають наявності середовища з трансляторами, редакторами, контролерами версій і т. п.;

– предметні абстракції в МП базуються на визначених користувачем функціях, класах, структурах даних, програмах і КПВ з бібліотек.

Модель ПрО специфікована мовою DSL є загальною моделлю GDM і відбиває таке:

– словник понять, онтологію відповідних їм об`єктів та характеристики ПрО і членів сімейства в просторі проблем;

– набір членів сімейства і їхніх специфікацій у мовах типу DSL, RSL, CSP та інші;

– завдання ПрО в просторі задач для їхньої реалізації компонентами з наступною їхньою композицією у конфігурацію для окремого члена сімейства або усього сімейства;

– знання про конфігурацію (Configuration knowledge) з характеристиками і їхніми сполученнями для членів сімейства;

– модель характеристик із загальними, змінюваними і незмінними характеристиками членів сімейства і правилами конструювання систем сімейства із компонентів та КПВ;

– архітектуру (каркас) сімейства систем та опис компонентів архітектури у підходящій МП.

Генерування членів сімейства здійснюють за специфікацією специфічних особливостей домену в DSL. Опис домену в цієї мові використовують у середовище GMD розробки членів сімейства за компонентами, КПВ, аспектами, об'єктами тощо. Їхній опис трансформують до МП загального призначення (Java, C++ та ін.), які одночасно можуть використовуватися для опису окремих елементів членів сімейства, що розв'язують відповідні завдання домену.

1.3. Трансформація предметно-орієнтованих DSL описів

Модель домену або предметної області $M_{про}$ це така загальна модель:

$$M_{про} = \{M_{про1}, M_{про2}, ..., M_{проN}\}, \qquad (1)$$

яка складається з декількох моделей окремих підобластей, що відповідають деяким членам сімейства.

Кожна з цих моделей відображає поняття і специфіку відповідного члена сімейства із сімейства систем домену. На їхньому перехресті можуть знаходитися загальні поняття, характеристики і обме-

ження, які відображені у моделі характеристик ПрО. Опис кожної моделі $М_{проі}$ виконують відповідною предметно-орієнтованою DSL_i-мовою. Цей опис за правилами перетворення трансформують безпосередньо у відповідну $МП_i$ як мову реалізації цієї моделі (рис. 2), або в іншу DSL_j-мову, яка потім трансформується у потрібну МП.

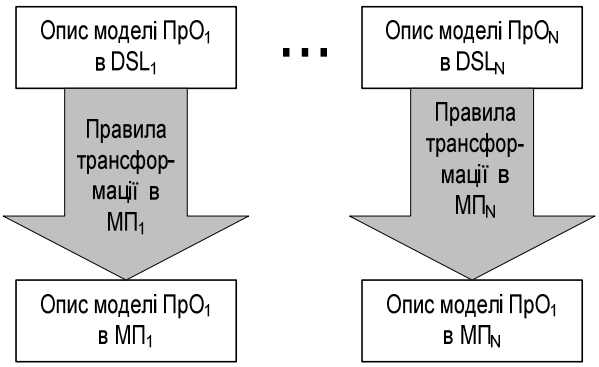

Рис. 2. Схема трансформації моделей ПрО

Після трансформації опису моделі ПрО у МП як проміжний опис, його транслюють до вихідного коду того середовища, де цей код буде функціонувати. Крім того, КПВ і знову розроблені компоненти, що реалізують окремі задачі з простору проблем домену, мають опис різними МП і тим є елементами реалізації завдань у просторі задач. Поступова перебудова опису моделі ПрО до вихідного коду залежить від платформи виконання ПС.

Подання моделей ПрО виконується за відомою модельно-керованою розробкою – MDD (Model Driven Development). Архітектуру системи за цією моделлю моделюють на двох рівнях – платформи незалежного рівня за моделлю PIM (Platform Independent Model) і платформи залежного рівня за моделлю PSM (Platform Specific Models). Концепції дворівневого моделювання архітектури MDA (Model Driven Architecture) і відображення PIM→PSM безпосередньо відповідають наведеній ідеології побудови сімейства систем за моделлю прикладних систем (Application model) у DSL–мові (рис. 3) [9, 10].

Відповідно до підходу MDD моделі окремих прикладних систем як членів сімейства можуть входити до складу сімейства ПС . Такі члени сімейства можуть

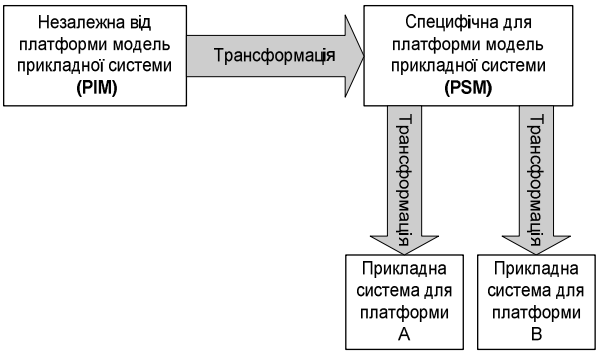

Рис. 3. Схема трансформації моделі прикладної системи до платформ A, B

мати спільні задачі й розрізнятися платформами реалізації, які відмічаються точками варіантності у моделі сімейства систем, отриманої автоматичною трансформацією PIM→PSM і без участі розробника. У такому випадку до *моделі* ПрО додають точки для прийняття альтернативних концепцій і моделювання особливостей понять генерувальної моделі GDM з такими функціями:

– зберігання або сортування інформації у деяких спеціальних компонентах;

– збирання даних, що впливають на виконання функцій компонентів;

– швидкодіюча певна конфігурація компонентів з аспектами взаємодії;

– відображення різних поглядів щодо загальних сумісних інтерфейсів, які можуть мати деякі організаційні наміри без будь-яких вимог.

