МАГІСТЕРСЬКА РОБОТА

МР. ІПм - 20.00.00.000 ПЗ

Група ІПм-23-3

Перцович Руслан

2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Перцович Руслан Васильович

(прізвище, ім'я, по батькові)

УДК 004.942

(індекс)

МАГІСТЕРСЬКА РОБОТА

Методологія специфікації та перевірки моделі системи у високо інтегрованих проєктах

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Перцович Р.В.

(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Крихівський Михайло Васильович, к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту

Завідувач кафедри

доц. Бандура В.В.

(посада) (підпис) (дата) (ініціали та прізвище)

Нормоконтроль

доц. Вовк Р.Б.

(посада) (підпис) (дата) (ініціали та прізвище)

Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Івано-Франківськ – 2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень магістр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
Зав. кафедрою **ІПЗ**
доц. В.В. Бандура
“ 04 ” вересня 2024 р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Перцовичу Руслану Васильовичу

(прізвище, ім’я, по-батькові)

1. Тема магістерської роботи “ Методологія специфікації та перевірки моделі системи у високо інтегрованих проєктах ”

керівник проекту (роботи) Крихівський Михайло Васильович, к.т.н., доцент

затверджені наказом закладу вищої освіти від “ 22 ” листопада 2024 р. № 781/7

2. Строк подання студентом проекту (роботи) 15 грудня 2024 р.

3. Вихідні дані до проекту (роботи) Теоретичні концепції та формальні моделі побудови та функціонування інформаційних технологій високо інтегрованих проєктів

4. Зміст розрахунково - пояснювальної записки(перелік питань, які потрібно розробити)

1. Дослідження предметної області перевірки моделей у високо інтегрованих проєктах
2. Особливості побудови алгоритмічних схем та нотацій засобами мов моделювання систем
3. Дослідження концепцій та понять системної інженерії на основі моделей
4. Представлення методології перевірки моделі системи у високо інтегрованих проєктах

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

1. Три постулати підходу MBSE (рис. 1.1)
2. Діаграма випадків використання системи спостереження (рис. 1.3)
3. Приклад вимог SysML (рис. 1.4)
4. Приклад діаграми вимог SysML (рис. 1.5)
5. Блок зі своїми обов’язковими та блоковими характеристиками (рис. 1.6)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата
Перевірка на плагіат	доц., к.т.н. Вовк Р.Б.	

7. Дата видачі завдання 04 вересня 2024 р.

Керівник _____
(підпис)

Завдання прийняв до виконання_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назви етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Підбір і вивчення літератури по темі магістерської роботи	15.09.2024	виконано
2	Аналіз концепцій та алгоритмів предметної області	29.09.2024	виконано
3	Дослідження предметної області перевірки моделей у високо інтегрованих проєктах	15.10.2024	виконано
4	Особливості побудови алгоритмічних схем та нотацій засобами мов моделювання систем	08.11.2024	виконано
5	Дослідження концепцій та понять системної інженерії на основі моделей	20.11.2024	виконано
6	Представлення методології перевірки моделі системи у високо інтегрованих проєктах	01.12.2024	виконано
7	Затвердження пояснювальної записки роботи завідувачем кафедри	15.12.2024	виконано

Студент – магістр _____
(підпис)

Керівник роботи _____
(підпис)

АНОТАЦІЯ

Магістерська робота: 84 с., 42 рис., 52 джерела.

Тема: Методологія специфікації та перевірки моделі системи у високо інтегрованих проєктах

Об'єкт дослідження: процеси розробки, інтеграції та перевірки моделей систем у високо інтегрованих проєктах.

Мета роботи: розробка методології специфікації та перевірки моделей систем у високо інтегрованих проєктах з використанням сучасних підходів до моделювання для забезпечення їх інтероперабельності, інтеграції та надійності.

Предмет дослідження: методи та методологія специфікації та перевірки моделей систем із застосуванням мови моделювання SysML у високо інтегрованих проектах.

Результати дослідження

В роботі запропоновано вдосконалену методологію специфікації та перевірки моделей систем для високо інтегрованих проєктів на основі використання SysML та MBSE підходу і досліджено методи інтеграції моделей у складних системах через процеси трансформації моделей та їх адаптації до різнорідних середовищ розробки.

Висновок

Застосування онтологічної веб-мови (OWL) та семантичних веб-технологій дозволяє інтегрувати знання та забезпечити автоматизацію процесів моделювання. Це відкриває нові можливості для побудови систем, здатних до автоматизованого оброблення та аналізу даних.

ВИСОКО ІНТЕГРОВАНІ ПРОЄКТИ, МОДЕЛЮВАННЯ СИСТЕМ, МІЖОПЕРАЦІЙНІСТЬ, ТРАНСФОРМАЦІЯ МОДЕЛІ, ОНТОЛОГІЧНА ВЕБ-МОВА, OWL, СИСТЕМНА ІНЖЕНЕРІЯ.

ABSTRACT

Master Thesis: 84 pp., 42 fig., 52 sources.

Thesis Subject: Methodology of system model specification and verification in highly integrated projects

Research object: processes of development, integration and verification of system models in highly integrated projects.

The purpose of the work: development of a methodology for the specification and verification of system models in highly integrated projects using modern modeling approaches to ensure their interoperability, integration and reliability.

Research subject: methods and methodology of specification and verification of system models using the SysML modeling language in highly integrated projects.

Research results

The work proposes an improved methodology for the specification and verification of system models for highly integrated projects based on the use of SysML and the MBSE approach and explores the methods of integrating models in complex systems through the processes of model transformation and their adaptation to heterogeneous development environments.

Conclusion

Application of Ontological Web Language (OWL) and Semantic Web technologies allows integration of knowledge and automation of modeling processes. This opens up new opportunities for building systems capable of automated data processing and analysis.

HIGHLY INTEGRATED PROJECTS, SYSTEMS MODELING, INTEROPERABILITY, MODEL TRANSFORMATION, ONTOLOGICAL WEB LANGUAGE, OWL, SYSTEMS ENGINEERING.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

AEC - Architecture Engineering Construction

MBSE - Model Based Systems Engineering

INCOSE - International Council on Systems Engineering

OMG - Object Management Group

SysML - System Modeling Language

PSM - Parametric Systems Modeling

RDF - Resource Description Framework

RDFS - Resource Description Framework Schema

OWL - Web Ontology Language

BDD - Block Definition Diagram

DSL - Domain Specific Language

DSM - Domain Specific Modeling

QVT - Query/View/Transformation language

BIM - Building Information Modeling

DBB - Design Bid Build

CDE - Common Data Environment

ICT - Information and Communications Technology

CAD - Computer-Aided Design

IFC - Industry Foundation Classes

GUID - Globally Unique Identifier

STEP - Standard for the Technical Exchange of Product Model Data

ISO - International Standards Organization

XML - Extensible Mark-Up Language

XSLT - eXtensible Stylesheet Language Transformations

HVAC - Heating, Ventilation and Air Cooling

URN - Uniform Resource Name

SDE - Systems Development Environment

MOF - Meta Object Facility

ВСТУП

Актуальність теми.

Сучасні високо інтегровані проєкти в таких галузях, як аерокосмічна, оборонна, автомобільна промисловість, інформаційні технології та енергетика, відзначаються складністю архітектури, багатокомпонентністю та взаємозв'язаністю їхніх систем і підсистем. У таких проєктах ефективне управління моделями, їх узгодженість, інтеграція та перевірка є критично важливими для досягнення надійності та функціональності кінцевих продуктів.

Застосування традиційних підходів до проєктування та розробки стає недостатньо ефективним через високий ступінь взаємодії між різними підсистемами та постійно зростаючу кількість вимог до систем. У цьому контексті модельно-орієнтована системна інженерія (MBSE) дозволяє перейти до системного підходу, де модель виступає як основний артефакт управління складними системами на всіх етапах їхнього життєвого циклу. Це не лише дозволяє забезпечити узгодженість та цілісність системи, але й значно покращує процеси інтеграції та виявлення помилок на ранніх етапах розробки. SysML (Systems Modeling Language) як універсальна мова моделювання дозволяє описувати різні аспекти системи, зокрема її структуру, поведінку, вимоги та параметри. Висока гнучкість та здатність до адаптації SysML під специфічні вимоги робить її ідеальним інструментом для високо інтегрованих проєктів, де потрібна точна інтеграція різних систем та компонентів. Крім того, інтеграція семантичних веб-технологій, зокрема OWL (Ontology Web Language), стає важливим кроком у забезпеченні автоматизації та інтероперабельності між різними системами. Вона дозволяє ефективніше працювати з великими обсягами даних та знань, що є ключовим фактором у сучасних складних проєктах.

Таким чином, актуальність теми дослідження зумовлена необхідністю вдосконалення процесів специфікації, інтеграції та перевірки моделей у

високо інтегрованих проєктах, де від точності моделей і їхньої міжопераційності залежить успіх реалізації проєкту. Використання MBSE, SysML та OWL створює нові можливості для ефективного управління складними системами, забезпечуючи їх узгодженість, надійність та гнучкість у масштабуванні.

Мета дослідження - розробка методології специфікації та перевірки моделей систем у високо інтегрованих проєктах з використанням сучасних підходів до моделювання для забезпечення їх інтероперабельності, інтеграції та надійності.

Об'єкт дослідження - процеси розробки, інтеграції та перевірки моделей систем у високо інтегрованих проектах.

Предмет дослідження - методи та методологія специфікації та перевірки моделей систем із застосуванням мови моделювання SysML у високо інтегрованих проектах.

Відповідно до мети роботи було сформовано наступні **задачі:**

- Дослідити особливості високо інтегрованих проєктів та процеси управління моделями у таких системах.
- Проаналізувати можливості використання SysML як мови моделювання для опису архітектури систем.
- Розробити методологію інтеграції та трансформації моделей у середовищах високо інтегрованих проєктів.
- Описати процеси перевірки моделей на основі діаграм використання та вимог.
- Оцінити можливості застосування семантичних веб-технологій та OWL для інтеграції знань у моделях складних систем.

Методи дослідження.

У роботі використовувались методи моделювання систем за допомогою мови SysML, аналізу та трансформації моделей, методи системного аналізу, а також семантичні технології для інтеграції знань та перевірки моделей.

Наукова новизна отриманих результатів

Запропоновано вдосконалену методологію специфікації та перевірки моделей систем для високо інтегрованих проєктів на основі використання SysML та MBSE підходу і досліджено методи інтеграції моделей у складних системах через процеси трансформації моделей та їх адаптації до різнорідних середовищ розробки.

Практичне значення магістерської роботи полягає в ….

Отримані результати можуть бути використані в розробці складних високо інтегрованих проєктів, зокрема у галузях будівництва, автомобільної промисловості, де критично важливо забезпечити надійність, інтеграцію та ефективну перевірку моделей систем. Використання запропонованих методів може значно підвищити якість та швидкість проєктування, зменшити ризики помилок на етапах розробки та інтеграції.

Структура магістерської роботи. Робота складається зі вступу, трьох розділів та висновків. Загальний обсяг роботи становить 84 сторінку, і містить 42 рисунки, список використаних джерел із 52 найменувань.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ПЕРЕВІРКИ МОДЕЛЕЙ У ВИСОКО ІНТЕГРОВАНИХ ПРОЄКТАХ

1.1. Особливості високо інтегрованих проєктів

Високо інтегрований проект - це такий проект, в якому різні його компоненти, системи та процеси тісно взаємодіють між собою, створюючи єдине ціле. Це означає, що всі частини проекту працюють синхронно та ефективно, досягаючи загальної мети.

Ключові характеристики високо інтегрованого проекту наступні:

- Системний підхід. Всі елементи проекту розглядаються як частина єдиної системи, а не як окремі компоненти.

- Міждисциплінарна співпраця. У проекті беруть участь фахівці з різних галузей, які тісно співпрацюють між собою.

- Єдиний інформаційний простір. Використовується єдина інформаційна система, яка забезпечує доступ до всіх даних проекту для всіх учасників.

- Автоматизація процесів. Максимально автоматизовані процеси, що знижує ризик помилок та підвищує ефективність.

- Гнучкість: Проект здатний адаптуватися до змін зовнішнього середовища.

Приклади високо інтегрованих проектів:

- Будівництво великих споруд: Всі стадії будівництва, від проектування до експлуатації, інтегровані в єдину систему.

- Розробка програмного забезпечення: Різні модулі програмного забезпечення тісно взаємодіють між собою, забезпечуючи безперебійну роботу всієї системи.

- Виробництво автомобілів: Всі етапи виробництва, від розробки дизайну до збірки готового автомобіля, інтегровані в єдиний виробничий процес.

Високо інтегровані проекти є майбутнім управління проектами. Вони дозволяють досягти більш високих результатів при менших витратах і ризиках. Однак, для успішної реалізації таких проектів необхідна висока кваліфікація учасників та використання сучасних інструментів управління проектами.

Вибір інструментів для високо інтегрованих проектів, особливо в таких галузях як будівництво, залежить від багатьох факторів, включаючи масштаб проекту, складність, бюджет та конкретні вимоги. Ось деякі категорії інструментів, які зазвичай використовуються на прикладі моделювання інформаційних моделей будівель (BIM):

- Revit: Один з найпопулярніших інструментів BIM, що дозволяє створювати детальні 3D-моделі будівель.

- ArchiCAD: Ще один потужний інструмент BIM, який широко використовується в архітектурі та будівництві.

- Tekla Structures: Спеціалізований інструмент для сталевого конструкційного будівництва.

Наведемо риклад інтегрованого рішення для будівництва:

1. BIM-модель: Створюється детальна 3D-модель будівлі в Revit.

2. База даних: Вся інформація про будівлю зберігається в базі даних PostgreSQL (MySQL, MS SQL).

3. Система управління проектами. Використовується Asana або Jira для управління завданнями і проектами.

4. Веб-додаток. Створюється веб-додаток на основі JavaScript і React для візуалізації даних з BIM-моделі і бази даних.

5. Мобільний додаток. Створюється мобільний додаток для доступу до інформації про проект з мобільних пристроїв.

Важливі фактори при виборі інструментів:

- Масштаб проекту. Для великих проектів потрібні більш потужні і масштабовані інструменти.

- Складність проекту. Чим складніший проект, тим більше функціоналу потрібно від інструментів.

- Бюджет. Вибір інструментів залежить від доступного бюджету.

- Наявні навички команди. Вибір інструментів повинен враховувати наявні навички команди.

- Інтеграція з іншими системами. Інструменти повинні легко інтегруватися з іншими системами, які використовуються в проекті.

1.2. Модель системи як первинний артефакт MBSE-підходу

Системна інженерія на основі моделі (MBSE) — це формалізована програма моделювання для підтримки процесу системної інженерії. Його використання в цьому процесі полягає в підтримці таких дій, як системні вимоги, проектування, аналіз, перевірка та валідація, які мають місце, починаючи з фази концептуального проектування, продовжуючись протягом розробки та наступних фаз життєвого циклу.

Підхід MBSE наголошує на використанні моделей для виконання системної інженерної діяльності, яка традиційно виконується з використанням підходу на основі документів. Незважаючи на численні переваги цього традиційного підходу, він має деякі фундаментальні обмеження. На жаль, доступ до повноти, узгодженості, зв’язків і взаємозв’язків між вимогами, проектом, інженерним аналізом і тестовою інформацією важко отримати, оскільки інформація розподілена по кількох документах. На відміну від цього підходу, MBSE інтегрує цю інформацію для узгодженого вирішення багатьох аспектів системи, що розробляється, замість того, щоб мати справу з розрізненою колекцією окремих моделей.

З цією метою MBSE використовує свій основний артефакт, яким є «модель системи». Ця модель формально представляє всі аспекти проблеми системної інженерії, що покращує специфікацію та якість проектування; повторне використання специфікацій системи та артефактів дизайну; і

спілкування між командами розробників. Крім того, модель може бути інтегрована з іншими моделями аналізу та проектування для представлення інших аспектів системи. Наступна частина розділу охоплює ключові аспекти моделі системи та її впровадження в процес для ефективного застосування MBSE. Будучи основним артефактом MBSE-підходу, системна модель дозволяє проектувати систему, яка визначає свої вимоги та відповідає її загальним цілям. Вона забезпечує механізм визначення та інтеграції проектів підсистем і компонентів у системну модель, а також підтримує відстеження вимог до системи та компонентів. Враховуючи зростаючу складність сучасних проектів системної інженерії, оскільки вони обробляються територіально розподіленими командами проектувальників, об'єднаними цілями багатьох зацікавлених сторін і переповненими великою кількістю інформації. Модель системи може бути використана для представлення інформаційного центру, що розглядає міждисциплінарні залежності; отже, підтримка цілісного підходу до моделювання та обмін проектною інформацією між різними точками зору. Визначення та розвиток цієї моделі ґрунтується на трьох основних постулатах:

а) мова: для представлення системи, що розробляється;

б) метод: який визначає дії та артефакти,

с) інструмент: для впровадження мови та методу моделювання (рис. 1.1).

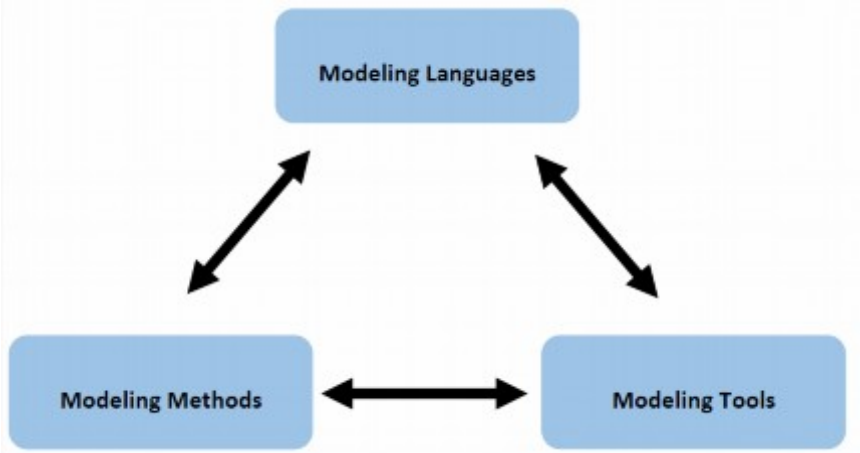


Рис. 1.1. Три постулати підходу MBSE

Мова, яка є ключовою частиною розробки моделі, базується на стандарті System Modeling Language (SysML). SysML було прийнято групою управління об’єктами (OMG) у 2006 році та підтримується провідними організаціями системної індустрії, включаючи Міжнародну раду з системної інженерії (INCOSE). SysML розширює UML, який спочатку був визначений як мова моделювання для підтримки програмного моделювання загального призначення. Він був розроблений у відповідь на недоліки UML щодо системної інженерії. Відповідно, кілька дослідників запропонували застосувати SysML для моделювання складних систем для ефективної реалізації MBSE. Наприклад, автор в [5] переносить SysML у висо інтегрований проект в області проектування будівель і розробляє метод під назвою параметричне моделювання систем (PSM) для мультидисциплінарної оптимізації проектування та співпраці при проектуванні. В [6] вказали, що впровадження SysML дозволяє відкритий підхід, що полегшує комунікацію між залученими організаціями, а отже, покращує управління проектом. Крім того, ці дослідники дійшли висновку, що SysML містить діаграми та моделі, які зменшують ймовірність неправильного спілкування, і забезпечує стандартну та комплексну парадигму для специфікації системи. В дослідженні [7] описали SysML як добре підходящу мову для визначення зв’язків високого рівня, які існують між вимогами, структурою та поведінкою.

На відміну від SysML-підходу, часто рекомендують MBSE-підхід, що включає графи RDF як мову замість SysML. У даній роботі RDF-графи використовуються для моделювання: індивідуальних проектних вимог; графіки вимог; характеристики окремих компонентів; і графіки компонентів дизайну. По суті, ці графіки разом із мовою веб-онтології (OWL) утворюють формальну логічну мову для представлення знань. На жаль, вони мають обмежене застосування в системній інженерії через відсутність послідовної графічної нотації, яку забезпечує мова SysML. Відповідно в [8] вказали на

поєднання обох способів, щоб отримати вигоду від привабливої графічної нотації SysML і формального обґрунтування RDF/OWL.

1.3. Особливості побудови алгоритмічних схем та нотацій засобами мов моделювання систем

Мова системного моделювання (SysML) — це мова візуального моделювання, яка надає повний набір діаграм і конструкцій для моделювання багатьох загальних аспектів проблем системної інженерії. По суті, цей підхід до моделювання дозволяє графічно відобразити вимоги, структуру та поведінку, щоб забезпечити повний опис системи, що розробляється, включаючи її компоненти та середовище. У той час як діаграми показують кілька видів системи для розуміння її аспектів як окремо, так і разом, опис складається з інтегрованої моделі системи для підтримки аналізу, специфікації, проектування, перевірки та валідації складних систем SysML надає дев’ять діаграм і може представляти наступні аспекти системи, компонентів та інших об’єктів:

а) структурний склад, взаємозв’язки та класифікацію;

б) поведінку на основі потоку, на основі повідомлень і на основі стану;

в) обмеження на фізичні та робочі властивості;

г) розподіл між поведінкою, структурою та обмеженнями;

д) вимоги та їхні зв’язки з іншими вимогами, елементами дизайну.

Наступні розділи охоплюють найбільш релевантні діаграми відповідно до мети дослідження, а саме: діаграма SysML і нотація в цілому, діаграма варіантів використання, діаграма вимог і діаграма визначення блоку - також добре відома як BDD.

1.3.1. Діаграма та нотація SysML

Діаграми та нотації SysML забезпечують механізм представлення різних поглядів на модель для конкретної мети. Хоча ці діаграми та

позначення базуються на діаграмах і позначеннях UML, вони не включають фундаментальні аспекти UML через те, що він не відповідає вимогам до систем моделювання. З цією метою SysML включив модифікації інших діаграм UML, таких як діаграма класів, складена діаграма та діаграма діяльності, а також додатково додав дві нові діаграми для вимог і параметрів. Зрештою це призвело до створення таксономії діаграми SysML, як показано на рисунку 1.2.

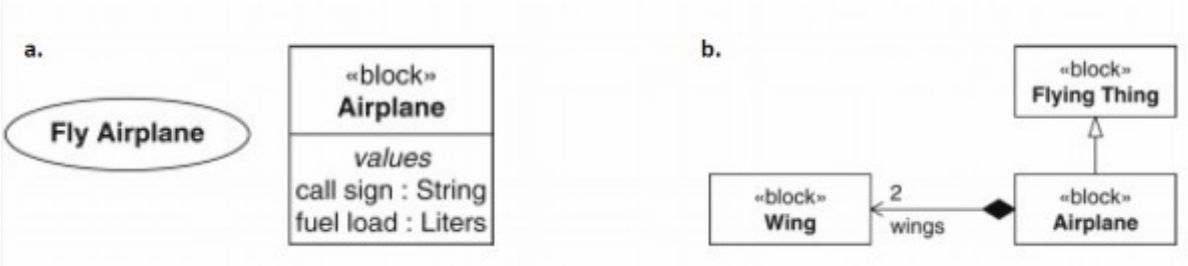


Рис. 1.2. Вузли a) та шляхи, що з’єднують вузли b)

На додаток до цієї таксономії діаграм SysML також підтримує табличні, матричні та деревоподібні представлення для моделі. Щодо зрозумілості та реалізації діаграм і нотацій для систем моделювання, кожна діаграма, показана в таксономії, повинна мати «фрейм діаграми», який охоплює вміст діаграми. Фрейм діаграми відповідає елементу моделі, який забезпечує контекст для вмісту діаграми. Структура кадру є прямокутником із заголовком діаграми, що містить інформацію у верхньому лівому куті кадру. Область вмісту включає елементи діаграми, які представляють цікаву модель. Що стосується діаграм SysML, то вони можуть складатися з вузлів і шляхів, як показано на рисунках 1.2 a) і 1.3 b). Вузли — це елементи діаграми, які зазвичай виглядають як фігури, наприклад прямокутники, овали чи інші багатокутники з текстовими мітками. Крім того, вузли можуть містити текстові рядки та/або графічні символи, які можуть відповідати іншим елементам моделі (рис. 1.2 a). Щодо шляхів, також відомих як ребра, це елементи діаграми, які зазвичай виглядають як лінії з додатковими прикрасами, такими як наконечники стрілок і текстові рядки. Лінії

реалізовані в різному стилі та мають кілька кінців залежно від концепції моделювання, яку вони представляють (рис. 2.2 b).

1.3.2. Діаграма варіантів використання

Діаграма варіантів використання описує зв'язки між системою, що розглядається, з її варіантами використання та акторами. Система представляє предмет, що розробляється, і надає своїм користувачам кілька функціональних можливостей. Ці функції відображаються та моделюються за допомогою варіантів використання, щоб відобразити, як користувачі використовують систему для досягнення своїх цілей. У цій ситуації варіант використання може охоплювати один або кілька сценаріїв, які відповідають тому, як система взаємодіє зі своїми користувачами за різних обставин. Користувачі описуються акторами, які представляють роль людини, організації або будь-яких зовнішніх систем, які беруть участь у використанні системи. Актори можуть взаємодіяти безпосередньо з системою або опосередковано через інших акторів.

