

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ЕРГОНОМІЧНОГО ПРОЄКТУВАННЯ ЛЮДИНО-МАШИННИХ
ІНТЕРФЕЙСІВ АСУТП
RESEARCH AND SOFTWARE IMPLEMENTATION OF ERGONOMIC
DESIGN METHODS FOR HUMAN-MACHINE INTERFACES IN AUTOMATED
CONTROL SYSTEMS

Виконав(ла): студент(ка) 2 курсу, групи КНТ-123м

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

КАМІНСЬКА Ж.К.

(ПРІЗВИЩЕ та ініціали)

Керівник СЕРДЮК С.М.

(ПРІЗВИЩЕ та ініціали)

Рецензент ПОЛЯКОВ М.О.

(ПРІЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

КАМІНСЬКОЇ Жанни Костянтинівни

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів
ергономічного проєктування людино-машинних інтерфейсів АСУТП. Research
and Software Implementation of Ergonomic Design Methods for Human-machine
Interfaces in Automated Control Systems

керівник проєкту (роботи) к.т.н., доцент, СЕРДЮК Сергій Микитович,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “07” жовтня 2024 року № 398

2. Строк подання студентом проєкту (роботи) 5 грудня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне
завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно
розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Розробка
багаторівневої онтології. 4. Експлуатація, тестування та експериментальне
дослідження розробленої онтології. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	СЕРДЮК С.М., доцент	підпис	підпис
Нормоконтроль	ДЕЙНЕГА Л.Ю., ст.викладач	підпис	підпис

7. Дата видачі завдання “ 28 ” лютого 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1-2 тижні	Завдання, ТЗ
2	Аналіз предметної області	3-4 тижні	Розділ 1
3	Розробка та удосконалення методів, моделей й алгоритмів вирішення задачі.	5-6 тижні	Розділ 2
4	Розробка архітектури програми.	7-8 тижні	Розділ 3
5	Розробка програми.	9 тижні	Розділ 4
8	Оформлення пояснювальної записки та документів до неї.	10 тиждень	
9	Нормоконтроль та рецензування.	11 тиждень	
10	Захист роботи.	12 тиждень	

Студент(ка)

(підпис) Жанна КАМІНСЬКА
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Сергій СЕРДЮК
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
100с., 6 табл., 38 рис., 1 дод., 20 джерела.

ДІЯЛЬНІСТЬ ПРОЄКТУВАЛЬНИКА, ТЕХНОЛОГІЧНИЙ ПРОЦЕС,
ЛЮДИНО-МАШИННИЙ ІНТЕРФЕЙС, БАЗА ЗНАНЬ, ОНТОЛОГІЯ,
ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ, SCADA.

Об'єкт дослідження – процес проєктування людино-машинних
інтерфейсів (ЛМІ) SCADA-систем.

Предмет дослідження – методи проєктування ЛМІ SCADA-систем.

Мета роботи – підвищення ергономічності ЛМІ SCADA-систем на основі
нової інформаційної технології проєктування.

Матеріали, методи та технічні засоби: методи ергономіки; методи
представлення знань; метаонтології та онтології предметних областей; редактор
онтологій Protege; мова запитів SPARQL; операційна система Windows 10 або
вище.

Результати. Вперше запропоновано інформаційну технологію
інтелектуальної підтримки процесу ергономічного проєктування ЛМІ, що
розробляються за допомогою SCADA-систем. Проведено аналіз структури ЛМІ
та виявлено властивості його типових елементів, які можуть бути представлені у
вигляді відомих категорій кодування інформаційних моделей.

Висновки. Використання розробленого програмного забезпечення
дозволяє підвищити ергономічність ЛМІ SCADA-систем та обізнаність
операторів щодо поточної виробничої ситуації.

Галузь використання – проєктування ЛМІ SCADA-систем.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 100 pages, 6 tables, 38 figures, 1 appendix, 20 sources.

DESIGNER ACTIVITY, TECHNOLOGICAL PROCESS, HUMAN-MACHINE INTERFACE, KNOWLEDGE BASE, ONTOLOGY, INFORMATION TECHNOLOGY, SCADA.

The object of the study is the process of designing human-machine interfaces (HMI) of SCADA systems.

Subject of the study is the methods of designing HMI of SCADA systems.

The purpose of the work is to improve the ergonomics of HMI of SCADA systems based on a new information technology for design.

Materials, methods, and tools: ergonomic methods; knowledge representation methods; meta-ontologies and ontologies of subject domains; Protege ontology editor; SPARQL query language; Windows 10 operating system or higher.

Results. An information technology for intelligent support of the ergonomic design process for HMI developed with SCADA systems is proposed for the first time. An analysis of HMI structure was conducted, identifying the properties of its typical elements, which can be represented through well-known categories for coding information models.

Conclusions. The use of the developed software improves the ergonomics of HMI for SCADA systems and enhances operator awareness of the current production situation.

