УДК 004.9:005.94 DOI: 10.20837/0135-1710.2023.179.058

А.Є. КОЖАНОВ

ДОСЛІДЖЕННЯ МОДЕЛЕЙ ФОРМАЛЬНОГО ОПИСУ ЗНАНЬ ПРО ФУНКЦІОНАЛЬНІ ВИМОГИ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ З ВИКОРИСТАННЯМ EER-ДІАГРАМИ

Розглянуто моделі формального опису знань про функціональні вимогі до інформаційної системи. Досліджено формальний апарат, що використовує фреймову модель для відображення опису функціональних вимог на рівні знань з використанням діаграми класів та ER-діаграми, та запропоновано формальний апарат відображення спільних елементів діаграми класів, ER-діаграми та EER-діаграми. Проаналізовані отримані результати.

1. Вступ

Однією з найважливіших проблем, що виникають при спробах описати вимоги до інформаційної системи (ІС) формальними способами, є практично постійна неповнота вимог. Ця неповнота викликана, насамперед, прагненням Споживача отримати тільки ті ІТ-послуги, експлуатація яких забезпечить якнайшвидшу появу прибутки від процесу, що автоматизується. При цьому Споживач не замислюється про необхідність додаткових ІТ-послуг для забезпечення ефективної злагодженої роботи основних ІТ-послуг. Крім того, слід пам'ятати про можливість змін процесу, що автоматизується, із часом (наприклад, внаслідок реінжинірингу або поступового поліпшення).

Вирішення цієї проблеми стає можливим у тому випадку, якщо припустити, що Постачальник і Споживач у кожному конкретному проєкті створення ІС мають справу не з простою множиною вимог до ІС, повністю або частково впорядкованих певним чином, а з загальнішою сукупністю вимог. При цьому до такої сукупності мають входити не тільки вимоги до ІС, висунуті Споживачем і прийняті до виконання Постачальником, але й так звані «забуті» (невиявлені або несвоєчасно виявлені) вимоги, які Постачальник може висунути до ІС, виходячи зі свого бачення предметної області (ПрО); усі ці вимоги можуть раніше або пізніше стати елементами множини вимог до ІС у ході чергової ітерації процесів, що безпосередньо працюють із вимогами. Тому перед побудовою системи дуже важливо зафіксувати вимоги. Позначення вимог за допомогою точної й однозначної моделі дозволяє гарантувати, що всі зацікавлені особи розуміють вимоги й згодні з ними.

Множина вимог правовласників є хронологічно першим системним описом створюваної ІС і являє собою систему не як єдиний цілісний продукт, а як набір не пов'язаних один з одним описів окремих елементів системи. Дана множина містить у собі різні вимоги, на основі яких формується множина сценаріїв, що описують окрему функціональну можливість створюваної системи. Вимоги правовласників і сценарії, що ідентифікують функціональні можливості системи, виражаються в термінах моделей (текстових або формальних), які орієнтовані на мету й поведінку системи і описують її в контексті середовища й умов функціонування.

На основі множини вимог правовласників і сценаріїв, яка ідентифікують функціональні можливості системи, у ході виконання процесу аналізу вимог формується системний опис, що являє собою множину системних вимог. Серед цих вимог можна виділити [1]:

а) функціональні вимоги;

б) експлуатаційні вимоги;

в) вимоги, пов'язані з такими властивостями системи, як здоров'я, безпека, захищеність, безвідмовність, готовність;

г) вимоги, пов'язані із властивостями систем забезпечення;

д) технічні показники й показники якості під час використання.

Крім того, множина системних вимог доповнюється множиною обмежень, серед яких слід особливо виділити множину функціональних меж системи в термінах її поведінки й властивостей, які повинні бути забезпечені. Рекомендації зі способу вираження системних вимог і обмежень в [1] не наводяться.

Аналіз прийнятих у сучасних стандартах системних описів створюваної ІС на ранніх стадіях її життєвого циклу (ЖЦ) дозволяє зробити такі висновки:

а) рекомендовані [1] системні описи, сформовані на ранніх стадіях ЖЦ ІС, орієнтовані на представлення створюваної системи як множини окремих вимог, функцій і архітектурних елементів, а не єдиного цілісного проєкту;

б) у ході виконання процесів формування вимог правовласників, аналізу вимог і проєктування архітектури задачі системної інтеграції не вирішуються й не ставляться;

в) використання при формуванні системних описів досвіду, накопиченого в ході виконання попередніх проєктів створення аналогічних систем, не розглядається;

г) рішення про доцільність початку ІТ-проєкту створення ІС має ухвалюватися на основі множини вимог правовласників і не базується ані на досвіді створення ІС аналогічного призначення, ані на представленні ІС як єдиного цілісного продукту, що серйозно спотворює оцінки обсягу робіт зі створення ІС.

Дані висновки дозволяють стверджувати, що в ході створення ІС існують серйозні проблеми на ранніх стадіях її ЖЦ, що значно підвищує витрати на створення ІС й збільшує кількість інцидентів, які можуть виникнути в ході розробки частини забезпечення даної ІС. У той самий час практика виконання проєктів (у тому числі – ІТ-проєктів) свідчить [2], що зміни ІС за вимогами зацікавлених сторін найефективніші й вимагають менших витрат переважно на ранніх стадіях її ЖЦ. Тому вирішення зазначених проблем є одним з найперспективніших напрямків науково-прикладних досліджень в ІТ-сфері.

Результати аналізу сучасного представлення ІС показують, що вимоги до ІС є основною вихідною інформацією для проєктування архітектури ІС, на основі якої виконуються роботи з розробки систем забезпечення. Тому необхідно проаналізувати сучасні представлення вимог до ІС із метою виділення їх недоліків, що призводять до збільшення витрат на створення або модернізацію ІС.

Результати сучасних процесів візуального моделювання вимог до ІС, призначених для автоматизації керування підприємствами й організаціями, є одними з найважливіших факторів, що визначають успішність виконання всіх наступних робіт із проєктування й розробки в рамках або створення нової ІС, або модифікації наявної системи. В основі цих процесів лежить формування опису ІС, що створюється або модифікується, як сукупності вимог до системи з подальшим перетворенням цих описів спочатку в опис архітектури ІС, а згодом – в описи специфікацій на розробку елементів комплексів забезпечення.