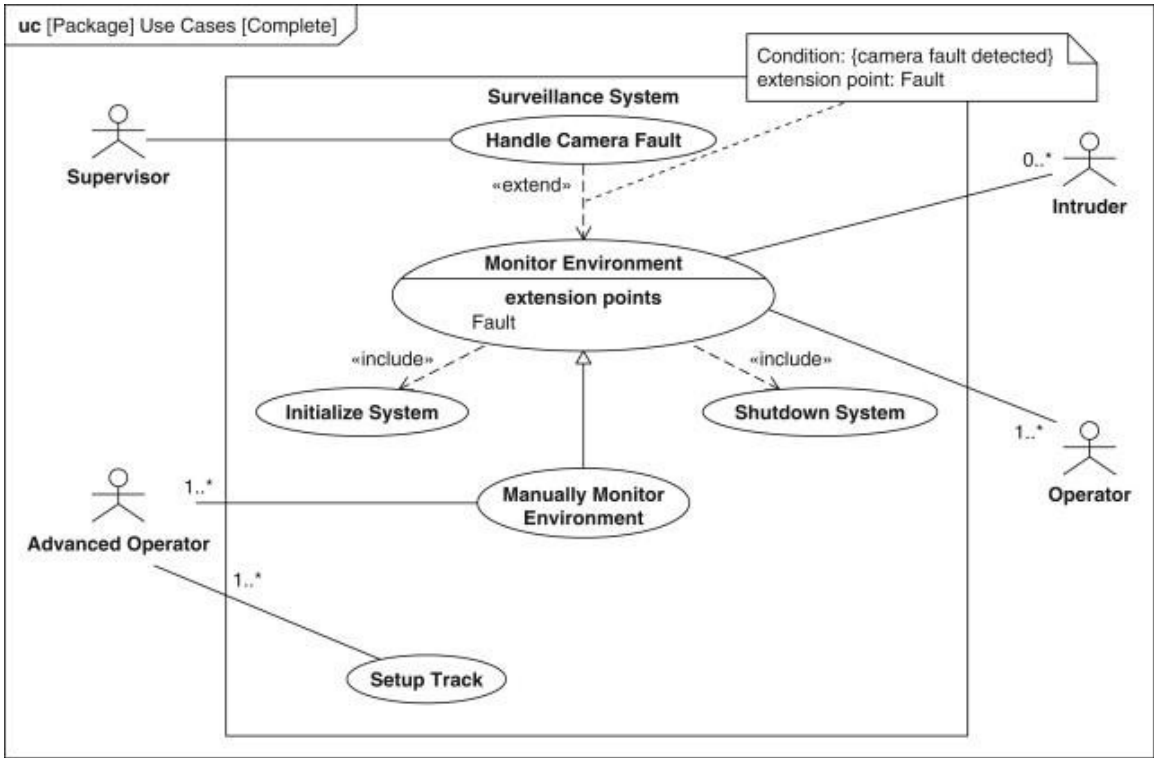


Рис. 1.3. Діаграма випадків використання системи спостереження

На діаграмі варіантів використання, показаній на рисунку 2.3, система представлена блоком із назвою, розташованим у верхній частині. У цьому випадку назва системи – система спостереження. Варіанти використання показані овалами з назвами варіантів використання всередині них. Актори зображені або як фігурки з іменем актора під ними, або як прямокутники, що містять ім’я актора під ключовим словом <<актор>>, залежно від використовуваного інструменту чи методу. Що стосується взаємодії між системою та акторами, це визначається асоціацією між акторами та варіантами використання за допомогою стандартної нотації асоціацій. Ця асоціація також включає нотацію множинності (0..* або 1..*), що описує кількість випадків використання, до яких актор або актори можуть бути залучені в будь-який момент часу.

Крім того, відносини, які зазвичай використовуються, варіанти використання також можуть бути пов’язані з іншими випадками за допомогою таких відносин: класифікація, включення та розширення. Класифікація варіантів використання може бути застосована за допомогою стандартних зв’язків узагальнення SysML. На рисунку 2.3 показано приклад того, як сценарій використання «Середовище моніторингу» далі класифікується як у варіантах використання «Середовище моніторингу вручну» та «Автоматичний моніторинг середовища». Включення або розширення для випадків використання можна застосувати за допомогою пунктирних ліній із відкритою стрілкою на включеному та розширеному кінцях відповідно.

У той час як зв’язки включення дозволяють одному варіанту використання, який називається базовим варіантом використання, включати функції іншого варіанту використання; відносини розширення дозволяють розширювати функціональні можливості, які не вважаються частиною функціональних можливостей базового варіанту використання. Крім того, діаграми взаємодії, діяльності та/або автоматів стану можна використовувати для моделювання детальної інформації про випадки використання.

Наведена діаграма (рис. 1.3) випадків використання (use case diagram) описує функціональність системи спостереження з погляду різних користувачів (акторів). Вона відображає основні дії, які користувачі можуть виконувати з системою, та взаємозв'язки між цими діями.

Актори:

- Supervisor (Супервізор): Має найвищий рівень доступу, може виконувати всі дії.

- Operator (Оператор): Має менший рівень доступу, ніж супервізор. Може виконувати більшість дій, але не всі.

- Advanced Operator (Досвідчений оператор): Може виконувати додаткові дії, такі як налаштування треків (Setup Track).

- Intruder (Злочинець): Не є легітимним користувачем системи, але його присутність впливає на роботу системи.

Випадки використання:

- Monitor Environment (Моніторити середовище): Основна функція системи, яка передбачає постійний контроль за об'єктом спостереження.

- Initialize System (Ініціалізувати систему): Підготовка системи до роботи.

- Shutdown System (Вимкнути систему): Закінчення роботи системи.

- Manually Monitor Environment (Ручний моніторинг середовища): Додаткова функція, яка дозволяє оператору вручну керувати камерою або іншими пристроями системи.

Setup Track (Налаштувати трек): Додаткова функція для досвідченого оператора, яка дозволяє налаштувати автоматичне відстеження об'єктів.

Handle Camera Fault (Обробити несправність камери): Реакція системи на виявлення несправності камери.

Інтерпретація діаграми:

- Система спостереження призначена для моніторингу об'єкта.

- Супервізор має повний контроль над системою.

- Оператор виконує основні завдання з моніторингу.

- Досвідчений оператор може виконувати додаткові налаштування.

- Система реагує на несправності обладнання.

1.3.3. Діаграма вимог

Діаграма вимог графічно зображує ієрархію вимог, що визначають специфікацію системи або компонента. Цей спосіб моделювання вимог значно покращує керування вимогами протягом усього життєвого циклу системи, забезпечуючи точне відстеження між текстовими вимогами та елементами моделі, які представляють дизайн системи, аналіз, реалізацію та тестові випадки. Захоплення вимоги в SysML представлено стереотипом «вимоги», що включає відсіки для імені, ідентифікатора та текстового рядка, що описує текстові вимоги (рис. 1.4). Крім того, вимоги можна налаштувати шляхом додавання таких властивостей, як метод перевірки, статус перевірки, критичність, ризик і категорія вимог.

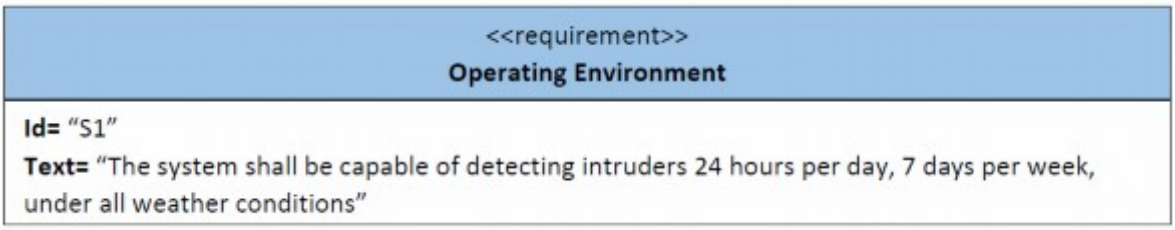


Рис. 1.4. Приклад вимог SysML

Після того, як вимоги зафіксовані, вимога SysML забезпечує зв’язки, які можна використовувати для визначення структури вимог, де проявляється взаємодія між вимогами та іншими вимогами та елементами моделі. Розрізняють такі відносини: задоволення, перевірка, уточнення, отримання, копіювання, відстеження та обмеження. У той час як зв’язки отримання та копіювання можуть лише пов’язувати одну вимогу з іншою, зв’язки задовольняють, перевіряють, уточнюють і відстежують зв’язок вимог з іншими елементами моделі. Крім того, їх реалізація може бути зображена за допомогою таких нотацій: пряма нотація, нотація компартменту, нотація

виноски та обґрунтування. Пряма нотація зображує пряме відношення, включаючи пунктирну стрілку з назвою відношення, що відображається як ключове слово, наприклад, «satisfy», <<verify>>, <<refine>>, <<deriveReqt>>, <<copy> > і <<trace>>. На відміну від цієї нотації, нотацію компартмента можна використовувати шляхом включення компартмента в блок вимог як показано на рисунку 1.5.

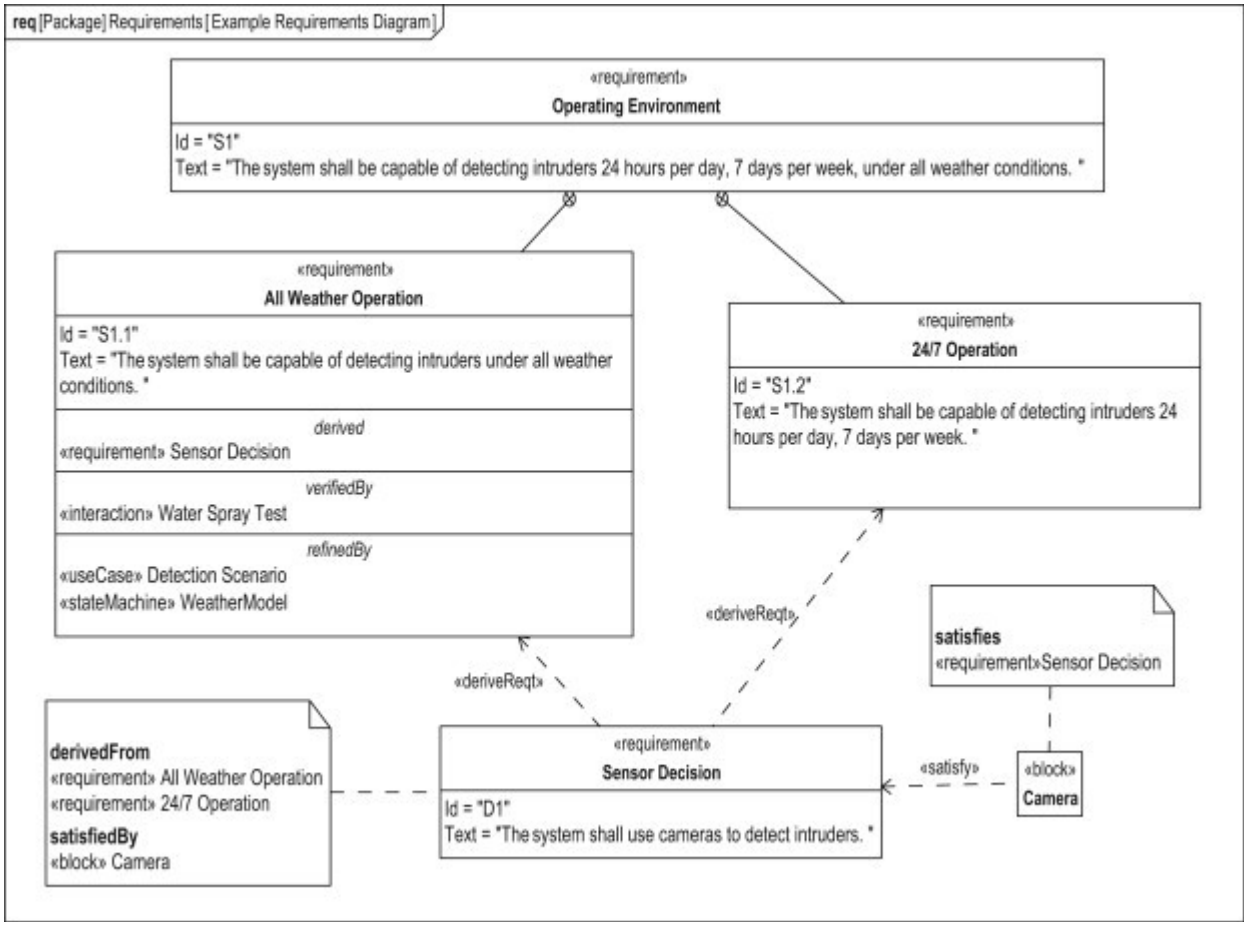


Рис. 1.5. Приклад діаграми вимог SysML

Зосереджуючись на цій діаграмі, можна побачити, що вимога «All Weather Operation» включає три додаткові відсіки, що описують наступні відносини: derived, verifiedBy та refineBy. З іншого боку, виноска зображується як символ примітки, графічно пов’язаний з елементом моделі. Символ виноски посилається на модель на іншому кінці зв’язків. Нарешті, обґрунтування виражається за допомогою символу примітки з ключовим

словом «обґрунтування». Текст у символі примітки може або безпосередньо надавати обґрунтування, або посилатися на зовнішній документ іншої частини моделі.

1.3.4. Діаграма визначення блоків (модулів)

Діаграму визначення блоків можна використовувати для моделювання та представлення структури систем з точки зору їх функцій, ієрархії та взаємозв'язку. Для опису цих структур діаграма використовує «блоки» як фундаментальну модульну одиницю. На основі блоків можна визначити та представити кілька систем, компонентів, взаємозв'язків компонентів або елементів, які проходять через систему. Крім того, можуть бути створені зовнішні та концептуальні сутності або логічні абстракції. Блок характеризується як прямокутник, який сегментований на ряд відділень. Для визначення блоку можна використовувати функції блоку. У верхній частині символу блоку з'являється відсік імені, який є єдиним обов'язковим відсіком. За бажанням можна використати необов'язкове ключове слово <<блок>> перед розділом імені. Функції блоку, як частини, операції, властивості значень і порти, можуть бути представлені в інших відсіках символу блоку. Ці відсіки мають ярлики, які вказують на те, яку функцію вони містять. Ці мітки зображуються курсивом у нижньому регістрі, мають множину та містять пробіл між словами (рис. 1.6).

Рис. 1.6. Блок зі своїми обов'язковими та блоковими характеристиками

Структурні особливості можна розглядати як властивості, що визначають характеристики блоку. Визначають наступні категорії властивостей: властивості частин, властивості посилань, властивості значень, обмеження, порти та потоки.

Властивості частин описують зв’язки композиції між блоками. Цей зв’язок також називають зв’язком «ціле-частина». Частина описує екземпляр або екземпляри блоку в контексті його складеного блоку. Потенційна кількість екземплярів визначається кратністю частини, яка визначається як: (а) нижня межа (мінімальна кількість екземплярів) і (б) верхня межа (максимальна кількість екземплярів). На певній блочній схемі деталь може бути показана або у відсіку деталей, або як складена асоціація. Частина є властивістю блоку, і як така може бути перерахована в окремому відділенні деталей у блоці (рис. 1.6). Відсік деталей позначено ключовим словом parts і містить один запис для кожної частини в блоці. Композитна асоціація показана у вигляді лінії між двома блоками з ромбовидною прикрасою блоку на всьому кінці та відкритим наконечником стріли на кінці частини.

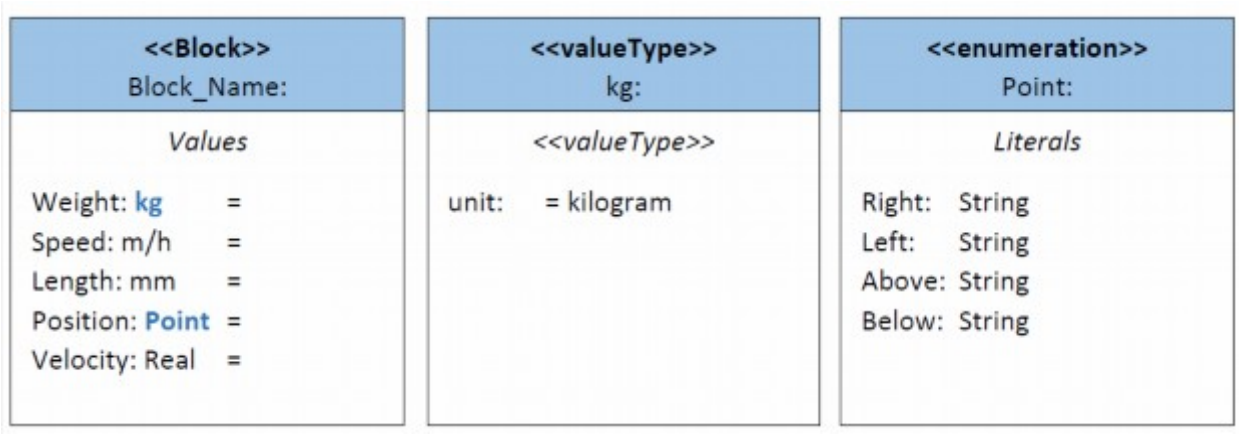
Властивості значень використовуються для моделювання кількісних характеристик блоку, таких як його вага, швидкість, положення або швидкість. Вони можуть мати множинність і відображаються у відсіку свого блоку власності, подібно до інших властивостей. Відсік значень містить значення етикетки (рис 1.7 a). Крім того, вони засновані на типі значення, яке визначає діапазон дійсних значень, які властивість може приймати під час опису екземпляра блоку-власника. Типи значень використовуються для опису значень кількостей. У цьому випадку такі властивості значень, як загальна вага та вага компонента, можуть бути введені за допомогою типу значення, що називається кілограмами (кг), яке може бути будь-яким дійсним числом, більшим або рівним 0 (рис. 1.7 b). Тип значення може бути описано структурою даних для представлення кількості та вказує його допустимий набір значень.

Визначено такі категорії типу значень:

а) примітивний тип, який підтримує скалярні значення визначення, такі як Integer, String, Boolean та Real;

б) перерахування, визначає набір іменованих значень, які називаються літералами, такими як кольори або дні тижня;

в) структурований тип, який представляє специфікацію структури даних, яка включає більше одного елемента даних, кожен з яких представлений властивістю значення.



а) б) в)

Рис. 1.7. Блок, що включає властивості значення а); тип значення б); перерахування с)

На діаграмі визначення блоку тип значення представлено символом рамки з суцільною межею. Відділення імені містить ключове слово <<valueType>>, яке передує його імені (рис. 2.7 б). Символ перерахування має один відсік, позначений літералами, у якому перераховано всі літерали переліку та ключове слово «enumeration» перед його назвою у відсіку імені (рис. 2.7 в). Символ структурованого типу має один відсік, позначений значеннями, які перераховують вкладені властивості значень типу значення, використовуючи ту саму нотацію відсіку, що й для інших властивостей значень.

Обмеження можна використовувати для обмеження властивостей блоку. Вони являють собою математичне співвідношення (рівняння або

нерівність), яке накладається на набір властивостей значення. Обмеження можна показати в спеціальному відсіку, позначеному обмеженнями. Обмеження також може бути показано як символ примітки, доданий до елементів моделі, який воно обмежує, з текстом обмеження, показаним у тілі примітки.

Взаємодії між різними частинами системи можуть бути визначені через порти та потоки, які забезпечують абстрактне уявлення про взаємодію. Порти являють собою точку доступу на межі блоку та на межі будь-якої частини або посилання, введеного цим блоком. SysML розрізняє два типи портів, які називаються повними портами (<<full>>) і проксі (<<proxy>>). Тоді як повний порт еквівалентний частині на межі батьківського блоку, яка стає доступною як точка доступу до блоку та з нього. Проксі-порт не є частиною свого батьківського блоку, а натомість забезпечує зовнішній доступ до функцій свого батьківського блоку або частин.

Висновки до розділу

У даному розділі проведено дослідження предметної області перевірки моделей у високо інтегрованих проєктах. Високо інтегровані проєкти характеризуються складністю систем, які включають безліч компонентів, взаємопов'язаних між собою, що ускладнює процес їх перевірки та тестування. Особлива увага приділяється важливості інтеграції на рівні компонентів та системного підходу для забезпечення надійності й відповідності системи технічним вимогам. Використання моделей систем у MBSE (Model-Based Systems Engineering) дозволяє розглядати систему як первинний артефакт, що забезпечує формалізацію процесу розробки та перевірки проєктів. Модель системи є важливим елементом, який спрощує управління складними проєктами та дозволяє узгоджувати всі етапи життєвого циклу системи.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ КОНЦЕПЦІЙ ТА ПОНЯТЬ СИСТЕМНОЇ ІНЖЕНЕРІЇ НА ОСНОВІ МОДЕЛЕЙ

2.1. Дослідження доменно-специфічної мови за допомогою SysML

SysML можна налаштувати для визначення доменно-орієнтованої семантики за допомогою доменно-орієнтованої мови (DSL). У цьому випадку кілька DSL можна визначити в SysML і, отже, створити міст до багатьох інших доменів, за допомогою чого можна підвищити продуктивність програміста та покращити ефективне спілкування між експертами домену. Визначаючи ці DSL, кілька дослідників продемонстрували свої переваги розширюваним конструкціям SysML, таким як «профілі». Наприклад, в дослідженні [12] розглядали стереотипи SysML, щоб абстрагувати та охопити спеціалізовану семантику конкретної області застосування, щоб дозволити інтегрувати моделі та симуляції безперервної динаміки в SysML. Стереотипи використовуються для розширення конструкцій SysML, щоб забезпечити хороший баланс між перетворенням деякої неявної семантики Modelica у семантику SysML. Крім того, в [13] також вказали на свої переваги щодо профілів SysML і, отже, метамоделей для визначення предметно-специфічного опису різних дисциплін і мов, для створення та інтеграції кількох представлень вбудованих систем у SysML. У їх дослідженні формальне визначення доменів, задіяних у системі, визначено через мета-моделі, а профілі SysML були створені для включення DSL. У цьому підході загальні компоненти системи, такі як вимоги, гідравлічна система та система керування, визначаються за допомогою профілів SysML та відповідно інтегруються з Microsoft Excel, MS Visio та MagicDraw.

2.1.1. Профіль SysML

Профіль SysML надає конструкції, які розширюють і додають нові можливості самої мови моделювання. Це особливий вид пакету, що містить

набір «стереотипів» і допоміжні визначення. Його використання служить для визначення набору концепцій, які підтримують нову область, або набору концепцій, які додають нову інформацію до моделі в області, яка вже підтримується. Більш складні профілі можуть містити або підпрофілі, або підпакети, які додатково поділяють загальний домен на підмножини пов'язаних концепцій домену. Його зображення можна показати на пакетній діаграмі з «Профілем» як елементом моделі, який відповідає фрейму діаграми. На рисунках 2.1 і 2.2 показано два створені профілі, що стосуються предметної області, включаючи стереотипи, що визначають її концепції.