1.4. Моделювання архітектури сімейства за характеристиками

Моделювання архітектури сімейства виконують за підходом, орієнтованим на характеристики (feature-oriented approach) її членів. Архітектура сімейства систем з окремих членів, наслідує еталонну архітектуру в просторі рішень (реалізацій) ГП. Цю архітектуру формують у просторі проблем таким чином, щоб різні комбінації характеристик цієї архітектури могли бути застосовані як компоненти реалі-

зації простору рішень у кінцевому програмному продукту сімейства. Архітектура сімейства може бути подана предметно-орієнтованим шаблоном у цілях генерування, який визначає архітектурний стиль ПС сімейства за її інваріантами, механізмами мінливості та еволюційного розвитку через подання комбінацій відповідних шаблонів проектування цієї архітектури [11].

При цьому доменну модель ПС сімейства розширюють шляхом залучення до неї інваріантних понять та характеристик членів системи відповідно до певних їх критеріїв та предметно-орієнтованого опису цієї моделі засобами відповідного DSL для опису проблем або задач, що створюють область перспектив домену.

При аналізі завдань домену досліджують інформаційні джерела простору проблем, будують модель характеристик (feature model) або особливостей (M_x), що відображає загальні й змінні характеристики членів систем у сімействі, їх властивості та сполучення, а також опис проблем мовою DSL (рис. 4). Цю модель використовують для подання архітектури системи і визначення компонентів для деяких задач домену.

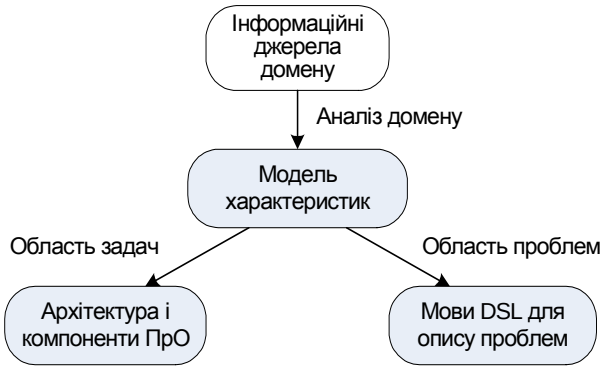

Рис. 4. Схема побудови моделі характеристик ПрО

Характеристика – це властивість членів сімейства, яка доступна користувачу системи й можуть бути обов'язковою, необов'язковою або альтернативною. До них належать інтерфейсні, проектні, продуктивні та реалізаційні характеристики. Вони визначають спільні для ПрО властивості понять, а також тих, якими можна варіювати для окремих понять та їх взаємозалежностей.

У модель характеристик входять варіабельні властивості, що визначають інформацію задану явно або неявно у конструкціях мови DSL [8, 10]. Спільні чи обов'язкові властивості цієї моделі можуть бути присутні у всіх об'єктах системи і визначати примітиви DSL та порядок обчислень. Усі характеристики можуть відображатися у моделі характеристик відповідними діаграмами орієнтованого графа домену.

Ця модель відбуває область дії сімейства систем і здатність до змінювання архітектури та її компонентів, а також знань з конфігурації системи, необхідної при автоматизованому виробництві членів сімейства.

Доменна модель в (1) за сутністю є метамодель, вона містить моделі членів сімейства, їх предметно-орієнтовані описи у DSL з механізмами підвищення рівня абстракцій і запровадження їх в реалізацію окремих членів сімейства, а також модель характеристик M_x, а саме:

$$M_{про} = \{(M_{про1}, M_{про2}, .., M_{проN}), M_x\}. \quad (2)$$

Інструментальні засоби генерувального середовища забезпечують відображення характеристик M_x членів сімейства, які відповідають поняттям з простору проблем в реалізовані завдання простору рішень.

Розробку архітектури домену виконують у просторі проблем і є їхньою проекцією на модель характеристик. Цей простір конкретизують на прикладних і аспектних характеристиках, які виконують компоненти з простору рішень. У моделі характеристик прикладними елементами можуть бути архітектурні шаблони, які слугують реалізації архітектури сімейства систем.

1.5. Методологія розробки домену

Методологія розробки домену MDD базується на моделі GDM [2, 7] та на:

– *модельно-керованої архітектурі* MDA, що використовує мову моделювання

UML для подання архітектури системи і її конкретні профілі для різних платформ;

– *модельно-інтегрованої обробки* MIC (Model-Integrated Computing) для реалізації елементів системи і їхніх зв'язків за допомогою засобів предметно-орієнтованої мови моделювання DSML (Domain-Specific Modeling Language) і перетворення цього опису в платформозалежні артефакти.