The field of use is the design of HMI of SCADA systems.

ЗМІСТ

Перелік скорочень та умовних позначок.....	8
Вступ.....	9
1 Аналіз предметної області	11
1.1 Аналіз проблем ергономічного проєктування людино-машинних інтерфейсів АСУТП	11
1.2 Аналіз онтологій, та їхня роль в інтеграції знань для людино-орієнтованого проєктування інтерфейсів SCADA.....	15
1.3 Постановка задач дослідження.....	19
1.4 Висновки до першого розділу	21
2 Матеріали і методи	23
2.1 Розробка концептуальної моделі предметної області «Проєктування ЛМІ АСУТП»	23
2.2 Розробка ієрархічної структури онтології.....	29
2.3 Ерготехнічна система та її структура	31
2.4 Висновки до другого розділу.....	33
3 Розробка багаторівневої онтології	34
3.1 Розробка ієрархії класів та зв'язків метаонтології	34
3.2 Розробка онтології «Технологічний процес».....	37
3.3 Розробка онтології «Ерготехнічна система».....	42
3.4 Розробка онтології «Людино-машинний інтерфейс»	47
3.5 Вирівнювання онтології	50
3.5.1 Вирівнювання онтології «Технологічний процес» з DUL... ..	51
3.5.2 Вирівнювання онтології «Ерготехнічна система» з DUL....	57
3.6 Висновки до третього розділу	61
4 Експлуатація, тестування та експериментальне дослідження розробленої онтології.....	62
4.1 Приклад онтології технологічного процесу виробництва газобетону	62

4.2 Приклад реалізації аварійної ситуації.....	67
4.3 Реалізація технології проектування ЛМІ.....	71
4.4 Приклад реалізація кодування інформації у зоні	75
4.5 Висновки до четвертого розділу	84
Висновки	85
Перелік джерел посилання	87
Додаток А. - Копії слайдів презентації	90

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

DUL	– DOLCE + DNS UltraLite;
HMI	– Human Mashine Interface;
SCADA	– Supervisory Control and Data Acquisition;
АСУТП	– автоматизована системи управління технологічним процесом;
ВКВ	– вапняно-кремнеземисто в'язучий;
ЕТС	– ерготехнічна система;
ЛМІ	– людино-машинний інтерфейс;
ЛМС	– людино-машинна система;
ОКУ	– об'єкти контролю та управління
ТП	– технологічний процес

ВСТУП

В умовах сучасної економіки одним із ключових завдань стає підвищення ефективності автоматизованих систем управління технологічними процесами (АСУТП), в яких людина-оператор, як і раніше, залишається важливим елементом. Незважаючи на високий ступінь автоматизації, людина не виключена з контурів управління технологічними процесами і найчастіше є вразливою ланкою. Статистика, що принаймні 70% аварій на виробничих підприємствах відбуваються через людську помилку, підтверджується в багатьох дослідженнях [1]. Відповідно до даних European Agency for Safety and Health at Work, цей показник ще вищий - 80-90% [2]. Ці помилки також значущі у сферах, пов'язаних із введенням даних та управлінням складними системами, що негативно позначається на безпеці, продуктивності та точності виконання завдань.

SCADA-системи, що широко застосовуються в промисловості, відіграють ключову роль в управлінні та моніторингу технологічних процесів. Однак, незважаючи на їхню важливість, проєктування людино-машинних інтерфейсів (ЛМІ) SCADA-систем часто не враховує ергономічні та психологічні аспекти взаємодії. Це призводить до підвищення когнітивного навантаження операторів, збільшення ймовірності помилок та зниження ефективності роботи.

В даний час інструментальні SCADA-системи надають широкий спектр можливостей для створення інтерфейсів, але в більшості випадків вони не оснащені інтелектуальними модулями, які можуть допомагати проєктувальникам ЛМІ з точки зору урахування вимог інженерної психології та ергономіки. Розробка інформаційної технології для підтримки цього процесу могла б суттєво покращити процес створення людино-машинних інтерфейсів, зменшуючи вплив людського фактора та підвищуючи безпеку та надійність роботи систем.

Таким чином, актуальним є дослідження та програмна реалізація методів ергономічного проєктування людино-машинних інтерфейсів АСУТП.

Об'єктом дослідження даної дипломної кваліфікаційної роботи магістра є процес проєктування людино-машинних інтерфейсів SCADA-систем.

Предмет дослідження – методи проєктування ЛМІ SCADA-систем.

Мета роботи полягає у створенні інформаційної технології інтелектуальної підтримки проєктувальників SCADA-інтерфейсів, що дозволить підвищити якість їхньої розробки та мінімізувати кількість помилок операторів. Впровадження такої технології може суттєво покращити взаємодію людини із системою та сприяти створенню більш ефективних та безпечних виробничих процесів.