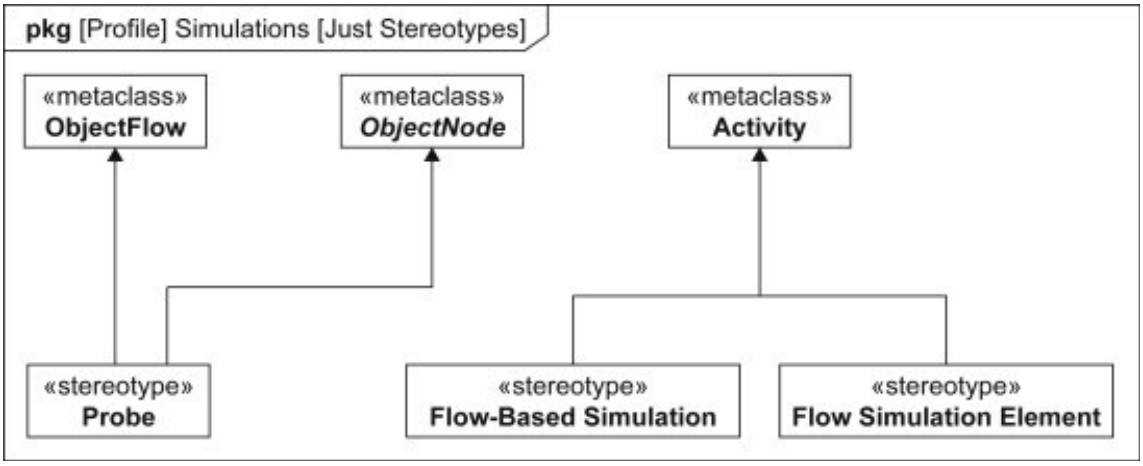


Рис. 2.1. Профіль SysML для моделювання на основі потоку

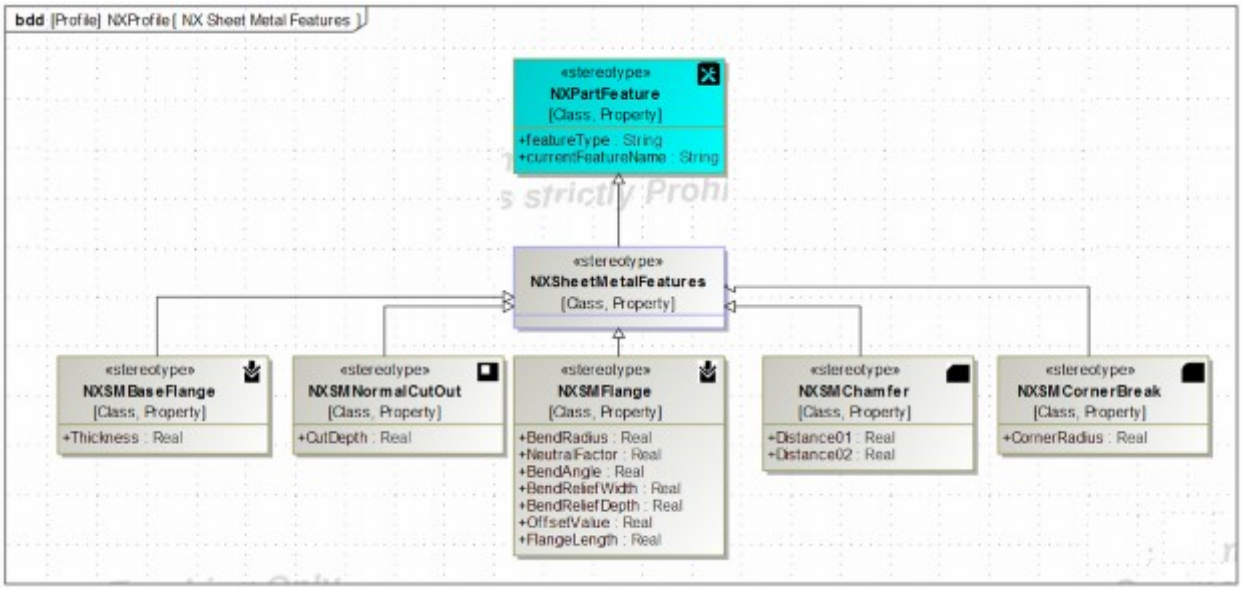


Рис. 2.2. Профіль SysML для прикладного розширення

З іншого боку, стереотипи використовуються для створення профілів і нових елементів моделі з існуючих, включаючи детальні атрибути, придатні для предметно-спеціальних програм. Наприклад, один або більше мета-класів у еталонній метамоделі можна розширити за допомогою стереотипів (рис. 2.1), або існуючий стереотип можна спеціалізувати на інших стереотипах (рис. 2.2). Стереотип представлений прямокутником із ключовим словом «стереотип» у центрі вгорі, після якого йде назва стереотипу. Зв’язок розширення зображений у вигляді лінії із зафарбованим трикутником на кінці мета-класу, а зв’язок узагальнення зображений у вигляді лінії з порожнистим трикутником на загальному кінці.

2.1.2. Перспективи в SysML

Моделі SysML містять вичерпну описову інформацію про систему, що розробляється, та її середовище. Цю інформацію можна представити у вигляді схем і таблиць. Через значний обсяг інформації, яку можна включити, важливо мати можливість налаштувати подання цієї інформації для підтримки різноманітного набору споживачів. У цьому випадку SysML надає точки зору, які можуть уточнювати як використання процесу для створення артефактів, так і те, як артефакти мають бути представлені зацікавленим сторонам. Ця інформація може включати малюнки, таблиці, графіки, цілі документи, слайди презентації або відео.

2.2. Інтеграція SysML у середовище розробки систем за допомогою трансформації моделі

Модель SysML може представляти кілька аспектів системи, що розробляється, на досить абстрактному рівні. По суті, вона визначає систему та її компоненти аж до певного рівня системної ієрархії. Таким чином, його можна використовувати з іншими моделями, які представляють більш детальні аспекти системи або можуть представляти інші аспекти системи, які

не розглядаються моделлю SysML. Одночасно керування цією великою кількістю моделей різними мовами, розроблених різними зацікавленими сторонами, може створити проблеми, включаючи повідомлення про неоднозначність і непослідовність інформації. Тому інтеграція моделі SysML з іншими моделями та структурованими даними, розробленими іншими інженерними дисциплінами, є дуже важливою та необхідною для забезпечення узгоджених рішень на основі моделі. З цієї причини зв'язок між даними в різних моделях має підтримуватися, щоб зменшити невідповідності та покращити цілісність дизайну та якість.

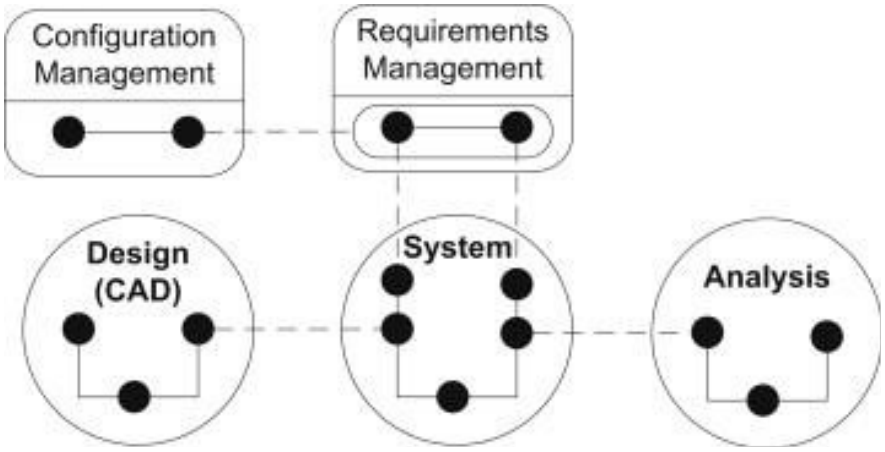


Рис. 2.3. Різноманітність сховищ і моделей, включаючи інтеграцію між ними

Відповідно, цього можна досягти шляхом забезпечення інтерфейсу даних між інструментом моделювання системи та іншими середовищами розробки систем (SDE) (рис. 2.3). Як правило, це з'єднання можна створити за допомогою механізмів обміну даними. Вони узагальнили такі механізми:

а) ручний обмін, який передбачає повторне введення даних з одного інструменту в інший;

б) обмін файлами з використанням власного формату файлів або стандартного формату обміну (наприклад, XMI);

в) обмін на основі взаємодії з використанням прикладного програмного інтерфейсу (API);) перетворення моделі (рис. 2.4).

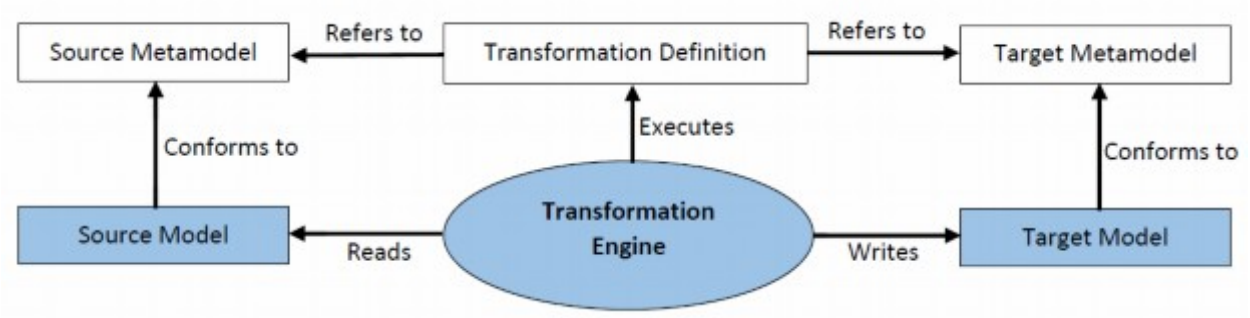


Рис. 2.4. Основні поняття трансформації моделі

Розглядаючи перші два механізми, зазначено, що в таких випадках, коли моделі не можуть бути повністю перетворені за допомогою стандартів обміну, підтримка узгодженості зазвичай залишається на виборах користувачів, що призводить до значних зусиль без додаткової вартості та потенційної помилки. Щодо третього, то висловили занепокоєння щодо відсутності стандарту для обміну на основі API, який відображає використання додатків «точка-точка» для полегшення обміну, оскільки кожен інструмент має власний API. На відміну від цих механізмів, трансформація моделі дуже підходить, коли дві сторони обміну є різними і трансформація необхідна для підтримки обміну даними. Вони заявили, що сфера трансформації моделі стає все більш важливою як стандарт, оскільки підходи, засновані на моделях, і предметно-орієнтовані мови стають все більш поширеними. Відповідно, кілька дослідників включили підходи до трансформації моделі у свої дослідження для інтеграції особливо SysML з іншими системними середовищами.

Наприклад, в [15] використовували перетворення моделі для інтеграції SysML і Modelica. У цьому прикладі перетворення моделі наближається до граматики потрійних графів (TGG), а правила перетворення графів реалізовано для підтримки двонаправленого відображення між конструкціями SysML і відповідними моделями Modelica. Тут застосували трансформацію моделі у своєму дослідженні для генерації предметно-спеціальних представлень у SysML за допомогою доменних інструментів EPLAN і Modelica. На відміну від попереднього підходу, діаграми сюжетів

тут використовуються для візуального визначення правил перетворення між різними видами. Ці правила можна порівняти з правилами, що використовуються в підході TGG відповідно до них. Однак для двонаправленого перетворення потрібні були додаткові правила в сюжетних діаграмах у кожному напрямку перетворення. В іншому випадку, в [21] використали трансформацію моделі для інтеграції SysML і мови веб-онтології (OWL). У цьому підході мова Query/View/Transformation (QVT) реалізована для перетворення профільованих моделей SysML в онтології OWL і навпаки. В [22] використано трансформацію моделі для інтеграції SysML і інструменту Siemens NX. У цьому підході створюється та впроваджується інноваційна методологія перетворення від моделі до моделі, яка програмно об’єднує дві різні структури даних шляхом відтворення метамоделі програми САПР через представлення на основі графів у SysML.

2.3. Особливості інформаційного моделювання у високо інтегрованих проектах

Цифровізація трансформувала широкий спектр галузей промисловості, що призвело до надзвичайного зростання продуктивності, якості та різноманітності продукції. Для порівняння, автомобільна промисловість зазнала переходу до оцифрованої розробки продукції та виробництва на основі моделей, що дозволило їй досягти значного підвищення ефективності.

Довгий час будівельна промисловість стикалася з низькою продуктивністю, відсталістю процесів. У відповідь на ці проблеми галузь сьогодні стикається з технологічним і цифровим переходом через прийняття інформаційного моделювання будівель (BIM - Building Information Modeling) у процесі розробки проекту.

Використання цифрових інструментів та інформації в промисловості по всьому ланцюжку процесів значно відстає від інших областей. Наприклад, надто часто цінна інформація переважно передається у формі креслень або у

вигляді фізичних друкованих зображень на папері, або в цифровому, але обмеженому форматі. З цієї причини більш відповідним підходом є впровадження нової технології та супутніх змін у процесах. Відповідно, вони проілюстрували модель зрілості BIM, розроблену BIM Task Group, яка визначає чотири дискретні рівні впровадження BIM (рис. 2.5). Відповідно до них визначаються такі рівні.

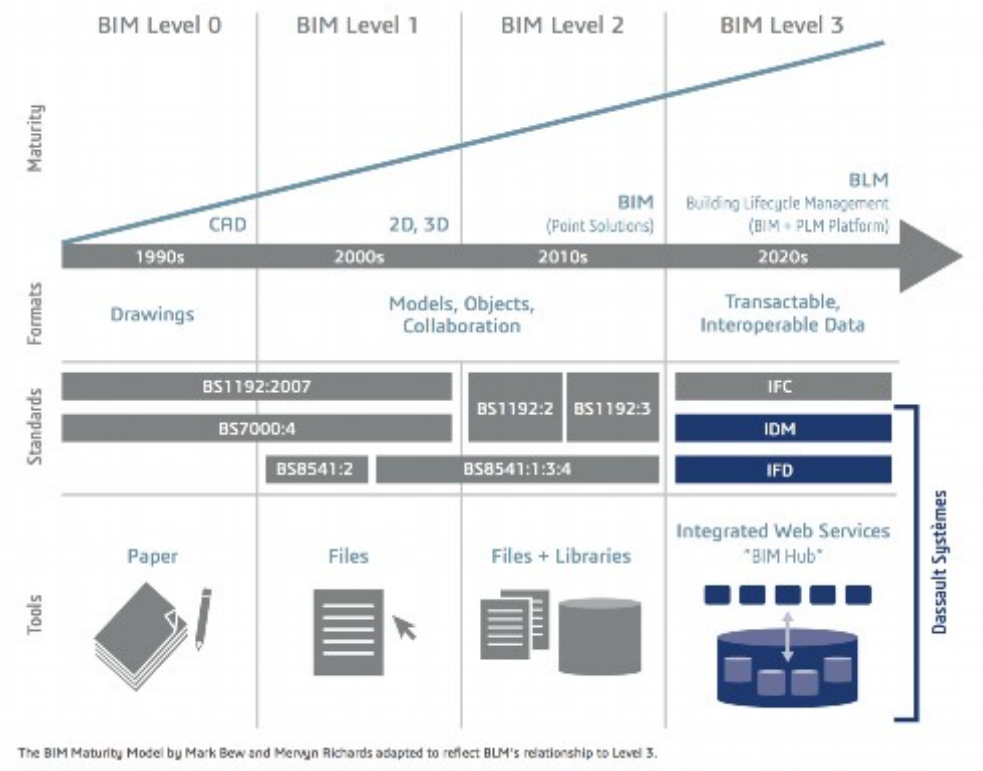


Рис. 2.5. Моделі зрілості Building Information Modeling

• Рівень 0 описує звичайну робочу практику на основі 2D CAD, за якої обмін інформацією відбувається на паперових кресленнях;

• Рівень 1 включає часткове 3D-моделювання об’єкта (здебільшого складні геометрії), тоді як більша частина проекту все ще реалізується за допомогою 2D-креслень. Обмін інформацією здійснюється через окремі файли, а центральна платформа проекту не використовується;

• Рівень 2 визначається використанням програмного продукту BIM для створення цифрових моделей, за допомогою якого кожна з різних дисциплін, залучених до процесу, розробляє власну модель. Після цього кожну модель дисципліни об’єднують і перевіряють на наявність конфліктів чи інших

розбіжностей. Взаємна узгодженість досягається протягом процесу шляхом періодичних координаційних сесій або зустрічей. На цьому рівні все ще циркулюють 2D-креслення, отримані з моделей. Обмін інформацією все ще здійснюється на основі файлів (власні формати) або відкритих стандартів (не попит на цьому рівні), і керується на центральній платформі, яка називається загальним середовищем даних (CDE);

• Рівень 3, націлений на майбутнє, базується на концепції повністю інтегрованого BIM. Реалізація базується на BIG Open BIM, тобто для обміну даними та опису процесів використовуються стандарти ISO, а глибоко інтегровані цифрові моделі використовуються протягом усього життєвого циклу. Щоб підтримувати дані проекту безперервно та послідовно протягом життєвого циклу будівель, використовуються хмарні сервіси.

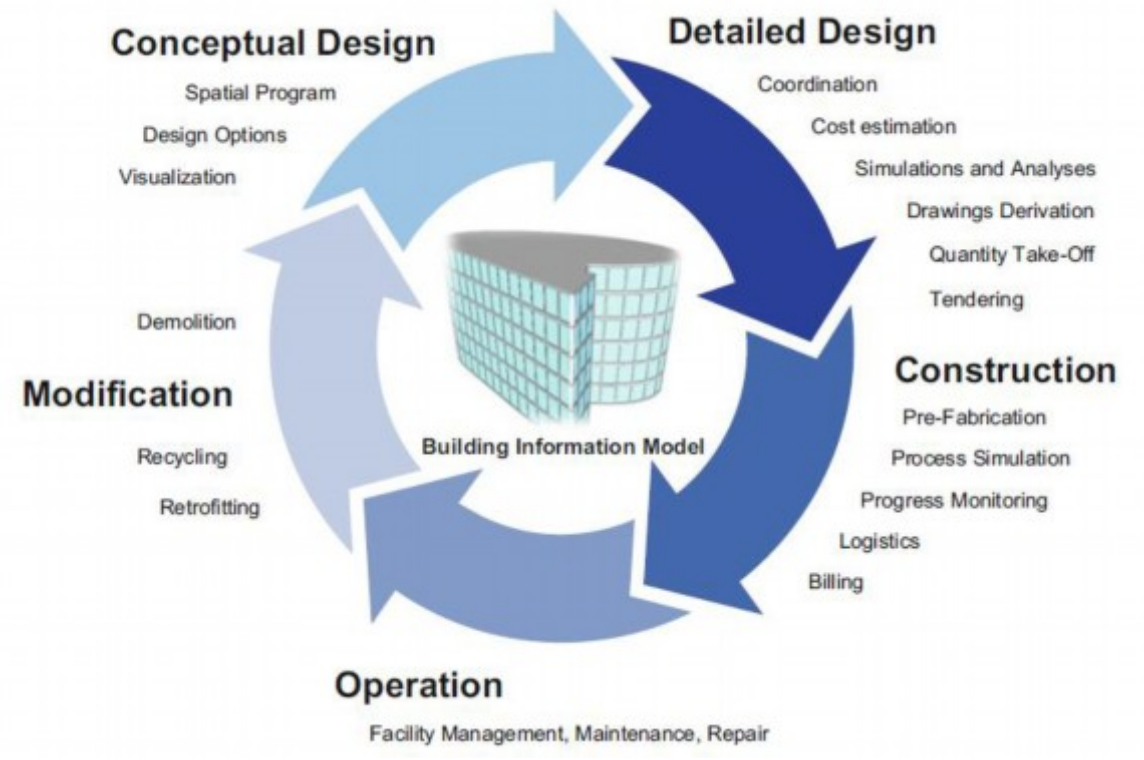


Рис. 2.6. Комплексний підхід BIM, що охоплює весь життєвий цикл

Інформаційне моделювання будівель (BIM) можна визначити як процес створення та використання цифрових моделей для проектування, будівництва та/або експлуатації проектів.

BIM також можна визначити в трьох вимірах:

а) як продукт (інформаційна модель будівлі), який є структурованим набором даних, що описує будівлю;

б) як процес (інформаційне моделювання будівлі), який є актом створення інформаційної моделі будівлі;

с) як система (управління інформацією про будівлі), яка включає бізнес-робочу та комунікаційну структуру, що підвищує якість та ефективність протягом життєвого циклу процесу будівництва. Рисунок 2.13 ілюструє комплексний підхід BIM, що охоплює весь життєвий цикл.

BIM можна розглядати як процес, як цифровий і управлінський інструмент, який використовується для покращення візуалізації проекту, потоку інформації, об'єднання системи, планування, калькуляції витрат і взагалі будь-якого прогнозування, яке стосується цілей проекту.

Впровадження BIM як продукту здійснюється за допомогою використання тривимірної геометрії та об'єктно-орієнтованої CAD-моделі. Ця модель має повний набір інтегрованих додаткових послуг, розроблених для різних функцій проекту, включаючи аналіз будівлі, структурну інженерію, кількісну оцінку, аналіз витрат, аналіз придатності до будівництва, 4D-планування та управління об'єктами (рис. 2.7). Крім того, це тривимірна модель, яку можна охарактеризувати як Building Information Model — це інтелектуальне, багате даними, за своєю суттю параметричне та об'єктне представлення об'єкта, що проектується та будується.

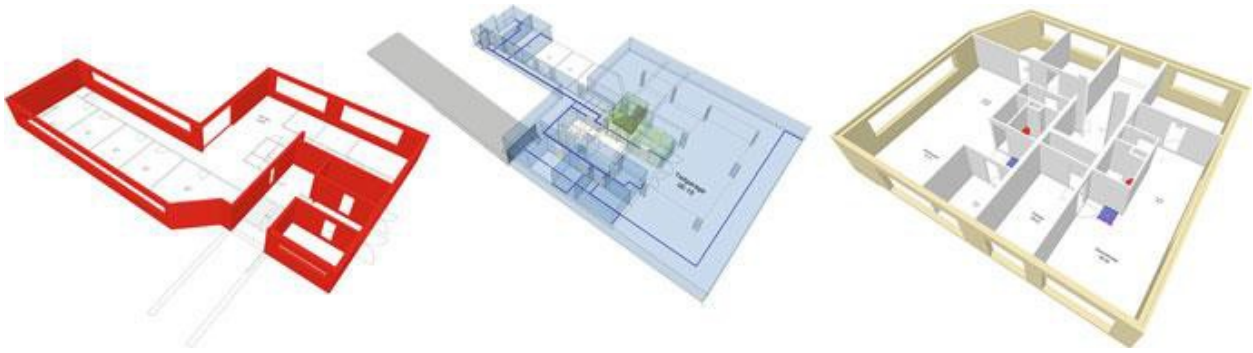


Рис. 2.7. Моделі BIM для різних функцій проекту

2.4. Міжопераційність та концепція побудови великої відкритої моделі

Як показано на рисунку 2.8 а, індустрія будівництва характеризується дуже фрагментованим процесом із численними різними та незалежними учасниками, які використовують широкий спектр програмних засобів для виконання своїх завдань. На жаль, більшість із них використовують пропрієтарні формати для обміну інформацією з інструментами, які все ще є мало автоматизованими. Це означає, що ці інструменти мають лише обмежену підтримку обміну даними між окремими програмами, що ускладнює взаємодію інтегрованої інформації про будівлі в галузі. Отже, дані та інформацію, які вже існують у цифровій формі, необхідно повторно вводити вручну, що є трудомістким і схильним до затримок, нових помилок і витрат через надлишкове введення даних, зайві IT-системи та IT-персонал, а також неефективний бізнес процеси. У цій ситуації сумісність або стандарт сумісності між інформаційними системами пропонує потенціал для значної економії та фінансової вигоди. Міжопераційність або інтероперабельність можна розглядати як здатність передавати дані між програмами, а також можливість кількох програм спільно виконувати поточну роботу (рис. 2.8 б). Цей підхід усуває необхідність вручну копіювати дані, вже згенеровані в іншій програмі, що призводить до обміну даними без втрат між програмними продуктами різних постачальників.

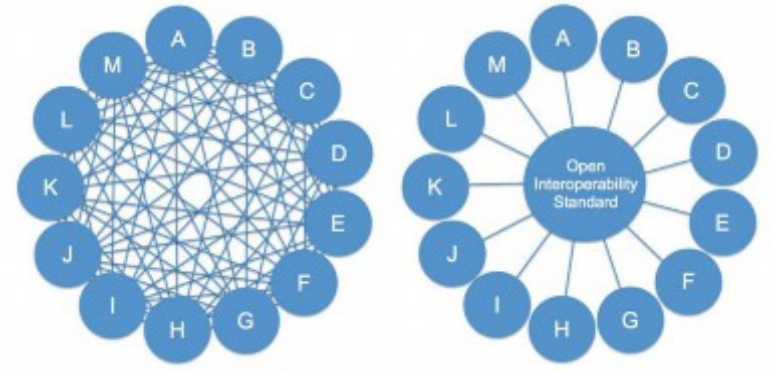


Рис. 2.8. Обмін інформацією між учасниками проекту (а); відкритий стандарт інтероперабельності для обміну даними між учасниками проекту (б)

2.4.1. Чоритьохрівнева архітектура основних класів галузі

На рисунку 2.9 показана чотирирівнева архітектура основних класів галузі (IFC - Industry Foundation Classes). Структура, яка є водночас великою та складною, складається з кількох шарів, щоб покращити її ремонтопридатність та розширюваність. Кожен рівень включає ряд класів (сутностей), які призначені для визначення або посилання на інші рівні для певних цілей.