2. Аналіз систем класифікації вимог і постановка проблеми дослідження

У ході аналізу розглянуті класифікації вимог до ІС, що використовуються в таких методологіях і роботах дослідників:

а) методологія SSADM [3];

б) дослідження, виконані Д. Леффінгуеллом і Д. Уідригом [4], а також К. Вігерсом [5];

в) методологія, що лежить у основі пакета керування вимогами Rational DOORS [6];

г) модель класифікації вимог FURPS і її сучасний варіант FURPS+ (компанія «Hewlett-Packard») [7], [8];

д) звід знань з аналізу бізнес-процесів (БП) сучасних підприємств (Business Analysis Body of Knowledge, BABOK) [9], у тому числі – з метою визначення доцільності автоматизації даних БП.

Дані варіанти описують усі основні точки зору на вимоги до ІС як до технічного ІТ-продукту, необхідного для досягнення економічних або техніко-економічних цілей.

За результатами аналізу можна зробити такі висновки [10]:

а) єдиної системи класифікації вимог до ІС не існує;

б) у теперішній час більшість варіантів класифікації вимог розглядають вимоги до системи як проміжну ланку між бізнес-вимогами й вимогами до програмного забезпечення (ПЗ);

в) нефункціональні вимоги розглядаються як спосіб уточнення інших вимог у ході їх аналізу;

г) у процесі проєктування архітектури системи увага зосереджується, головним чином, на проєктуванні архітектури ПЗ.

Хоча загальноприйнятні визначення поняття «вимога до системи» вже запропоновані, єдиної системи класифікації вимог досі не існує. Тому способи формального опису вимог до ІС багато в чому визначаються тими методами збору вимог, які застосовуються під час виконання процесу визначення вимог правовласників.

У ІТ-спільноті сформована думка, відповідно до якої жодне представлення вимог тільки одним способом не дає їх повної картини. Цю думку обумовлюють використанням ієрархії DIKW (Data, Information, Knowledge and Wisdom) для опису процесів, що безпосередньо працюють із вимогами, а також процесів проєктування архітектури системи. Порівняння представлень вимог, отриманих різними фахівцями під час різноманітних досліджень, допомагає виявити невідповідності, неясності, допущення й недогляди, які важко виявити, коли вимоги представлені в одному форматі. Слід зазначити, що класична модель DIKW припускає чітку спрямованість перетворень даних в інформацію, інформації – у знання, а знань – у мудрість. Однак у процесах, що безпосередньо працюють із вимогами, часто основою для моделювання вимог до ІС є представлення цих вимог на рівні інформації, виражене у вигляді їх текстових описів або візуальних моделей (ВМ). При цьому процес проєктування архітектури системи припускає можливе уточнення ВМ вимог описом цих вимог на рівні даних у вигляді тієї або іншої атрибутивної моделі [11].

Водночас використання атрибутивних моделей, або аналогічних їм, для представлення вимог до ІС на рівні даних ускладнене, насамперед, через прагнення об'єднати в рамках однієї атрибутивної моделі вимог до ІС набори атрибутів, що використовуються у принципово різних цілях. Тому виникає необхідність розгляду кожної вимоги до ІС як первісного різноманіття представлень цієї вимоги у вигляді даних, інформації й знань. Існування такого різноманіття обумовлене такими чинниками [12]:

а) представлення вимог до ІС на рівні інформації призначене для опису різними способами елементів об'єктів або процесів, що автоматизуються, елементів розроблюваної ІС

або ІС у цілому з метою забезпечення можливості виконання проєкту створення ІС, до якої висунуто вимогу шляхом виконання послідовності перетворень «неформалізований опис вимоги звичайною мовою – частково формалізований опис вимоги – формальний опис вимоги у вигляді набору цільових показників, значення яких характеризують ступінь задоволення вимоги»;

б) представлення вимог на рівні даних призначене для формування описів вимоги до ІС і її окремих представлень, взаємно узгоджених одна з одною, з метою забезпечення можливості здійснення стандартних операцій уведення, відображення, редагування й виділення вимоги до ІС і її окремих представлень у рамках інформаційної технології (ІТ) формування й аналізу вимог до ІС, а також керування окремими вимогами в ході виконання проєкту створення ІС;

в) представлення вимог до ІС на рівні знань призначене для виявлення знань про об'єкти або процеси, що автоматизуються, розроблюваних елементах ІС або ІС у цілому з метою забезпечення можливості повторного використання цих знань у проєктах створення інших ІС.

Основні положення концепції представлення вимог до ІС визначають загальні формалізовані описи вимог. Вони мають містити в собі:

– формалізовані описи вимог до ІС (як відомих, так і невідомих сторонам);

– формалізовані описи методів формування вимог до ІС;

– формалізовані описи представлень вимог до ІС на рівнях даних, інформації й знань.

Крім того, загальні формалізовані описи вимог до ІС додатково мають містити в собі [10]:

– формалізовані описи методів, прийомів і способів перетворення представлень різних вимог, виконаних на тому самому рівні, самих у себе й одна в одну, що дозволить установити й відображати взаємозв'язки між окремими вимогами групи, обумовлені особливостями процесу, що автоматизується;

– формалізовані описи методів, прийомів і способів перетворення представлень вимог до ІС, виконаних на одному рівні, у представлення вимог, виконаних на іншому рівні.

Розглянуті умови створення формалізованих описів вимог до ІС змушують відмовитися від класичних теоретико-множинних описів вимог. Математичний апарат, на основі якого слід розробляти формалізовані описи груп вимог до ІС, має дозволяти:

– розглядати вимоги до ІС як єдину цілісну систему – прообраз розроблюваної ІС – і одночасно як множину пов'язаних один з одним елементів, що перетворюються один в інший, відповідно до методології й технологій формування й аналізу вимог;

– враховувати вплив на представлення вимог до ІС конкретних методів, прийомів і способів формування й перетворення представлень цих вимог.