Для досягнення поставленої мети у даній роботі необхідно вирішити такі задачі:

- проаналізувати проблеми ергономічного проєктування ЛМІ АСУТП;
- проаналізувати сучасні онтології, та їх роль в інтеграції знань для людино-орієнтованого проєктування інтерфейсів SCADA;
- розробити концептуальну модель предметної області «Проєктування ЛМІ АСУТП»;
- розробити предметні онтології «Технологічний процес», «Людино-машинний інтерфейс» та «Ерготехнічна система» на базі метаонтології DUL;
- провести вирівнювання розроблених онтологій;
- розробити систему запитів до онтології;
- перевірити адекватність розроблених моделей на прикладі проєктування ЛМІ для АСУТП виробництва газобетону.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз проблем ергономічного проектування людино-машинних інтерфейсів АСУТП

Диспетчерське управління та збір даних (SCADA - Supervisory Control And Data Acquisition) на сьогодні є основним методом автоматизованого управління складними динамічними системами в критичних з точки зору безпеки і надійності областях. Це пов'язано зі значним прогресом в галузі обчислювальної техніки, програмного забезпечення та телекомунікацій, що збільшує можливості і розширює сферу застосування автоматизованих систем.

У сучасних АСУТП, розроблених із використанням SCADA-систем, робота вимагає від оператора граничної мобілізації своїх можливостей: психологічних, емоційних, вольових. Тому зростає частка помилок у системі «людина-машина» з вини людини та їх ціна.

Одна з причин цієї тенденції - старий традиційний підхід до побудови складних систем моніторингу та управління, тобто орієнтація на застосування новітніх технічних досягнень і недооцінка необхідності побудови ефективного ЛМІ, орієнтованого на людину-оператора.

Якісно спроектовані ЛМІ надають операторам важливі можливості, що підвищують ефективність процесу, безпеку та час безвідмовної роботи. Ключові можливості включають [3]:

- моніторинг у реальному часі. ЛМІ дозволяють операторам контролювати стан процесу у реальному часі. Це включає в себе відображення даних датчиків, сигналів тривоги, виробничих показників, стану обладнання та інших ключових показників ефективності;

- взаємодія з контролерами. Оператори можуть переглядати дані контролерів та надсилати команди, такі як запуск/зупинка циклу, зміна швидкості та інші функції прямого керування;

- візуалізація. Графічні інтерфейси чітко становлять стан процесу за допомогою графіків, діаграм, схем, віджетів візуалізації та інших інструментів;

- управління сигналами тривоги. Сповіщають операторів візуально та звуком про проблеми, які потребують уваги чи втручання, щоб уникнути простоїв. Надійні ЛМІ мають історію та аналіз сигналів тривоги;

- реєстрація даних та трендинг. Можливості Historian дозволяють операторам відстежувати продуктивність з часом та аналізувати тенденції процесу;

- керування доступом користувачів. Для підвищення безпеки забезпечується керування дозволами для різних класів користувачів;

- віддалений моніторинг. Моніторинг та керування процесами з будь-якого підключеного місця замість потреби перебувати на робочому місці на заводі.

Ці основні можливості дозволяють людині-оператору продуктивно та безпечно взаємодіяти зі складними машинами, установками та процесами. Інтерфейси ЛМІ підвищують здатність оператора розуміти те, що відбувається, і застосовувати свої знання при необхідності.

Більшість з існуючих на сьогоднішній день систем розробки ЛМІ є дорогими в експлуатації і вимагають значних апаратних ресурсів. Велика вартість таких систем в першу чергу обумовлена складністю їх впровадження і налагодження, а також необхідністю оновлення програмних та апаратних засобів на термінальних станціях при заміні об'єктів контролю та управління. Це вимагає значних економічних та часових витрат, призводить до зупинки технологічного процесу (ТП), наслідком чого є зменшення ефективності виробництва.

Для розробки сучасних АСУТП використовують інструментальні SCADA-системи, наприклад: WinCC (Siemens); Intouch (Wonderware); Citect (Schneider Electric); TRACE MODE (AdAstra Research Group). Важливою складовою АСУТП є людино-машинний інтерфейс за допомогою якого оператор здійснює керування технологічним процесом.

У складі інструментальних SCADA-систем обов'язковим є наявність графічних редакторів з розвинутою функціональністю. З їх допомогою проєктувальник як у конструкторі складає інтерфейс майбутньої автоматизованої

системи. У якості приклада, на рис. 1.1 та рис. 1.2 наведено приклади (обведено червоним) графічних елементів, з яких проєктувальник будує ЛМІ.