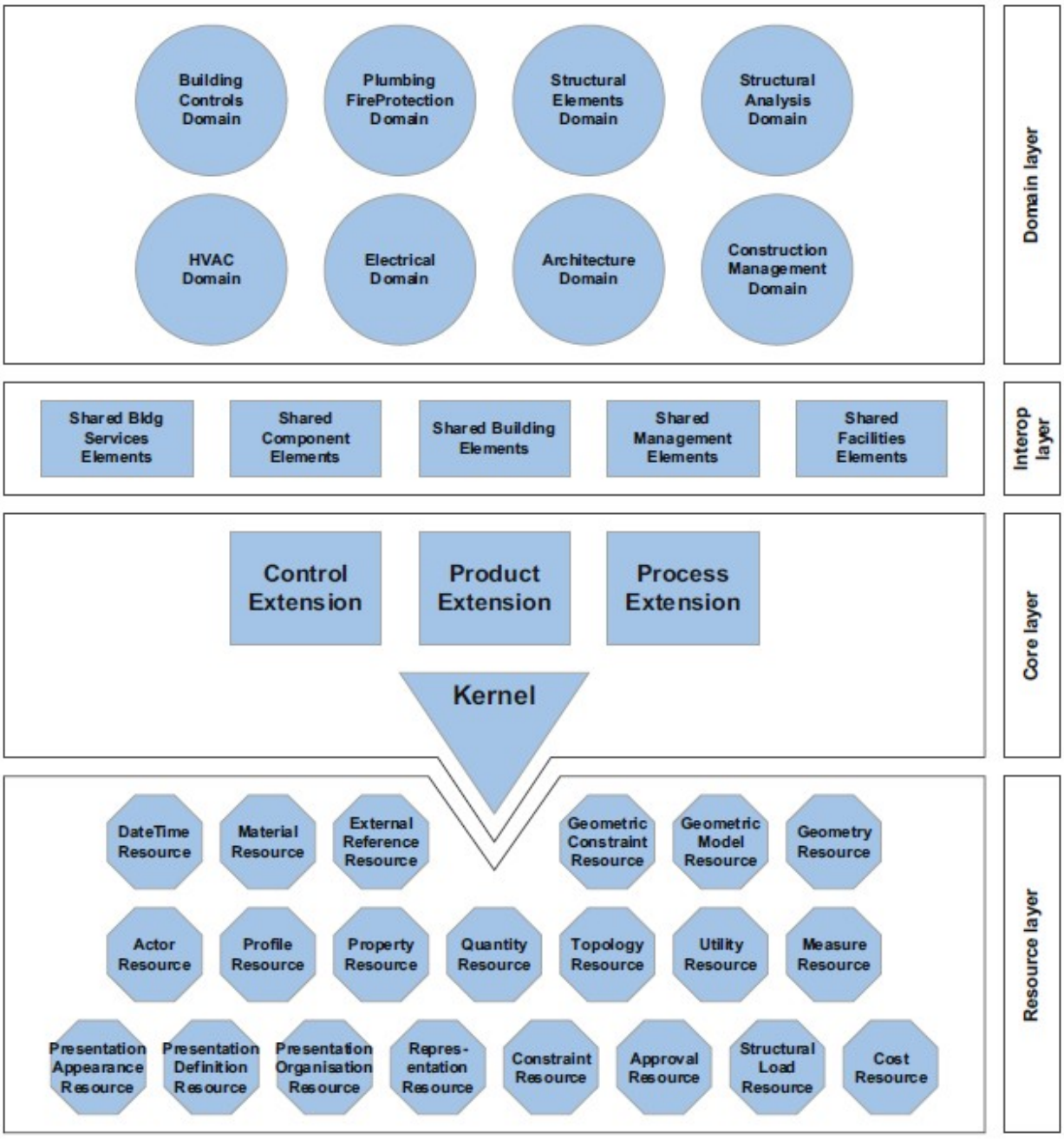


Рис. 2.9. Чоритьохрівнева архітектура основних класів галузі

Рівень ресурсів надає загальні класи, які можна використовувати для загального визначення будь-яких інших рівнів вищого рівня. На відміну від наведених вище рівнів, ці класи не мають власної ідентифікації. Крім того, вони описують такі елементи, як геометрія, матеріал, суб'єкти вимірювання, ролі, власність, презентація та вартість.

Основний рівень містить найелементарніші класи, які визначають базові структури, ключові відносини та загальні поняття. На ці класи можуть посилатися всі рівні вище, повторно використовувати та визначати саме класи на верхніх рівнях. Це робить класи на цьому рівні та вище надтипом, які надаються унікальною ідентифікацією, назвою, описом та інформацією про керування змінами. Крім того, рівень складається з ядра та трьох спеціальних розширень.

Ядро містить основні абстрактні класи та три розширення, які складаються з розширення продукту, розширення процесу та розширення керування. Схема розширення продукту описує фізичні та просторові об'єкти будівлі та їхні відповідні зв'язки.

Схема розширення процесу містить класи для опису процесів і операцій. Він надає базові засоби для визначення залежностей між елементами процесу для їх зв'язування з ресурсами. Розширення керування визначає базові класи для об'єктів керування, а також можливості розподілу цих об'єктів за фізичними та просторовими об'єктами.

Рівень взаємодії, який також відомий як спільний рівень, представляє рівень взаємодії між ядром і доменно-спеціальними схемами. Цей рівень забезпечує модульні класи, які є похідними від класів у базовому рівні, які можуть використовуватися більш ніж однією моделлю домену.

Доменний рівень охоплює вузькоспеціалізовані класи, які застосовуються лише до конкретного домену і утворюють листкові вузли в ієрархії успадкування. На класи в цьому останньому не може посилатися інший рівень або інша доменно-спеціальна схема.

2.4.2. Ієрархія успадкування

Стандарт IFC, окрім розширюваної моделі даних, також є об'єктно-орієнтованою. В об'єктно-орієнтованих моделях даних ієрархія успадкування відіграє вирішальну роль, що також стосується стандарту IFC. Ієрархія успадкування визначає зв'язки спеціалізації та узагальнення, а отже, які атрибути яких класів можуть бути успадковані іншими класами. В основному він дотримується семантичного підходу. На рисунку 2.10 показана частина IFC-моделі, що показує найважливіші класи ієрархії успадкування.

Як видно на цьому рисунку, клас ifcRoot є початковою точкою та коренем дерева успадкування IFC. Як згадувалося раніше, усі класи, за винятком класів на рівні ресурсів, повинні прямо чи опосередковано походити від ifcRoot. Цей клас забезпечує базову функціональність для унікальної ідентифікації об'єкта за допомогою глобального унікального ідентифікатора (GUID), для опису права власності та походження об'єкта, а також для відображення історії внесених до нього змін. Крім того, кожному об'єкту можна дати назву та опис. Безпосередньо з ifcRoot наступний рівень рівня успадкування слідує за класами IfcObjectDefinition, IfcPropertyDefinition і IfcRelationship.

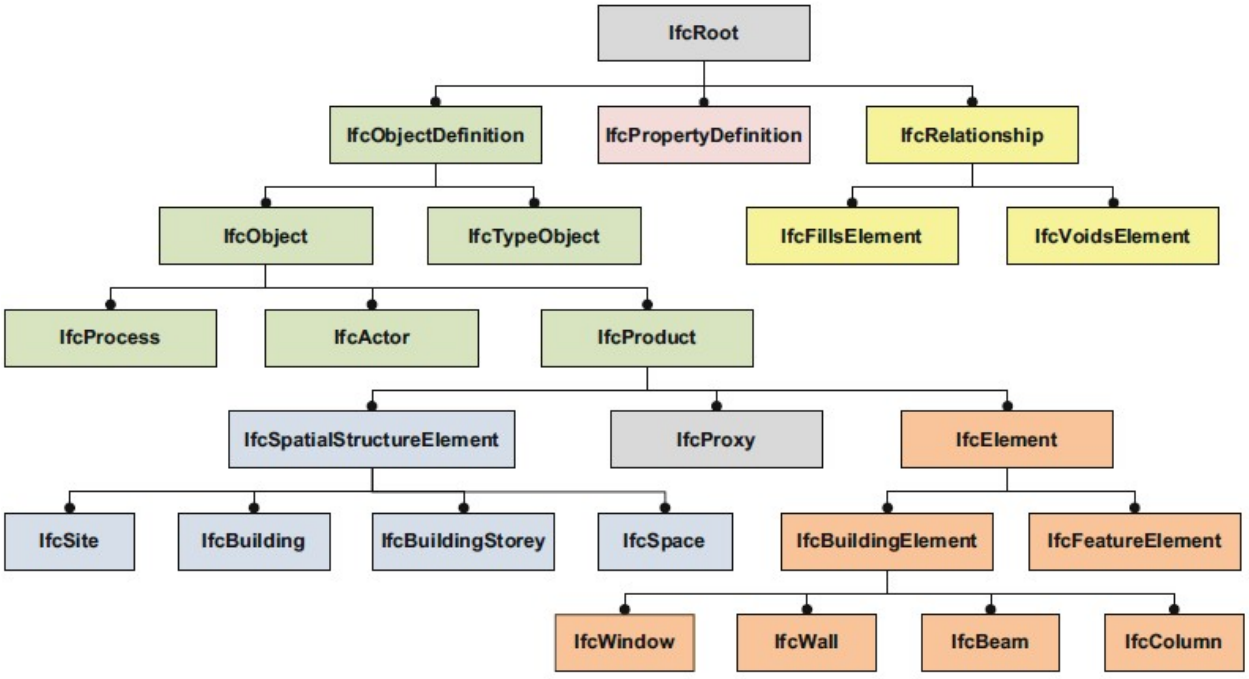


Рис. 2.10. Ієрархія успадкування IFC, включаючи класи ключів

Клас ifcObjectDefinition є абстрактним суперкласом для всіх класів, які представляють фізичні об’єкти (наприклад, будівельні елементи), просторові об’єкти (наприклад, отвори та простори) або концептуальні елементи (наприклад, процеси, витрати тощо). Він також містить визначення для опису інших осіб, які беруть участь у будівельному проекті. Ці класи представлені трьома підкласами ifcObjectDefinition, а саме a) IfcObject: який представляє окремий об’єкт як частину будівельного проекту; b) IfcTypeObject: тип об’єкта та c) IfcContext: це узагальнення контексту проекту, в якому визначаються об’єкти, об’єкти типу, набори властивостей і властивості.

Клас ifcPropertyDefinition визначає властивості об’єкта, які ще не є частиною моделі даних IFC. Ці властивості, більш відомі як набори властивостей або P-множини, являють собою набори властивостей, які використовуються разом для визначення матеріалу, певного типу продуктивності та контекстних властивостей, таких як вітрові, геологічні чи погодні дані.

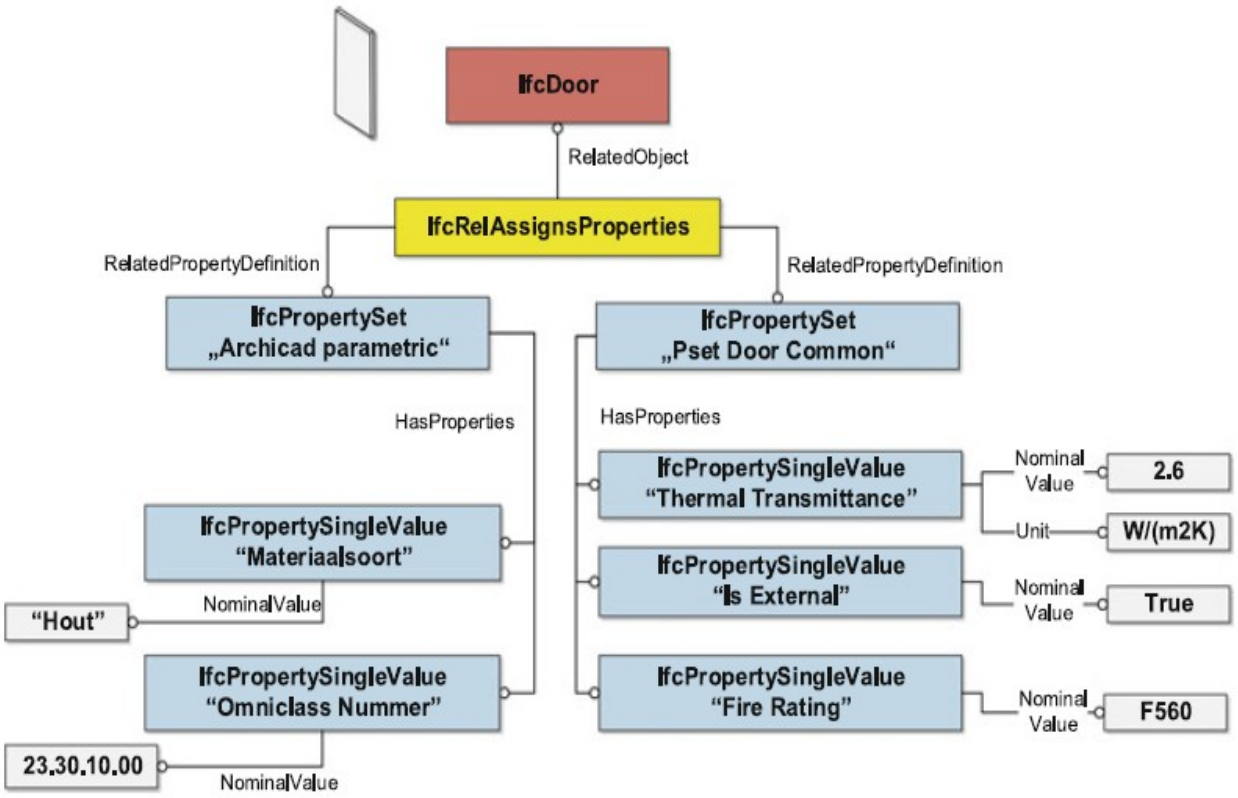


Рис. 2.11. Властивості призначені ifcDoor

Крім того, вони зібрані P-набори для багатьох будівельних об’єктів, таких як звичайний дах, стіна, віконне скління, армування вікон і балок, а також багато властивостей, пов’язаних з різними характеристиками матеріалу, наприклад для термічного матеріалу, продуктів згоряння, механічних властивостей, паливо, бетон, арматура та інші. На рисунку 2.11 показано приклад властивостей, призначених дверям.

З іншого боку, існують також ключові характеристики об’єктів, які можна визначити безпосередньо в схемі IFC-моделі за допомогою атрибутів у визначенні сутності. Наприклад, стандартні двері та вікна з їхньою абсолютною висотою та шириною можна вказати за допомогою атрибутів OverallHeight та OverallWidth.

Клас IfcRelationship і його підкласи описують об'єктивовані відносини. Відповідно до [18], такі зв’язки є важливою частиною та потужною функцією моделі даних IFC. Причина полягає в його здатності описувати зв’язки разом із семантичною класифікацією об’єктів, що є фундаментальним аспектом «інтелектуальної» інформаційної моделі будівлі, яка не лише записує елементи будівлі як ізольовані тіла; однак він підкреслює їхню функцію та взаємодію з іншими об’єктами.

Відносини формуються не шляхом прямої асоціації, а за допомогою спеціального об’єкта-посередника, який представляє сам зв’язок. Приклад проілюстровано на рисунку 2.12, де показано принцип об’єктних зв’язків на прикладі стіни, отвору та вікна.

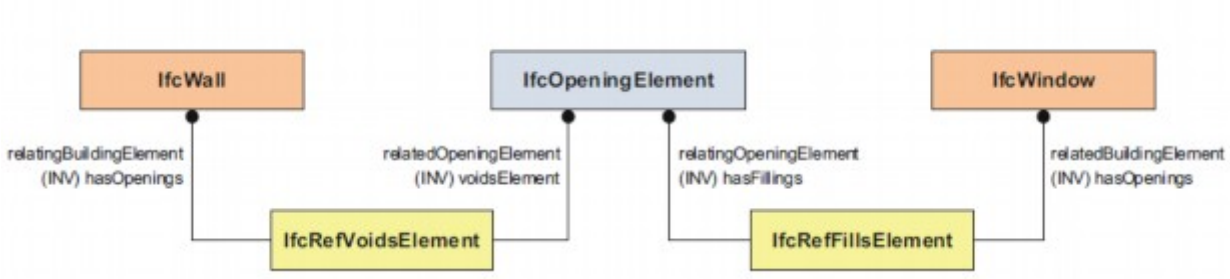


Рис. 2.12. Проміжні об’єкти, що представляють відносини

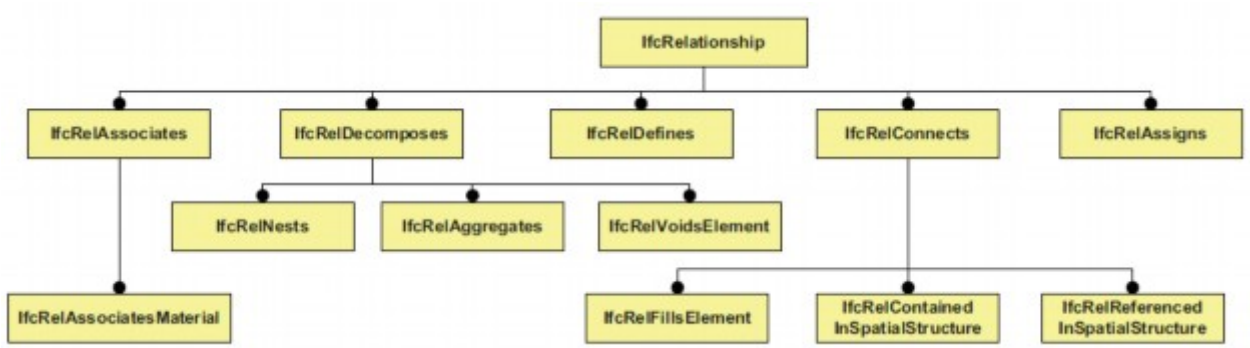


Рис. 2.13. Дерево успадкування зв'язку об'єкта

Крім того, на рисунку 2.13 показано дерево успадкування зв’язку об’єкта, де елемент IfcRelationship є кореневою формою, і кожен зв’язок може мати неформальний опис, який детально описує точну мету використання цього зв’язку.

2.4.3. Формальна мова моделювання даних

Мова моделювання даних EXPRESS є декларативною та репрезентативною мовою, на якій можна визначати об’єктно-орієнтовані моделі даних. Ця мова визначена на основі визначень бібліотеки STEP, розроблених ISO. STEP — це комплексний стандарт ISO, який надає інформацію про цифрові дані, представлення та обмін. Він забезпечує механізм, який здатний описувати дані продукту протягом життєвого циклу продукту незалежно від будь-якої конкретної системи. Цей стандарт був розроблений для вирішення питань передового обміну даними майбутніх об’єктних моделей на той час, що призвело до розробки EXPRESS-мови як основного продукту стандарту STEP. Відповідно, EXPRESS став центральним механізмом для підтримки моделювання продуктів у широкому спектрі галузей.

На додаток до текстової нотації EXPRESS, яка є машиночитаною мовою та чудово підходить для обчислювального використання, хоча й важко використовувати людиною; була розроблена версія мови з графічним

відображенням під назвою EXPRESS-G. Одночасно був розроблений EXPRESS-I, щоб забезпечити засоби відображення екземплярів елементів даних EXPRESS. Представляючи та визначаючи модель даних IFC, EXPRESS-схема дозволяє описувати інформацію, пов'язану з конструюванням, через мережу сутностей, атрибутів, простих і вибраних типів, перерахувань, колекцій і зв'язків. Ці мовні елементи використовуються для побудови конкретної EXPRESS-схеми.

Незважаючи на те, що IFC на даний момент вважається однією з найбільш відповідних схем для покращення обміну інформацією та сумісності в галузі AEC для: автоматичного розрахунку вартості будівництва, моделювання чотиривимірного графіка будівництва, аналізу шляхів евакуації, моделювання пожежі та евакуації та ін. . Його розгорнута мова має деякі обмеження щодо обмеженого діапазону вираження, труднощів у порціонуванні інформації та кількох описів тієї самої інформації. В [25] розглядають низку проблем у використанні схеми, а саме: неоднорідні процеси перекладу та зв'язування IFC; обмеження в швидкій адаптації схеми; і труднощі в розширенні схеми. У цьому випадку вони досліджували, чи можуть семантичні веб-технології надати альтернативні рішення.

2.5. Представлення онтологічної веб-мови (OWL)

Модель даних IFC, яка визначається як EXPRESS-схема та транслюється в XML-схему, також доступна як RDF-схема та OWL-онтологія, щоб забезпечити сумісні переваги для AEC-індустрії. Стандарт RDF/OWL — це мова моделювання для опису будь-якої інформації, не обмежуючись Всесвітньою павутиною (WWW). Таким чином, інформація неявно стає сумісною між різними середовищами, незалежно від того, чи є ці середовища веб-сторінками, повними програмними середовищами чи іншими. Стандарт є частиною домену Semantic Web, розробленого

Консорціумом всесвітньої павутини (W3C), щоб розширити поточну мережу, в якій інформації надається чітко визначене значення, щоб дозволити комп’ютерам і людям ідеально співпрацювати.

У будівельній індустрії семантична мережа привернула підвищену увагу через її застосування тут, для впровадження семантики в інформаційне моделювання побудови за допомогою технологій і методів семантичної мережі. Ця семантика необхідна для інструментів міркування, для перевірки узгодженості, відношень субсумпції та запитів. Враховуючи відсутність формальної семантики IFC та її EXPRESS-схеми, ці зусилля призвели до спільної міжнародної стандартизації ifcOWL: представлення онтологічної веб-мови класів Industry Foundation під егідою організації стандартизації BuildingSMART.

Ця стандартизація, яка є інтеграцією між семантичними веб-технологіями та стандартом IFC, дозволить:

а) продовжувати використовувати усталений стандарт IFC для представлення будівельних даних;

б) використовувати інструменти семантичних веб-технологій з точки зору розподілу даних, розширюваності моделі даних, запитів і міркувань;

в) повторно використовувати реалізацію програмного забезпечення загального призначення для зберігання даних, узгодженості, перевірки та знань

2.5.1. Семантичні веб-технології

Семантична павутина була представлена у всесвітній павутині як розширення традиційної павутини для можливої обробки веб-інформації за допомогою комп’ютера Більшість інформації в Інтернеті призначена лише для споживання людьми замість того, щоб додатково дозволити її інтерпретацію комп’ютерами. Ця традиційна мережа, також відома як мережа документів, кодує інформацію за допомогою природної мови, зображень, відео та інших, які роблять значення інформації неявним або

прихованим у контексті. На жаль, ця інформація зрозуміла лише людям, незважаючи на її неоднозначність.

Однак для правильної інтерпретації та розуміння машиною інформація має бути додатково описана. Інакше, як наслідок, неможлива та неефективна співпраця між комп'ютерами та людьми.

У відповідь на цю неефективність нова бізнес-модель Інтернету вимагала від організацій пошуку рішень для забезпечення глибокої сумісності та інтеграції між своїми системами та додатками. Ці зусилля призвели до нового рішення, яке визначало інформацію в Інтернеті за допомогою семантики та онтології таким чином, щоб покращити взаємодію та інтеграцію. Таким чином, семантична мережа була створена шляхом поступових змін у мережі, шляхом додавання машинозчитуваних описів до даних і документів, які вже є в мережі.

Одночасно семантичну мережу можна визначити як семантичну мережу, в якій інформація або домени представлені та об'єднуються у вигляді спрямованих мічених графів. Ця мережа складається з взаємозв'язаних даних, доступних у стандартному форматі, доступних і керованих за допомогою автоматизованих інструментів. Графи складаються з вузлів, які представляють поняття або об'єкти у світі, і дуг, які представляють логічні зв'язки між двома з цих понять або об'єктів. Щоб легко представити цю структуру, Semantic Web використовує стандарт Resource Description Framework (RDF) як мову, а для покращеної семантичної структури – словники або онтології RDF. Ці технології разом з іншими, показаними на рисунку 2.14, утворюють стек інтегрованих технологій Semantic Web для підтримки Semantic Web і Linked Data.

Поверх універсальних ідентифікаторів ресурсів (URI), уніфікованих імен ресурсів (URN) і XML, як показано на рисунку 2.14 стеку семантичного вебу інтегрованих технологій лежить мова RDF. Ця мова розроблена для машинного опису будь-якої мережі та ресурсу під час обміну інформацією. Хоча XML був розроблений для покращення інтеграції програм і систем;

цього було недостатньо, оскільки дані, якими обмінювалися, не мали чіткого опису свого значення, що означає, що інтеграція програм також повинна включати семантичну інтеграцію. За його словами, ця ситуація призвела до розробки RDF, щоб забезпечити більш повну інтеграцію та взаємодію даних між спільнотами та доменами.