При формулюванні вимог до ІС усе різноманіття синтаксичних конструкцій і можливих синонімів повинне зводитися до єдиного зрозумілого для всіх сторін представлення цього поняття у вигляді даних, інформації або знань.

На практиці найчастіше зустрічається, що розробники переходять від моделей, що описують БП підприємства, до моделей, що описують ІС керування цим підприємством, а від них, у свою чергу, – до моделей, що описують окремі види забезпечень ІС. Можливі й зворотні переходи – наприклад, у ході реверсінжинірингу (зворотного проєктування) бази даних (БД) ІС.

Із прикладної точки зору з одного боку передбачається формування й реалізація якогось загального алгоритму або бази правил перетворення початкового представлення вимоги до ІС на кінцеве представлення цієї ж вимоги або множини інших вимог. При цьому мається на

увазі, що формат текстових конструкцій або елементів ВМ, які описують ці вимоги, апріорно співпадає з форматом вихідних даних алгоритму або системи правил і практично не змінюється з часом. Тому даний спосіб найбільше підходить для таких описів, що зводять до мінімуму кількість елементів алфавіту мови опису або моделювання вимог до ІС. До подібних описів можна віднести більшість розповсюджених ВМ ІС і її компонентів – діаграми потоків даних, діаграми «сутність – зв'язок» (Entity-Relation Diagram, ERD), діаграми класів (ДК) тощо. З іншого боку передбачається створення й постійний розвиток якоїсь множини алгоритмів або якоїсь системи правил перетворення початкового представлення вимоги до ІС у кінцеве представлення цієї ж вимоги або множини інших вимог. При цьому темпи зміни цієї множини алгоритмів або систем правил визначаються темпами появи нових або зміни існуючих понять тезауруса початкового представлення вимоги до ІС. Поява кожного нового, раніше невідомого поняття в такому представленні вимоги до ІС (у вигляді даних, інформації або знань) вимагає доповнення існуючого варіанта реалізації новими алгоритмами або правилами, що встановлюють спосіб і форму опису такого нового поняття в кінцевому представленні цієї вимоги.

Як випливає з моделей сформульованих вимог до ІС і універсума вимог до ІС, процеси розробки ІС [13], що безпосередньо працюють із вимогами, можуть розглядатися як послідовні перетворення представлень вимог у вигляді даних, інформації й знань під час виявлення й формулювання вимог до ІС відповідно до обраної методології розробки ІС. Однак ця модель залишає відкритим питання про вид представлень окремих вимог до ІС, методах і способах формування цих представлень і перетворень цих представлень самих у себе й одне в одне.

Функціональні вимоги, здебільшого, висуваються до таких елементів ІС, як інформаційне забезпечення (ІЗ) та ПЗ. Зазвичай, для опису архітектур цих частин, використовують ERD та ДК, відповідно. Та під час розробки, у деяких випадках, виникає проблема сумісності ІЗ та ПЗ через те, що діаграми виконані з лише таких точок зору на відповідні види забезпечення, для яких візуальні описи були створені. Ця проблема може бути вирішена уніфікацією цих описів. Існуючі технології розробки й коректування ВМ передбачають конвертацію окремих типів ВМ одна в одну відповідно до особливостей технологічних ланцюжків. Як такі технології і засоби їх реалізації можна відзначити технології SSADM, SADT, мову UML [14] і спеціалізований інструментальний засіб моделювання ARIS. Більшість технологій, що реалізують об'єктно-орієнтовані ВМ, припускають використання як допоміжних моделей структурних ВМ БП об'єкта автоматизації й ВМ потоків даних. Структурні ВМ застосовуються, головним чином, для визначення меж БП досліджуваного об'єкта автоматизації, а також для визначення меж проєктованої ІС.

Моделі слугують корисним інструментом аналізу проблем, обміну інформацією між усіма зацікавленими сторонами (користувачами, аналітиками, проєктувальниками тощо), проектування ПЗ та підготовки документації. Моделювання сприяє повнішому засвоєнню вимог, поліпшенню якості системи й підвищенню ступені її керованості. Під терміном «моделювання» розуміється процес створення формалізованого опису системи у вигляді сукупності моделей. Під моделлю ПЗ в загальному випадку розуміється формалізований опис системи ПЗ певного рівня абстракції. Кожна модель визначає конкретний аспект системи, використовує набір діаграм і документів заданого формату, а також відбиває точку зору і є об'єктом діяльності різних людей з конкретними інтересами, ролями й задачами.

Графічні моделі являють собою засоби для візуалізації, опису, проєктування й документування архітектури системи. За ISO/IEC/IEEE 42010 «Systems and Software Engineering – Architecture Description», під архітектурою розуміються «фундаментальні поняття та властивості системи в навколишньому середовищі, втілені в її елементах, відносинах, а також у принципах її проєктування та розвитку.

Архітектурне представлення – це спрощений опис системи з конкретної точки зору, що охоплює певне коло інтересів й опускає об'єкти, несуттєві з даної точки зору.

Архітектурно значимі елементи – це елементи, що мають значний вплив на структуру системи та її продуктивність.

Погляд на архітектуру може бути представлений набором архітектурних моделей (Architecture Model). Кожна модель будується відповідно до конвенцій, установлених для даного виду моделі, причому ці конвенції, зазвичай, визначаються як частина архітектурної точки зору. Моделі забезпечують засоби для обміну деталями між поглядами на архітектуру й використанням множинних позначень в одному погляді на архітектуру.

Вид моделі (Model Kind) визначає конвенції, що задають правила побудови архітектурних моделей.

Мова моделювання включає [15]:

– елементи моделі – фундаментальну концепцію моделювання і семантики;

– нотацію (систему позначень) – візуальне представлення елементів моделювання;

– посібник з використання – правила застосування елементів у рамках побудови тих або інших моделей ПЗ.

Методологія візуального моделювання допомагає подолати деякі організаційні труднощі, що виникають перед командами розробників, групами підтримки якості й менеджерами, що беруть участь у створенні складного ПЗ масштабу підприємства.

