УДК 004.9:33:62
І-74

Авторський колектив

Устенко С. В., д.е.н., проф., *Рамазанов С. К.*, д.е.н., д.т.н., проф., *Степаненко О. П.*, д.е.н., проф., *Іванченко Г. Ф.*, к.т.н., проф., *Сендзюк М. А.*, к.е.н., проф., *Ситник Н. В.* к.е.н., проф., *Шевченко К. Л.*, д.т.н, проф., *Ріппа С. П.* д.е.н., проф., *Тішков Б. О.*, к.е.н., доц., *Галузинський Г. П.,* к.т.н., доц., *Городній О. В.*, к.т.н., доц., *Денісова О. О.*, к.е.н., доц., *Гордієнко І. В.*, к.е.н., доц., *Краснюк М. Т.*, к.е.н., доц., *Курков М. С.*, к.е.н., доц., докторант, *Помазун О. М*, к.е.н., доц., *Зінов'єва І. С.*, к.е.н., доц., *Держжук О. В.*, к.е.н., доц., *Кривошеєв К. В.*, к.т.н., доц., *Бібко О. О.*, асис., *Кустаровський О. Д.*, асп., *Ріппа М. Б.*, к.е.н., доц., *Гіваргізов І. Г.*, асп., *Погореловська І. Д.*, к.е.н., доц., *Бадер Омар Ахмад Далайін*, к.е.н. (Йорданія)

Рецензенти

Дивак М. П. докт.техн.наук, проф.
Тернопільський національний економічний університет

Горбовий А. Ю. докт.техн.наук, проф.
Інститут інформаційних технологій Державної фіскальної Служби України

Джаладова І. А. докт.фіз.-мат.наук, проф.,
Київський національний економічний університет імені Вадима Гетьмана

Рекомендовано до друку Вченою радою КНЕУ
Протокол № 10 від 30.05.2019

І-74 **Інформаційні управляючі системи та технології /** За заг. ред. докт. екон. наук, професора Устенко С. В. — Київ : КНЕУ, 2019. — 419 с., [5] с.
ISBN 978-966-926-307-0

У монографії розглянуто концепції та методології проектування та розвитку інформаційних управляючих систем на підприємствах, у наукових інститутах, ІТ-сфері, виробництві, логістиці, освітньому процесі; методи, моделі та алгоритми вирішення завдань управління високотехнологічних виробничих систем, систем еволюційного прогнозування синергетичного ефекту злиття та поглинання підприємств, інтелектуальних систем прийняття рішень; принципи та стратегії проектування інформаційних систем з використанням технологій криптоекономіки та блокчейн, інформаційно-комунікаційних технологій підтримки інформаційної безпеки систем, створення гібридних інтелектуальних систем; упровадження технічних аспектів підвищення надійності пам'яті, методів отримання вимірювальної інформації, розроблення спеціалізованих систем цифрової обробки інформації.

Для викладачів, наукових працівників, аспірантів, студентів, спеціалістів, які займаються проблемами впровадження сучасних технологій при проектуванні та впроваджені інформаційних систем в управлінні підприємствами, організаціями, а також в ІТ-бізнесі, освіті.

УДК 004.9:33:62

ISBN 978-966-926-307-0

ЗМІСТ

В П. Куприяновский, П. В. Куприяновский, С. А. Синягов. — International Journal of Open Information Technologies ISSN: 2307-8162 vol. 4, no. 1, 2016. — С. 4–11.

8. Accenture Technology Vision 2015. Digital Business Era: Stretch Your Boundaries. 2015 Accenture. — [Electronic resource] — Access mode: https://www.accenture.com/sg-en/_acnmedia/Accenture/Conversion-Assets/Microsites/Documents11/Accenture-Technology-Vision-2015.pdf — Name from the screen.

9. Monitoring the Digital Economy & Society 2016–2021 [Electronic resource]. — Access mode: http://ec.europa.eu/eurostat/documents/341889/725524/Monitoring+the+Digital+Economy+%26+Society+2016-2021/7df02d85-698a-4a87-a6b1-7994df7fbeb7 — Name from the screen.

10. The 2018 Digital University Staying Relevant in the Digital Age [Electronic resource]. — Access mode: https://www.pwc.co.uk/assets/pdf/the-2018-digital-university-staying-relevant-in-the-digital-age.pdf — Name from the screen.

11. *Лук'яненко Д. Г.* DIGITAL UNIVERSITY: Проект розбудови цифрового університету в ДВНЗ «Київський національний економічний університет імені Вадима Гетьмана» / Д. Г. Лук'яненко, О. П. Степаненко // зб. мат. Національної наук.-метод. конф. «Цифрова економіка», 4–5 жовтня 2018 р., м. Київ. — К.: КНЕУ, 2018. — С. 245–249.

12. *Stepanenko Olga P.* Development of information and communication technologies in the digital economy / Olga P. Stepanenko // Моделювання та інформаційні системи в економіці. Збірник наукових праць. Випуск 96'2018. — К.: КНЕУ, 2018. — С. 189–202.

***Денісова О. О.*, к.е.н., доц.**

КОНЦЕПЦІЯ ТА МЕТОДОЛОГІЇ СИСТЕМНОГО ІНЖИНІРИНГУ

Початок формування дисципліни «системний інжиніринг» («системна інженерія», «system engineering») зазвичай відносять до 30−40-х років XX століття, коли виникла необхідність визначення та маніпулювання властивостями цілісної системи, що можуть відрізнятись від суми властивостей складових цієї системи, у відповідь на появу численних складних інженерних проектів, передусім у військовій галузі. За словами С. Рамо [1], автора першого визначення, системний інжиніринг — це дисципліна,

що вимагає розгляду проблеми загалом, урахування всіх граней і змінних, що стосуються соціальних і технічних аспектів. За визначенням Х. Ейснера [2], системний інжиніринг — це ітеративний процес синтезу «згори-донизу», розробки та оперування системою реального світу, що задовольняє новим оптимальним чином повний набір вимог до цієї системи.

Системний інжиніринг розвивався впродовж минулих десятиліть (див. табл. 1), і нині це — визнана зріла дисципліна [2], яку розглядають як належний підхід для реалізації успішних систем.

Таблиця 1

ВАЖЛИВІ ДАТИ В СТАНОВЛЕННІ ДИСЦИПЛІНИ СИСТЕМНОГО ІНЖИНІРИНГУ

1937	Робота Британської міждисциплінарної команди з аналізу протиповітряних захисних систем
1939–1945	Bell Labs підтримує військовий проект США зі створення зенітного ракетного комплексу з наведенням NIKE
1951–1980	MIT керує програмою створення SAGE — системи напівавтоматичної координації дій перехоплювачів шляхом програмування їх автопілотів по радіо комп'ютерами, розташованими на землі, — однієї з перших великомасштабних глобальних комп'ютерних мереж
1956	Розробка методів системного аналізу корпорацією RAND
1962	Публікація книги «Методологія системного інжинірингу» А. Д. Холла
1968	Зародження практики моделе-керованого системного інжинірингу під час інженерних робіт корпорації TRW у рамках програми розробки міжконтинентальних балістичних ракет Повітряних сил США
1990	Створення INCOSE (International Council on Systems Engineering) — Міжнародної ради із системного інжинірингу
2002	Випуск стандарту ISO/IEC 15288: 2002(E) — Systems engineering — System life cycle processes (Системний інжиніринг — Процеси життєвого циклу систем), одного з базових стандартів системного інжинірингу [6]

Водночас формування відповідних методологій ще не завершено, жодна з них не стала домінуючою, тому виникає потреба в їх аналізі, визначенні перспективних тенденцій розвитку й проблем, що вимагають вирішення.