Модель MIC поєднує:

– предметно-орієнтовані мови моделювання DSML, як засіб формалізації структури, поведінки і вимог до систем усередині ПрО, визначення зв'язків між її поняттями, семантики й обмежень цих понять;

– трансформаційні процесори і генератори, які аналізують задані аспекти моделей домену і синтезують вихідний код у мову типу XML, включаючи схеми розгортки, узгодження між реалізаціями і зафіксованими вимогами до функцій і якості системи.

При проектуванні домену за каркасом або моделлю GDM можуть задаватися додатково механізми змінювання, синхронізації, безпеки засобами аспектного програмування.

Технологія розробки сімейства систем має такі базові процеси [10]:

– розробку унікальних, одиночних програм ПрО;

– інженерію повторного використання ресурсів;

– менеджмент домену.

Розробка ПрО належить до конвеєрної із базових ресурсів повторного використання, а саме: компоненти, генератори, DSL-описи, модель ПрО, КПВ, документація й інше. Розробка предметної області – це більш складний виробничий процес і містить у собі такі загальні етапи: аналіз, проектування і впровадження в ПрО одиночних програм або КПВ.

Аналіз області це аналіз систем сімейства, які треба побудувати, визначаючи для них загальні й відмінні риси, створити структурні та поведінкові специфікації членів сімейства. Аналіз ПрО починають з вибору вимог і їхньої специфікації для системи і членів сімейства. Специфікація вимог – це вхідні дані для ручного або автоматичного створення домену з готових ресурсів [9]. Найважливішим виходом аналізу є термінологія та базована на ній спільна мова фіксації знань щодо ПрО, яка залишається неформальною, хоча базується на певній термінології та змісті понять цієї ПрО.

Інженерія повторного використання – це пошук і аналіз КПВ для їхнього повторного використання під час розв'язання задач ПрО. Вони відображаються в загальній архітектурі сімейства і її окремих членів (тобто прикладних систем) сімейства цієї ПрО. Такими членами можуть бути одиночні прикладні системи, що вироблені методами прикладної інженерії, або готові КПВ.

Впровадження або реалізація систем сімейства виконують за готовими ресурсами: компонентами, одиночними програмами, генераторами, DSL–описами й ін. Об'єднання цих ресурсів у зв'язану сукупність виконуються за допомогою генераторів або конфігураторів готових ресурсів. Згенеровані продукти сімейства можуть включати і не програмні артефакти, наприклад, інструкції з користування DSL s продуктами домену.

Менеджмент домену – це керування конвеєрною розробкою з повторним використанням ресурсів, яку планують і контролюють при підборі типових для ПрО ресурсів, їхньої оцінки та перевірки відповідності вимогам до систем сімейства. У задачу менеджменту входить також перевірка можливості застосування тих чи інших готових ресурсів для реалізації специфіки ПрО і програмування компонентів з простору задач відповідно до потреб клієнтів домену.

Таким чином, *інженерія домену* складається з таких процесів:

– аналізу домену, виявлення об'єктів і зв'язків між ними та побудова $M_{про}$;

– визначення області дій об'єктів, членів сімейства ПрО;

– визначення загальних функціональних і змінюваних характеристик, побудова моделі M_x з встановленням залежностей між різними членами сімейства;

– створення базису для інженерії виробництва конкретних прикладних членів сімейства з механізмами змінювання незалежно від засобів їхньої реалізації;

– підбору і підготовки КПВ для реалізації задач ПрО;

– генерації окремих членів сімейства або домену в цілому.

Процеси в цій схемі забезпечують формування моделі ПрО і моделі M_x, як елементів простору проблем. Їх трансформують в архітектуру системи й опис її компонентів. При цьому здійснюють підбір КПВ і їхню генерацію для отримання програм, що розв'язують задачі ПрО у просторі рішень. Іншими словами, дана схема призводить до генерації моделі GMD для сімейства ПС, зміни її членів і породженню готових підсистем або окремих членів сімейства [1, 2, 4].

Змінюваність деяких членів сімейства досягається шляхом [10]:

– коректування процесів виробництва через включення у систему нових проектних рішень або КПВ;

– моделювання залежностей між компонентами, формування сутності зміни деяких елементів моделі GMD, об'єктної моделі або моделі взаємодії шляхом додавання нових вимог і деяких властивостей понять з фіксацією їх у моделі характеристик і в конфігурації системи;

– фіксації залежностей між додатковими характеристиками понять і об'єктів з уточненням моделі $M_{про}$ і конфігураційної бази сімейства систем;

– розроблення інфраструктури КПВ, включаючи засоби репозитарію для опису, збереження, пошуку, оцінювання, а також об'єднання їх у нові системи сімейства;

– забезпечення безпеки, захисту даних для окремих членів сімейства, а також взаємодії КПВ у складі сімейства систем, які функціонують у різних середовищах.