Візуальне моделювання являє собою застосування для фіксування ескізів нотацій з розвиненою семантикою, графікою й текстовим змістом. Нотація, така як UML, дозволяє збільшити рівень абстракції, забезпечуючи повноцінний синтаксис і семантику. Таким чином вона поліпшує зв'язок у колективі, який працює над ескізом, і забезпечує точно виражену основу для реалізації.

Практика розробки ІС різноманітного призначення показує, що найсерйозніші протиріччя при спільному використанні структурного й об'єктного підходів до моделювання ІС виникають при стикуванні інформаційного й програмного комплексів ІС.

Інформаційний комплекс являє собою сукупність інтегрованих інформаційних, програмних, технічних, організаційних і лінгвістичних рішень, що забезпечують необхідні умови прийому, тривалого зберігання інформації й видачі її користувачам. Основу даного комплексу становлять рішення щодо структури БД ІС, організації запитів на видачу даних і процедур введення даних у базу. Для опису інформаційного комплексу найчастіше використовують ERD. Дані діаграми є достатньо поширеними й присутні в реалізаціях переважної більшості структурних технологій візуального моделювання ІС.

Існує також модель EER [16]. Вона була розроблена для точнішого відображення властивостей і обмежень, що є в складних БД, які застосовуються в галузях інженерного дизайну й виробництва, телекомунікаціях, пошукових системах, date mining, у складних програмних системах і геоінформаційних ІС. Дана модель є розширенням класичної ER-моделі за рахунок введення:

– субтипів й супертипів (інша назва – субкласи й суперкласи);

– спеціалізації та генералізації;

– спадкування атрибутів і зв'язків таблиць;

– типу категорій.

Нотацією EER-моделі є розширена діаграма «сутність – зв'язок» (Enhanced/Extended Entity-Relationship Diagram, EERD).

Програмний комплекс слід розглядати як об'єднання інтерфейсного й розрахункового комплексів. Він являє собою сукупність інтегрованих програмних, інформаційних, технічних і лінгвістичних рішень, що забезпечують необхідні умови взаємодії користувачів із проєктованою ІС і реалізацію бізнес-логіки, адекватну реальним БП об'єкта автоматизації. Основу даного комплексу становлять, зазвичай, програмні продукти, що реалізують введення, обробку й відображення необхідної користувачеві інформації. Основні складнощі під час їх стикування виникають, як правило, у ході розробки й коректування за результатами тестування численних локальних реалізацій окремих елементів програмного комплексу, для чого потрібна досить значна кількість часу. Для опису програмного комплексу використовується цілий ряд ВМ, серед яких найзначущою є діаграма класів (ДК). Дана діаграма також є однією з найпоширеніших серед ВМ програмних систем і присутня в більшості об'єктно-орієнтованих технологій візуального моделювання ІС.

3. Мета і задачі дослідження

Метою дослідження є вирішення задачі уніфікації візуальних описів архітектур ІЗ та ПЗ ІС. Така уніфікація дозволить замість декількох ВМ, що описують специфікації вимог для розробників окремих видів забезпечень (ІЗ і ПЗ), використовувати єдину ВМ.

Для досягнення поставленої мети необхідно вирішити такі задачі:

– провести аналіз існуючих підходів до формування вимог і створення описів архітектури складних ІС;

– провести аналіз існуючих ВМ, що використовуються у процесах проєктування ІС;

– розробити чи модифікувати модель, яка буде використовуватися для формального опису знань про функціональні вимоги до ІС у процесах проєктування архітектури системи та реалізації елементів ІЗ та ПЗ системи;

– розробити формальний апарат синтезу варіантів описів архітектури ІЗ та ПЗ ІС з використанням запропонованих технологій;

– розробити формальний апарат формування опису архітектури ІЗ і ПЗ ІС, заснований на результатах проведеного в роботі дослідження.

4. Моделі, що використовуються для формального опису знань про функціональні вимоги до інформаційної системи

Представленням вимоги до ІС на рівні знань пропонується вважати [10] онтологію елемента керованого об'єкта або процесу, елемента ІС або ІС у цілому, до яких висувається вимога. У той самий час прагнення повторного використання вимог до ІС обумовлює необхідність додаткового виділення онтологій елементів ІС або ІС у цілому, реалізованих у виконаних раніше проєктах створення ІС. Загалом пропонується розглядати такі види онтологій:

– онтології ПрО, які являють собою знання про об'єкти або БП, що автоматизуються, добуті під час виявлення й аналізу вимог до ІС;

– онтології реалізованих вимог до ІС, які являють собою знання про структури даних і процесах ІС, ІТ-послуг і ІТ-сервісів, створених у рамках попередніх проєктів;

– онтології вимог до створюваної ІС, які являють собою знання про структури даних і процесах ІС, ІТ-послуг і ІТ-сервісів, виокремлені під час формування й аналізу вимог у рамках поточного проєкту створення ІС.

Формування цих онтологій передбачається проводити на основі фреймової моделі знань. Застосування даної моделі обумовлене такими міркуваннями:

– використання фреймової моделі знань дозволяє застосовувати єдиний математичний апарат для опису знань про ПрО для формального представлення вимог до ІС, а також для опису знань, реалізованих в елементах ІС у вигляді моделей ПЗ даної системи;

– використання фреймової моделі дозволяє реалізувати взаємо-однозначне відображення представлень вимог до створюваної ІС на рівні знань в елементи ПЗ цієї ІС.

Для дослідження використовуються існуючі ВМ опису знань про функціональні вимоги, їх формальні описи, дослідження фреймової моделі представлення знань, дослідження формального апарату візуального опису функціональних вимог на рівні знань та формального апарату перетворення візуального опису функціональних вимог на рівні знань у описи ІЗ та ПЗ ІС.