За результатами проведеного аналізу різних концепцій [1–5] було визначено такі базові постулати системного інжинірингу:

– системне мислення — виявлення, вивчення, діагностика і діалог, спрямовані на осягнення, моделювання та обговорення

систем реального світу з метою кращого розуміння, визначення та роботи з ними;

– інтерактивність процесів з підтримкою навчання, виявлення справжніх вимог і безперервне вдосконалення;

– спрямованість на складні системи — виявлення неочікуваних властивостей і непрогнозованої поведінки та мінімізація небажаних наслідків. Для особливо великих проектів уживають термін «система систем», який позначає набір систем, що взаємодіють і таким чином продукують результат, недосяжний для відокремленої системи;

– ефективне керування змінами — зменшення ризику під час створення нових або модифікації старих складних систем;

– охоплення технічних та управлінських процесів з акцентом на гарному прийнятті рішень, першочергово — на ранніх стадіях життєвого циклу системи;

– міждисциплінарний характер і загальна зорієнтованість, оскільки доцільність і можливість застосування існуючих підходів і методологій не залежать від галузі або варіанта життєвого циклу;

– залежність системного інжинірингу від комунікацій та взаєморозуміння учасників.

Якщо від самого початку системний інжиніринг знайшов застосування в аерокосмічній та оборонній галузях, то згодом до них приєднались усі види транспорту, енергетика, медицина, телекомунікації, фінанси, освіта та ін.

У 1968 році під егідою Наукового Комітету НАТО було проведено конференцію, у назві якої вперше було вжито термін «інжиніринг програмного забезпечення» (програмна інженерія, software engineering), що дало поштовх розвитку однойменної дисципліни. У програмах світових університетів відповідна навчальна дисципліна з'явилась лише наприкінці 80-х років XX ст. [4].

Програмну інженерію визначають як науку побудови комп'ютерних програмних систем на інженерній основі за методами, засобами та інструментами програмування, сучасними стандартами процесів ЖЦ, менеджменту та керування якістю [7].

Результати порівняння системного та програмного інжинірингу наведено в табл. 2.

По суті, кодування становить менше 10 % часу роботи програмного інженера [4], а решта — припадає на наукові, інженерні, управлінські, економічні та виробничі роботи.

Таблиця 2

ПОРІВНЯННЯ ФУНКЦІЙ І ХАРАКТЕРИСТИК СИСТЕМНОГО ТА ПРОГРАМНОГО ІНЖИНІРИНГУ

ФУНКЦІЇ І ХАРАКТЕРИСТИКИ		
Системний інжиніринг		Програмний інжиніринг
Інтерпретація високорівневих, неясних вимог	Визначення потреб Визначення вимог	Розробка й деталізація до нижнього рівня точних вимог
Гарантування відповідності проектів підсистем і ПЗ	Архітектурні рішення для системи/ПЗ Проектування системи/ПЗ	Формальні методи, процеси розробки ПЗ, методи та засоби, кодування
Перевірка коректності зовнішніх інтерфейсів, інтерфейсів між підсистемами і ПЗ	Розпізнавання стейкхолдерів, бачень і точок зору Установлення інтерфейсів, визначення правил і характеристик	Уведення в дію інтерфейсів між програмними модулями, даними й шляхами комунікацій
Оркестрування дисциплін Пошук компромісів на рівні системи	Пошук компромісів	Компроміси між повторним, новим та оновленим
Інтерфейс з клієнтом (не програмний)	Шляхи взаємодії з клієнтом (визначення вимог до ПЗ і проектування)	Інтерфейс з клієнтом (кодування)
Технічні і бізнес-цілі	Формування нефункціональних вимог	Атрибути якості ПЗ
ЖЦ придбання, V-модель зі зворотнім зв’язком	Гнучка (Agile) розробка	Спринти, релізи
Визначення підходів до верифікації/валідації	Визначення та виконання верифікації й валідації системи/ПЗ	Виконання верифікації та валідації ПЗ
Менеджмент програм	Керування швидкими змінами	Менеджмент проектів ПЗ
Не стосується деталей		Не стосується технічного забезпечення
Спрямованість на широке охоплення		Спрямованість на глибину (ПЗ)

Спільною тенденцією системного та програмного інжинірингу є використання моделе-зорієнтованого підходу. Природна мова з усією її виразністю та зрозумілістю для позатехнічних фахівців не підходить належним чином для автоматизованого інжинірингу через

багатозначність, неточність і важкість формалізації. Тому з кінця 60-х років XX ст. для інжинірингу стало звичним використання цифрових моделей, але нині цей підхід значно змінився [8] — неявні моделі стали явними, вторинні моделі перетворились на основні, а описові — набули обов'язкового характеру. Нинішні моделі є настільки точними й повними, що стали звичним засобом для аналізу, комунікацій, порозуміння та узгодження. Крім цього, їх використання дає змогу поліпшити якість проекту за рахунок ранньої ідентифікації проблемних вимог, співвіднесення вимог з компонентами програмного та технічного забезпечення, ретельного відстеження виконання вимог, верифікації вимог і валідації проекту, зменшення помилок і підготовки узгодженої документації. Додатково до цього відбувається підвищення продуктивності, зменшення ризикованості та поліпшення точності оцінок вартості проекту.

Моделе-зорієнтований системний інжиніринг (Model-based Systems Engineering, MBSE) — це формалізоване застосування моделювання для підтримки формування вимог, проектування, аналізу, верифікації та валідації упродовж всього життєвого циклу системи [1]. При цьому модель є абстракцією системи й може подавати будь-який аспект системи з будь-якої точки зору. Отже, модель, по суті, складається з множини подань (описів, views) з використанням обраних нотацій, текстових описів, таблиць, засобів формалізації. Кожне подання складається із сукупності елементів, що є екземплярами елементів онтології в моделі. Водночас модель не є простим набором подань, а має точну структуру, визначену фреймворком (framework) через певну кількість точок зору (способів подань, методів опису, viewpoint). Точку зору тут можна розглядати як шаблон подань у рамках моделі. Кожна точка зору також складається з певної кількості відповідних елементів. Як подання має узгоджуватись з точкою зору, з якою його асоційовано, так і елементи подань візуалізують елементи точок зору і мають узгоджуватись з ними.

Серед обмежень MBSE можна назвати складність застосування модельного підходу до нефункціональних вимог. Незважаючи на декларовану простоту сучасних мов моделювання, не можна розраховувати на абсолютне розуміння моделі всіма зацікавленими особами. Слід враховувати також, що реалізація MBSE вимагає дисциплінованої та гарно підготовленої команди проекту та зрілих процесів організації.

У [9] визначено як провідні такі MBSE-методології, що визначають набори взаємопов'язаних процесів, методів і засобів: Harmony-SE, OOSEM, RUP SE, Vitech MBSE, JPL SA, OPM.

Harmony-SE від IBM Telelogic — підмножина процесів розробки інтегрованих систем (див. рис. 1) і програмного забезпечення Harmony (деякі постачальники вживають слово «процес» як синонім слова «методологія») [9].