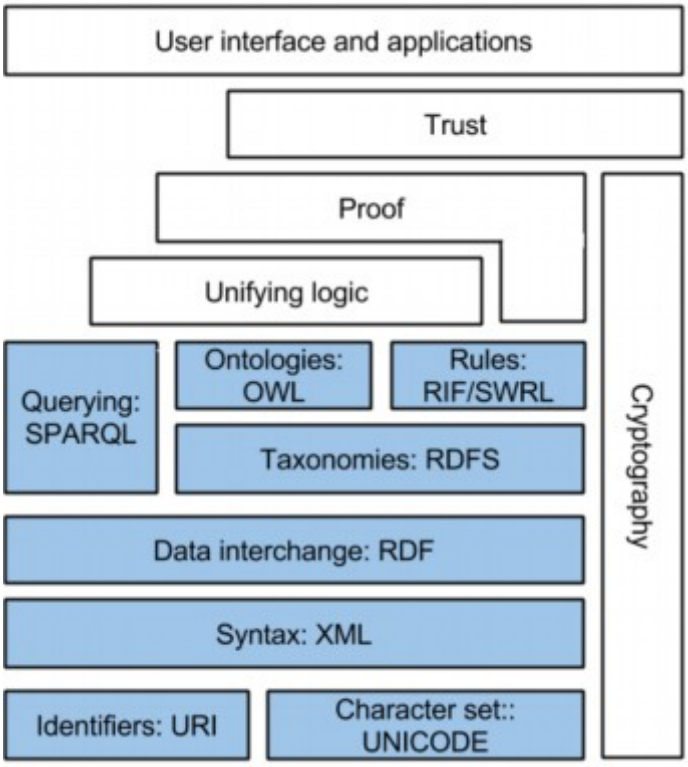


Рис. 2.14. Стек семантичної мережі

RDF прагне додати формальне визначення до ресурсів, надаючи модель даних, засновану на «суб'єкт-прогнозований-об'єкт», широко відомому як «трійка» RDF, як показано на рисунку 2.15 a). Набір трійок можна розглядати як висловлювання, що містять поняття або об'єкти у світі та їхні зв'язки, в результаті чого утворюється RDF-граф. Кожне поняття та відношення унікально визначаються URI, що робить RDF-граф явно позначеним, як це показано далі. Отриманий графік можна перетворити на текстове представлення, яке відповідає певному синтаксису. На рисунку 2.15 b)

показано приклад такої трійки, на якій зображено предмет: Вікно X; з властивістю: загальна висота; і значення: 2100 мм.

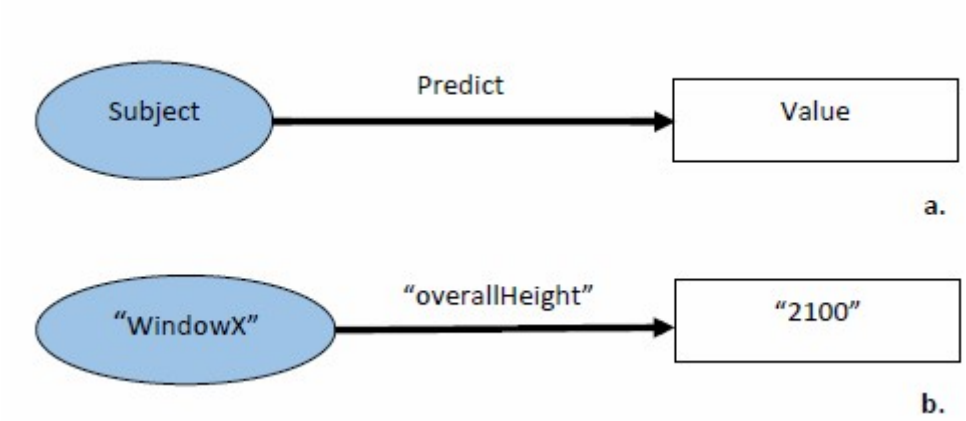


Рис. 2.15. RDF трійка: «суб’єкт прогнозує об’єкт» (a); приклад такої трійки під назвою «Вікно X: загальна висота 2100» (b)

Як згадувалося раніше, RDF-графу можна надати покращену семантичну структуру за допомогою словників або онтологій RDF. Найважливіші базові елементи для опису таких онтологій доступні у словнику RDF-схеми (RDFS). Він надає графу RDF тип специфікації системи для побудови об’єктних моделей, з яких посилаються на фактичні дані, і речі можуть бути семантично визначені. Ця специфікація складається з класів, підкласів, коментарів і типів даних, які дозволяють користувачам визначати ресурси.

Крім, клас можна розглядати як структуру подібних речей і успадкування дозволено. Ресурси можуть бути визначені як екземпляри одного або кількох класів. Це дозволяє організовувати класи в ієрархічний спосіб. Властивість можна розглядати як атрибут класу. Властивості RDFS можуть успадковуватися від інших властивостей, а обмеження домену та діапазону можуть бути застосовані, щоб зосередити їх використання. Обмеження домену використовується для обмеження того, який клас або класи може мати певна властивість, а обмеження діапазону використовується для його можливих значень.

На вершині RDFS лежить мова веб-онтології (OWL), що є розширенням словника RDF-концепцій для сприяння кращій сумісності машин. Це дозволяє створювати більш складні оператори RDF, такі як обмеження потужності, обмеження типу та складні вирази класу. Крім того, це формальна онтологія, розроблена для семантичної мережі, щоб визначити більш потужну мову для опису семантики. В Інтернеті OWL є найпоширенішою мовою розмітки для публікації та обміну даними за допомогою онтологій.

2.5.2. Конструкції для моделювання

Як описано раніше, конструкції моделювання, що використовуються в специфікації визначення моделі IFC, визначеної мовою EXPRESS, перекладаються в еквіваленти зі словників моделювання RDF(S) і OWL, в результаті чого створюється мета-модель OWL для будівель. Більшість ініціатив щодо формалізації IFC в онтології були мотивовані метою забезпечення семантично насиченої та незалежної від платформи структури, яка може підтримувати інтеграцію програмних інструментів та обмін даними в системі, заснованій на знаннях як людиною, так і людиною. читаються та використовуються машинами.

Відповідно, на основі низки помічених випадків використання визначено, що фокус розробок насправді полягає не в заміні існуючих технологій, а в поєднанні інформації про будівництво з відповідною інформацією в інших областях. Розглядаючи, наприклад, деякі розробки, зокрема, у сфері вимог до проекту та перевірки ефективності будівлі, представили середовище перевірки семантичних правил для проектування та будівництва будівель на основі тесту акустичних характеристик. Вони коротко показали, як можна подолати обмеження поточної схеми IFC при розгортанні мов семантичної мережі

У цьому випадку інформація про побудову, визначена як EXPRESS-схема, була перетворена в онтологію OWL і ряд наборів правил, розроблених

у N3Logic. Потім обидва додаються до бази знань для виконання перевірки правил. Базована на логіці графова структура семантичного вебу дозволяє проектувати та впроваджувати значно покращені середовища перевірки правил.

Залишаючись у цій галузі перевірки продуктивності побудови, досліджуються наслідки автоматизованої перевірки специфічних вимог клієнта за допомогою методів перевірки правил і стандартів семантичного вебу. У цьому підході він розробив відкриту систему перевірки вимог, яку можна багаторазово використовувати. За його словами, використання семантичного вебу робить можливою автоматизовану перевірку, за допомогою якої знання про елементи будівлі можуть бути отримані легким і відстежуваним способом. Зосереджуючись на іншому варіанті використання, можна покращити управління вимогами до будівельної техніки. У результаті є розроблений інструмент, що використовує принципи Semantic Web і Linked Data, який забезпечує зв'язок між вимогами структурної інженерії та компонентами BIM-моделі. Тут також перетворюють інформацію про будівництво та вимоги в онтологію OWL, щоб, можливо, зв'язувати, повторно використовувати та отримувати бажану інформацію через запити.

Підсумовуючи, онтологія ifcOWL запропонована та підтримується як друга альтернативна схема. Фокус його формалізації полягає в безпосередньому перетворенні з мови ifcEXPRESS на онтологію OWL. У цьому перетворенні мережа сутностей, атрибутів, простих і вибраних типів, перерахувань, колекцій і зв'язків схеми EXPRESS транслюється в класи, властивості об'єктів, властивості типів даних, домени, діапазони схеми RD.

Висновки до розділу

У другому розділі досліджуються концепції та ключові поняття системної інженерії на основі моделей, з особливим акцентом на застосуванні SysML та інших інструментів для інформаційного моделювання.

Дослідження доменно-специфічної мови за допомогою SysML демонструє можливості налаштування цієї мови для специфічних потреб окремих галузей. Профіль SysML забезпечує гнучкість та можливість розширення, що дозволяє адаптувати мову для різних типів проєктів. У перспективі SysML продовжує розвиватися як універсальний інструмент для моделювання складних систем, враховуючи нові вимоги та технічні виклики.

Інтеграція SysML у середовище розробки систем за допомогою трансформації моделі полегшує передачу даних між різними етапами розробки та підсистемами, що забезпечує безперервність та узгодженість моделювання. Цей підхід дозволяє підвищити ефективність проєктування та спрощує інтеграцію різнорідних компонентів у складних системах. Інформаційне моделювання у високо інтегрованих проєктах потребує особливої уваги до питання міжопераційності, яка забезпечується завдяки чітко структурованим моделям та трансформаціям даних між різними рівнями архітектури систем. Використання чотирьохрівневої архітектури та ієрархії успадкування сприяє побудові масштабованих та стійких моделей. Концепція побудови великої відкритої моделі ґрунтується на міжопераційності та можливості взаємодії між різними системами та технологіями. Використання формальної мови моделювання даних допомагає забезпечити уніфікований підхід до побудови моделей, що є особливо важливим у складних багаторівневих системах.

Представлення онтологічної веб-мови (OWL) та семантичних веб-технологій дозволяє ефективно інтегрувати знання та забезпечувати зв'язок між різними типами інформації. OWL надає гнучкі конструкції для моделювання, що дозволяє формалізувати дані та знання в контексті складних системних моделей.

РОЗДІЛ 3. ПРЕДСТАВЛЕННЯ МЕТОДОЛОГІЇ СПЕЦИФІКАЦІЇ ТА ПЕРЕВІРКИ МОДЕЛІ СИСТЕМИ У ВИСОКО ІНТЕГРОВАНИХ ПРОЄКТАХ

3.1. Розробка дизайну моделі

Розробка моделі включає розробку моделі інтеграції SysML-BIM для покращення систематичної ідентифікації, захоплення та перевірки функціональних вимог у високо інтегрованих проектах. Відповідно, цей розділ охоплює: а) дизайн моделі (рис. 3.1): вказує, як можна інтегрувати SysML і BIM; б) реалізацію моделі: зображення її реалізації через варіант використання для практичного застосування; і в) тестування прототипу, що забезпечує автоматичне перетворення моделі в модель.

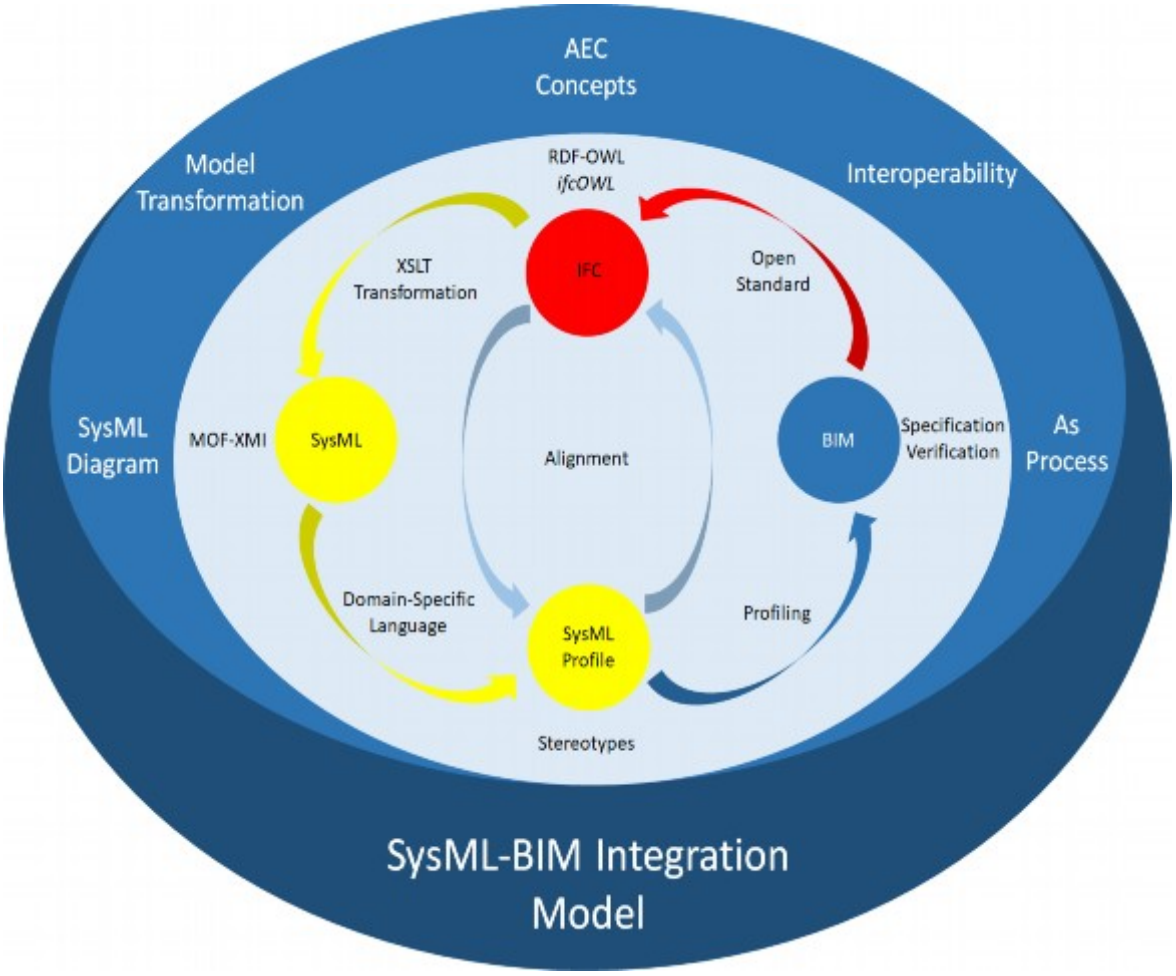


Рис. 3.1. Дизайн моделі

Дизайн моделі відображає інтеграційний підхід, у якому SysML і BIM можуть бути інтегровані, щоб забезпечити інтегровану та засновану на моделі структуру для специфікації та перевірки складних високо інтегрованих проектів. Фреймворк передбачає інтеграцію на рівні процесів, що охоплює весь життєвий цикл проекту, завдяки чому сумісність забезпечує можливість спільної співпраці незалежно від відмінностей, що стосуються домену; чітке визначення понять; співпраця на основі моделі через перетворення моделі; та покращення комунікації за допомогою діаграм та моделей.

3.1.1. Визначення взаємо залежностей в моделі

Перша частина композиції моделі зображує два ключових компоненти SysML і BIM як два взаємозалежні підходи, які сприяють узгодженості функціональних вимог під час інструктажу та процесу проектування. Перш за все, SysML як мова візуального моделювання надає повний набір діаграм і конструкцій для моделювання їхньої структури та їх взаємодії з іншими аспектами системи, що розробляється - в даному випадку будівлі. У цьому підході до моделювання діаграми, як діаграма варіантів використання, можуть бути реалізовані для відображення та моделювання функцій системи на основі варіантів використання, щоб зобразити, як користувачі використовують систему для досягнення своїх цілей.

Одночасно діаграму вимог можна використовувати для графічного зображення ієрархії вимог, що визначають специфікації системи та її компонентів. Крім того, діаграму визначення блоку можна застосувати для моделювання та представлення структури системи з точки зору її функцій, ієрархії та взаємозв’язків. Крім того, SysML - як графічне позначення - покращує комунікацію між учасниками проекту та зменшує ймовірність непорозумінь. Крім того, через своє походження від UML, його можна виразити як метамодель MOF-XMI або інші, що забезпечують взаємодію між комп’ютерними системами. З іншого боку, BIM як процес забезпечує

інтегрований підхід до визначення, проектування, будівництва та експлуатації будівель за допомогою цифрових моделей. У рамках цього підходу та інших рівнів створення, управління та передача інформації між зацікавленими сторонами є принципово важливими. При цьому моделі створюються учасниками проекту та використовуються в різний час для кількох варіантів використання. Наприклад, одним із таких варіантів використання є специфікація та перевірка функціональних вимог протягом усього життєвого циклу будівлі. Однак дослідження обмежується етапом інструктажу та проектування.

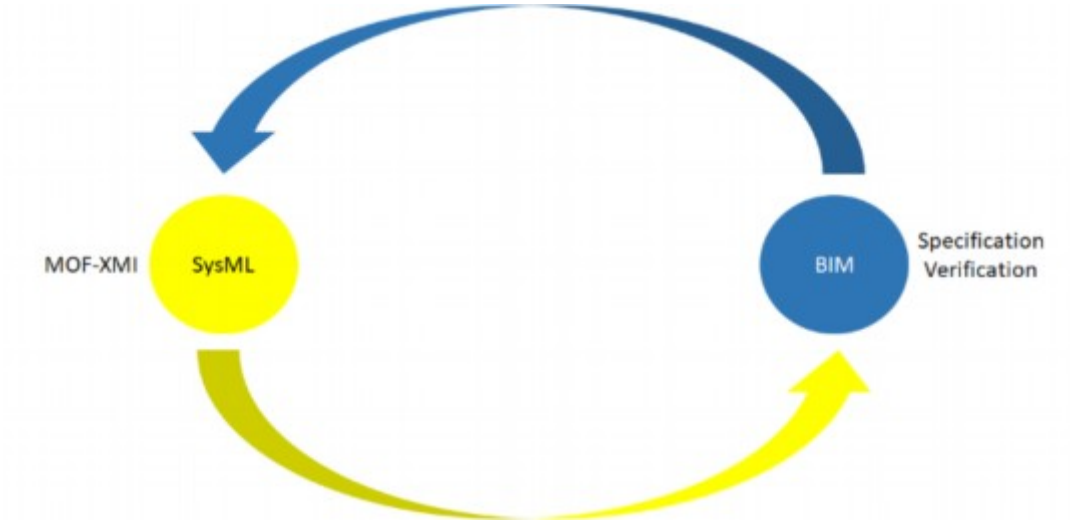


Рис. 3.2. SysML і BIM як два взаємозалежні підходи

3.1.2. Інтероперабельність моделі

Друга частина композиції ілюструє, як два підходи, у цьому випадку SysML і BIM, можуть взаємодіяти з іншими предметно-спеціальними підходами. У випадку SysML, предметно-спеціальну мову можна розробити шляхом створення профілів SysML, які надають конструкції, які розширюють і додають нові можливості самій мові моделювання. Відповідно, стереотипи можуть бути використані для створення профілів і нових елементів моделі з існуючих, включаючи детальні атрибути, придатні для предметно-спеціальних програм.

У випадку BIM класи Industry Foundation Classes (IFC) можна використовувати як відкритий стандарт сумісності для сумісності між кількома підходами домену. Стандарт IFC представляє як геометрію, процес, характеристики матеріалу, виготовлення та інші властивості. Іншими словами, він забезпечує повну семантичну структуру будівельних моделей, яка є важливою основою для досягнення мети BIG Open BIM. У цій моделі її мовне розгортання базується на технології семантичної мережі RDF-OWL для семантичної взаємодії. Цей підхід, добре відомий як ifcOWL, розроблений для доповнення обмежень формальної та альтернативної схеми IFC.

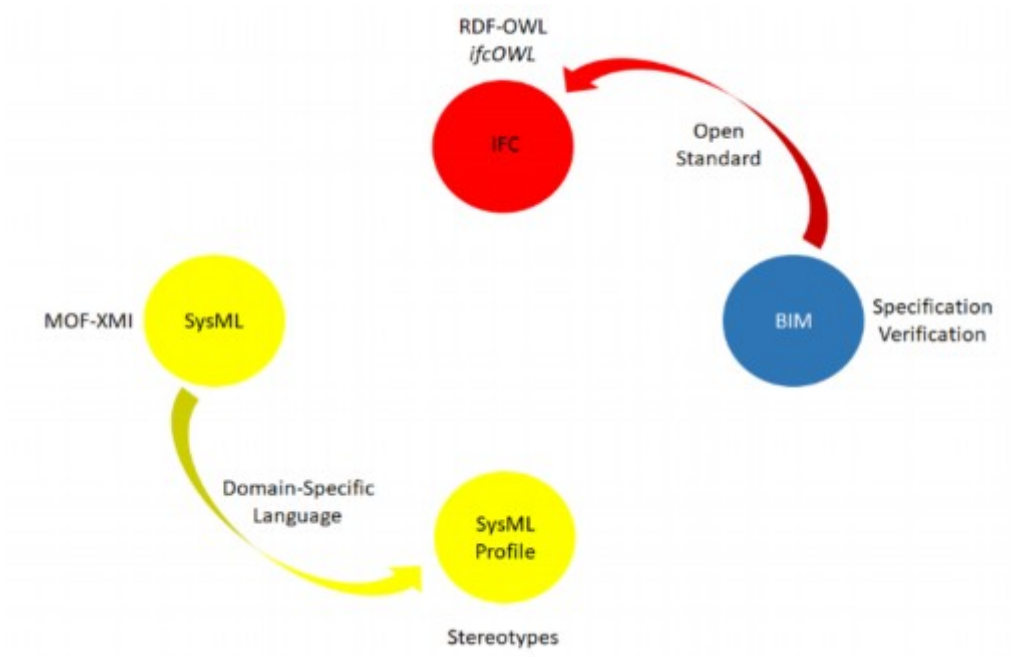
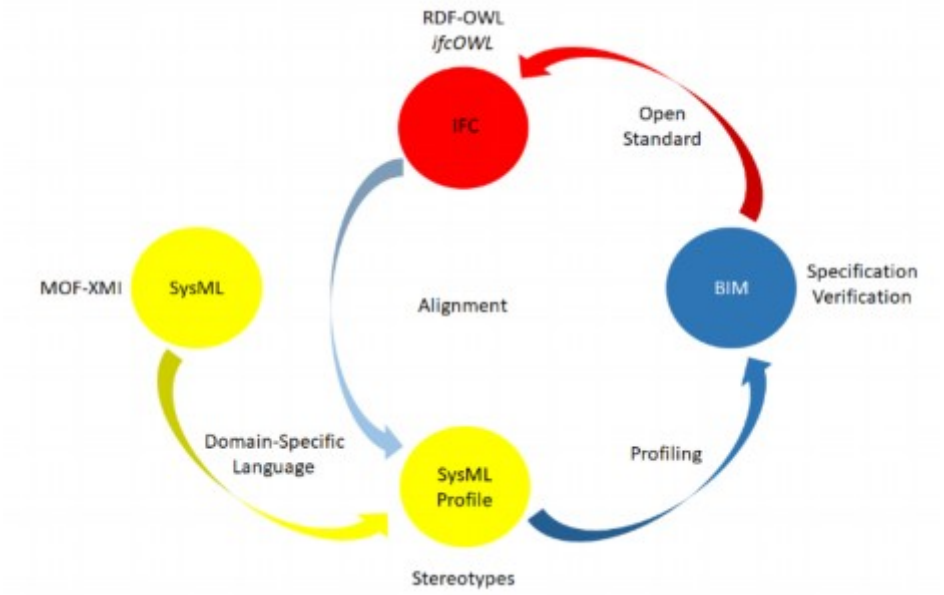


Рис. 3.3. Інтероперабольність підходів

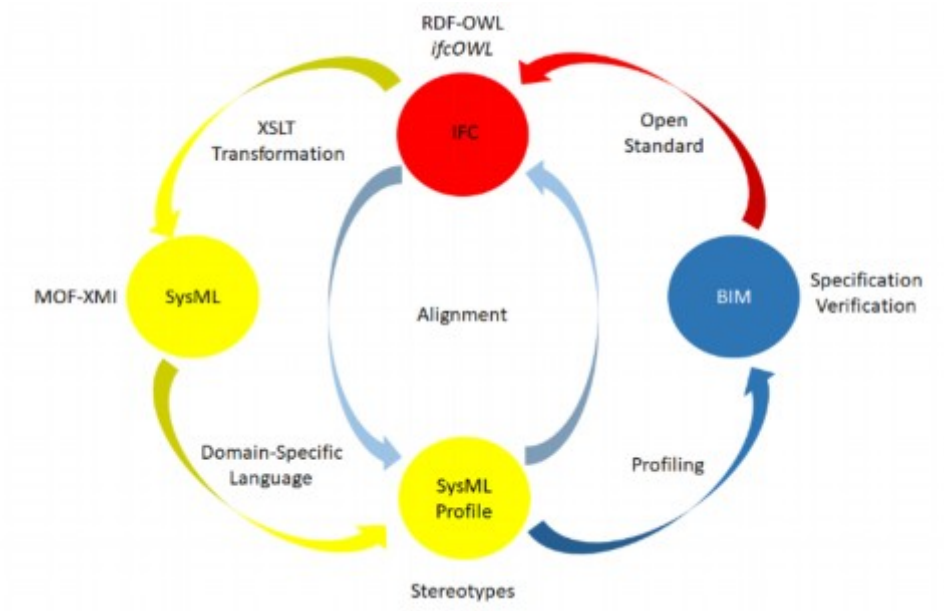
3.1.3. Процеси інтеграції в моделі

Третя частина композиції показує, як точно реалізована інтеграція між SysML і BIM. Як показано на рисунку 3.4, пропущені з'єднання були включені, щоб зв'язати компоненти разом, щоб завершити коло в пов'язаному колі. Зосередження уваги на взаємодії між профілем SysML і BIM (рис. 3.4 а) це посилання створюється через Profiling-link, яке

встановлюється для подолання вимог випадків використання BIM-процесу. У цьому дослідженні випадки використання обмежені специфікацією та перевіркою функціональних вимог.