Як ВМ узяті ДК, ERD і EER. Порівняльний аналіз елементів цих моделей дозволяє зробити такий висновок: описи зазначених елементів мають однакову структуру. Підмножина може бути відображена в множину класів . Однак, деталізуючи опис класів і сутностей, можна побачити, що деяка підмножина множини атрибутів ДК не може бути адекватно відображена в множину атрибутів ERD. Це пов'язане з тим, що атрибут в ERD зобов'язаний бути атомарною одиницею, а в ДК атрибут може мати тип і не бути атомарним. Але на рівні знань атомарність можна не враховувати, оскільки архітектори БД при розробці ІЗ враховують атомарність первинних ключів, а розробники ПЗ, у свою чергу, працюють зі своєю специфікою.

Пропонується застосовувати другий з описаних вище підходів. Взаємооднозначне відображення термінів ПрО в таблиці БД ІС вимагає застосування відповідної методики побудови, згідно з якою БД проєктується як набір взаємозалежних вітрин даних, реалізованих у вигляді окремих схем даних. У кожній із цих вітрин таблицею фактів є кореневий елемент конкретної ієрархії фреймів, а консольним таблицям і таблицям вимірів відповідають уточнюючі поняття ПрО. При цьому наслідувані фрейми представляються як слабкі сутності, що залежать від кореневого поняття відповідної ієрархії фреймів онтології ПрО, а кожна окрема вітрина даних являє собою багатомірний куб, який може бути згорнутий до одного універсального відношення, що містить усю множину можливих атрибутів конкретної вітрини даних.

У цьому випадку назви таблиць відповідають фреймам, атрибути таблиць – атрибутам фреймів, а приєднані процедури фреймів відображаються в збережувані процедури й функції, а також у тригери реляційної схеми даних. Ієрархічні зв'язки відображаються в набори сутностей, пов'язаних між собою ідентифікуючими зв'язками, а зв'язки-асоціації – у посилальні обмеження цілісності даних, реалізовані із застосуванням первинних і зовнішніх ключів.

Оскільки дослідження та обговорення отриманих результатів проводиться на прикладі використання реляційної БД, треба розглянути особливості реалізації спадкування у подібних

БД докладніше. Для розв'язання даної задачі пропонується виділення абстрактної узагальненої сильної сутності, яка відповідає кореневому елементу конкретної ієрархії фреймів і є універсальною для різних ПрО. Атрибути такої сутності носять загальний характер, а її застосування припускається при реалізації вимог, характерних для декількох ІС. Для фреймів другого й вище рівнів ієрархії створюються слабкі сутності, які повністю залежать від загальної й універсальної сутності попереднього рівня ієрархії й доповнюють її, оскільки містять додаткову інформацію, характерну для конкретнішого класу об'єктів ПрО ІС. Така слабка сутність як ключовий атрибут (PK) містить зовнішній ключ (FK), що реалізує зв'язок «1:0..1» з узагальнюючим сильним типом сутності. При цьому у випадку, коли відповідний їй фрейм має нащадків, така сутність у свою чергу є сильною для сутностей, відповідних до наступного рівня ієрархії.

Для спрощення роботи з БД ІС реалізуються віртуальні таблиці – представлення, які містять усю інформацію про об'єкт ПрО, поєднуючи дані з декількох таблиць (аналогічно тому, як в об'єктно-орієнтованому програмуванні класи-потоки успадковують усі атрибути класів батьків). Використання представлень дозволяє сховати розгалуженість схеми даних від користувачів і забезпечити відповідність логічного представлення даних конкретній ПрО.

Незважаючи на залежність слабкої сутності від сильної (така залежність виражається в тому, що записи в слабкій сутності не можуть існувати без відповідних записів у сильній сутності), розподіл набору атрибутів одного реального об'єкта ПрО між декількома таблицями, зв'язаними ідентифікуючими зв'язками й відношеннями «1:0..1» і створеними з метою реалізації в реляційній БД ієрархічними зв'язками, утворює необхідність забезпечення цілісності даних на рівні декількох таких таблиць. Подібна цілісність даних нехарактерна для класичної ситуації розміщення таких атрибутів в одній таблиці (на етапі концептуального проєктування таблиці, що мають зв'язки «1:1», завжди поєднуються в одну таблицю).

5. Результати дослідження

Якщо говорити про відображення ДК в EER, то, оскільки мова йде про рівень знань, то при перенесенні опису класу ДК в опис сутності EER можна перенести тільки знання (знання про назви класів відображаються у знання про назви сутностей, їх зв'язки, назви і типи змінних – у назви і типи атрибутів, тощо). Відображення операцій класу з діаграми класів на EER описане нижче. Варто вказати, що деякі досліджувачі зазначають, що в EER використовуються уточнення та узагальнення на основі нотації й термінології UML [17], яка є метамоделлю ДК.

Таким чином, можна говорити про можливість такого відображення:

$$\begin{cases} at^{j}_{C_k} \rightarrow at^{j}_{E_i} \\ mt^{b}_{C_k} \rightarrow nmt^{b}_{E_i}, descmt^{b}_{E_i} \end{cases}, \quad (1)$$

де $at^{j}_{C_k}, j = 0,\ldots,N$ – елемент множини атрибутів класу C_k; $at^{j}_{E_i}$ – елемент множини атрибутів Atr_e сутності E_i; $mt^{b}_{C_k}, b = 0,\ldots,M$ – опис b-ї операції класу C_k; $nmt^{b}_{E_i}$ – елемент множини атрибутів Atr_e сутності E_i, який зберігає назву операції $mt^{b}_{C_k}$; $descmt^{b}_{E_i}$ – елемент множини атрибутів Atr_e сутності E_i, який зберігає текстовий опис операції $mt^{b}_{C_k}$.

Оскільки робота з EERD йде на рівні знань, то можна говорити, що, дотримуючись задуму її авторів [16], дана діаграма досить широко висвітлює знання про ПрО.

В основу запропонованих моделей покладено багатомірну модель даних, яка дозволяє найкращим способом зберігати й обробляти дані й метадані, що характеризують ПрО, раніше розроблені ІС і створювану ІС. Водночас запропоновані моделі можуть бути трансформовані в класичні схеми реляційних БД для спрощення практичної реалізації інтелектуальної ІТ формування й аналізу вимог до ІС.