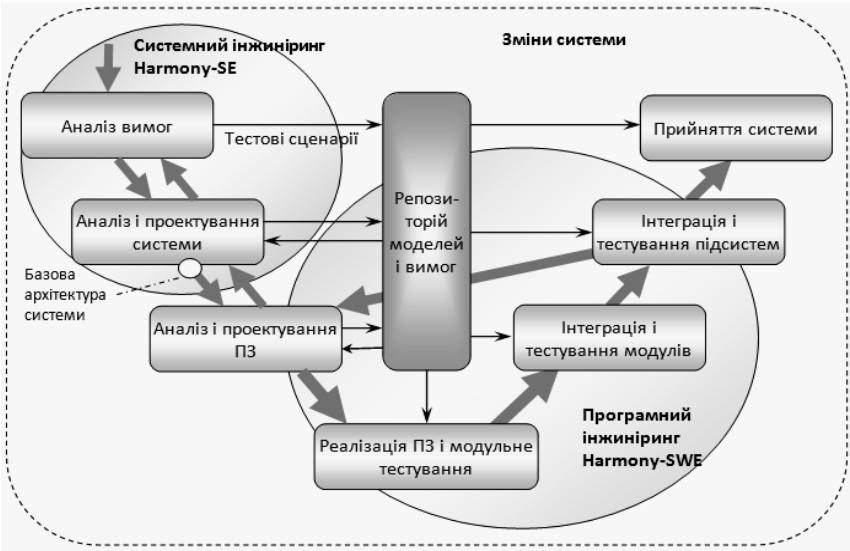


Рис. 1. Процес розробки інтегрованих систем
і програмного забезпечення Harmony

Harmony-SE охоплює аналіз вимог, функціональний аналіз системи та архітектурне проектування й переслідує три головні цілі: виявлення потрібної функціональності системи, визначення асоційованих станів і режимів системи, а також розташування системних станів/режимів за фізичною архітектурою. Harmony-SE ґрунтується на класичній V-моделі життєвого циклу, маючи своєю особливістю створення репозиторіїв вимог, моделей, випробувань, що оновлюються на кожному етапі.

Harmony-SE використовує підхід до моделювання «керований запитом сервіс» (service request-driven) з артефактами мови SysML. Структура системи подається за допомогою структурних діаграм, базовим елементом яких є блок. Комунікації між блоками ґрунтуються на повідомленнях (запитах сервісу). Надані сервіси та зміни стану/режиму або операції (діяльність) описуються операційними контрактами. Функціональна декомпозиція проводиться через декомпозицію операційних контрактів діяльності.

SysML (The Systems Modeling Language, мова моделювання систем) є мовою загального призначення, що підтримує специфікацію, аналіз, проектування, верифікацію та валідацію систем. SysML є розширенням підмножини мови UML з використанням механізму профілів [1, 10].

Поява SysML є результатом рішення, прийнятого INCOSE у січні 2001 року щодо необхідності пристосування мови UML для потреб системного інжинірингу. Першу специфікацію було випущено в 2004 році як результат відкритого проекту за участю багатьох фахівців і постачальників інструментальних засобів (SysML Partners). З моменту затвердження специфікації OMG SysML v. 1.0 у 2006 році й до сьогодні тривають роботи з удосконалення мови за результатами її практичного використання, зокрема щодо мета-моделі та забезпечення сумісності MBSE-засобів.

Крім того, що UML орієнтована на моделювання програмних систем, а SysML — на широкий спектр застосувань системної інженерії, остання має й інші переваги:

– гнучкіша й виразніша семантика за рахунок двох додаткових діаграм — діаграми вимог, на основі якої відбувається інжиніринг вимог, і діаграми параметрів, що використовується для проведення аналізу продуктивності та чисельного аналізу. Як результат — SysML придатний для моделювання як програмного, так і технічного забезпечення, інформації, процесів, персоналу, обладнання тощо;

– компактність — має меншу кількість як діаграм, так і конструктів;

– підтримка моделей, подань і точок зору для керування моделюванням.

Важливою особливістю SysML є механізм розподілу (allocation), який дає змогу з'єднувати елементи різних моделей. Зокрема, може здійснюватись прив'язка поведінки/функцій до компонентів, які цю поведінку/функції реалізують, прив'язка структури, що з'єднує логічні компоненти з фізичними, і прив'язка потоків об'єктів.

У SysML передбачено створення дев'яти діаграм, що мають чотири «опори»: структуру системи, її архітектурні елементи (логічні й фізичні) та їх взаємозв'язки; поведінку системи, включно зі зміною станів, послідовностями дій, функціями та взаємодією; вимоги до системи; параметри, тобто обмеження, що накладаються на систему (див. рис. 2).

При цьому діаграми вимог і параметрів є новими порівняно з UML, діаграми визначення блоків, внутрішніх блоків і діяльності є модифікаціями діаграм UML 2, а решта діаграм не мають особливостей.

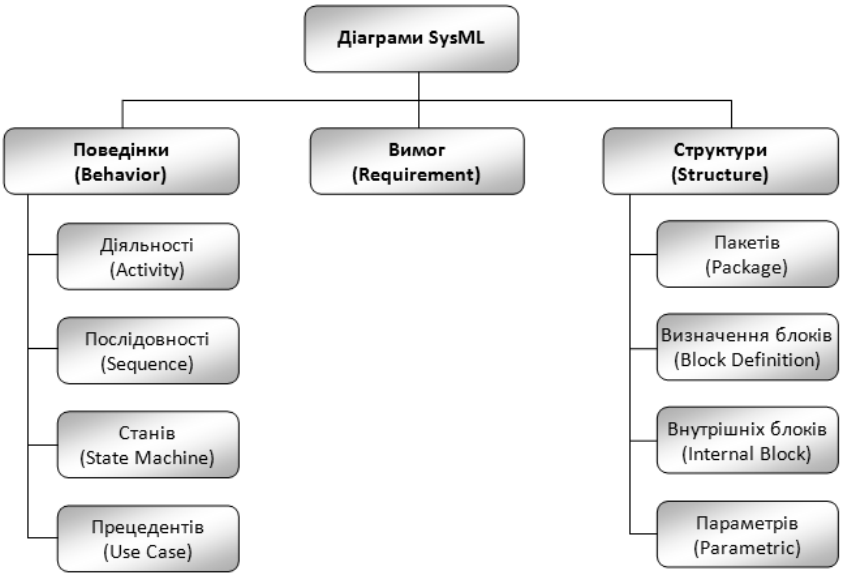


Рис. 2. Діаграми мови SysML

Діаграма визначення блоків (Block definition diagram) описує системну ієрархію та класифікацію систем і компонентів. Фактично блоки є аналогом класів UML. Діаграма внутрішніх блоків (Internal block diagram) подає внутрішню структуру системи в термінах частин, портів і конекторів. Діаграма пакетів використовується для організації моделі.

Діаграма варіантів використання (Use case diagram) містить високорівневий опис функціональності системи. Діаграма діяльності (Activity diagram) подає потоки даних і контролю між діями. Порівняно з UML, потік контролю доповнено керуючим оператором, створено можливість моделювати безперервні системи, з потоком можна асоціювати ймовірність і розширено правила моделювання. Діаграма послідовностей (Sequence diagram) відображає взаємодію між частинами системи, що працюють сумісно. Діаграма станів (State machine diagram) описує переходи між станами та дії, які їх система або її частини мають здійснити у відповідь на події.