а)



б)

Рис. 3.4. Частини інтеграції: Profiling-Link а); XSLT Transformation-Link б)

Профіль і стереотипи SysML, використовуючи концепції IFC як основу, можна використовувати для визначення та створення нових

концепцій для виконання цих випадків використання в середовищі SysML. З іншого боку, взаємодія між IFC і SysML (рис. 3.4 б) створюється за допомогою XSLT Transformation-link. Цей зв'язок може бути виконаний через міст «RDF(S)-MOF», що дозволяє трансформувати концепції RDF(S) у концепції MOF і навпаки відповідно до них. Ця взаємодія — використання профілю SysML, що має вирішальне значення для співпраці між моделями, що підтримує послідовний зв'язок між даними з IFC у SysML.

3.2. Реалізація моделі

Як згадувалося раніше, реалізація моделі виконується через конкретний варіант використання для практичного застосування та оцінки. Цей варіант використання під назвою IfcDoor зосереджується на одному конкретному об'єкті з процесу предметної області будівництва: дверному об'єкті. Двері як один із основних елементів у будівлях можна визначити як будівельний елемент, який переважно використовується для забезпечення контрольованого доступу для людей і товарів. Реалізація охоплює основні частини інтеграції, як показано на рисунку 3.4 і описано в попередньому пункті про інтеграцію, а саме: (а) зв'язок між профілем SysML і BIM через профілювання з IFC як вхідні дані; б) зв'язок між IFC і SysML через перетворення XSLT із профілем SysML як вхідні дані.

3.2.1. Створення спеціалізованих профілів SysML

Більшість поточних інструментів моделювання SysML надають можливість створювати профілі SysML для певного домену. У цьому дослідженні використовується як інструмент моделювання. Перший крок до інтеграції можливий шляхом профілювання концепцій з BIM-процесу. Розглядаючи об'єкт двері як варіант використання, для створення класу дверей використовується діаграма профілю під назвою Building Element: IfcDoor (рис. 3.5). У цьому випадку блок стереотипу використовується та

розширюється через «відношення узагальнення » для розвитку стереотипу IfcDoor. Крім того, стереотип включає відсіки, які спрямовані на врахування характеристик класу (IfcDoor). У цьому прикладі відсік містить властивості, прийняті зі специфікації схеми IFC, визначеної BuildingSMART для узгодження з IFC відповідно до проекту моделі.

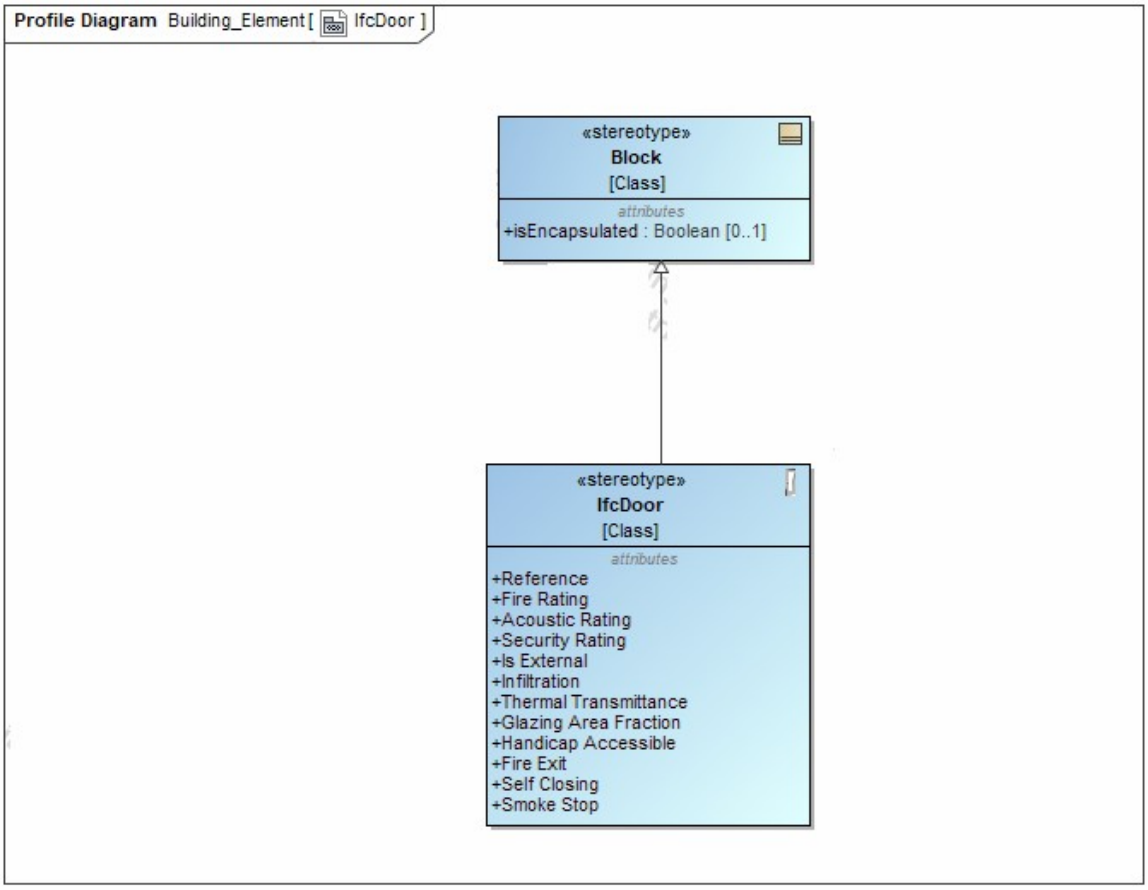


Рис. 3.5. Діаграма профілю, що визначає стереотип IfcDoor

Як наслідок, на рисунку 3.6 зображено діаграму визначення блоку BDD (Block Definition Diagram) під назвою Building Element, що показує дверний об’єкт, зафіксований у стереотипі <<IfcDoor>>, визначеному раніше. Верхній стереотип показує клас IfcDoor, за яким нижче слідує зв’язок узагальнення, що ілюструє IfcDoor_id та його характеристики як потенційні екземпляри цього класу. Зосереджуючись на цьому початковому зв’язку з процесом інтеграції, ця частина ілюструє здатність визначати та моделювати предметно-спеціальні об’єкти з процесу BIM, виконувати сценарії

використання як специфікацію та перевірку функціональних вимог у BIM-процесах.

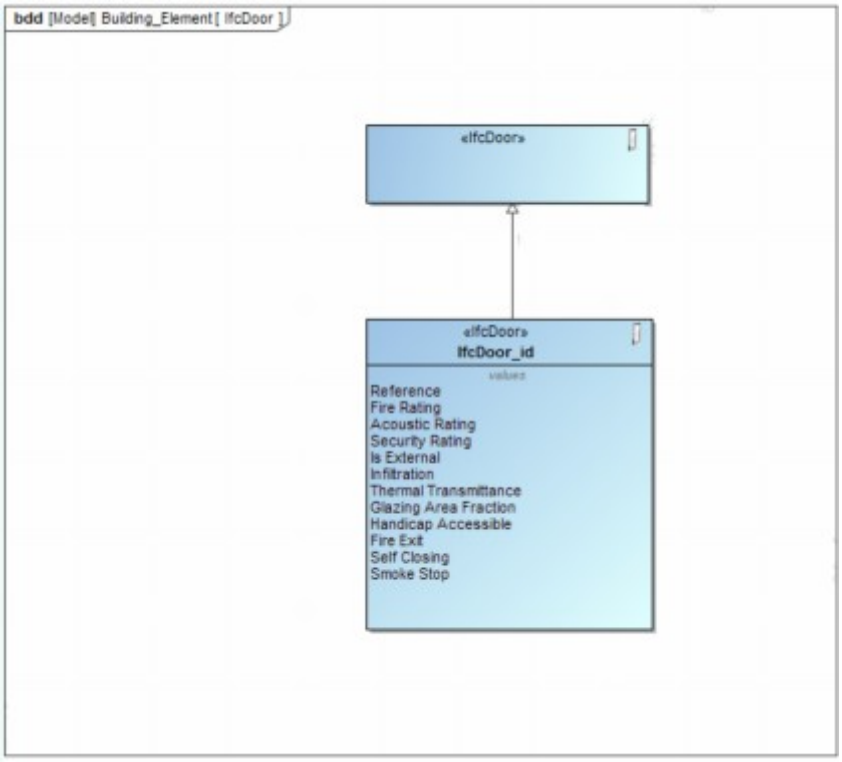


Рис. 3.6. Діаграма, що показує стереотип <<IfcDoor>> і його потенційні екземпляри (IfcDoor_id)

3.2.2. Трансформація моделі

Друга частина процесу інтеграції є практичною на основі підходу трансформації моделі, як описано в розділі 3.1.3. і показано на рисунку 3.4 б). Відповідно, розроблено підхід до трансформації, зображений на рисунку 3.7, який складається на основі основних концепцій трансформації моделі. Як описано раніше, трансформація моделі дуже підходить, коли дві сторони обміну є різними, і трансформація необхідна для підтримки обміну даними, що має місце в цьому дослідженні. Як показано в другому розділі і переробленому рисунку 3.7, перетворення з IFC на SysML можна розглядати як механізм трансформації, який читає вихідну модель (IFC-модель) і записує її в цільову модель (SysML BDD). Вихідною моделлю, використаною в цьому тестовому прикладі, є добре відомий дуплексний будинок. Будинок

виражається відповідно до запропонованого представлення OWL для IFC-схеми: ifcOWL.

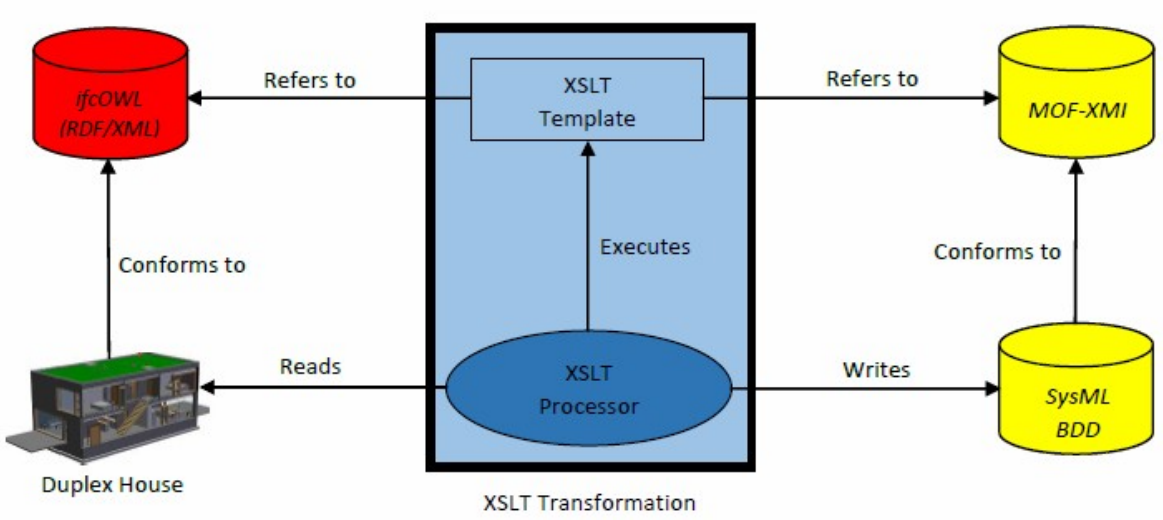


Рис. 3.7. Модельно-трансформаційний підхід

Представлення ifcOWL створюється, а потім форматується у форматі TURTLE (.ttl). З іншого боку, цільовою моделлю є SysML BDD, який можна виразити у стандарті метамоделі: MOF-XMI. Більшість інструментів SysML постачаються разом із цим сумісним стандартом. Що стосується перетворення, воно базується на перетворенні XSLT, у якому механізм перетворення представлено процесором XSLT, а визначення перетворення записується в шаблоні XSLT.

Нарешті, через обсяг схеми та для відповідності варіанту використання, це дослідження охоплює трансформацію випадкового об’єкта ifcDoor, включаючи дві його властивості. У наступних розділах описано вилучення та створення графа ifcDoor, а також його перетворення на SysML BDD за допомогою перетворення XSLT.

3.3. Опис процесів трансформації моделі

Як описано, перетворення фокусується лише на одному випадковому об’єкті ifcDoor із двома його властивостями. Отже, першим кроком було

виділення графічного відношення, в якому об’єкт ifcDoor пов’язаний зі своїми властивостями. З цією метою SPARQL використовується для запиту цього графіка, після чого може бути вилучено та створено новий частковий граф. Відомо, що запити SPARQL визначаються за допомогою функцій «CONSTRUCT», «WHERE», «LIMIT 2» (лістинг 3.1) для вилучення та створення нового графіка з основного.

Лістинг 3.1. Процес вилучення та аналізу конкретної частини інформації за допомогою спеціалізованої мови запитів SPARQL

```
PREFIX ifcowl: <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#>
PREFIX inst: <http://linkedbuildingdata.net/ifc/resources/20190418_114625/>
PREFIX list: <https://w3id.org/list#>
PREFIX express: <https://w3id.org/express#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

CONSTRUCT {

        ?Door a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .
        ?RelDefines <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatedObjects_IfcRelDefines> ?Door .
        ?RelDefines a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcRelDefinesBy Properties> .
        ?RelDefines<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatingPropertyDefinition_IfcRelDefines
        ByProperties> ?PropertySet .
        ?PropertySet a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySet> .
        ?PropertySet <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#hasProperties_IfcPropertySet>
        ?SingleValue . ?SingleValue a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcProperty
        SingleValue> .
        ?SingleValue <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty> ?Name .
        ?Name a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier> .
        ?Name <https://w3id.org/express#hasString> ?N .
        ?SingleValue<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_IfcPropertySingleValue>
        ?Value .
        ?Value <https://w3id.org/express#hasString> ?V .

}

WHERE {

        ?Door a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .
        ?RelDefines <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatedObjects_IfcRelDefines> ?Door .
        ?RelDefines a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcRelDefinesByProperties> .
        ?RelDefines<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatingPropertyDefinition_IfcRelDefines
        ByProperties> ?PropertySet .
        ?PropertySet a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySet> .
        ?PropertySet <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#hasProperties_IfcPropertySet>
        ?SingleValue .
        ?SingleValue a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySingleValue> .
        ?SingleValue <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty> ?Name . ?Name a
        <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier> .
        ?Name <https://w3id.org/express#hasString> ?N .
        ?SingleValue<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_IfcPropertySingleValue>
        ?Value .
        ?Value <https://w3id.org/express#hasString> ?V .

}

limit 2
```

Як показано на рисунку 3.8, інструмент SPARQL Apache Jena Fuseki використовується для визначення запитів і створення часткового графіка.

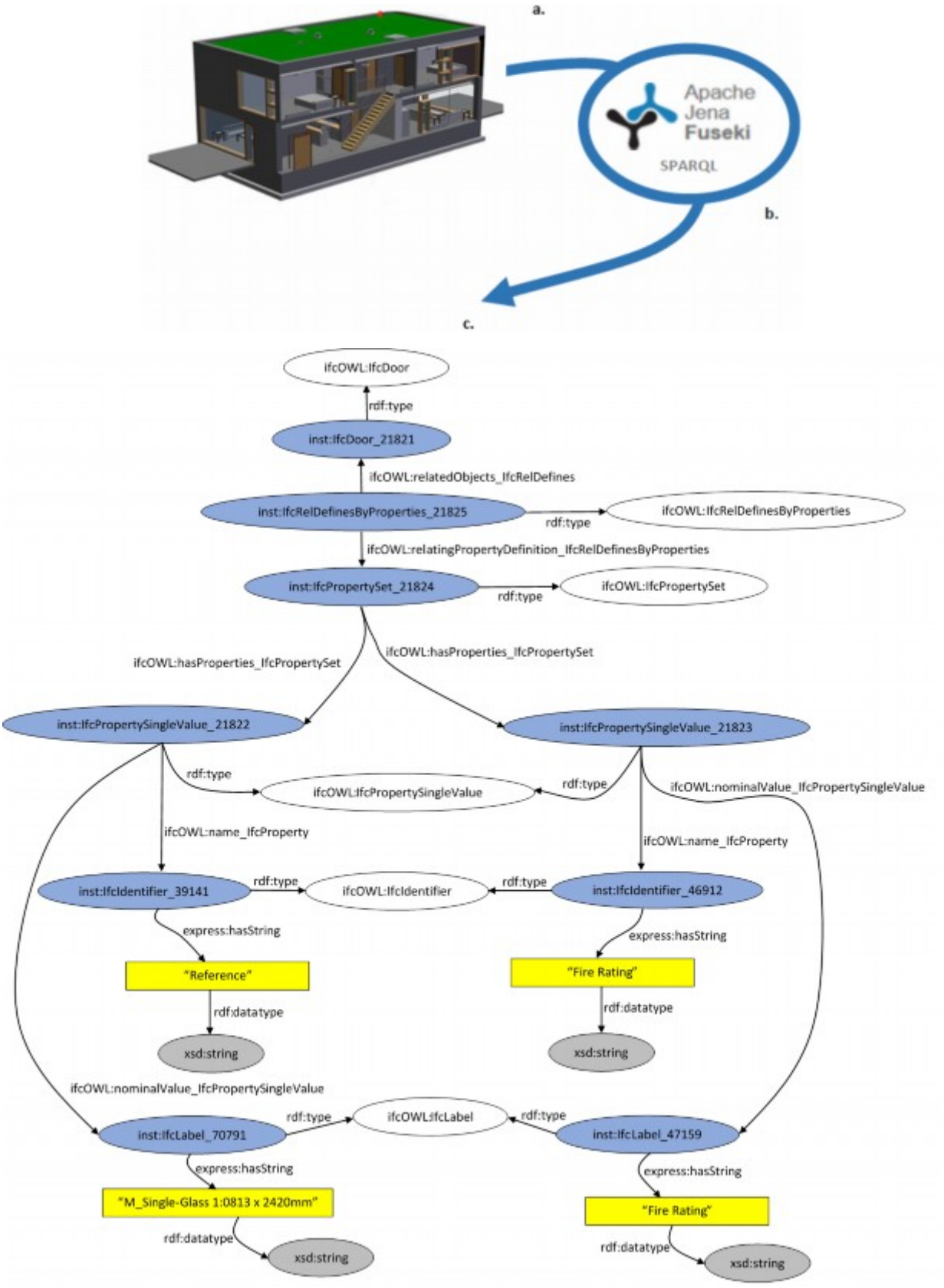
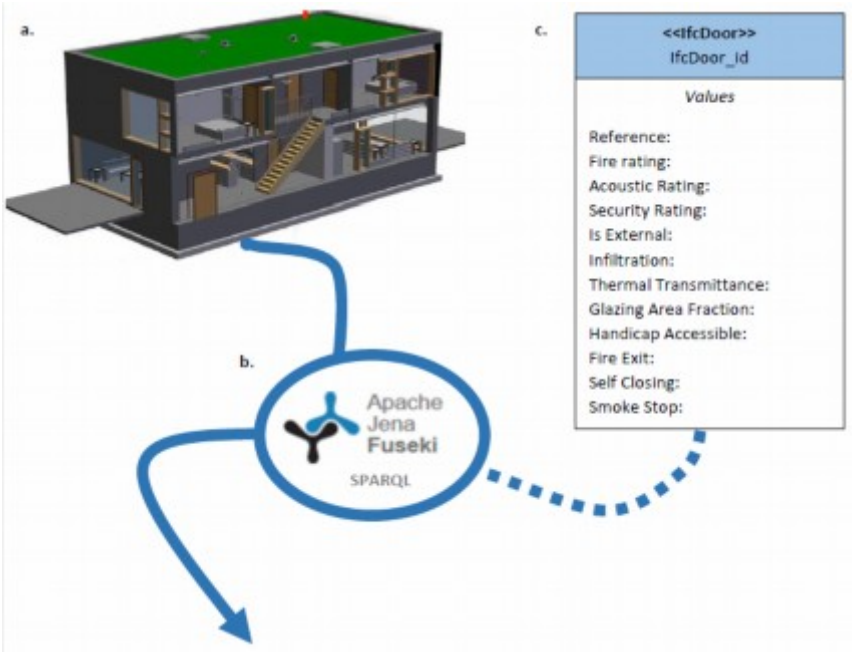


Рис. 3.8. Тестовий приклад (будинок) a), Apache Jena Fuseki для вилучення графа IfcDoor b), витягнутий (попередній) граф IfcDoor c)



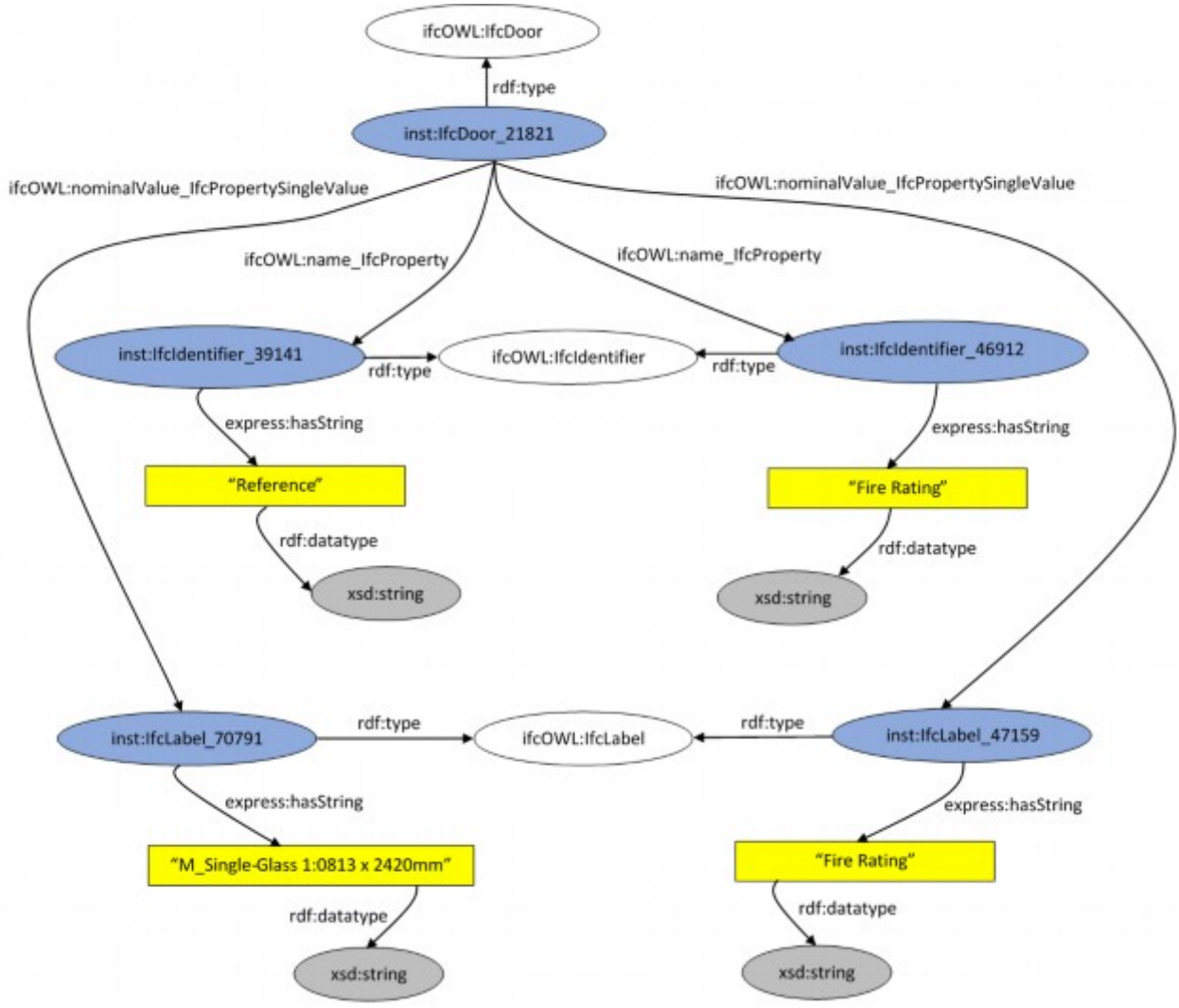


Рис. 3.9 Тестовий приклад (будинок) a), Apache Jena Fuseki для вилучення та створення нового графа IfcDoor b), профіль SysML як вхідні дані c), граф IfcDoor

Як видно на рисунку 3.8, екземпляр IfcDoor_21821, який є типом класу IfcDoor, витягується разом із двома його властивостями під назвою Reference та Fire Rating. Властивості пов’язані з дверним об’єктом через зв’язок успадкування, що робить граф надзвичайно великим і, отже, ускладнює перетворення до MOF-XMI. З цієї причини тут використовується підхід, який передбачає розробку більш простого та зрозумілого графіка для легкої навігації через запити та перетворення.