Слід зазначити, що представлення моделей структурних патернів проєктування вимог до ІС у вигляді кортежів дозволяє розглядати ці патерни як формалізовані висловлення певних вимог до способів організації описів вимог до ІС як об'єктів, автоматизоване оброблюваних у рамках інтелектуальної ІТ формування й аналізу вимог до ІС.

Результати взаємного відображення елементів ДК, EER та ER, які описують ІС на рівні знань, представлено у таблиці 1.

Таблиця 1

Результати взаємного відображення елементів діаграм класів, EER та ER

Елемент ДК	Елемент EER	Елемент ER
Змінна (назва, тип)	Атрибут (назва, тип)	Атрибут (назва, тип)
Назва класу	Назва типу сутності	Назва типу сутності
Операція	Атрибут *A* [назва_операції] Підатрибут *B* [опис_операції], *B* є підатрибутом *A*	Атрибут *A* [назва_операції] Атрибут *B* [назва_операції]_[опис_операції]
Пакет	Домен	Домен
Наслідування	Наслідування	Базовий клас і дочірні класи стають сутностями, пов’язаними між собою зв’язком типу «асоціація» (або асоціативною сутністю)

Для позначення й подальшого формалізованого опису такої розширеної сукупності вимог до ІС пропонується використовувати існуюче поняття «універсум». Це поняття найчастіше позначає фіксовану систему (або системи) об'єктів, до яких належать твердження (висловлення) якої-небудь теорії. Пропонується така інтерпретація поняття «універсум»: «універсум – це «множина всіх можливих систем», з яких лише одна – досліджувана система – реальна, а всі інші (у тому числі й проєктовану ІС) можливо осмислити тільки логічним шляхом, тобто несуперечливим чином описуючи можливі факти або зв'язки досліджуваної системи». Розгортаючи дану інтерпретацію, поняття «універсум» у теорії й практиці створення, впровадження, експлуатації й модернізації ІС можна сформулювати таким чином: «універсум – сукупність даних, інформації та знань про досліджувану систему, об'єкт або процес, як відомих, так і невідомих дослідникові, у розпорядженні якого є скінченна множина методів отримання й обробки цих даних, інформації та знань». Дане означення породжує цілий

ряд висновків, з яких для формалізованого опису вимог до ІС і процесів проєктування архітектури важливі, насамперед [12]:

– точність опису досліджуваної системи, об'єкта або процесу наближується до максимуму в тому випадку, якщо обсяг невідомих дослідникові даних, інформації та знань про цю систему, об'єкт або процес наближується до мінімуму;

– дослідник завжди має припускати, що сукупність невідомих йому даних, інформації та знань про досліджувану систему, об'єкт або процес не є порожньою;

– для практичного застосування Споживачем універсум має мати не максимальну точність, а таку, яка дозволить на підставі сукупності відомих даних, інформації та знань ухвалювати рішення з бажаними для Споживача універсуму характеристиками ефективності і якості.

6. Обговорення отриманих результатів

Одній вимозі фізично відповідають набір класів ПЗ, таблиць БД і мережа фреймів, що описують вимогу у вигляді знань. Отже, фізична реалізація вимоги вимагає її декомпозиції на окремі підвимоги, що фізично подається у вигляді окремих класів, таблиць тощо.

Відомості про особливості механізмів і алгоритмів обробки даних, регламентованих вимогою до ІС, можуть бути витягнуті з візуальних структурних і об'єктних моделей, що описують вимоги на рівні інформації. Їхній опис у вигляді знань може бути реалізований із застосуванням каузальних сценаріїв. У такому випадку інформаційні одиниці мережі фреймів, яку описує вимога, можуть бути доповнені сигнатурами приєднаних процедур фреймів, описаними у вигляді сценаріїв.

Наявність ієрархічних зв'язків між об'єктами ПрО обумовлює утворення фреймами дерев, які додатково можуть бути з'єднані між собою горизонтальними зв'язками. Отримана в такий спосіб мережа фреймів, яка описує вимоги, матиме вигляд «низькорослого лісу» або «чагарнику», у відповідності зі зв'язками об'єктів ПрО, для якої проєктується ІС.

Під час аналізу ПрО й формування вимог до ІС здійснюється декомпозиція цих вимог на підвимоги й виділення окремих об'єктів ПрО, представленнями яких має оперувати ІС. Реалізовані вимоги перед виконанням робіт зі створення елементів ІС, як правило, групуються навколо одного кореневого поняття ПрО й можуть включати дерева понять, що деталізують базове дерево й пов'язані з ним горизонтальними зв'язками. При цьому вимога може вважатися новою, а не уточнюючою існуючу, якщо в мережі фреймів вона є кореневим вузлом нового дерева об'єктів ПрО. Таке дерево деталізується під час подальшого аналізу ПрО й формування вимог, що уточнюють це дерево. У ПЗ це відображається ієрархією класів-спадкоємців одного базового (часто абстрактного) класу, в ІЗ – схемами даних типу «зірка» або «сніжинка», у яких консольні таблиці й таблиці вимірів згруповані навколо базової таблиці фактів, яку вони розширюють і уточнюють. Для зручності роботи з такими наборами таблиць вони можуть поєднуватися в одну схему даних (яка належить одному користувачеві).

Характерною властивістю подібної схеми даних є можливість денормалізації й об'єднання всіх таблиць схеми в одне універсальне відношення, що описує базове поняття ПрО, відповідне до таблиці фактів вітрини даних. Наприклад, таке універсальне відношення може бути створене у вигляді представлення (віртуальної таблиці). Застосування універсальних відношень дозволяє масштабувати моделювання вимог до ІС, оперуючи реалізованими вимогами на рівні підвимог, окремих таблиць БД і класів ПЗ, окремих вимог до ІС, фізично представлених ІТ-сервісів або їх компонентів.

Для формування мережі фреймів, відображуваних згодом в елементи ПЗ та ІЗ ІТ-сервісів ІС, потрібне застосування прийомів класичного об'єктно-орієнтованого аналізу й проєктування (аналогічно сучасним методологіям розробки ПЗ). Сформована в такий спосіб онтологія ПрО придатна для двостороннього відображення представлення вимог на рівні знань в описи ПЗ та ІЗ ІС.