Діаграма параметрів (Parametric diagram) містить обмеження на значення параметрів системи, такі як продуктивність, надійність, маса тощо, необхідні для інженерного аналізу. По суті, це спеціалізований тип діаграми внутрішніх блоків, що подає блоки

обмежень, їх параметри та властивості блоків, до яких обмеження й блоки відносяться. Для підтримки такого моделювання передбачено окремий тип блока — ConstraintBlock (блок обмежень), який визначає множину параметрів та одне чи більше обмежень для них. За замовчуванням, параметри є не зорієнтованими й не мають позначень причинно-наслідкових зв'язків. ConstraintBlock можна використовувати для побудови математичних виразів, обчислення статистичних показників або функції корисності. SysML також дає змогу визначити типи значень, зазначивши одиниці виміру, розмірність і розподіл ймовірностей для опису властивостей блоків.

Діаграма вимог (Requirement Diagram) подає ієрархію вимог і відношення залежності, такі як походження (derive), задоволення (satisfy), верифікації (verify), уточнення (refine), відстеження (trace) з можливим обґрунтуванням (rationale). Для забезпечення можливості повторного використання вимог передбачено відношення копіювання (copy) між вимогою-«хазяїном», тобто оригіналом, і вимогою-«рабом», тобто копією, випадком повторення вимоги-«хазяїна». Відношення надають можливість зіставити вимоги між собою й пов'язати їх з проектними моделями та контрольними прикладами. Для зручності сприйняття вимоги в SysML можуть також подаватись у вигляді таблиць.

Діаграма вимог є засобом зв'язку між традиційними засобами керування вимогами й системними моделями. Оскільки не завжди всі архітектурні, дизайнерські та тестувальні артефакти подаються на основі SysML, може відбуватись переведення (меппінг) вимог SysML у формат RIF (ReqIF, Requirements Interchange Format) — XML-сумісний формат файлів для обміну вимогами між програмами різних розробників, що підтримується OMG і використовується в багатьох галузях [11]. Водночас відбувається узгоджування RIF, SysML та AP 233.

AP 233 — це частина ISO 10303, який неформально називають стандартом обміну даними моделей продукту (STandard for the Exchange of Product model data, STEP) [12]. По суті, ISO 10303 є сімейством стандартів, що визначають надійну перевірену часом методологію опису даних продукту впродовж його життєвого циклу. STEP широко використовується для систем автоматизованого проектування (Computer-Aided Design, CAD) і керування даними продукту та життєвим циклом (Product data/lifecycle management, PDM/PLM) у багатьох галузях, зокрема аерокосмічній та автомобільній, а також кораблебудуванні. Протоколи додатків (Application Protocol, AP), що входять до складу ISO

10303, стосуються специфічних продуктів і процесів. AP 233 спрямований на системний інжиніринг і проектування, має модульну структуру (див. рис. 3) й охоплює поведінку та структуру системи, моделювання системи, підтримку рішень, вимоги, аналіз і закупки, менеджмент програм і проектів, верифікацію й валідацію, керування ризиком і проблемами.

Рис. 3. Модулі AP 233

Його розроблено як нейтральну інформаційну модель для обміну даними між інструментальними засобами системного інжинірингу, автоматизованого проектування, опису архітектури системи та іншими, взаємопов'язаними з ними.

AP 233 та SysML відрізняються термінологією та обсягами, їх визначено на основі різних фреймворків. Водночас, оскільки SysML є сумісним з UML, а AP 233 базується на структурі мови моделювання даних про виробництво EXPRESS, то сумісність EXPRESS та UML 2 уможливлює обмін даними між AP 233 та SysML. Слід також зазначити, що під егідою OMG було проведено роботи з меппінгу AP 233 та SysML [13].

Об'єктно-орієнтований метод інжинірингу систем від INCOSE (Object-Oriented Systems Engineering Method, OOSEM) [14] — заснований на моделях SysML метод підтримки специфікації, аналізу, проектування та верифікації систем. OOSEM поєд-

нує об'єктно-орієнтовану концепцію з традиційними методами інжинірингу «згори-донизу» та іншими методами моделювання з метою полегшення інтеграції з об'єктно-орієнтованою розробкою програмного забезпечення, розробкою технічного забезпечення й тестування, а також підтримки повторного використання систем та еволюції проекту.

Основними видами діяльності за OOSEM є аналіз потреб стейкхолдерів, визначення вимог до системи, логічної архітектури, синтез варіантів архітектур, підготовлених на основі логічної, оптимізація та оцінювання альтернатив, валідація й верифікація системи. При цьому передбачається використання таких специфічних прийомів і методів, як каузальний аналіз, моделювання підприємства, опрацювання контексту, аналіз варіацій вимог, системна та логічна декомпозиція, критерії розмежування, розподіл вузлів.

Раціональний уніфікований процес системного інжинірингу (Rational Unified Process for Systems Engineering, RUP SE) від IBM — методологія RUP, адаптована для системного інжинірингу в частинах специфікації, аналізу, проектування й розроблення систем з підтримкою моделе-керованої розробки (Model-Driven Systems Development, MDSD) [15]. Базові принципи RUP, такі як паралельність проектування та ітеративної розробки, було збережено й доповнено придатними для конфігурування шаблонами потоків робіт і дисциплін з визначення компонентів технічного й програмного забезпечення, а також робочих ролей з інжинірингу системи. Так само як і RUP, RUP SE має на меті підтримку систематичного визначення, організації, обговорення та керування вимогами. Обидві методології підтримують керування змінами та контроль якості. Ключовими елементами, доданими в RUP SE, є такі:

– нові ролі — додатково до архітекторів, розробників, тестувальників та ін. передбачається робоча роль системного інженера, пов'язана передусім зі специфікацією та розгортанням усієї системи, а також оперуванням загальносистемними вимогами;

– нові артефакти й потоки робіт — якщо RUP охоплює такі сфери, як зручність використання (юзабіліті), можливість підтримки, продуктивність, масштабованість, то RUP SE доповнено доменом системного інжинірингу, що стосується безпеки, навчання та підтримки логістики;

– акцент на бізнес-моделюванні — RUP SE не вносить змін до бізнес-моделювання RUP, але рекомендує аналізувати бізнес-прецеденти з пов'язаними з ними бізнес-акторами й потоками бізнес-подій задля коректного визначення системних вимог, а також виведення останніх з бізнес-вимог;

– точки зору для системного інжинірингу — RUP SE передбачає вимірювання архітектури системи за способами подання (viewpoint) і рівнями моделювання (model level) (див. табл. 3).