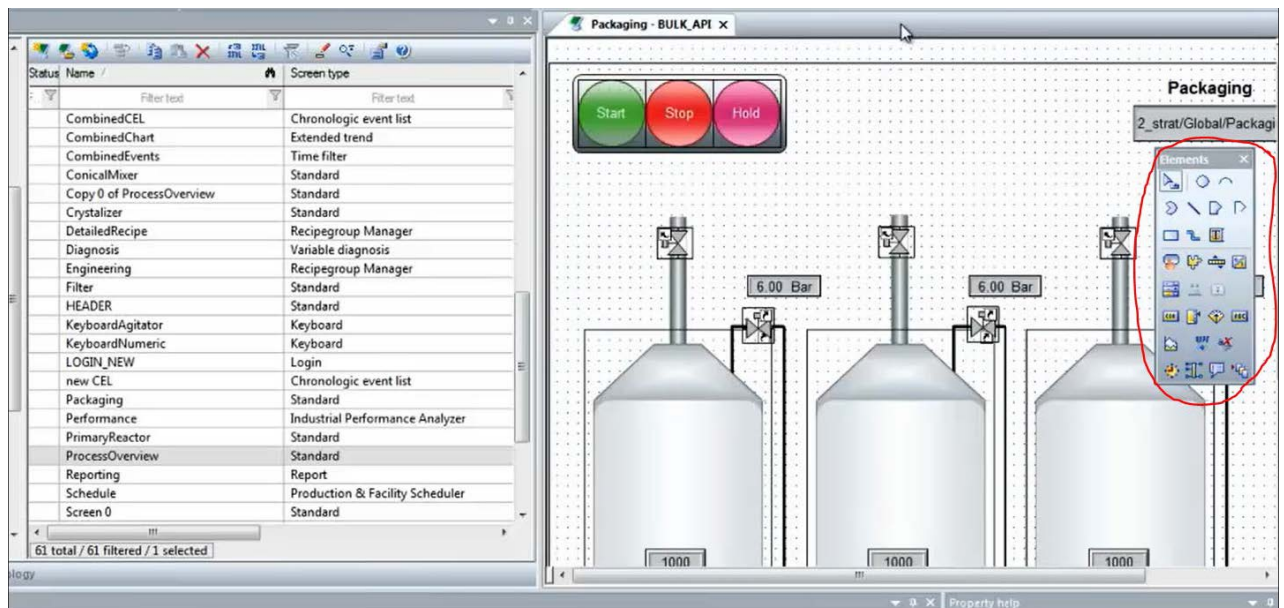


Рисунок 1.1 – Приклад бібліотеки елементів zenon SCADA [4]

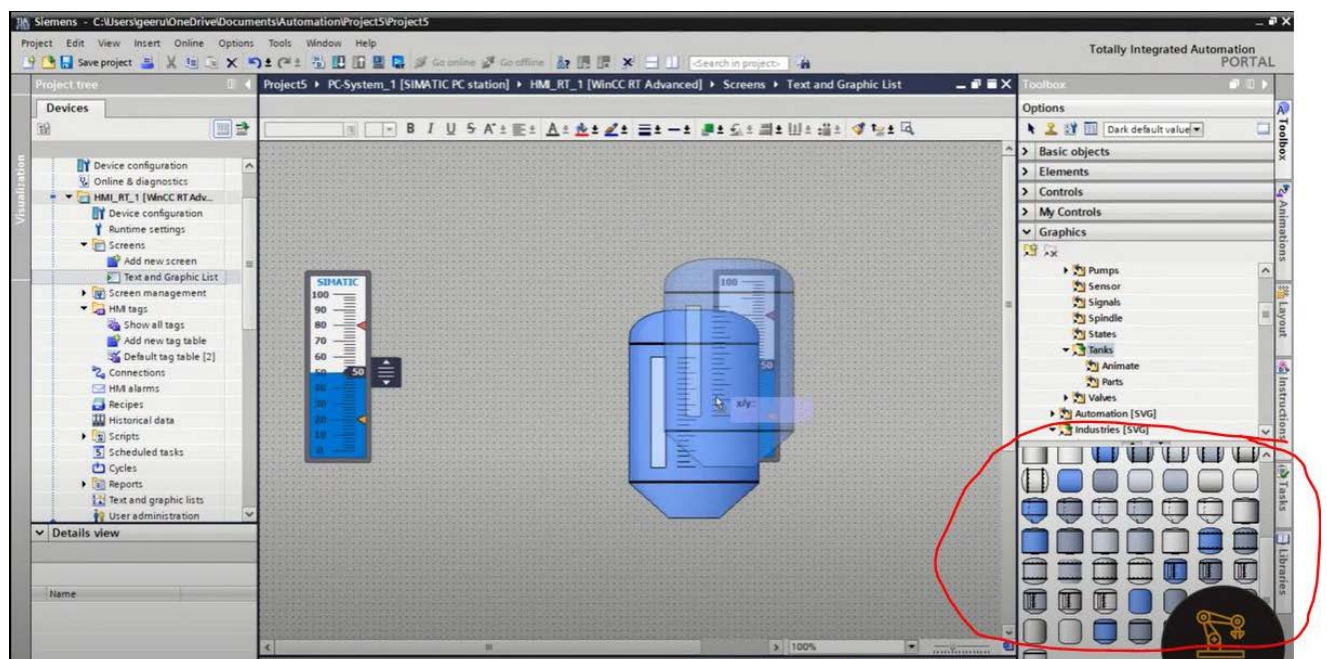


Рисунок 1.2 – Приклад бібліотеки елементів WinCC SCADA [5]

Аналіз сучасних інструментальних SCADA-систем виявив, що жодна з них не має в своєму складі модулів інтелектуальної підтримки процесу ергономічного проєктування ЛМІ.