Застосування об'єктно-орієнтованого аналізу дозволяє виявити об'єкти ПрО та зв'язки між ними, у тому числі сформувати ієрархії класів об'єктів.

Об'єктно-орієнтований аналіз дозволяє формувати ієрархії класів ПрО необмежено великої довжини. Формально в сучасних мовах програмування також немає обмежень на кількість рівнів спадкування класів об'єктів. Однак на практиці при проєктуванні й візуальному моделюванні ПЗ не рекомендується застосовувати ієрархії спадкування, що мають більш ніж 5-6 рівнів, бо це утрудняє сприйняття діаграм класів людиною. Крім того, необхідність відображення мережі фреймів у БД ІС також накладає обмеження у вигляді складності реалізації багаторівневих ієрархій у вигляді наборів таблиць БД.

Внаслідок цього найскладнішою задачею при формуванні онтології ПрО у вигляді мережі фреймів є вибір фреймів, що є кореневими елементами дерев онтології, і ухвалення рішення про спадкування фреймів від існуючих або про формування нового кореня дерева фреймів. При роботі з деревами вимог з метою порівняння онтологій різних ПрО й ухвалення рішення про розширення й повторне використання вимоги в створюваній ІС у рамках відповідної ІТ-послуги найскладнішою є задача оцінки подібності об'єктів різних ПрО.

Як один з результатів застосування розробленої технології пропонується розглядати здійснення взаємо-однозначного відображення мережі фреймів, яка описує онтологію ПрО проєктованої ІС у вигляді патернів вимог до ІС, в елементи ПЗ та ІЗ ІС для наступного виконання проєктних робіт з фізичної реалізації системи. Теорія фреймів є основою сучасних об'єктно-орієнтованих мов програмування, і тому відображення патернів вимог, описаних у вигляді мережі фреймів, у сутності БД і класи ПЗ розроблюваної ІС зводиться до розв’язання задачі об’єктно-реляційного відображення.

Оскільки мережа фреймів і діаграма класів засновані на застосуванні загального математичного апарата, однакових типів зв'язків і структур даних, відображення мережі фреймів онтології ПрО в діаграму класів ПЗ здійснюється природним чином: фрейми відображаються в класи з такими самими назвами й наборами атрибутів і методів, зв'язки між фреймами – в однойменні їм зв'язки генералізації й асоціації класів. Стосовно ІЗ ІС основна складність полягає у відсутності в найпоширенішій у даний момент реляційної моделі даних ієрархічних зв'язків і механізму спадкування. Для реалізації задачі об'єктно-реляційного відображення зараз існує кілька типових підходів, найпоширенішими з яких є відображення кожного дочірнього класу (включаючи наслідувані атрибути) в окремі таблиці БД, а також емуляція механізму спадкування за рахунок застосування ідентифікуючих зв'язків між сильними (батьківськими) і слабкими (дочірніми) сутностями.

Внаслідок цього саме необхідність відображення ієрархії фреймів у реляційні схеми вітрин даних обумовлює необхідність застосування правила класифікації фреймів як кореневих самостійних понять і як результат – представлення мережі фреймів у вигляді «низькорослого лісу».

Оскільки ані цілісність сутностей, ані посилальна цілісність повною мірою не забезпечує розв’язання даної задачі, пропонується використання тригерів. Тригери реалізують складні

обмеження цілісності даних, які нереалізовані описовими обмеженнями, що встановлюються при створенні таблиць.

Можна виділити чотири основні типи тригерів, що забезпечують цілісність даних в узагальненій й уточнюючих таблицях:

а) тригери, що забезпечують сюр'єктивне відображення даних у двох таблицях (обов'язкова наявність відповідних записів у головній і залежній таблицях, зв'язок «1:1», оскільки наявність ідентифікуючого зв'язку забезпечує тільки наявність зв'язку «1:0..1»);

б) тригери, що забезпечують унікальність значень комбінацій атрибутів, розміщених у різних таблицях (аналог унікального ключа в одній таблиці);

в) тригери, що забезпечують контроль внесення даних у дочірні таблиці, коли сильна сутність пов'язана з декількома слабкими, а дані в слабких сутностях не мають перетинатися між собою (наприклад, сильна сутність «люди», слабкі – «викладач» і «студент»);

г) тригери, що забезпечують заповнення (аналог not null option) необов'язкових полів сильних сутностей, якщо це потрібно для залежних від них слабких сутностей (наприклад, для абстрактної сильної сутності «людей» поле «контактний телефон» може бути необов'язковим, а для наслідуваної від неї сутності «співробітник» це саме поле, фізично розташоване в головній таблиці, є обов'язковим).

Основні етапи побудови мережі фреймів під час аналізу вимог і ПрО створюваної ІС, а також відповідні їм етапи відображення й реалізації вимог в описи елементів ПЗ та ІЗ ІС наведено у [11].

Оскільки EER-діаграма схожа з діаграмою класів та ER-діаграмою у математиці, наведеній у роботах [18-19], то мережа фреймів працює і з EER-діаграмою.

Вимога на рівні знань може бути описана у вигляді фрейму f, який може бути розділений на окремі структури даних, описувані інтерфейсами фрейму. Як і в об'єктно-орієнтованому програмуванні, кілька різних фреймів, що описують різні вимоги, можуть реалізовувати той самий інтерфейс, тобто мати спільні структурні елементи. Застосування інтерфейсів фреймів дозволяє застосовувати в різних фреймах єдині описи типової структури даних. При цьому клас або фрейм може одночасно реалізовувати кілька інтерфейсів. Крім того, більшість сучасних мов об'єктно-орієнтованого програмування не надають можливість реалізації множинного спадкування, реалізація якого викликає неоднозначності (наприклад, якщо в наслідуваних класах існують однойменні атрибути). Застосування інтерфейсів дозволяє здійснювати не тільки вертикальний аналіз ієрархічної структури класів або фреймів, але й горизонтальний аналіз структурних елементів окремих класів, що компенсує відсутність механізмів множинного спадкування. Комбінації двох зазначених видів аналізу структур фреймів дозволяє здійснювати відображення фреймів, що описують вимоги, у різні елементи ПЗ.