Таблиця 3

АРТЕФАКТИ RUP SE ЗА РІВНЯМИ МОДЕЛЮВАННЯ І ТОЧКАМИ ЗОРУ

Рівень моделювання	Точка зору					
	Виконавська (worker)	Логічна (logical)	Інформаційна (information)	Дистриб'юції (distribution)	Процесна (process)	Геометрична (geometric)
Контекст (Context)	Визначення ролей, моделювання діяльності	Специфікація діаграми прецедентів	Подання даних підприємства	Подання, специфічні для предметної області		Подання, специфічні для предметної області
Аналіз (Analysis)	Поділ системи	Логічна декомпозиція продукту	Концептуальна схема даних продукту	Подання локалізації продукту	Подання процесів продукту	Макети
Проектування (Design)	Інструкції	Проектування компонентів ПЗ	Схема даних продукту	Проектування засобів медіаконтролю	Часові діаграми	Автоматизоване проектування (CAD)
Реалізація (Implementation)	Конфігурація технічного та програмного забезпечення					

Комірки таблиці відповідають поданням. Точки зору відповідають певному набору проблем стейкхолдерів. Зокрема, подання дистриб'юції описує, як функціональність системи поширюється за фізичними ресурсами. Локалізація в цьому контексті є частиною системи, узагальненим або абстрактним поданням фізичних ресурсів. Рівень моделювання визначається як підмножина архітектурних моделей, що подають певний рівень специфікації (від абстрактного до конкретного), нижчі рівні більше відображають технологічну специфіку. Рівень моделювання не є рівнем абстракції, більше того, рівень моделювання може містити кілька рівнів абстракції. На одному рівні моделювання групуються артефакти з однаковим рівнем деталізації;

– поліпшення масштабованості — системна архітектура подається множиною UML/SysML-діаграм для опису її за різними

точками зору й рівнями моделювання. Відповідні нові артефакти дають можливість розділити систему на підсистеми і за локалізаціями, де безпосередньо відбувається оброблення. До кожної підсистеми разом з її локалізацією висувається окремий набір вимог, що за рахунок масштабування дає змогу працювати з дуже великими та складними проектами;

– закріплені й похідні вимоги — у RUP SE передбачено два типи системних вимог: функціональні, що відповідають прецедентам, і додаткові, що охоплюють нефункціональні вимоги (показники якості), такі як надійність та супроводжуваність. Вимога до підсистеми або локалізації вважається закріпленою за цією підсистемою або локалізацією, якщо останні несуть виключну відповідальність за виконання цієї системної вимоги. Якщо вимогу визначено в результаті вивчення взаємодії окремої підсистеми або локалізації з іншими для виконання системної вимоги, її називають похідною;

– деталізований розгляд діяльності на рівні підсистем — RUP SE регламентує виведення системних вимог з бізнес-вимог через деталізацію прецедентів, а додатково до цього — специфікацію потоку подій на рівні підсистем, такий додатковий крок необхідний для прийняття рішень щодо місця події та співвіднесення подій і процесів;

– підтримка проектування додаткових компонентів — якщо RUP зосереджено на програмному забезпеченні, то RUP SE підтримує різні компоненти, зокрема технічне забезпечення, їх окреслення підтримується аналізом описів прецедентів підсистем і локалізацій, що генеруються до розробки проектних рішень.

Методологія моделе-керованого системного інжинірингу від Vitech (Vitech MBSE) — корпорація Vitech пропонує свою методологію як набір підручників із прив'язкою до власного набору інструментів CORE [16]. В основу MBSE-методології Vitech покладено чотири взаємопов'язані паралельні види діяльності з системного інжинірингу, що підтримуються спільним репозиторієм системного проекту (рис. 4):

– аналіз початкових вимог;

– функціональний/поведінковий аналіз;

– валідація та верифікація проектних рішень (validation & verification, V&V);

– синтез/архітектура.

Кожен з названих видів діяльності має контекст у вигляді однойменного домену.

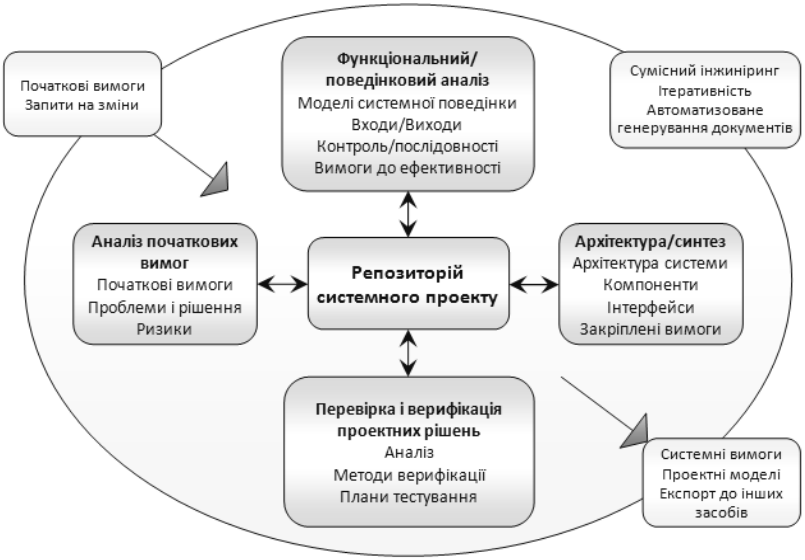


Рис. 4. Основні види діяльності за Vitech MBSE

Базовими положеннями методології Vitech MBSE є такі:

– моделювання проблеми й простору рішень з використанням спеціальної мови моделювання та семантично-значущої графіки для забезпечення відстежуваності моделі, послідовності графічних зображень, автоматичного документування, перевірки синтаксису й семантики артефактів, динамічної верифікації та імітування, генерування графічних подань і звітів, автоматичного контролю узгодженості, а також підтримки технічних комунікацій аналітиків вимог, системних дизайнерів і розробників. Як наголошується в методології, мова визначення системи (System Definition Language, SDL) має бути структурованою, загально-орієнтованою, точною, контекстно та інструментально незалежною;

– використання репозиторію проектних рішень системи;

– використання інструментальних засобів для рутинної роботи й зосередження працівників на творчих завданнях;

– «горизонтальний» інжиніринг має передувати «вертикальному». На виконання цієї вимоги запропоновано «модель цибулі» (Onion Model) — інкрементний процес з ітераціями основних паралельних видів діяльності на кожному рівні. Коли роботи на певному рівні проекту системи завершено, слід зняти цей «шар цибулі» й перейти до наступного. Проектування вважається заве-

ршеним, коли досягнуто бажаний рівень деталізації. Команда не може повернутись назад більш як на один рівень. Якщо дієвого узгодженого рішення немає на жодному з рівнів, слід зменшити жорсткість умов та обговорити модифікації проектних рішень, які можна внести на попередньому рівні. «Модель цибулі» підтримує дві часові лінії системного інжинірингу — проектування «згори-донизу» і зворотний інжиніринг.

Як необхідні й достатні для повного опису системи визначено три моделі: модель контролю (функціональної поведінки), модель інтерфейсу (введення/виведення) і модель фізичної архітектури (компонентів). Вимоги до продуктивності та ресурси співвідносять з частинами або комбінаціями названих трьох моделей. Завершення розробки цих моделей означає закінчення системного інжинірингу — розроблено проектні специфікації всіх системних компонентів, плани валідації та верифікації визначено й повністю відстежено.

Хоча із самого початку для функціонального аналізу (аналізу поведінки) у Vitech MBSE було передбачено підтримку візуальних моделей поведінки й конструкцій виконуваною графічною мовою EFFBDs (Enhanced Function Flow Block Diagrams, розширені діаграми функцій, потоків, блоків), а також діаграм FFBD, N2-діаграм і діаграм поведінки, згодом було додано підтримку DoDAF і SysML, зокрема засобами CORE Spectrum і GENESYS.

Аналіз станів JPL (JPL State Analysis, SA) — розроблена Лабораторією реактивного руху (Jet Propulsion Laboratory, JPL) MBSE-методологія, що використовує переваги моделе- та станозорієнтованих підходів. Стан визначається як подання зафіксованого в певний момент становища системи, що розвивається, а моделі описують, як змінюються стани [8].