Таким чином, модель $M_{про}$ сімейства ПС, її модель характеристик і набір компонентів, реалізуючих задачі домену, є базисом автоматизованої генерації окремих членів сімейства і сімейства у цілому, що специфіковані мовою DSL або МП.

2. Розробка теоретичних і прикладних аспектів ГП

Сформульовані вище основні положення ГП показують, що в його межах склалися достатньо важливі концепції, моделі і методи, орієнтовані на інженерію виробництва сімейства систем. Але ці напрацьовані положення потребують розвитку формального апарата щодо СПС, засобів тестування згенерованих членів сімейств, гарантування й оцінювання їхній якості та життєздатності функціонування у сучасних середовищах генераційного типу. На розв'язання цих проблем зорієнтована наукова тема проекту "Розробка теоретичного апарату генеруючого програмування і інтегрованого середовища його підтримки" Інституту програмних систем НАН України (2007–2011 рр.).

Головні напрями основ формального подання СПС у ГП направлені на розробку:

– змісту, меж ПрО, мовних засобів опису і розв'язання задач ПрО як членів сімейств за їхній генерацією до умов сучасних середовищ і платформ;

– метазасобів подання DSL-мовою зі специфічними поняттями домену та моделі характеристик, з їхніми властивостями, взаємозв'язками та залежностями від задач членів сімейства;

– апарата інфраструктури підтримки опису інтерфейсів КПВ, їхнього сховища та методів пошуку необхідних КПВ для застосування у нових системах;

– технологічних базових положень сімейства процесів життєвого циклу, кожний з яких забезпечує спеціалізовану трансформацію деякого члена сімейства систем та різних її складових;

– новітніх підходів до верифікації і тестування готових КПВ, їхніх сукупностей та складових сімейства домену;

– нової концепції експертного оцінювання елементів процесів розроблення, реалізуючих різні задачі членів сімейства домену, а також вдосконалених методів оцінювання якості згенерованих продуктів для систем сімейства;

– інструментального середовища з репозитарієм артефактів (КПВ, компоненти, аспекти тощо) для вибору готових, розроблення нових компонентів і породження з різних шаблонів (templates) програм і інших складових систем сімейства;

– нових характеристик, що забезпечують життєдіяльність згенерованих членів сімейства систем, що розташовані та функціонують у різних середовищах.

Поставлені завдання у даному проекті ґрунтуються на наукових і прикладних результатах, отриманих у попередньому фундаментальному проекті "Розробка теоретичних основ та методологічних засад компонентного програмування" (2001–2006 рр.), що був призначений для побудови теорії компонентного програмування [1, 12], а саме, формалізації опису інтерфейсів різних КПВ для їхнього зберігання у репозитаріях і визначення операцій інтеграції різномовних компонентів для нових компонентних систем. За цим проектом сформульована теорія КПВ і компонентів як основи ГП, оскільки КПВ – це головні „деталі" виробництва компонентних систем і СПС, і з нашої точки зору, вона є первинним теоретичним фундаментом, який надалі буде розвиватися у ГП.

Дамо характеристику базового фундаменту компонентного програмування і новим підходам з розв'язання вищеперелічених задач ГП.

2.1. Об'єктно-компонентний метод як базовий фундамент ГП

Головне досягнення попереднього фундаментального проекту – новий метод комплексного аналізу і компонентної побудови програмних систем [12]. У методі узагальнено поняття об'єктів, як елементів реального світу з відповідними властивостями і характеристиками, послідовне їхнє визначення на рівнях об'єктного аналізу (узагальненого, структурного, характеристичного, поведінкового) засобами математичних формалізмів опису та уточнення вибраних при проектуванні домену різних об'єктів, з завданням їм достатніх властивостей та характеристик, що відрізняють один об'єкт від іншого. Доведення приналежності властивостей об'єктів ПС виконано за теорією Фреге [13].

Розроблений об'єктно-компонентний метод програмування відображає принципи об'єктного аналізу ПрО з побудови об'єктної моделі (ОМ), а також формальний перехід від об'єктів до компонентів з інтерфейсами. Тобто цей метод встановлює з формальної точки зору теоретичний і практичний зв'язок між об'єктним аналізом і компонентним розробленням ПС, вміщує операції об'єктного аналізу, зовнішній та внутрішній компонентної алгебри [12].

Об'єктний аналіз виконується на множині базових функцій зміни денотат і концептів об'єктів під час декомпозиції ПрО на об'єкти $O=(O_1, O_2, . On)$ за такими операціями:

– декомпозиційне змінювання денотата однорідних і неоднорідних об'єктів – $decds\ (O_i)$: $O_i \rightarrow (O_{i1}, \dots O_{ik})$ для $\forall j\ Con_{ij} = Con_i, \cup\ Den_{ij} = Den_i$ чи $\forall j\ Con_{ij} = \varnothing, \cup Den_{ij} = Den_i$;

– композиційне змінювання денотата однорідних і неоднорідних об'єктів $comds\ (O_{i1}, \dots O_{ik})$: $(O_{i1}, \dots O_{ik}) \rightarrow O_i$, для $\forall j\ Con_i = Con_{ij}$, $Den_i = \cup Den_{ij}$ і $Con_i = \varnothing$, $Den_i = \cup Den_{ij}$;

– розширення концепту об'єкта такими умовами $(P_t \in P)$, $(P_t \notin Con_i)$, де $P_t(O_i)$ істинно і $conexp\ (O_i, P_t) : O_i \rightarrow O_i`$, $Con_i \cup \{P_t\} = Con_i$;

– звуження концепту $connar(O_i, P_t)$: $O_i \rightarrow O_i`$ при $O_i` = O_i`(Name_i, Den_i, Con_i`)$, $Con_i` = Con_i \setminus P_t$.