При такому підході, якість розробленого інтерфейсу (тобто його сумісність з психофізіологічними характеристиками людини-оператора) буде залежати безпосередньо від кваліфікації проєктувальника. Згідно [4], «Інженер може використовувати елементи, створені графічними художниками у зовнішніх програмах з допомогою XAML в редакторі zenon. Потім їх потрібно лише пов'язати з потрібною функцією, і вони готові до використання».

Таким чином, для розробки якісних інтерфейсів, проєктувальнику необхідно знати великий перелік різноманітних нормативних документів, ергономічних та інженерно-психологічних правил та рекомендацій, що враховують особливості та обмеження людини-оператора. Це свідчить про актуальність розробки інноваційних технологій у області ергономічного (людино-орієнтованого) проєктування ЛМІ.

Однак, проєктування інтерфейсів у системах автоматизованого управління (SCADA) є складним завданням, що вимагає інтеграції знань відразу з декількох суміжних дисциплін. Проєктувальникам необхідно враховувати не тільки технологічні процеси, що відображаються та контролюються через інтерфейси, а й специфіку людино-машинної взаємодії, а також принципи ергономіки та людино-орієнтованого дизайну.

У зв'язку з цим виникає необхідність у систематизації та структуруванні цих знань, що дозволить автоматизувати процес їх використання та підвищити ефективність проєктування. Для вирішення цього завдання актуальною стає розробка онтологічної моделі, яка формалізує знання про три ключові предметні області: технологічні процеси, людино-орієнтоване проєктування інтерфейсів та людино-машинні системи. Така модель стане основою нової інформаційної технології, яка забезпечить інтелектуальну підтримку проєктувальників інтерфейсів SCADA систем. Вона дозволить інтегрувати та логічно пов'язувати дані з різних джерел та дисциплін, пропонуючи проєктувальникам структуроване та цілісне уявлення про процес розробки інтерфейсів.

1.2 Аналіз онтологій, та їхня роль в інтеграції знань для людино-орієнтованого проєктування інтерфейсів SCADA

При розробці інтелектуальних систем підтримки проєктувальників ЛМІ для SCADA систем важливо вибрати правильний метод для формалізації знань. Одним з таких методів є використання онтологій, які можуть ефективно описувати складні системи, процеси та взаємодії між людиною та машиною. Однак, враховуючи багаторівневий характер проєктування та різноманітні галузі знань, які необхідно інтегрувати, існує необхідність використання онтологій верхнього та середнього рівня.

Онтології верхнього рівня є високоабстрактними структурами, призначеними для опису загальних концепцій, таких як агенти, події, процеси, ролі та відносини. Приклад таких онтологій є DOLCE+DnS Ontology (DUL) і SUMO. Ці онтології створюють універсальні фреймворки для опису сутностей та його зв'язків, що допомагає організувати знання найвищому рівні абстракції. Однак для специфічних завдань, таких як проєктування ЛМІ, їх необхідно доповнити онтологіями середнього рівня, які пропонують більш детальні концепти, що належать до конкретних предметних областей.

Метаонтології, або онтології верхнього рівня, відрізняються один від одного щодо виразності мови уявлення знань, що кодує їх вміст, та щодо онтологічних виборів, припущень та аксіом, які вони приймають [6]. Існує декілька дослідних проєктів, спрямованих на розробку стандартних метаонтологій. Найбільш відомими з них є SUMO, Upper Cys та DOLCE.

Використання метаонтологій обґрунтоване тим, що вони забезпечують єдиний каркас для інтеграції різних галузей знань, що особливо важливо у завданнях, де потрібні дані з таких різнорідних дисциплін, як технологічні процеси, людино-орієнтоване проєктування інтерфейсів та людино-машинні системи. Такі онтології дозволяють уникнути дублювання понять, допомагають забезпечити логічну відповідність між різними концептуальними рівнями та підвищують когерентність даних, структурованих із різних джерел. Таким чином,

метаонтології відіграють ключову роль, створюючи сполучну основу для подальшої детальної формалізації та автоматизації знань про людино-машинні взаємодії.