SA передбачає процес фіксування вимог до системи й ПЗ у формі явних моделей, зменшуючи відмінності між вимогами до ПЗ, сформульованими системними інженерами, і реалізацією цих вимог програмними інженерами за рахунок прямого перекладання вимог на ПЗ і кращого розуміння програмістами потрібної поведінки системи. При цьому розрізняють «стан системи» й «знання про цей стан». Справжній стан системи може бути вкрай складним, але зазвичай знання про нього фіксуються в простіших абстракціях, корисних і достатніх для його характеристики, — «змінних стану». Тоді відомий стан системи — це значення змінних стану в певний момент часу. Визначення стану й моделі надає змогу оперувати системою, прогнозувати її майбутній стан, спрямовувати її до бажаного стану та оцінювати ефективність.

Ключові характеристики:

– стан є явним — знання про стан системи, що контролюється, подається набором змінних стану;

– оцінювання стану відокремлюється від його контролю — їх пов'язують лише змінні стану. Ця вимога сприяє об'єктивному оцінюванню стану системи й модульності, забезпечує послідовне використання стану по всій системі, спрощує проектування, прискорює реалізацію в ПЗ;

– технічні адаптери забезпечують виключний інтерфейс між системою, що контролюється, і контрольною системою. Вони формують межу архітектури, забезпечують всі вимірювання та абстракції команд, що використовуються для контролю й оцінювання, відповідають за трансляцію та керування входами й виходами;

– усеохоплюючі моделі — моделі використовуються як для виконання (оцінювання та контролю стану), так і для високорівневого планування (керування ресурсами). Методологія SA вимагає, щоб моделі документувались в явній формі, найзручнішій для певного додатку;

– ціле-зорієнтований процес замкнутого циклу — замість опису бажаної поведінки в термінах низькорівневих команд відкритого циклу, SA використовує цілі, що є обмеженнями для змінних станів у певні інтервали часу;

– пряме переведення в програмне забезпечення — для контрольованих елементів установлюється пряма відповідність з компонентами модульної архітектури. Прикладом такої архітектури є Система даних місії — архітектура вбудованого програмного забезпечення, роботизованих кораблів-розвідників [17].

Методологія SA також має три керівні принципи:

– контролю підлягають усі аспекти оперування системи. Він здійснюється лише на основі моделей системи, що контролюється. При цьому контрольна система має чітко відокремлюватись від системи, що контролюється;

– моделі системи, що контролюється, мають бути недвозначно визначені й використані в спосіб, який забезпечує згоду системних інженерів. Розуміння стану є фундаментальною умовою для успішного моделювання. Усі необхідні знання та всі потрібні дії мають бути виражені в термінах стану системи, що контролюється;

– створення моделей з мінімізацією подальших перетворень, необхідних для проектувальників і розробників.

Методологія SA визначає ітеративний процес виявлення та моделювання станів з належною розробкою моделей упродовж усього життєвого циклу проекту. Рекомендується використання

бази даних станів (State Database), що містить інформацію, зазвичай оформлену у різних артефактах системного інжинірингу. Додатково описано механізми використання моделей для проектування артефактів ПЗ та оперування системою. Таким чином, SA спрямована на три головні види діяльності — моделювання поведінки на основі станів, що зумовлює проектування ПЗ на основі станів і ціле-орієнтований операційний інжиніринг.

Підхід SA може здаватись значним відходом від документо- і моделе-зорієнтованого проектування, хоча насправді ця методологія доповнює функціональний аналіз моделе-зорієнтованого процесу. Ітеративний процес декомпозиції як частина традиційного функціонального аналізу має за результат ієрархії функцій, фізичних компонентів (структура розбивки продукту) і вимог, пов'язаних між собою (рис. 5).

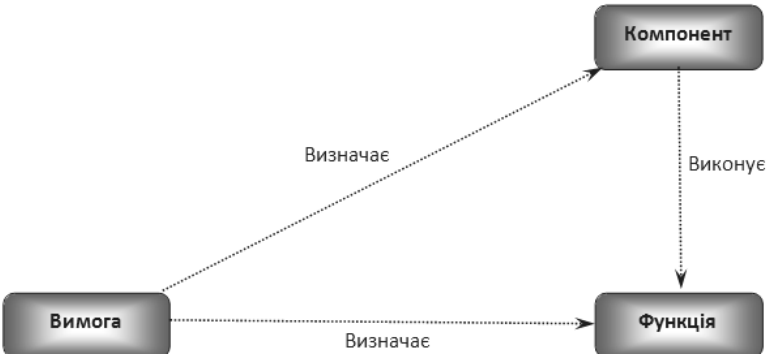


Рис. 5. Елементи та відношення за функціональним аналізом

SA доповнює їх змінними станів, командами й вимірами (рис. 6).

Об'єктно-процесна методологія Дорі (Dori Object-Process Methodology, OPM) — концептуальний підхід до розробки систем, підтримки життєвого циклу та еволюції [18]. OPM-методологія містить формальні візуальні моделі — діаграми об'єктів-процесів (Object-Process Diagrams, OPDs) та описів функцій, структури й поведінки обмеженою природною мовою OPL (Object Process Language, мова об'єктів і процесів). Кожен OPD-конструкт виражається семантично еквівалентним OPL-реченням або частиною речення й навпаки. OPL є мовою з подвійним призначенням, вона зорієнтована як на людину, так і на машину. Методологію прийнято як стандарт ISO/PAS 19450 «Automation systems and integration — Object-Process Methodology (OPM)».

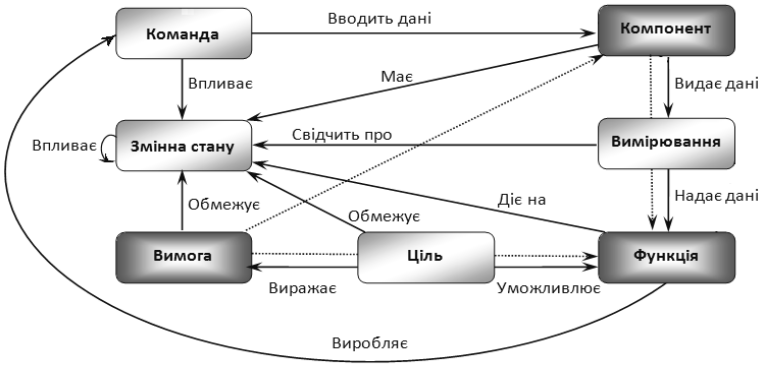


Рис. 6. Елементи та відношення за аналізом станів, синтезованим з функціональним аналізом

OPM ґрунтується на передумові, що все в світі, зрештою, є або об'єктом, або процесом, тому в методології об'єкти й процеси є будівельними блоками вищого рівня, що узагальнено називаються речами. На рівні моделювання в OPM передбачено три типи сутностей:

– об'єкти — речі, які існують реально або потенційно, фізично або ментально;
– процеси — зразки трансформації об'єкта;
– стани — ситуації, в яких може бути об'єкт.

Об'єкти існують, а процеси трансформують їх шляхом генерування, споживання або впливу. Стани використовуються для опису об'єктів і є невід'ємними від останніх. Кожної миті кожен об'єкт може мати лише один стан.