Ці операції забезпечують адекватне подання ОМ ПрО.

Теоретичні аспекти компонентного програмування. Складовими нової теорії компонентного програмування є моделі компонента, інтерфейсу, середовища та алгебра (зовнішня та внутрішня). Ці моделі – інструмент переходу від ОМ до компонентної моделі ПрО за операціями зовнішньої і внутрішньої алгебри.

Зовнішня алгебра: $\Psi = \{CSet, CESet, \Omega 1\}$, де $\Omega 1 = \{CE_1, CE_2, CE_3, CE_4\}$ – операції зовнішній алгебри:

– інсталяція (розгортання) $CE_2 = Comp \oplus CE_1$;

– об'єднання компонентних середовищ $CE_3 = CE_1 \cup CE_2$;

– видалення компонента з компонентного середовища $CE_2 = CE_1 \setminus Comp$;

– заміщення компоненти $Comp_1$ на $Comp_2$ – $CE_2 = Comp_2 \oplus (CE_1 \setminus Comp_1)$.

Внутрішня алгебра: φ = { CSet, CESet, Ω2}, де $\Omega 2 = \{O_{refac}, O_{Reing}, O_{Rever}\}$ – *операції* внутрішній алгебри, призначені для еволюційного змінювання об'єктів до компонентів, а також безпосередньо самих компонентів:

– операції рефакторінгу O_{refac} = ={*AddOImp, AddNImp, ReplImp, AddInt*};

– операції реінженерії O_{reing}= ={*rewrite, restruc, adop, supp, conver* };

– операції реверсної інженерії O_{Rever}= ={*visual, metric, restruc, design, rewrite*}.

Зовнішня і внутрішня алгебра разом складають алгебру компонентного програмування: Ξ = {Ψ ∩ φ} = { CSet, CESet, Ω1} = { CSet, CESet, Ω2}= {CSet, CESet, Ω1, Ω2}.

Компонентна алгебра – новий базис формалізованої інженерії виробництва ПС з компонентів та КПВ.

Розвиток формальних і мовних аспектів в ГП. Формальний апарат ГП буде створюватися з використанням теоретичного апарата компонентного програмування, що включає формальні моделі опису і подання компонентів, frameworks ПрО, а також операцій відповідної компонентної алгебри. До цього апарата будуть додаватися формальні засоби (синтаксис і семантика) предметно-орієнтованої мови DSL, глосарій і онтологія доменів, нові можливості подання інфраструктури репозитарієв КПВ [14 –18], моделі якості [19–21] й життєздатності (живучості) виготовлених систем [23–25]. Мова опису понять і задач ПрО буде залежать від засобів подання вимог (наприклад, UML [14]), моделей ПрО і методів їх трансформації до виконавчого коду, патернів (заготовок) деяких компонентів з операціями їхній генерації і використання у сімействі [3, 11], та інших новітніх концепцій генерування і композиції різних об'єктів і компонентів у сімейство з урахуванням умов середовища ГП [1–3].

Передбачається подання моделі характеристик діаграмами засобами UML, для якого є досвід побудови алгебри об'єктного аналізу з правилами керування тими діаграмами UML [12, 14], як алгебра графічного подання характеристик деяких систем сімейства доменів та складова частина компонентної алгебри.

Додавання до алгебри інших операцій залежить від типів генераторів, трансформаторів, а також плагинів, що увійдуть до складу середовища як інструменти перебудови різних описів членів сімейства, генерації наявних шаблонів, цільових frameworks тощо. До складу операцій увійдуть операції і процедури, пов'язані з методами тестування згенерованого вихідного коду для членів сімейства і сімейства систем, а також новий теоретичний апарат з методами експертного і якісного оцінювання показників якості, живучості продуктів сімейства у середовищі ГП та аспекти теорії керування програмними проектами.

У межах ГП повторне використання КПВ отримує розвиток нових типів об'єктів та засобів їх опису і подання КПВ у репозитарій [17, 18], моделювання і виготовлення сімейств систем за цими КПВ з систематичним пошуком спільних рис між КПВ, інструментами, системами сімейства і готовими одиничними програмами з даної ПрО чи з іншої. На основі деяких спільних рис можуть бути створені окремі члени сімейства як повторно використані (наприклад, загальні архітектури, компоненти, моделі). Але нова форма абстракцій фреймворків і компонентів за предметно-орієнтованими мовами, відсутність чітких границь і різноманітних їхніх описів потребує, на наш погляд, додавання у метод генерації нових правил і операцій, які будуть також породжені залученими деякими плагинами й інструментами до середовища ГП.