Нижче наведено аналіз існуючих онтологій верхнього та середнього рівня, що підходять для задач людино-орієнтованого проєктування у SCADA системах. У якості критеріїв порівняння було обрано можливості онтологій по представленню знань щодо: технологічних процесів; людино-орієнтованому проєктуванні інтерфейсів SCADA систем та можливості опису людино-машинних систем. Розглянуто, як кожна онтологія структурує знання та підтримує дані предметні області. Результати порівняльного аналізу представлені у таблиці 1.1.

Таблиця 1.1 – Порівняльний аналіз онтологій

Онтологія	Технологічні процеси	Проектування інтерфейсів SCADA	Людино-машинні системи
1	2	3	4
DOLCE+DnS Ontology (DUL)	Підтримує моделювання процесів через структурні події та агентську взаємодію.	Описує відносини агента, артефакту та цілей, що може бути корисним для опису взаємодій людини з інтерфейсами.	Моделює складні людино-машинні системи через ролі, агенти, артефакти та події.
SUMO (Suggested Upper Merged Ontology)	Має потужну підтримку моделювання фізичних і технологічних процесів, з акцентом на формальні системи і причини зв'язку.	Може частково підтримувати проектування SCADA через моделювання агентів та їх взаємодію із системами.	Підтримує моделювання як фізичних систем, так і агентських взаємодій, включаючи людино-машинні системи.
PROV-O (Provenance Ontology)	Спеціалізується на відстеженні вихідних даних і процесів, може бути корисним для технологічних процесів у контексті даних.	Не призначена для проектування інтерфейсів SCADA, основний акцент на управлінні даними.	Підтримує опис взаємодій між системами та людьми через дані та події.

Продовження таблиці 1.1

1	2	3	4
SSN (Semantic Sensor Network Ontology)	Орієнтована на опис сенсорних мереж та даних від сенсорів, що може бути корисним для технологічних процесів, пов'язаних з контролем та вимірюваннями.	Може бути корисною в контексті інтеграції даних від сенсорів до інтерфейсів SCADA.	Описує сенсорні системи та взаємодії з користувачами через дані та сенсори.
Upper Cys	Здатна описувати високорівневі технологічні процеси, проте потребує специфічних уточнень більш вузької області.	Підтримує високорівневе моделювання взаємодій людини із системами, але потребує налаштування для SCADA.	Підтримує моделювання людино-машинних систем через агентні структури та ролі.
OPMW (Ontology for Provenance and Workflow Management)	Орієнтована на моделювання процесів управління та робочих потоків, але без глибокої технологічної специфікації.	Слабо пристосована для опису інтерфейсів користувача, більше фокусується на процесах.	Можна використовувати для відстеження взаємодій між системою та користувачами через керування робочими процесами.

Згідно з даними таблиці 1.1 можна зробити наступні висновки:

- метаонтології DUL та SUMO найбільш повно охоплюють опис як технологічних процесів, так і людино-машинних систем, включаючи аспекти проєктування інтерфейсів SCADA;
- SSN фокусується на взаємодії із сенсорними системами і може бути корисною для представлення знань SCADA в частині моніторингу;
- OPMW та PROV-O більше орієнтовані на управління процесами та даними, з меншим акцентом на взаємодію людини з інтерфейсами;
- Upper Cys може бути використана для високорівневого моделювання, але потребує налаштування для більш специфічних областей, таких як SCADA.

Остаточний вибір між DUL та SUMO був зроблений на користь DUL з кількох причин, особливо з урахуванням специфічних завдань, пов'язаних з

формальним описом трьох вищеназваних предметних областей. Розглянемо кожну з них.

Для формалізації предметної області «Технологічні процеси»:

- DUL пропонує більш гнучку структуру для опису подій, дій та агентів, що особливо важливо для моделювання складних технологічних процесів. У DUL концепція події та агента добре розвинена через зв'язки з концептами часу та просторових обмежень. Це спрощує опис деталізованих процесів за участю різних агентів, артефактів та цілей;

- SUMO також здатна описувати процеси та каузальні зв'язки, проте акцент зроблений більше на формальних і логічних системах. SUMO більш абстрактна в описі подій і потребує складнішого налаштування для вузькоспеціалізованих областей, таких як технологічні процеси SCADA.

Таким чином, DUL дозволяє більш чітко та деталізовано описувати процеси, де взаємодіють люди та системи, з урахуванням ролі кожного учасника у ланцюжку процесів.

Для формалізації знань з людино-орієнтованого проектування інтерфейсів SCADA:

- DUL безпосередньо фокусується на взаємодії між агентами (людьми), артефактами та цілями. У контексті людино-орієнтованого проектування інтерфейсів SCADA систем ця онтологія пропонує ясні способи моделювання взаємодій людини з інтерфейсами та системами управління, акцентуючи увагу на цілях користувачів та їх взаємодіях з фізичними та цифровими об'єктами;

- SUMO представляє загальні концепції взаємодії агента та системи, але її опис людино-машинних взаємодій менш специфічний для конкретних завдань, таких як проектування інтерфейсів SCADA. SUMO вимагатиме додаткового налаштування для деталізації процесів, пов'язаних з інтерфейсами користувача.