OPM підтримує багату семантику моделювання. Зокрема, усі можливі зв'язки розділено на структурні, які виражають постійні довготривалі відношення між об'єктами або процесами в системі, і процедурні, які виражають поведінку системи. Структурні зв'язки використовують для подання агрегації/участі, прояву/характеристики, узагальнення/спеціалізації, класифікації/виявлення прикладу. За допомогою процедурних зв'язків відображають стимулювання, трансформування, а також вплив подій, умов і викликів. Подані на одній діаграмі, структурні й процедурні зв'язки подають повну картину системи однією графічною моделлю із супроводжувальним текстом.

Додатково OPM підтримує керування складністю за допомогою механізмів уточнення/абстракції, що дають змогу описувати систему на будь-якому бажаному рівні деталізації без втрати чіткості й зрозумілості специфікації: розгортання/згортання

(unfolding/folding) структурної ієрархії, збільшення/зменшення (in-zooming/out-zooming) для показу/приховування внутрішніх деталей, вираження/замовчування стану (state expressing/ suppressing) для оперування показом станів об'єкта.

Оскільки термін «процес» є терміном методології і має специфічну семантику, в OPM для позначення життєвого циклу системи використовується словосполучення «еволюція системи» і «OPM-системний процес». Масштабування на рис. 7 показує, що розробка системи за Дорі складається зі стадій опису вимог, аналізу та проектування, реалізації, використання й підтримки. Усі ці процеси використовують єдину OPM-онтологію, що зменшує розриви між стадіями.

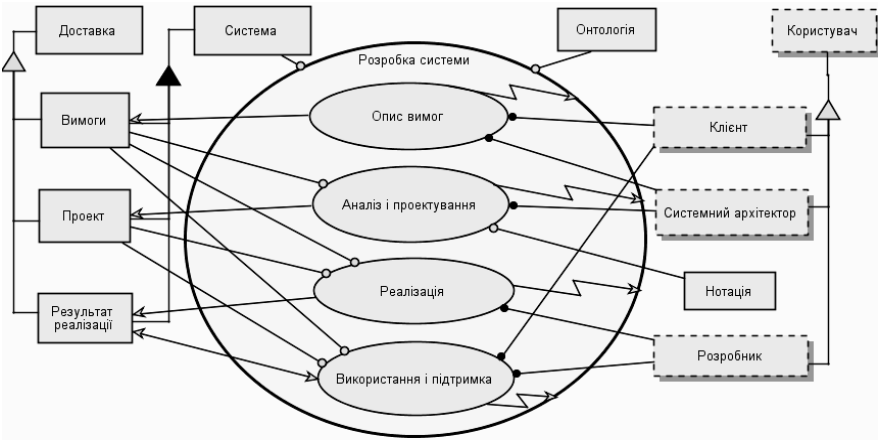


Рис. 7. Об'єктно-процесна діаграма «Розробка системи»

Для вдосконалення проекту й перевірки його на помилки, повноту та відповідність встановленій меті рекомендовано проводити статичне й динамічне тестування, зокрема анімацію системи, яка імітує поведінку системи з урахуванням подій, умов, розгалужень і циклів. Виявлені помилки та недоробки коригуються на рівні моделі до початку реалізації.

Стадія реалізації починається з визначення профілю, який містить, зокрема, цільову мову програмування. Після цього на основі OPL-скрипту та внутрішніх правил, які містять пари «тип OPL-речення — шаблон мовою програмування», автоматично генерується основа для реалізації, що містить як структурні, так і поведінкові аспекти системи. Під час відлагодження та тестування основа, модифікована в програмну систему, зіставляється

з вимогами і в разі їх повного й правильного виконання передається для використання. Під час підтримки клієнт збирає нові вимоги, що записуються за допомогою вбудованого механізму в OPM-форматі, для реалізації в новому поколінні системи.

За результатами проведеного аналізу зроблено висновок, що загальноприйнятою практикою є використання мови SysML, яка для системного інжинірингу стала стандартом де-факто, — практично всі сучасні методології або ґрунтуються на ній, або підтримують. При цьому слід зазначити, що SysML не замінює собою методологію, так само, як MBSE є незалежним від певного типу діаграм — вирішальним критерієм під час їх вибору є придатність способу подання системи для потреб користувача або аудиторії.

Альтернативою SysML є OPM. Їх порівняння наведено в табл. 4 [18].

Таблиця 4

ПОРІВНЯННЯ SYSML І OPM

Характеристика	SysML	OPM
Теоретичне підгрунтя	UML, об'єктно-орієнтований підхід	Мінімальна універсальна онтологія, об'єктно-процесна теорема
Обсяг документації (сторінки)	≈1670 (з них 1400 з UML)	≈180 (стандарт ISO 19450 з додатками)
Установа стандартизації	OMG	ISO
Кількість типів діаграм	9	1
Об'єкт верхнього рівня	Блок (об'єктний клас UML)	Річ (об'єкт або процес)
Керування складністю	Декомпозиція на основі аспектів	Декомпозиція за рівнем деталізації
Ієрархічна декомпозиція	На деяких типах діаграм	Так
Кількість символів	≈120	≈20
Графічне подання	Так	Так
Текстове подання	Ні	Так
Вбудована розрізнювання «фізичне — інформаційне»	Ні	Так
Розрізнювання «системне — зовнішнє»	Частково (використання границь)	Так
Логічні умови	Ні	Так
Моделювання ймовірності	Ні	Так
Можливості з виконання, анімаційної імітації, валідації, верифікації	Частково (у деяких засобах для деяких діаграм)	Так
Доступність автоматизованих засобів	Багато, частина — доступні безоплатно	OPCAT

Принциповими відмінностями OPM від SysML є:

– розгляд процесів на рівні з об'єктами, на відміну від об'єктно-орієнтованого підходу, згідно з яким процеси інкапсулюються в об'єктах. SysML розглядає процеси опосередковано, як прецедент, діяльність, дію, метод або послідовність, ураховуючи різні аспекти й конотації;

– як об'єкти, так і процеси можуть бути фізичними або інформаційними, системними (частиною системи) або екзогенними (частиною оточуючого середовища, що взаємодіє із системою);

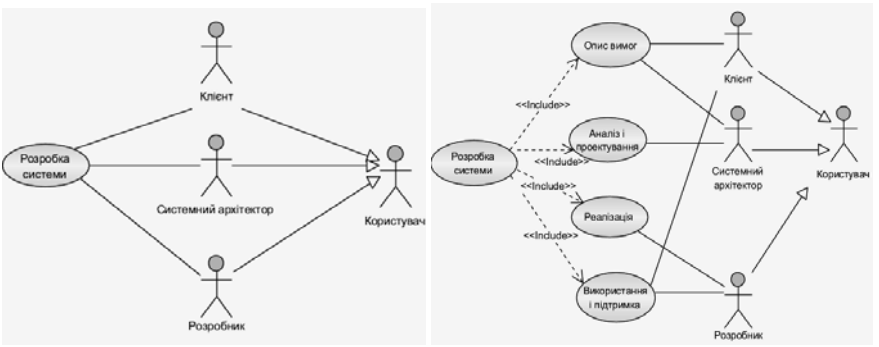
– єдиний тип діаграм — об'єктно-процесна, на відміну від SysML/UML, у якому кожен аспект подається окремим типом діаграм. Взаємопов'язані OPD-діаграми подають частини системи з різним ступенем деталізацій і становлять чітку ієрархію. Хоча OPM не передбачає таких важливих для інжинірингу елементів, як параметричні обмеження у SysML, їх можна подати атрибутами зі значеннями, якими оперують під час обчислень. З одного боку, мінімальність онтології дає змогу уникнути ускладнення моделі й зробити її максимально зрозумілою для всіх читачів, а за допомогою механізмів розгортання та масштабування можна вибрати потрібний ступінь деталізації. З іншого боку, підхід «структура-поведінка» робить проблематичним відображення множинності способів взаємодії компонентів системи. Водночас SysML, як і UML, передбачаючи діаграми для всебічного опису поведінки, збільшують відповідно їх кількість;