2.2. Інфраструктура КПВ

Як вище описано, одним з продуктів аналізу домену є пошук повторно використовуваних знань конкретні КПВ, що необхідні домену. У попередніх проектах

з компонентного програмування сформульовано концепцію подання кожного КПВ інформаційною моделлю у вигляді пошукового образу, в якому анотовані функції, інтерфейси, мова опису, середовище розробки тощо [17]. Стандартом їхнього подання є UML діаграми, що задані в термінах понять користувача. До інфраструктури належать насамперед накопичені у репозитарію КПВ, методичні, технічні і людські ресурси, що здi1йснюють зберігання і вибірку КПВ для подальшого їхнього застосування у деякої прикладної системі домену, що розробляється.

Сутність КПВ полягає у їхній незалежності від конкретної платформи, наявності інтерфейсу і параметрів настроювання на нове середовище, а також можливості взаємодії з іншими компонентами системи без внесення в них змін. Основні процеси розроблення ПС за КПВ такі:

– аналіз об'єктів і зв’язків у ПрО, яку реалізують з метою виявлення КПВ з загальними властивостями, притаманними групам об'єктів цієї області;

– адаптація наявних КПВ у базі репозитарію, розроблення нових процесів і функцій, не представлених у цій базі, і доведення їх до рівня повторного використання;

– розроблення інтерфейсів компонентів і розміщення їх у репозитарії інтерфейсів системи;

– інтеграція КПВ з їхніми інтерфейсами та іншими елементами ПС і формування конфігураційної бази системи.

У ГП в якості готових компонентів могуть бути різні програмні артефакти як універсальні методи і ресурси, так і інструменти, які необхідні для виконання процесів розроблення ПС.

Процес генерації систем за участю усіх видів готових ресурсів пропонується такий [16–18]:

1) генерація вимог під керуванням користувача;

2) пошук готових ресурсів проектування систем за вимогами;

3) прийняття рішення щодо визначення достатності знайдених готових ресурсів, засобів і інструментів у репозитарії та їх генерації у відповідному середовищі.

Головне завдання щодо готових ресурсів у проекті, це розроблення онтологічних моделей їхній специфікації, яка слугує поданню деякого ресурсу у формалізованому і загальному за формою шаблоні, здатному для генерації спеціальними інструментами за єдиною генераційною схемою.

2.3. Верифікація, тестування членів сімейства та їх оцінювання

Одним з напрямів дослідження проблем ГП є *інженерія оцінювання* об’єктів СПС і пов’язаних з нею питання перевірки правильності (верифікація, тестування) членів сімейства. Проблема оцінювання об’єктів ПС детально досліджена у попередніх проектах, а також наведена у звіті за 2008 рр. Створені нові концепції і методи експертного і кількісного оцінювання різних об’єктів і запропоновані нові проектні рішення з оцінювання показників якості для сімейства процесів у ГП [19, 20].

Іншим напрямом вирішення проблеми оцінювання – удосконалення існуючих і розроблення нових методів тестування різних об’єктів (звичайних компонентів, веб-компонентів і сервісів тощо) за онтологічною моделлю, затвердження їх відповідності загальноприйнятим стандартам і моделям якості ПС, визначеним для оцінювання генерованих членів сімейства ПрО. Наступним напрямом є побудова моделей архітектури сімейства ПС за новітніми підходами, сформульованими до інженерії якості ПС [7, 18] з можливістю зміни різних артефактів після генерації і отримання оцінок, що не задовольняють розробників сімейства систем.

Всі ці задачі і підходи орієнтовані на нове *сімейство процесів* ГП, а саме:

– оброблення описів КПВ та їхніх інтерфейсів, зберігання їх у різних сховищах (репозитаріях) для подальшого вибору і пристосовування у конфігурацію системи з сімейства;

– генерації членів сімейства за різними процесами середовища з метою отримання кінцевого продукту домену;

– підтримки набору процесів з формалізованого сімейства процесів з інженерії доменів, прикладних систем і доменів шляхом поступового конвеєрного вироблення на них відповідних продуктів для масового вживання користувачами;

– реалізації нового методу багатокритеріального експертного оцінювання об'єктів ЖЦ систем та механізмів забезпечення варіабельності архітектури СПС із застосуванням експертних методів оцінювання [18, 21, 22];

– підтримки графічного подання онтологій з тестування і використання спеціальних видів плагинів оцінювання продукту після тестування;

– моделювання нефункціональних характеристик СПС за комбінуванням різнотипних моделей ПрО, формальної верифікації моделей характеристик ПрО та ієрархічної моделі якості компонентів.

Одним з процесів сімейства буде процес експертного оцінювання, що подається системою експертиз з використанням моделі переваг класу аргументованого дерева цінності [20], інформаційне взаємопов'язаних онтологічним середовищем із визначеними у ньому відношеннями подібності, а також індивідуальними експертними оцінками на основі імовірності подій у процесі розроблення [22].