Основа для розробки лежить на вхідних даних із профілю SysML до IFC, як показано на рисунку 3.4, у якому профіль SysML представляє дверний об’єкт і його властивості, як показано на рисунку 3.9 c), у прямій компартментній асоціації. По суті, це можна розглядати як структуру, в якій можна створювати нові графіки для легкого перетворення, наприклад, між IFC і SysML. Таким чином, він прийнятий для створення графа, показаного на рисунку 3.9, за допомогою якого властивості безпосередньо пов’язані з класом IfcDoor, що робить граф більш компактним, меншим і стабільно придатним для інтеграції, ніж той, що показаний на рисунку 3.8 c). Подібно до попереднього підходу, запити SPARQL також визначені для вилучення дверного об’єкта та створення нового графа. Для цього використовуються такі функції, як «CONSTRUCT», «WHERE» і «LIMIT 2».

XSLT (eXtensible Stylesheet Language Transformations) використовується для перетворення графа ifcDoor, вираженого у форматі TURTLE (.ttl), у SysML BDD. Однак перед перетворенням графік виражається у форматі RDF/XML. Що виражає RDF-граф як документ XML, що робить його більш сумісним зі стандартом MOF-XMI, оскільки XMI є частиною стандарту на основі XML для мета-мета-моделі, мета-моделі та спільного використання моделі. Крім того, XSLT, його шаблон і процесор ідеально підходять для трансформації.

Зосереджуючись на підході до трансформації, зображеному на рисунку 3.7, графік ifcDoor представляє частковий графік із цілого, що визначає будинок. Продовжуючи XSLT-підхід, XSLT-шаблон визначає правила

виконання трансформації. Цей набір правил, визначений у синтаксисі XSL, як показано в Додатку IV, містить, по-перше, позначене червоним кольором посилання на файл RDF/XML як вхідні дані для вилучення та перетворення даних; по-друге, позначений зеленим кольором, простір імен XMI, що визначає вихідний файл як стандарт XMI; по-третє, виділені жирним шрифтом, ключові компоненти та властивості MOF-структури, у яку вхідний файл (RDF/XML) має бути перетворений і зафіксований у вихідному файлі (MOF-XMI); і, нарешті, позначені синім кольором, запити, визначені мовою XPATH для вилучення дверного об'єкта та його властивостей із вхідного файлу та включені у вихідний файл.

Після цього трансформація може бути виконана. Для цієї операції використовується процесор XSLT. Як визначено на рисунку 3.7, процесор виконує визначення перетворення, визначене в шаблоні, використовуючи файл RDF/XML як вхідні дані та трансформований у вихідний файл MOF-XMI, як показано в лістингу 3.2.

Лістинг 3.2. Графік IfcDoor, виражений як MOF-XMI (вихідний файл)

```
<?xml version="1.0" encoding="UTF-8"?>

<xmi:XMI xmlns:xmi="http://www.omg.org/spec/XMI/20131001"
xmlns="http://www.w3.org/2005/Atom"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:inst="http://linkedbuildingdata.net/ifc/resources/20190418_114625/"
xmlns:list="https://w3id.org/list#"
xmlns:ifcowl="http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:express="https://w3id.org/express#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

<uml:Package xmlns:uml="http://www.omg.org/spec/UML/20131001" name="ifcDoor"
xmi:id="_18_5_3_429b06e2_1554882936919_674848_14243" xmi:type="uml:Package">

<packagedElement xmi:type="uml:Class" xmi:id="_18_5_3_429b06e2_1554882987193_743807_14276"
        name="http://linkedbuildingdata.net/ifc/resources/20190418_114625/IfcDoor_21821">
<ownedAttribute xmi:type="uml:Property" xmi:id="_18_5_3_429b06e2_1554967218049_369151_14447"
        name="Reference" aggregation="composite">
<defaultValue xmi:type="uml:LiteralString" xmi:id="_18_5_3_429b06e2_1556611363126_30851_14363"
        value="M_Single-Glass 1:0813 x 2420mm" />
</ownedAttribute>
<ownedAttribute xmi:type="uml:Property" xmi:id="_18_5_3_429b06e2_1554967258730_934889_14450"
        name="Fire Rating" aggregation="composite"> <defaultValue xmi:type="uml:LiteralString"
        xmi:id="_18_5_3_429b06e2_1556611390513_544678_14364" value="Fire Rating" />
</ownedAttribute>
</packagedElement>
</uml:Package>
</xmi:XMI>
```

Як можна побачити, вихідний файл структурований за допомогою синтаксису MOF-XMI, що містить об’єкт та його властивості. Крім того, цей файл можна імпортувати в кожен інструмент SysML, який читає файли MOF-XMI. Як приклад, вихідний файл імпортується в Cameo Systems Modeler, де раніше був розроблений профіль для об’єкта. Результат зображено на рисунку 3.10, де показано блок-схему, що містить об’єкт IfcDoor_21821 і його дві властивості, визначені раніше. Нарешті, його можна застосувати до випадку використання конкретного домену, у цьому випадку до стереотипу <<IfcDoor>>, як показано в на рисунках 3.11 і 3.12.

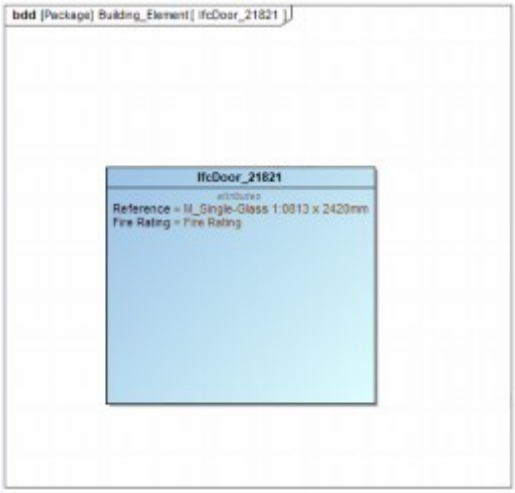


Рис. 3.10. Діаграма, що показує імпортований IfcDoor_21821

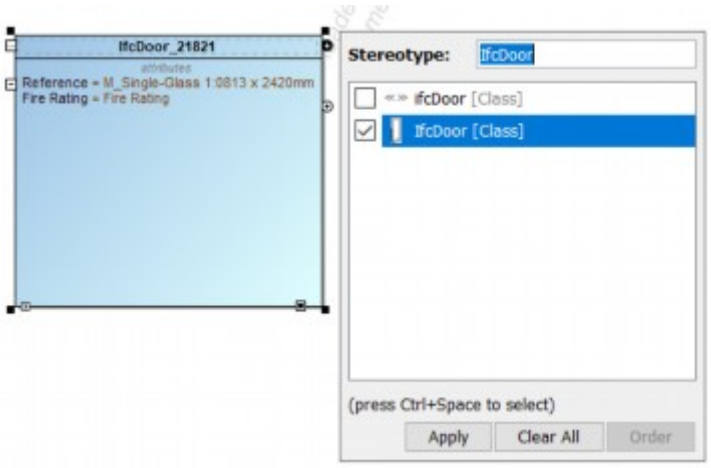


Рис. 3.11. Застосування IfcDoor_21821 до конкретного стереотипу; <<IfcDoor>>

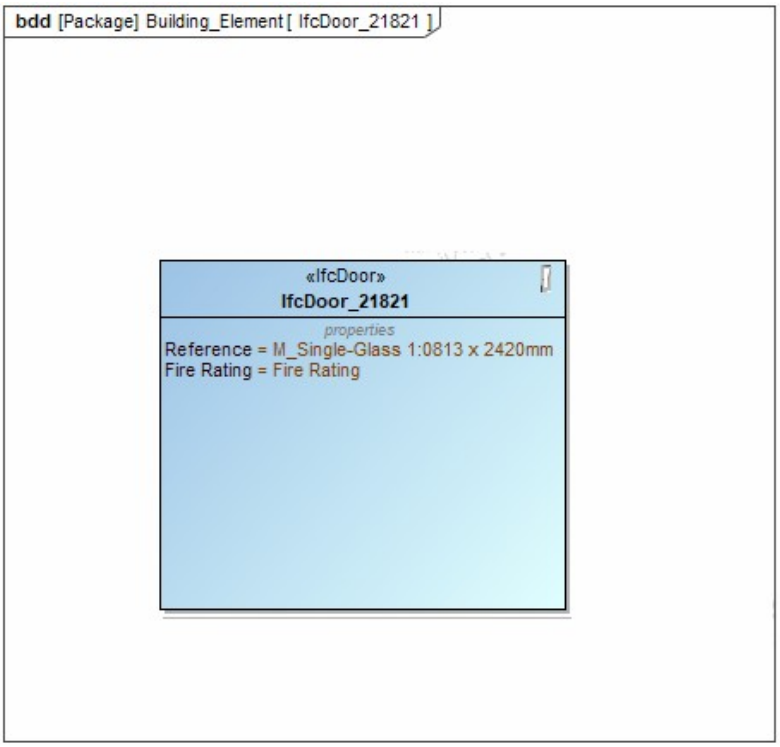


Рис. 3.12. Діаграма, що показує імпортований IfcDoor_21821 як стереотип <<IfcDoor>>

3.4. Перевірка моделі системи в умовах високо інтегрованого проекту

Модель процесу, зображена на рисунку 3.13, дає огляд того, як розроблену модель інтеграції SysML-BIM можна використовувати в контексті проекту житлового будинку. У підході до спільної роботи команда, що складається з системних інженерів і дизайнерів. Основою для цієї співпраці є розробка моделі системи для квартири (SM.010). Ця модель, розробка якої доручена групі системних інженерів, містить системну специфікацію квартири. Після процесу створення модель системи потім розповсюджується через сервер точки спільного доступу до команди проектувальників, формуючи основу для продовження командою розробки проекту (DM.010). Проект базується на комплексному підході BIM. Інтегрована BIM-модель, отримана в результаті цього підходу, потім

передається групі системних інженерів через сервер для ініціювання перевірки проекту (SM.020).

На цьому етапі проект перевіряється, щоб переконатися, що він відповідає специфікації моделі системи. При позитивному результаті можна розпочинати будівництво.

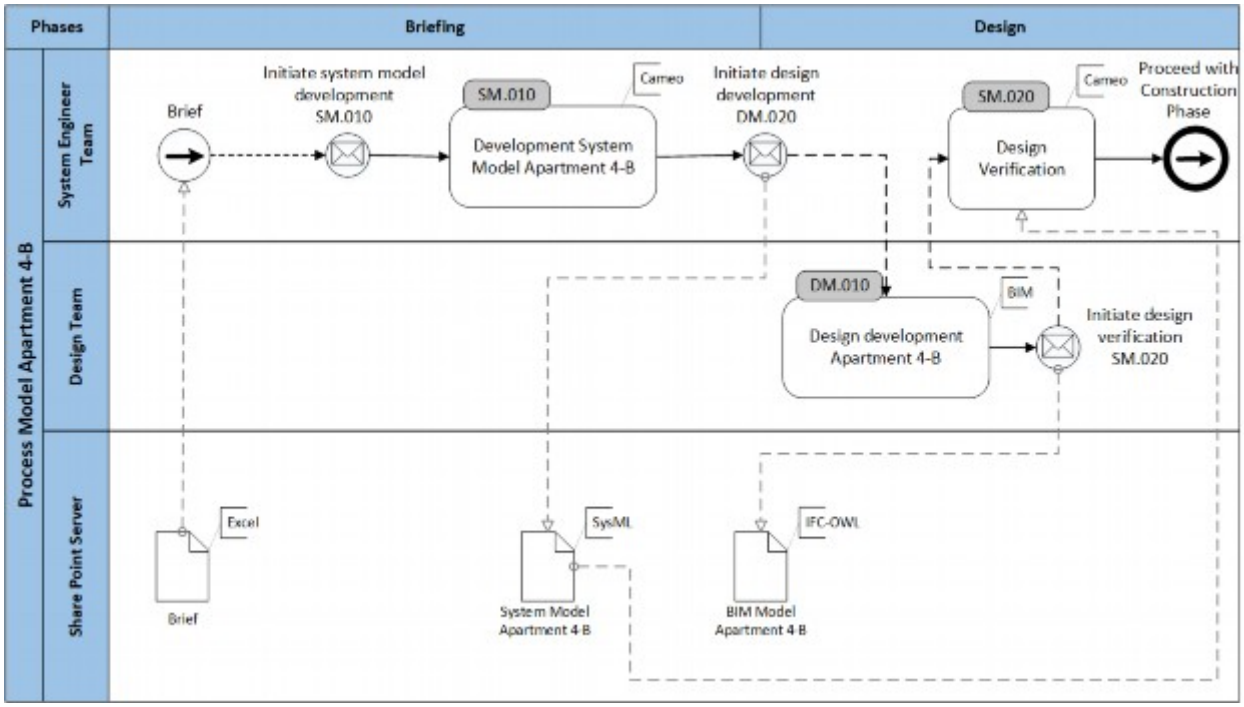


Рис. 3.13. Модель процесу

Як описано раніше, системна модель є основним артефактом MBSE-підходу, який дозволяє розробити цікаву систему для досягнення її загальних цілей. Розробка цієї моделі базується на трьох постулатах, а саме: a) мова моделювання, яка в даному випадку SysML; b) метод моделювання, за допомогою якого використовується методологія параметричного моделювання системи (PSM), але лише діаграму варіантів використання (uc), діаграму вимог (req) і діаграму визначення блоку (bdd) вибрано для досягнення мети дослідження і (c) інструмент моделювання, за допомогою якого інструмент SysML Cameo Systems Modeler вибрано як інструмент.

Процес створення системної моделі починається з визначення предметно-спеціальної мови на основі BIM за допомогою профілювання.

Діаграма профілю забезпечує цю функцію визначення, що визначає концепції BIM-IFC на основі блоків, для виконання варіантів використання як специфікація та перевірка функціональних вимог. На цій основі розроблено модель системи, що містить діаграму варіантів використання та функціональні можливості системи.

3.4.1. Діаграма використання: функції системи

Діаграма варіантів використання має на меті надати ілюстрацію системи, що розробляється, і її взаємодії з варіантами використання та акторами. У цьому випадку на схемі представлена одна з квартир житлового будинку проекту у вигляді блоку (рис. 3.14). У блоці варіанти використання показані у вигляді овалів, включаючи їхні назви всередині них, що зображують функціональні можливості, які система має надавати своїм мешканцям. Ця передумова відповідає функціональним вимогам, визначеним раніше, які вказують на те, що квартира має забезпечувати комфортне та безпечне середовище проживання для своїх мешканців.

Крім того, ця умова включає в себе інші функціональні можливості, які також має забезпечити система, щоб полегшити виконання базової умови. За допомогою «зв’язку включення» варіанти використання Sleep, Dine, Live, Cook і Move пов’язані з цією базовою умовою.

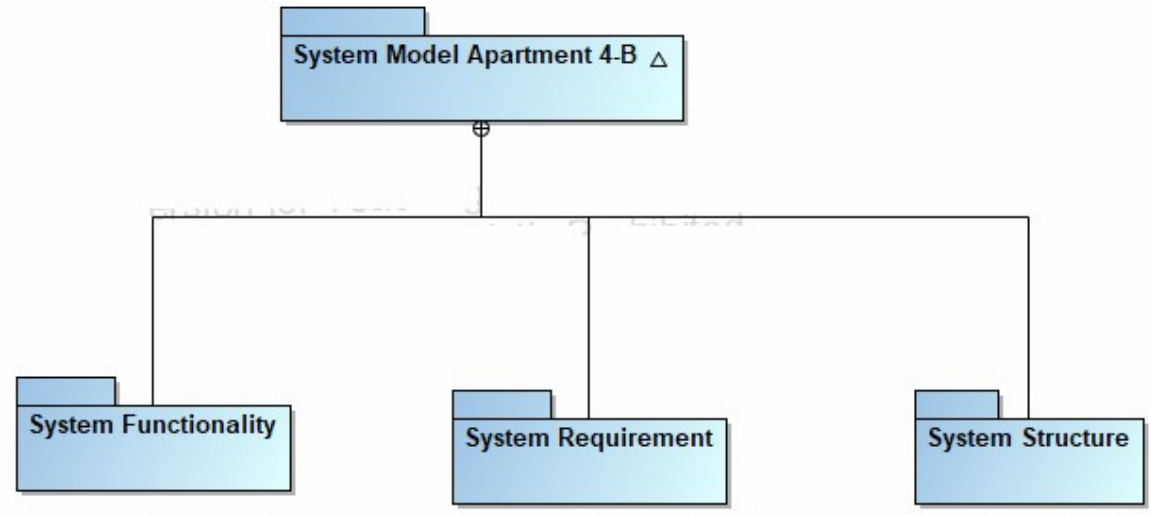


Рис. 3.13. Модель системи, що містить специфікації об’єкту

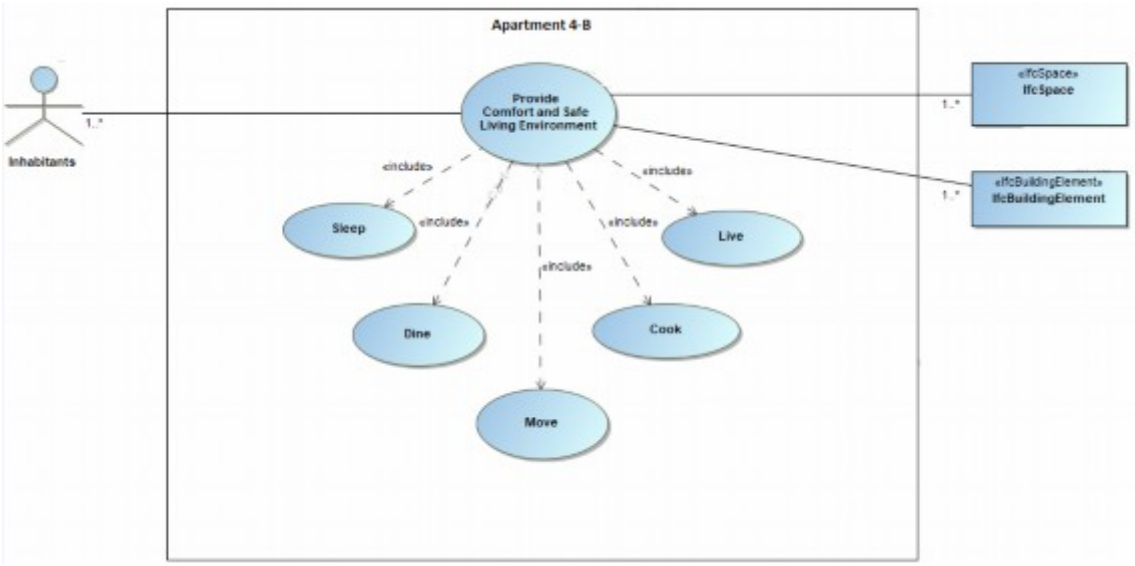


Рис. 3.14. Діаграма варіантів використання, що представляє функціональні можливості об’єкту

За межами системи знаходяться користувачі системи, які описуються акторами, які можуть представляти роль людини, організації або будь-якої зовнішньої системи, яка бере участь у використанні системи. У цьому прикладі актори мають на увазі мешканців і компоненти будівлі, які взаємодіють із системою через стандартні асоціації, пов’язані з варіантами використання. Зв’язок між мешканцями та варіантами використання показує відповідність функціональних можливостей мешканцям для забезпечення злагодженої взаємодії з системою. Множинність [1..*], що відноситься до «один до багатьох», позначає кількість акторів, залучених до випадків використання. З іншого боку, зв’язок між компонентами побудови та варіантами використання ілюструє ключову роль, яку компоненти відіграють у реалізації функцій системи. Множинність [1..*], що відноситься до «один до багатьох», позначає кількість компонентів, залучених у сценарії використання.

3.4.2. Діаграма вимог: системні вимоги

Діаграма вимог графічно зображує ієрархію вимог, що визначають специфікацію системи або компонента, як показано на рисунку 3.15. Після

того, як вимоги визначені, їх можна зафіксувати в стереотипі вимоги SysML «requirement» і пов’язати з іншими вимогами або елементами. Як видно на малюнку, кожна вимога включає попередньо визначені властивості для унікального ідентифікатора та для текстового рядка, що вказує на можливість або умову, якій має задовольняти функція, яку виконує система, або умову продуктивності, яку система повинна досягти. На основі зв’язку обмеження вимоги відповідним чином пов’язані одна з одною, утворюючи ієрархію вимог, що визначає систему на різних абстракціях.

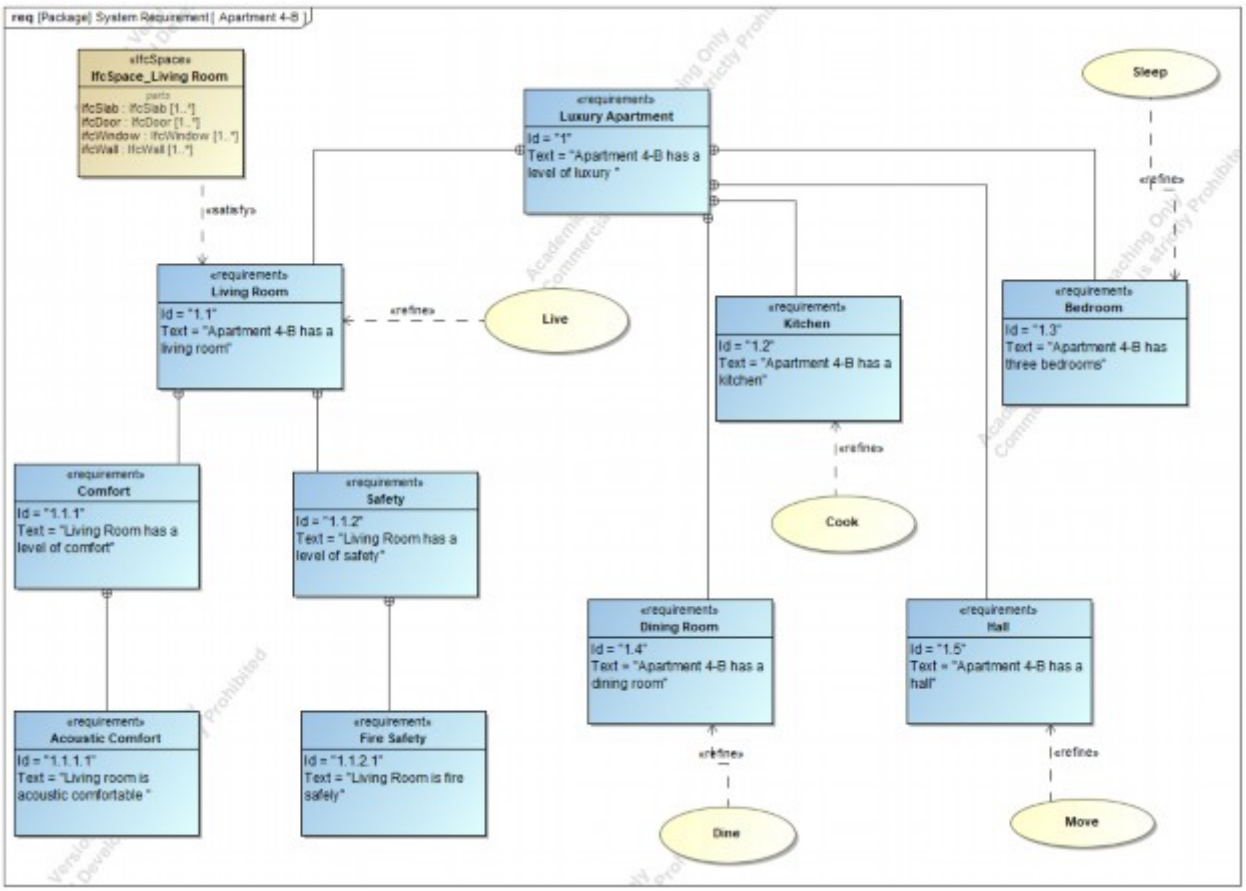


Рис. 3.15. Діаграма вимог, яка представляє системні вимоги

На найвищому рівні ієрархії розташована вимога "Розкішна квартира", яка визначає, що "Квартира 4-Б має рівень розкоші", посилаючись на систему, що розглядається, як показано на діаграмі випадків використання (рис. 3.14).

На другому рівні ієрархії слідують вимоги "Вітальня", "Їдальня", "Кухня", "Спальня" та "Передпокій", що вказують на бажані компоненти в Квартирі 4-Б. Крім того, вони також уточнюють випадки використання, зазначені на діаграмі випадків використання, за допомогою "відношення уточнення". Зосереджуючись на вітальні як об'єкті інтересу, ця вимога пов'язана з вимогами "комфорт" і "безпека", що стосуються необхідного комфортного та безпечного життєвого середовища, яке квартира повинна забезпечити своїм мешканцям. Мета полягає в тому, щоб вітальня була акустично комфортною та пожежно безпечною. Крім того, та сама вимога пов'язана з підсистемою "Вітальня", яка походить з діаграми визначення блоків, що представляє структуру Квартири 4-Б (рис. 3.18). За допомогою "відношення задоволення" підсистема задовольняє вимогу, відповідаючи можливості або умові, яку вимагає вимога. Нарешті, результати можуть бути відображені в матриці задоволення вимог та карті містичності вимог для кращого розуміння та огляду вимог (рисунки 3.16 та 3.17).