– використання природної мови. Тоді як використання текстових документів для проектування супроводжується ризиком неповного й помилкового формулювання проектних рішень, вони є невід'ємним засобом забезпечення комунікацій та обговорень різнорідними групами, до яких, крім інженерів, можуть входити споживачі, користувачі, стейкхолдери, менеджери, експерти та інші особи, які не мають спільних для всіх контексту й нотації. Генерування документів з OPD-моделей виключає помилки в них і залишає моделі єдиним джерелом знань про систему. Водночас типи OPL-речень можуть бути використані як шаблони для формулювання вимог до системи на основі аналізу природномовних документів і звірки вимог і моделей з документами предметної області.

Для того, щоб скористатися перевагами обох підходів, слід установити відповідність між діаграмами:

– для діаграми прецедентів фізичні екзогенні об'єкти кореневої об'єктно-процесної діаграми розглядають як актори, а процеси — як прецеденти. На рис. 8 подано варіанти діаграм прецедентів,

сформованих на основі OPD-діаграм (перший рівень OPD-діаграми представлений на рис. 7);



а) б)

Рис. 8. Діаграми прецедентів, сформовані за OPD-діаграмами «Розробка системи»: а) нульового рівня, б) першого рівня

– для діаграми визначення блоків добирають системні об’єкти та структурні відношення між ними та іншими елементами;

– діаграми діяльності можна скласти за всіма OPM-процесами, що містять підпроцеси, подані за допомогою збільшення. Під час визначення контрольних потоків слід аналізувати OPM-семантику — розташування процесів і типи зв’язків між ними.

Висновок. Керованість моделями стала невід’ємною характеристикою системного інжинірингу, що підтверджується численними стандартами та методологіями. Обрана за мету підтримка повного життєвого циклу й складність сучасних систем зумовлюють розмаїття вимог до системних моделей: простота й зрозумілість для різних користувачів з мінімумом попереднього навчання, абстрагування та повнота під час обговорення із зацікавленими особами й специфікації вимог, точність і високий рівень деталізації для забезпечення належної реалізації. Як результат — є необхідність у побудові різних моделей, зокрема різними мовами. Можливості для цього забезпечують алгоритми та автоматизовані засоби трансформації моделей одна в одну, а також формати обміну даними між різними програмними засобами. Водночас залишається потреба розвитку як методологій, так і мов моделювання за визначеними в результаті дослідження напрямами:

– створення метамоделі системного інжинірингу й забезпечення явних семантичних зв’язків між високорівневою архітек-

турною моделлю та деталізованими аналітичними моделями з підтримкою міжмодельних трансформацій;

– визначення алгоритмів і правил перекладу системних моделей, розроблених загально-орієнтованими мовами, на мови предметно-орієнтовані, зокрема під час програмної інженерії у виконувані моделі програмних систем із залученням відповідних знань;

– підтримка викладення системних моделей природною мовою та автоматичного генерування моделей обраною мовою моделювання на основі природномовних текстів;

– збільшення семантичної та прагматичної складових моделей для забезпечення автоматизованого узгодження вимог з інтересами стейкхолдерів, валідації, верифікації та аналізу проектних рішень, оцінювання та добору варіантів з їх обґрунтуванням, візуалізації моделей за інтересами й потребами окремих користувачів або їхніх груп.

Література

1. *Holt J., Perry S.* SysML for Systems Engineering: A model-based approach. –London, IET Publishing, 2013. — 876 p.
2. INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities. — INCOSE, 2015. — 315 p.
3. Guide to the Systems Engineering Body of Knowledge (SEBoK) — режим доступу http://www.sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)
4. *Laplante Ph. A.* What Every Engineer Should Know about Software Engineering. — CRC Press, 2007. — 298 p.
5. *Estefan J. A.* Survey of Model-Based Systems Engineering (MBSE) Methodologies — режим доступу http://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf
6. SE Standards — режим доступу https://www.incose.org/about-systems-engineering/se-standards
7. *Лавріщева К. М.* Програмна інженерія. — К.: Академперіодика, 2008. — 319 с.
8. *Holt J. and oth.* Foundations for Model-based System Engineering. From Patterns to Models. — The Institution of Engineering and Technology, 2012. — 360 p.
9. *Douglass B. P.* Harmony aMBSE Deskbook Version 1.00. Agile Model-Based Systems Engineering Best Practices with IBM Rhapsody — режим доступу https://docs.wixstatic.com/ugd/21dc4f_4991d336015d4207b2d42d3ba1fc8d28.pdf

10. About the OMG System Modeling Language Specification. Version 1.5. — режим доступу https://www.omg.org/spec/SysML/
11. About the Requirements Interchange Format Specification. Version 1.2 — режим доступу https://www.omg.org/spec/ReqIF/1.2/
12. *U'Ren J.* An Overview of AP233 STEP's Systems Engineering Standard — режим доступу https://ndiastorage.blob.core.usgovcloudapi.net/ndia/2003/systems/slides.ppt
13. SysML and AP233 Mapping Activity — http://www.omgwiki.org/OMGSysML/doku.php?id=sysml-ap233:mapping_between_sysml_and_ap233
14. INCOSE Object-Oriented Systems Engineering Method (OOSEM) — режим доступу http://www.omgwiki.org/MBSE/doku.php?id=mbse:incoseoosem
15. Rational Unified Process for Systems Engineering. RUP SE1.1. — режим доступу ftp://ftp.software.ibm.com/software/rational/web/whitepapers/2003/TP165.pdf
16. *Long D., Scott Z.* A Primer for Model-based Systems Engineering. — режим доступу http://www.ccose.org/media/upload/MBSE_Primer_2ndEdition_full_ Vitech_2011.10.pdf
17. *Dvorak D. and oth.* Software Architecture Themes In JPL's Mission Data System — режим доступу http://citeseerx.ist.psu.edu/viewdoc/download?doi= 10.1.1.459.3789&rep=rep1&type=pdf
18. *Dori D.* Model-based Systems Engineering with OPM and SysML. — Springer, 2016. — 411 p.

***Краснюк М. Т.,* к.е.н., доц.**
***Кустаровський О. Д.,* асп.**

КОНЦЕПЦІЯ ТА МЕТОДОЛОГІЇ ПРОЕКТУВАННЯ АДАПТИВНОЇ ЛОГІСТИЧНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Головна мета проектування та реінжинірингу логістичної інформаційної системи (ЛІС) — поліпшення ефективності функціонування логістичних процесів підприємства, оптимізація всіх його бізнес-процесів.

Зокрема, без ефективної та адекватної ЛІС менеджменту вітчизняної логістичної компанії складно оптимально та оперативно здійснювати управління такими класичними прийомами логістики, як: інтеграція ланцюжка створення вартості, оцінювання ефективності маршрутів, використання інтелектуальних інформаційних технологій, оптимізація комунікацій, прогнозування