Тобто, у проекті системи ГП ставиться задача забезпечення теоретичної, нормативно-методологічної та програмної підтримки розв'язку проблем верифікації, тестування й моделювання якості членів сімейства і самих сімейств процесів ГП.

2.4. Концептуальні основи побудови системи генерації і середовища ГП

Головні концепції такі:

– визначення змісту, моделей і меж ПрО з формування спеціалізованого середовища ГП, а також набору мовних і програмних засобів опису і розв'язку завдань доменів [24];

– вибір підходящих frameworks відкритого типу для підтримки процесів розроблення, генерації і оцінювання якості та живучості програмних продуктів для різних типів доменів;

– реалізація готових шаблонів і компонентів з різних доменів та розміщення їх у середовищі ГП;

– визначення архітектурних і проектних рішень щодо середовища ГП для підтримки інженерії доменів за типом конвеєрного виробництва.

Застосування DSL як засобу подання специфіки ПрО потребує наявності у середовищі ГП:

– інструментів трансформації опису моделей доменів шляхом перетворення первісної моделі у вихідну (цільову) згідно з сформульованими операціями алгебри ГП з перетворення як для однієї мови, так і для сукупності конструкцій мов її подання.

– інструментальних засобів (компілятор, редактор, аналізатор, верифікатор, генератор тощо), які при змінах ПрО теж можуть змінюватися або замінюватися;

– метаінструментів для оброблення моделей ПрО, які зможуть адаптуватися до нових мов DSL або тих, які відображають частини вже використаних мов DSL;

– засобів побудови нових версій згенерованих програмних продуктів або окремих їх частин у системах сімейства, що описані МП. Їми можуть бути також мета інструменти з механізмами настроювання і створення нових конфігурацій систем сімейства;

– універсальні засоби трансформації, придатні для будь-якої ПрО і незалежних від ПІ;

– технологічні процеси життєвого циклу, здатні послідовно виконувати обробку різних описів (вимог, специфіки, специфікації тощо).

Тож задача інформаційного забезпечення системи генерації – запропонувати на кожній стадії розробки засобів підтримки трансформації описів і моделей.

Усі перераховані процеси створюють базове ядро системи ГП. В якості інструментального середовища цієї системи обрано Eclipse [6], яка є відкритою системою і надає базові frameworks, типові інструменти, а також засоби й інструменти, за якими додаються нові можливості з забезпечення генерації різних СПС.

2.5. Проблема забезпечення живучості систем сімейства

Одним з важливих понять у сімействі систем є життєздатність (viability – живучість), тобто властивість кожної з систем виконувати свої функції в процесі супроводу в відповідному середовищі з урахуванням різних зовнішніх і внутрішніх випадків з нестабільного виконання. Важливою умовою життєздатності деякої системи є можливість при вказаних випадках змінювати й адаптувати свою модель шляхом корегування проектних характеристик архітектурного каркасу і структурних інваріантів, поданих засобами теорії живучості LST (Living System Theory) [23], моделюванням архітектури MDA і механізмами інтеграційної обробки MIC. Теорія LST дає інтегрований концептуальний підхід LSPA (Living Systems Process Analysis) проведення процесу аналіза живучості майбутній системи за розробкою і супроводом. Основні положення цієї теорії досліджені і вивчені в межах попереднього згаданого проекту, які отримують розвиток для сімейства систем ГП.

За теорією живучості створюють модель живучості VSM (Viable System Model) [24], яка містить у собі такі важливі механізми, як зменшення (atttnuation) або збільшення (amplification) головних її операцій, у випадку, коли виникають деякі нерегулярні ситуації при функціонуванні системи і потребують перенастроювання вхідних параметрів і характеристик моделі життєздатності системи на подальше продовження її роботи [25].

До складу цієї моделі включають деякі показники стандартної моделі якості, а саме, адаптивність, змінність, реактивність тощо. Крім того, ця модель поповниться деякими властивостями з моделі характеристик системи генерувального програмування, які будуть зорієнтовані на операції внесення змін і адаптацію чинників і зв'язків між різними характеристиками цієї моделі. Всі ці задумки будуть спиратися на засоби UML моделювання окремих систем сімейства, нові моделі обчислювань [15], шаблони (pattern-oriented architecture) архітектури, механізми керування ними тощо. Ця модель буде відображена у конфігураційній моделі СПС, що керує розгортанням окремих її членів.

Тобто, властивість живучості тлумачиться нами як комплексна характеристика ПС, яка вміщує у собі різні характеристики, властивості й показники, що накопичені в інших сучасних моделях, отримали досвід і будуть адоптовані до потреб розроблення, розгортання та функціонування систем сімейства.

Таким чином, у рамках даного проекту вирішується проблема забезпечення живучості окремих ПС із сімейства з використанням шаблонів архітектури, набору сучасних моделей та відповідних процесів із сімейства ГП. Залучення шаблонів проектування до процесу розроблення складних систем за моделями живучості й якості сприяє реалізації їх більш якісними і життєздатними у середовищі генерувального програмування.