Таким чином, DUL пропонує більш конкретні та деталізовані засоби для моделювання людино-орієнтованих систем, що є критично важливим для розробки SCADA інтерфейсів.

Для формалізації знань з предметної області «Ерготехнічні системи» (сюди включені людино-машинні системи):

- DUL розроблена з метою моделювання складних систем, що включають агентів, артефакти та різні події, що робить її ідеальною для опису взаємодій у людино-машинних системах. DUL дозволяє точно моделювати ролі учасників та артефактів у системі, а також події, до яких вони залучені, що підходить для опису людино-машинних систем, таких як SCADA;

- SUMO надає багатий верхньорівневий набір концепцій для моделювання фізичного світу та взаємодій, але потребує додаткового доопрацювання для детального опису людино-машинних систем. SUMO складніше у використанні для конкретних застосувань, оскільки потребує більше ресурсів для адаптації до специфічних завдань.

Таким чином, DUL найкраще підходить для детального моделювання складних людино-машинних систем, надаючи багаті можливості для опису різних аспектів взаємодії людини з машинами та інтерфейсами.

Загальні висновки:

- DUL більш гнучка та орієнтована на завдання, що потребують точного моделювання подій та взаємодій між агентами (людьми) та системами. Її структура інтуїтивна для опису таких систем, як SCADA, де важливі технологічні процеси та людино-машинні взаємодії;

- SUMO хоч і багата на концепції, але є більш загальною онтологією, яка потребує налаштування для специфічних областей застосування. Це робить її менш зручною для завдань, де потрібно точно описувати конкретні процеси та взаємодії у людино-машинних системах.

1.3 Постановка задач дослідження

Існує проблема інтелектуальної підтримки процесу ергономічного (людино-орієнтованого) проєктування ЛМІ АСУТП. Для вирішення цієї проблеми необхідно розробити відповідну інформаційну технологію

проектування (ІТП), яка надалі послужить основою для практичної реалізації у вигляді системи інтелектуальної підтримки проєктувальника. Важливою складовою даної ІТП є база знань, яка має формалізувати знання щодо системи «проєктувальник-ЛМІ-технологічний процес-середовище». Як було зазначено вище, необхідно представлення знань про наступні предметні області: «Технологічний процес»; «Проектування ЛМІ АСУТП» та «Ерготехнічна система».

Необхідність формалізації знань перших двох областей очевидна. Розглянемо необхідність представлення знань щодо предметні області «Ерготехнічна система». Справа у тому, що людина-оператор (або кілька людей-операторів) технологічного процесу є складовим елементом системи «людина—машина», або ергатичної системи. У загальному випадку, ергатична система — це складна ієрархічна система керування, в якій людина може брати участь на будь-якому рівні. Ергатичними системами є, наприклад, керування засобом транспорту, екскаватором, диспетчерська служба шахт, збагачувальних фабрик тощо [7].

Для представлення знань із різних галузей рекомендується використання метаонтологій [8]. Відображення предметних онтологій у цьому випадку стає набагато простіше, оскільки концепції у різних онтологіях, які необхідно відобразити, виводяться з однієї онтології.

Наприклад, щоб перевірити корисність фундаментальних онтологій в онтологічній інженерії, провели експеримент [11], у якому учасники мали обирати між розробкою «комп'ютерної онтології» з нуля або повторним використанням DOLCE чи BFO. Дослідження дійшло висновку, що переваги, отримані при використанні фундаментальних онтологій, компенсують час, витрачений на знайомство з ними. Набагато полегшить завдання також застосування спеціалізованих онтологій, наприклад, онтології PROV_O опису технологічних процесів.

Однак, як показує практика, інтеграція кількох предметних онтологій на основі метаонтології є досить важким завданням [9], [10]. Онтологічні категорії,

відношення, обмеження та аксіоми, визначені фундаментальними онтологіями, часто можуть мати невідповідність до онтологій доменів (предметних онтологій) та інших інформаційних артефактів (тобто інформаційних систем, баз даних, процесів інформаційних потоків). Тому важливим етапом усунення цих невідповідностей є зіставлення (вирівнювання) онтологій [12]. Воно відіграє особливо важливу роль у системах, агенти яких спілкуються за допомогою онтологій.

Виходячи із вищесказаного, для досягнення поставленої мети роботи необхідно:

- розробити концептуальну модель предметної області «Проектування ЛМІ АСУТП»;
- розробити предметні онтології «Технологічний процес». «Проектування ЛМІ АСУТП» та «Ерготехнічна система» на базі метаонтології DUL;
- провести вирівнювання розроблених онтологій;
- розробити систему запитів до онтології;
- перевірити адекватність розроблених моделей на прикладі проектування ЛМІ для АСУТП виробництва газобетону.