Головним недоліком отриманих результатів дослідження слід вважати обмеженість застосування отриманих результатів через спирання запропонованого підходу на фреймову модель (що вимагає додаткового аналізу придатності описаного у дослідженні до певних ПрО).

7 Висновки

Проведене дослідження базується на припущенні, що функціональні вимоги, здебільшого, впливають на такі компоненти ІС, як ІЗ та ПЗ. Зазвичай, для опису архітектур ІЗ та ПЗ використовують ER-діаграму та діаграму класів, відповідно. Але під час розробки у деяких випадках виникає проблема сумісності ІЗ та ПЗ через те, що діаграми виконані лише з таких точок зору, для яких були створені візуальні описи. Вирішення цієї проблеми полягає

в уніфікації цих описів, для чого слід застосувати формальний апарат, що використовує фреймову модель для відображення опису функціональних вимог на рівні знань з використанням діаграми класів та ER-діаграми.

У статті досліджені три візуальні нотації функціональних вимог, що описують архітектуру ІЗ (ER-діаграма та EER-діаграма) та ПЗ (ДК). За результатами дослідження спільних елементів діаграм та запропонованої мережі фреймів виявлено, що мережа фреймів може відображати опис функціональних вимог на рівні знань у EER-діаграму. Базуючись на отриманому результаті, запропоновано формальний опис взаємно-однозначних відображень спільних елементів ДК, ER-діаграми та EER-діаграми. Отримані результати дозволяють визначити EER-діаграму як таку ВМ, яка буде використовуватися для формалізації процесів візуального відображення описів функціональних вимог до ІС на рівні знань, і запропонувати механізм трансформації цієї ВМ у ER-діаграму та ДК, які, відповідно, описують архітектури ІЗ та ПЗ ІС.

Перелік посилань

1. ДСТУ ISO/IEC/IEEE 15288:2016 Інженерія систем і програмного забезпечення. Процеси життєвого циклу систем (ISO/IEC/IEEE 15288:2015, IDT). [Чинний від 2018-10-01]. Вид. офіц. Київ: ДП «УкрНДНЦ», 2018. 80 с.

2. Настанова до зводу знань з управління проєктами (Настанова PMBOK) Сьоме видання. Project Management Institute, PMI. 2021. 370 с.

3. SSADM Version 4 Reference Manual. Oxford: NCC Blackwell, 1990. 1400 p.

4. Leffingwell D., Widrig D. Managing software requirements: a unified approach/ Dean Leffingwell, Don Widrig. Boston: Addison-Wesley Longman Publishing, 1999. 491 p.

5. Wiegers Karl E., Beatty J. Software Requirements, 3rd Edition /Karl E. Wiegers, Joy Beatty. Redmond, WA: Microsoft Press, 2013. 673 p.

6. Hull E., Jackson K., Dick J. Requirements Engineering / Elizabeth Hull, Ken Jackson, Jeremy Dick. Springer London, 2011. 207 p.

7. Watson M. Managing Smaller Projects: A Practical Approach / M. Watson. Multi-Media Publications Inc., 2006. 240 p.

8. Capturing Architectural Requirements. URL: https://www.ibm.com/developerworks/rational/library/4706-pdf.pdf (дата звернення: 19.10.2023)

9. Babok: A Guide to the Business Analysis Body of Knowledge, v3. International Institute of Business Analysis, 2015. 512 p.

10. Евланов М.В. Модели, методы и информационная технология разработки архитектуры сложных информационных систем на основе функциональных требований: дис. … докт. техн. наук: 05.13.06. Харьков, 2017. 429 с.

11. Левыкин В.М. Паттерны проектирования требований к информационной системе: моделирование и применение / В.М. Левыкин, М.В. Евланов, М.А. Керносов: монография. Харьков: ООО «Компанія СМІТ», 2014. 320 с.

12. Евланов М.В. Концепция представления требований к информационной системе / М.В. Евланов // *Вісник національного технічного університету «ХПІ». Збірник наукових праць. Серія «Нові рішення в сучасних технологіях».* 2012. № 68 (974). С. 32-40.

13. Davis Alan M. 201 Principles of Software Development / Alan M. Davis. New York: McGraw-Hill, 1995. 240 p.

14. Fowler M. UML Distilled. A Brief Guide to the Standard Object Modeling Language 3rd Edition / Martin Fowler. Addison-Wesley Professional, 2004. 175 p.

15. He X., Ma Z., Shao W., Li G. A metamodel for the notation of graphical modeling languages. /He, Xiao; Ma, Zhiyi; Shao, Weizhong; Li, Ge. *31st Annual International Computer Software and Applications Conference (COMPSAC 2007),* Beijing, China, 2007. Vol. 1. P 219-224.

16. Teorey T. A logical design methodology for relational databases using the extended entity–relationship model. / Toby J. Teorey, Dongqing Yang, James P. Fry // *ACM Computing Surveys*. 1986. Vol. 18, № 2. P. 197–222.

17. Chapter 8. Enhanced EER Model. URL: http://comet.lehman.cuny.edu/jung/cmp420758/chapter8.pdf (дата звернення: 12.10.2023).

18. Левыкин В.М. Параллельное проектирование информационного и программного комплексов информационной системы / В.М. Левыкин. М.В. Евланов, В.С. Сугробов // *Радиотехника*. 2006. Вып. 146. С. 89-98.
19. The Enhanced Entity-Relationship Model. URL: https://www.is.informatik.uni-kiel.de/~thalheim/HERM/HERMindetail.pdf (дата звернення: 12.10.2023).

Надійшла до редколегії 27.10.2023 р.

Кожанов Адріан Євгенійович, аспірант кафедри ІУС ХНУРЕ, м. Харків, Україна, e-mail: adrian.kozhanov@nure.ua. ORCID: https://orcid.org/0009-0002-3586-6886/