Висновок

У роботі в межах проекту проведений аналіз основних положень сучасного ГП, сформульовані нові шляхи його розвитку щодо систематизації і удосконалення базових засобів подання ПС, сімейств систем за КПВ, які накопичують у системному репозитарію ГП. Сформульовані підходи до трансформації і генерації різних описів членів сімейства. Визначені нові підходи з їх тестування, проведення експертиз, оцінювання якості й забезпечення живучості генерованих систем у середовищі ГП. Розроблено загальна структура процесів сімейства та системи ГП за вище наведеними напрямами.

1. *Лаврищева Е.М.* Методы программирования. Теория, инженерия, практика.– Киев: Наук. думка, 2006. – 450 с.
2. *Kang K.C., Cohen S.G., Hess J.A.., Novak W.E. and Peterson A.S.* Feature-oriented domain analysis (FODA) feasibility study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, 1990.

3. *Чернецки К., Айзенекер У.* Порождающее программирование. Методы, инструменты, применение.– М. – СПб. – Харьков. – Минск. – Изд. дом Питер, 2005. – 730 с.
4. *Crarnetcki K.* Owerview of Generative Software Development // Canada, www.crarnetcki.fcm.org
5. *Mernik M., Heering J., Sloane A.M.*: When and how to develop domain-specific languages. Technical Report SEN-E0309, CWI, Amsterdam, 2003, Available from http://www.cwi.nl/ftp/CWIreports/SEN/SEN-E0309.pdf
6. *Гамма Э., Бек К.* Расширения Eclipse: принципы, шаблоны и подключаемые модули.: Пер. с. англ. – М.: КУДИЦ-ОБРАЗ, 2005.– 384 с.
7. *Consel C.* From a program family to a domain-specific language // Symposium on Principles of Programming Languages, Charleston, SC, USA, ACM Press, 1993.– P. 19–29.
8. *Jeff Grey et al*. OOPSLA'02 Workshop on Domain-Specific Visual Languages, 2002. http://www.cis.uab.edu/info/OOPSLA-DSVL2
9. *Model* Driven Development and Software Product Lines. BigLever Software Inc. – 2007. http://www.biglever.com/technotes/mdd_spl.html?source=mdd
10. *Mezini, M., Ostermann, K.* Variability management with feature-oriented programming and aspects. In: Foundations of Software Engineering (FSE-12), ACM SIGSOFT 2004.– P. 123–130.
11. *Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж*. Приемы объектно-ориентированного проектирования. Паттерны проектирования. – СПб: Питер, 2001. – 368 с.
12. *Грищенко В.Н.* Метод об'єктно-компонентного проектування програмних систем // Проблеми програмування.– 2007.– № 2.– С. 113–125.
13. *Фреге Г.* Логика и логическая семантика.– М.: Аспект–пресс, 2000. – 512 с.
14. *Буч Г.,Рамбо Д., Джекобсон А.* Язык UML. Руководство пользователя: Пер. с англ. – М.: ДМК, 2000. – 432 с.
15. *Меллор С.* Программная инженерия. Программы должны работать // Открытые системы. – 2008. – № 8. – С. 42–51.
16. *Jacobson I., Griss M., Johnson P.* Software Reuse: Architecture, Process and organization for Business Success – Addison Wesley, Reading , MA, May 1997.– 501 p.
17. *Бабенко Л.П.* Онтологический подход к спецификации свойств программных систем и их компонент // Кибернетика и системный анализ. – 2009. – № 1. – С. 30–37.
18. *Бабенко Л.П.* Спецификация прогнозируемой вариантности как инструмент управления изменениями программных продуктов UML // Кибернетика и системный анализ. – 2007. – № 3. – С. 156–163.
19. *Лавріщева К.М., Коваль Г.І., Коротун Т.М.* Підходи інженерії якості сімейств програмних систем // Проблеми програмування. – 2008. – № 2– 3. – С. 219 – 228.
20. *Лавріщева К.М.* Перспективні дисципліни програмної інженерії.–К.: Вісник НАН України, 2008. – С. 12–17.
21. *Слабоспицька О.О.* Моделі і методи експертного оцінювання у життєвому циклі програмних систем. Автореф. дис.… канд. физ.-мат. наук. – Ін-т кібернетики ім. В.М. Глушкова НАН України, Київ; 2008. – 21 с.
22. *Слабоспицька О.О.* Задачі, методи та засоби експертного оцінювання якості в інженерії програмних систем // Проблеми програмування. – 2007. – № 3. – С. 32–40.
23. *Miller J.G.* Living Systems. – Niwot, Colorado: University Press of Colorado, 1995. – 1102 p.
24. *Herring C.* Viable software. The intelligent control paradigm for adaptable and adaptive architecture. – 2002.– 343p. http://charles-herring.com/public/ViableSoftware.pdf
25. *Ігнатенко П.П., Бистров В.М.* Особливості забезпечения життєздатності програмних систем в умовах генеруючого програмування // Проблеми програмування. – 2008. – № 2 – 3.– С. 270–279.

Отримано 03.12.2008

Про автора:

Лавріщева Катерина Михайлівна,
доктор фізико-математичних наук,
професор.

Місце роботи автора:

Завідуюча відділом ІПС НАН України,
Тел.: 526 3470
e-mail – lem@isofts.kiev.ua