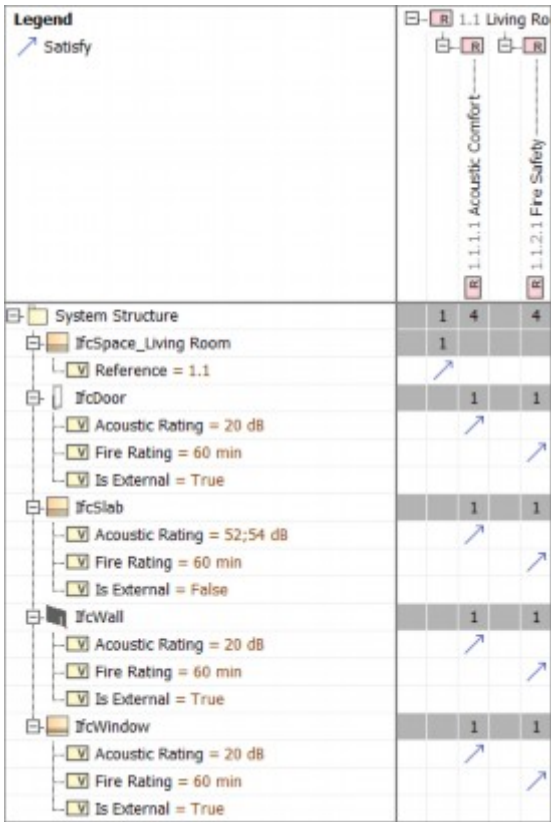


Рис. 3.16. Матриця вимог (SRM)

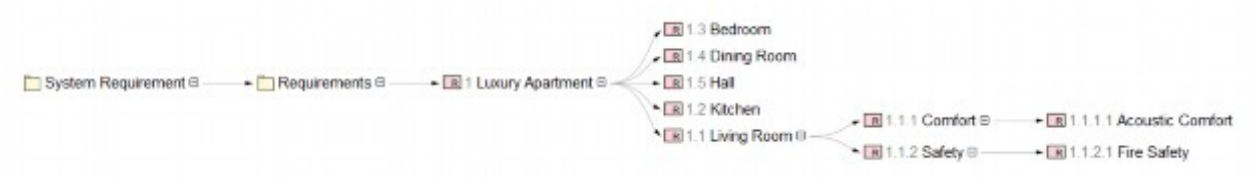


Рис. 3.17. Карта обмеження вимог (RCM)

Переходячи до наступного завдання, а саме перевірки проекту, прототип, можна використовувати для перетворення BIM-моделі в SysML (MOF-файл). Перед виконанням цього перетворення BIM-модель, виражену ifcEXPRESS, має бути перетворена в стандарт ifcOWL для кращої інтеграції даних і взаємодії.

Згодом трансформація з IFC на SysML може бути виконана за допомогою прототипу. Як визначено раніше, прототип обмежений вилученням одного випадкового дверного об’єкта, включаючи дві його ключові властивості. Передбачається, що об’єкти IfcSpace_Living Room, IfcSlab, IfcWindow та IfcWall вже витягнуті з BIM-моделі, при цьому об’єкт IfcDoor пропускається для завершення підсистеми перевірки проекту. Після виконання перетворення MOF-файл імпортується в інструмент Cameo Systems Modeler, у якому підсистема додатково компонується та перевіряється на відповідність вимогам. Результат цього перетворення зображено на рисунку 3.18, де блок, позначений синім кольором, представляє об’єкт IfcDoor_2039251 і його властивості Acoustic Rating=29 і Fire Rating=60, що завершує підсистему в BDD. Відповідно, цю підсистему можна порівняти та перевірити з підсистемою, визначеною раніше, щоб переконатися, що обидві однаково узгоджені, і так далі задовольняють і перевіряють вимоги.

Однак, порівнюючи діаграми, рисунок 3.18 відрізняється в деяких аспектах від оригінального. Причина цього відхилення полягає в тому, що рисунок 3.18 позначає конструкцію згідно з проектом. У цьому порівнянні специфікація вказує на те, що у IfcSpace_Living Room забагато зовнішніх дверей (кратність 1..*), тоді як у дизайні це стали одні двері. Незважаючи на

цю різницю, дизайн відповідає специфікації, оскільки кімната забезпечена зовнішніми дверима для виходу на балкон. Іншу різницю можна побачити щодо акустичного рейтингу, зазначеного для IfcDoor, який вказує на значення 20 дБ мінімум відповідно до будівельних норм. Конструкція показує значення 29, що представляє вище значення, ніж специфікація, що відповідає цій умові. Як останнє, варіація у зв’язку між IfcWall та IfcDoor_2039251. Специфікація містить відношення, засноване на еталонній композиції через припущення, що IfcDoor є частиною IfcWall через свій отвір. Що не так згідно з дизайном. Конструкція зображує навігаційну асоціацію, що посилається на встановлення дверей до стіни через порт, як показано на рисунку 3.18.

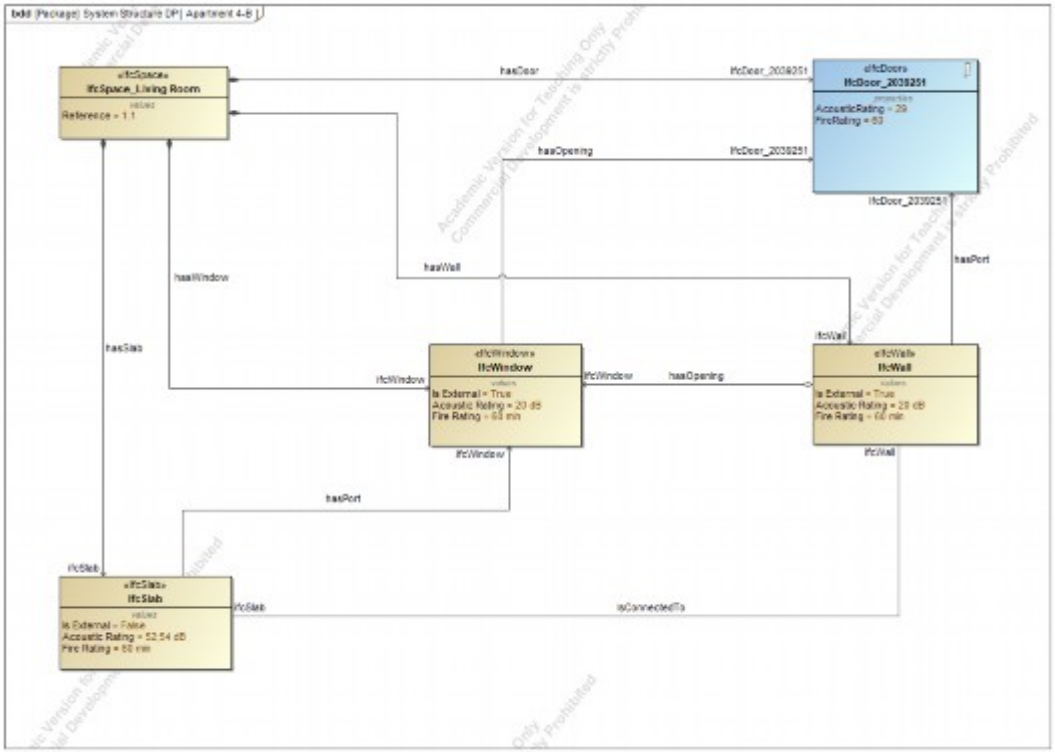


Рис. 3.18. Блок-схема, що представляє структуру підсистеми квартири 4-B на основі моделі проектування

Тим не менш, відхилення проекту від специфікації в цьому прикладі не впливає безпосередньо на виконання системних вимог. Тим фактом, що зв'язок між цими об'єктами непорушно існує, незважаючи на різницю в

асоціаціях. З іншого боку, це відхилення може вплинути на вимоги щодо архітектурних та естетичних умов житлового будинку, що не входить до сфери валідації.

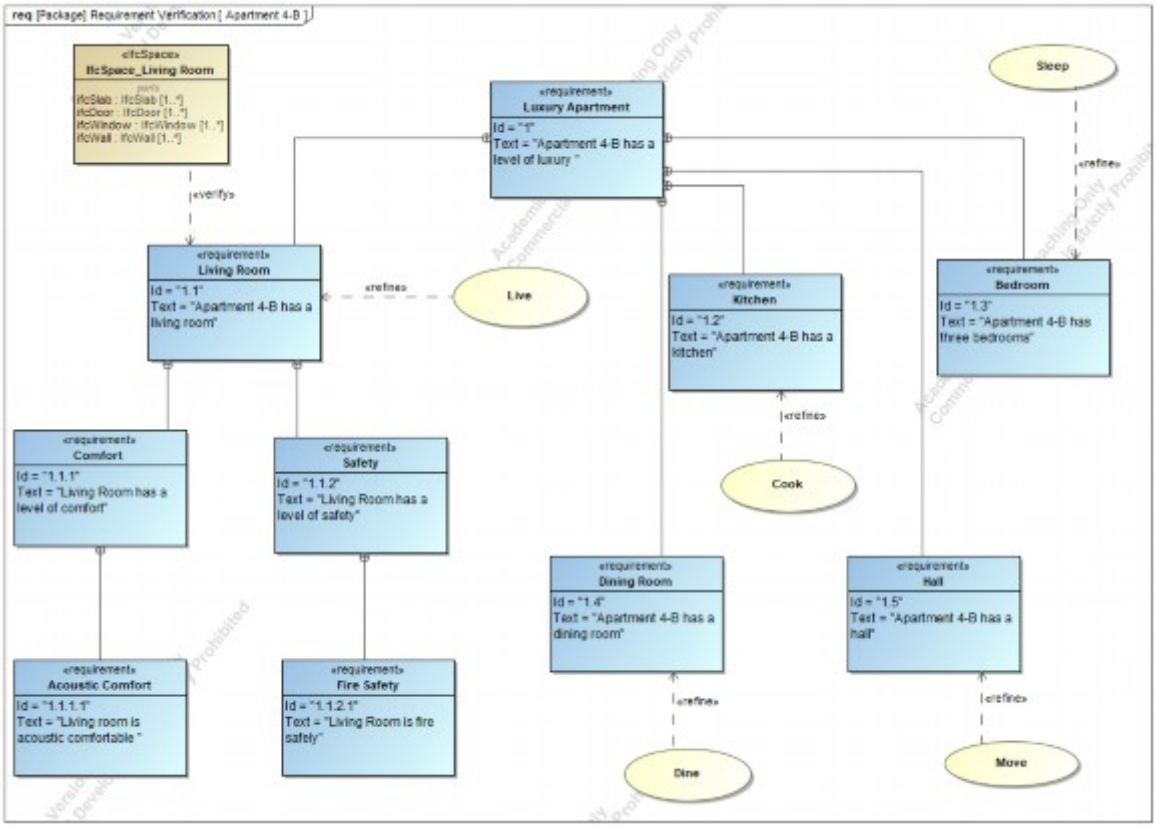


Рис. 3.19. Діаграма вимог, яка представляє системні вимоги об’єкту

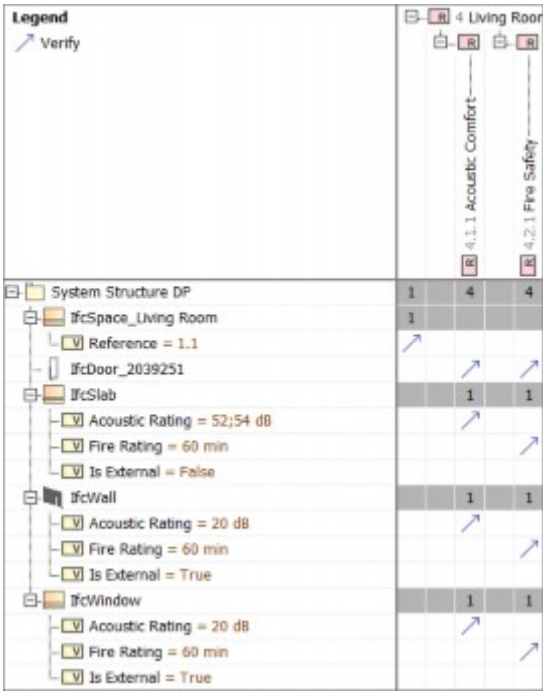


Рис. 3.20. Матриця вимог (SRM)

У підсумку можна побачити, що дизайн відповідає вимогам, зазначеним раніше на етапі інструктажу. Відношення satisfy між вимогами та IfcSpace_Living Room можна замінити зв’язком verify (рис. 3.19). Відповідно, результати можуть бути зафіксовані в матриці вимог перевірки (рис. 3.20) для кращої візуалізації та узгодженості їх складання нової моделі системи, яка включає будівельні та проектні специфікації для продовження етапу будівництва.

Висновки до розділу

Третій розділ присвячений розробці методології специфікації та перевірки моделей систем у високо інтегрованих проєктах. У цьому розділі описано процеси дизайну, реалізації та перевірки моделей з акцентом на забезпечення їхньої інтегрованості та функціональності.

Розробка дизайну моделі включає забезпечення інтероперабельності, яка дозволяє різним компонентам та підсистемам взаємодіяти у межах єдиної моделі. Процеси інтеграції відіграють ключову роль у поєднанні окремих частин системи в єдину функціональну структуру.

Реалізація моделі потребує створення спеціалізованих профілів SysML для врахування особливостей конкретного проєкту. Це дозволяє гнучко адаптувати інструменти моделювання під специфічні вимоги та структури систем. Процеси трансформації моделі описують етапи перетворення даних, необхідні для переведення інформації з однієї форми в іншу у рамках моделі. Це дозволяє зберегти цілісність даних та забезпечити узгодженість під час переходу між різними етапами розробки та інтеграції системи.

Перевірка моделі системи у високо інтегрованих проєктах є критично важливим етапом для виявлення можливих невідповідностей та забезпечення надійності системи. Діаграми використання та вимог допомагають чітко окреслити функції системи та її технічні вимоги, що дозволяє ефективно контролювати процес розробки та інтеграції.

ВИСНОВКИ

В магістерській роботі досліджено процеси перевірки моделей у високо інтегрованих проєктах з використанням підходів системної інженерії, зокрема моделювання на основі системної моделі (MBSE) та інструментарію SysML. Високо інтегровані проєкти характеризуються високим рівнем складності через велику кількість взаємопов'язаних компонентів і процесів. Для забезпечення узгодженості та надійності таких систем, критично важливим є застосування інтегрованих підходів до управління проєктами та використання моделей як основного інструменту для опису систем.

MBSE підхід визначає модель системи як первинний артефакт, що дозволяє формалізувати процеси розробки, інтеграції та перевірки систем. Цей підхід сприяє ефективній роботі з високо інтегрованими системами, забезпечуючи структуроване представлення архітектури системи на всіх етапах життєвого циклу.

Використання SysML як основної мови моделювання дозволяє представляти різні аспекти системи: функціональні, поведінкові та структурні. Множина діаграм SysML, таких як діаграми варіантів використання, вимог, блоків (модулів), дають змогу точно окреслити структуру системи, її вимоги та функціональність. Інтеграція SysML у середовище розробки систем через процеси трансформації моделі дозволяє забезпечити безперервність обробки даних і узгодженість між етапами розробки. Створення спеціалізованих профілів SysML для кожного конкретного проєкту сприяє гнучкості та адаптивності до специфічних вимог галузі.

У роботі також розглянуто питання міжопераційності та побудови великих відкритих моделей, що є особливо важливим для складних багатокомпонентних систем. Використання чотирьохрівневої архітектури класів та ієрархії успадкування полегшує структурування та моделювання даних у межах таких систем.

Застосування онтологічної веб-мови (OWL) та семантичних веб-технологій дозволяє інтегрувати знання та забезпечити автоматизацію процесів моделювання. Це відкриває нові можливості для побудови систем, здатних до автоматизованого оброблення та аналізу даних.

Загалом, дослідження підтверджує ефективність використання моделей систем для забезпечення узгодженості, інтеграції та перевірки складних високо інтегрованих проєктів, зокрема через використання SysML та супутніх технологій.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Acaroglu, L. (2017, September 7). Tools Of A System Thinker. Retrieved from Disrupt Design: https://medium.com/disruptive-design/tools-for-systems-thinkers-the-6-fundamental-concepts-of-systems-thinking-379cdac3dc6a
2. Akio. (2017, May 2). Leveraging the Full Value of BIM's Interoperability Potential. Retrieved from Dassualt Systemes: https://blogs.3ds.com/perspectives/leveraging-full-value-bims-interoperability-potential-2/
3. Alsharif, M. (2013, December 10). Semantic Web Core Technlogies. Retrieved from McKelvey School of Engineering: https://www.cse.wustl.edu/~jain/cse570-13/ftp/semantic/index.html
4. Barosan, I. (2017, 11 23). 7M900: Fundamentals of Building Information Modeling. Retrieved from Canvas: https://canvas.tue.nl/courses/4740/files/folder/Week1
5. Baundains, P., Bischop, S., Duffour, P., Marjanovic-Halburd, L., Psarra, S., & Spararu, C. (2014). A systems paradigm for integrated building design. Intelligent Building International(6), 201-214. doi:https://doi.org/10.1080/17508975.2014.935696
6. Beetz, J. (2009). Facilitating Distributed collaboration in the AEC/FM sector using Semantic Web Technologies. Eindhoven: Eindhoven University of Technology. doi:http://dx.doi.org/10.6100/IR652808
7. Beetz, J., Leeuwen van, J., & Vries de, B. (2009). IfcOWL: A Case of Transforming EXPRESS Schemas into Ontologies. Artificial Intelligence for Engineering Design, Analysis and Manufacturing(23), 89-101. doi:https://doi.org/10.1017/S0890060409000122
8. Beetz, J., Leeuwen, J. P., & Vries de, B. (2007). RDF-Based Distributed Part Specifications for the Facilitation of Service-Based Architectures. Eindhoven University of Technology: Design Systems group, 183-188.

Retrieved from https://www.researchgate.net/publication/250869030_RDF-Based_Distributed_Functional_Part_Specifications_for_the_Facilitation_of_Service-Based_Architectures

9. Berners-Lee, T., Hendler, J., & Lassila, O. (2001, May). The Semantic Web: A New form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Retrieved from https://www-sop.inria.fr/acacia/cours/essi2006/Scientific%20American_%20Feature%20Article_%20The%20Semantic%20Web_%20May%202001.pdf
10. Baier, C., & Katoen, J.-P. (2008). Principles of Model Checking. MIT Press.
11. Clarke, E. M., Grumberg, O., & Peled, D. A. (1999). Model Checking. MIT Press.
12. Harel, D., & Gery, E. (1997). Executable Object Modeling with Statecharts. IEEE Computer Society Press.
13. van Lamsweerde, A. (2009). Requirements Engineering: From System Goals to UML Models to Software Specifications. Wiley.
14. Jackson, D. (2012). Software Abstractions: Logic, Language, and Analysis. MIT Press.
15. Leveson, N. G. (2011). Engineering a Safer World: Systems Thinking Applied to Safety. MIT Press.
16. Broy, M., & Stølen, K. (2001). Specification and Development of Interactive Systems: Focus on Streams, Interfaces, and Refinement. Springer.
17. Patterson, D. A., & Hennessy, J. L. (2013). Computer Organization and Design: The Hardware/Software Interface. Morgan Kaufmann.
18. Ghezzi, C., Jazayeri, M., & Mandrioli, D. (2002). Fundamentals of Software Engineering. Prentice Hall.
19. Wing, J. M. (1990). A Specifier's Introduction to Formal Methods. IEEE Computer.
20. Harel, D. (1987). Statecharts: A Visual Formalism for Complex Systems. Science of Computer Programming.

21. Hoare, C. A. R. (1978). Communicating Sequential Processes. Communications of the ACM.
22. Lamport, L. (1978). Time, Clocks, and the Ordering of Events in a Distributed System. Communications of the ACM.
23. Lynch, N. A., & Tuttle, M. R. (1989). An Introduction to Input/Output Automata. CWI Quarterly.
24. Pnueli, A. (1977). The Temporal Logic of Programs. 18th Annual Symposium on Foundations of Computer Science.
25. Rushby, J. (1999). Model Checking and the Analysis of Critical Systems. In Proceedings of the NATO Advanced Study Institute on Model Checking and Software Verification.
26. Manna, Z., & Pnueli, A. (1992). The Temporal Logic of Reactive and Concurrent Systems. Springer.
27. Burch, J. R., Clarke, E. M., McMillan, K. L., Dill, D. L., & Hwang, L. J. (1992). Symbolic Model Checking: 10^20 States and Beyond. Information and Computation.
28. Holzmann, G. J. (2004). The Spin Model Checker: Primer and Reference Manual. Addison-Wesley.
29. De Moura, L., & Bjørner, N. (2008). Z3: An Efficient SMT Solver. Tools and Algorithms for the Construction and Analysis of Systems.
30. Boehm, B. W. (1988). A Spiral Model of Software Development and Enhancement. ACM SIGSOFT Software Engineering Notes.
31. Felderer, M., & Ramler, R. (2014). Integrating Risk-Based Testing in Industrial Test Processes. In Proceedings of the 2014 International Conference on Software Testing, Verification, and Validation Workshops (pp. 17-22).
32. Kumar, R., & Shankar, P. (2006). Verification of Real-Time Systems Using Timed Automata-Based Approaches: An Overview. Journal of Real-Time Systems.

33. Alur, R. (1999). Timed Automata. In Proceedings of the NATO Advanced Study Institute on Verification of Digital and Hybrid Systems.
34. Dong, J. S., & Zhu, H. (2013). Model Checking High-Level Petri Nets Using Logic Programming. The Computer Journal.
35. Jensen, K. (1997). Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Springer.
36. Hoeve ten, G. (2018). Improving Collaboration between Client and Contractor in Integrated Contracts in the Dutch Construction Sector. Delft : Delft University of Technology. Retrieved from http://resolver.tudelft.nl/uuid:7fbe5a20-53b7-4237-be67-67a6ea793502
37. Ilhan, B. (2014). An IFC-Based Framework for sustainable construction. Instabul, Turkey: Instanbul Technical University. Retrieved from https://polen.itu.edu.tr/bitstream/11527/14099/1/10048340.pdf
38. Jallow, A. K., Demian, P., Baldwin, A. N., & Chimay, A. (2014). An empirical study of the complexity of requirements management in construction projects. Engineering, Construction and Architectural Management(21), 505-531. doi:DOI 10.1108/ECAM-09-2013-0084
39. Jenkins, J. S., & Rouquette, F. N. (2012). Semantically-Rigorous Systems Engineering Modeling Using SysML and OWL. 5th International Workshop on Systems & Concurrent Engineering for Space Applications (pp. 1-8). Lisbon, Portugal: Jet Propulsion Laboratory, National Aeronautics and Space Administration. Retrieved from http://hdl.handle.net/2014/43338
40. Rushby, J. (1993). Formal Methods and the Certification of Critical Systems. In SRI International Computer Science Laboratory Technical Report.
41. Hall, A. (1990). Seven Myths of Formal Methods. IEEE Software.
42. Henzinger, T. A., & Manna, Z. (1991). The Temporal Framework for Concurrency. In Theoretical Computer Science.
43. Garlan, D., & Shaw, M. (1993). An Introduction to Software Architecture. In Advances in Software Engineering and Knowledge Engineering.

44. Borrmann, A., König, M., Koch, C., & Beetz, J. (2018). Building Information Modeling: Technology Foundations and Industry Practice. Germany: Springer International Publishing AG. Retrieved from https://link.springer.com/book/10.1007%2F978-3-319-92862-3
45. BuildingSMART. (1996-2006). IFC4 Add2 - Addendum 2. Retrieved from BuildingSMART International: http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/
46. Cardoso, J. (2007). Semantic Web Services: Theory, Tools, and Applications. Madeira, Portugal: Informantion Science Reference.
47. Cheng, J., Kumar, B., & Law, H. K. (2002). A Question Answering System for Project Management Applications. Advanced Engineering Informatics(16), 227-289. doi:https://doi.org/10.1016/S1474-0346(03)00014-4
48. Deshpande, A., Azhar, S., & Amireddy, S. (2014). A Framework for a BIM-based Knowledge Management System. Procedia Engineering(85), 113-122. doi:https://doi.org/10.1016/j.proeng.2014.10.535
49. Diederendirrix. (2019). Sixty 5. Retrieved from Diederendirrix: https://www.diederendirrix.nl/nl/projecten/sixty5/
50. Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2011). BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors (Second ed.). New Jersey: John Wiley & Sons, Inc. .
51. Friendenthal, S., Moore, A., & Steiner, R. (2015). A Practical Guide to SysML (Vol. Third edition). Waltham , USA: Elsevier Inc. doi:https://doi.org/10.1016/B978-0-12-800202-5.00003-5
52. Froese, T. M. (2010). The impact of emerging information technology on project management for construction. Automation in Construction(19), 531-538. doi:https://doi.org/10.1016/j.autcon.2009.11.004