1.4 Висновки до першого розділу

Ергономічне, людино-орієнтоване проектування інтерфейсів АСУТП має вирішальне значення для підвищення безпеки, ефективності та надійності роботи системи. Добре спроектовані інтерфейси допомагають мінімізувати помилки операторів, зменшують їх когнітивне навантаження та дозволяють швидше приймати рішення, особливо у стресових чи позаштатних ситуаціях. Такий підхід забезпечує адаптацію інтерфейсів до різних рівнів кваліфікації операторів, покращує взаємодію людини з машиною та робить управління технологічними процесами більш інтуїтивним та продуктивним.

Сучасні інструментальні SCADA-системи не мають у своєму складі засобів, що забезпечують інтелектуальну підтримку процесу ергономічного проектування ЛМІ. Тому, актуальна розробка інноваційних інформаційних технологій у цьому напрямку.

Важливою складовою інтелектуальних систем є база знань. Існує доволі багато різноманітних методів представлення знань. Однак, у випадку інтеграції знань з декількох предметних областей рекомендовано використовувати багаторівневі онтологічні моделі. Проведений аналіз метаонтологій та онтологій середнього рівня, дозволив обґрунтувати вибір онтології DUL у якості базової метаонтології.

Для досягнення поставленої мети роботи було сформульовано постановку певних задач дослідження.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Розробка концептуальної моделі предметної області «Проектування ЛМІ АСУТП»

Розробка концептуальної моделі предметної області «Проектування ЛМІ АСУТП» є необхідним етапом у рамках створення інтелектуальної інформаційної технології для ергономічного проектування людино-машинних інтерфейсів АСУТП з кількох ключових причин:

- формалізація знань. Проектування ЛМІ в АСУТП включає широкий спектр знань із різних галузей: ергономіка, інженерна психологія, технології процесів. Концептуальна модель дозволяє формалізувати та структурувати ці знання в єдину систему, що робить їх доступними для інтелектуальної обробки, аналізу та автоматизації проектувальних процесів;
- забезпечення інтелектуальної підтримки. Інтелектуальні системи можуть ефективно допомагати проектувальникам лише за наявності ясної та структурованої моделі знань. Концептуальна модель описує основні сутності та взаємозв'язки у предметній області, що необхідно для реалізації таких функцій, як автоматична генерація рішень, інтелектуальні рекомендації, перевірка на відповідність ергономічним стандартам та виявлення потенційних помилок в інтерфейсі;
- інтеграція людино-орієнтованих та технологічних аспектів. Концептуальна модель дозволяє врахувати як знання про технологічні процеси, так і особливості взаємодії людини із автоматизованою системою. Це важливо для створення інтерфейсів, які будуть ефективними та інтуїтивними для операторів з різним рівнем підготовки. Модель допомагає збалансувати вимоги до технологічних процесів та потреби людини, що підвищує зручність та безпеку роботи;
- прискорення проектування та уніфікація рішень. Концептуальна модель є основою для стандартизації та уніфікації проектних рішень для інтерфейсів АСУТП. Це спрощує та прискорює процес розробки, оскільки

дозволяє повторно використовувати перевірені шаблони та рекомендації. Також модель допомагає уникнути дублювання роботи та систематизувати підходи до вирішення подібних завдань;

– підтримка багаторівневого аналізу та валідації. Завдяки концептуальній моделі можливе проведення комплексного аналізу різних аспектів проектування – від ергономічних вимог до технічних параметрів систем. Це дозволяє розробникам більш точно верифікувати та валідувати проєктні рішення, гарантуючи їхню відповідність як ергономічним стандартам, так і технічним вимогам.

Таким чином, концептуальна модель предметної області «Проектування ЛМІ АСУТП» є критично важливим інструментом для створення інтелектуальної інформаційної технології, яка зможе надавати ефективну підтримку проєктувальникам, підвищуючи якість та надійність інтерфейсів, що проєктуються.

Основи інформаційної технології проектування ЛМІ АСУТП були закладені автором у [13], [14] та вимагають подальшого розвитку в рамках цієї роботи.

Концептуальну модель предметної області «Проектування ЛМІ АСУТП» у нотації мови UML представлено на рис. 2.1. Ієрархічна структура предметної області відображена через відношення агрегування, а характер інших зв'язків між класами показаний з використанням іменованих спрямованих асоціацій.

Згідно з рис.2.1, проєктувальник виконує покрокову розробку проєкту ЛМІ, використовуючи набір стандартних елементів інтерфейсу «Palette». Проектування ЛМІ відбувається у робочому просторі «WorkSpace», який включає інструменти, шаблони та методи, необхідні для виконання проєкту. Фінальний результат роботи проєктувальника – інтерфейс ЛМІ SCADA-системи.

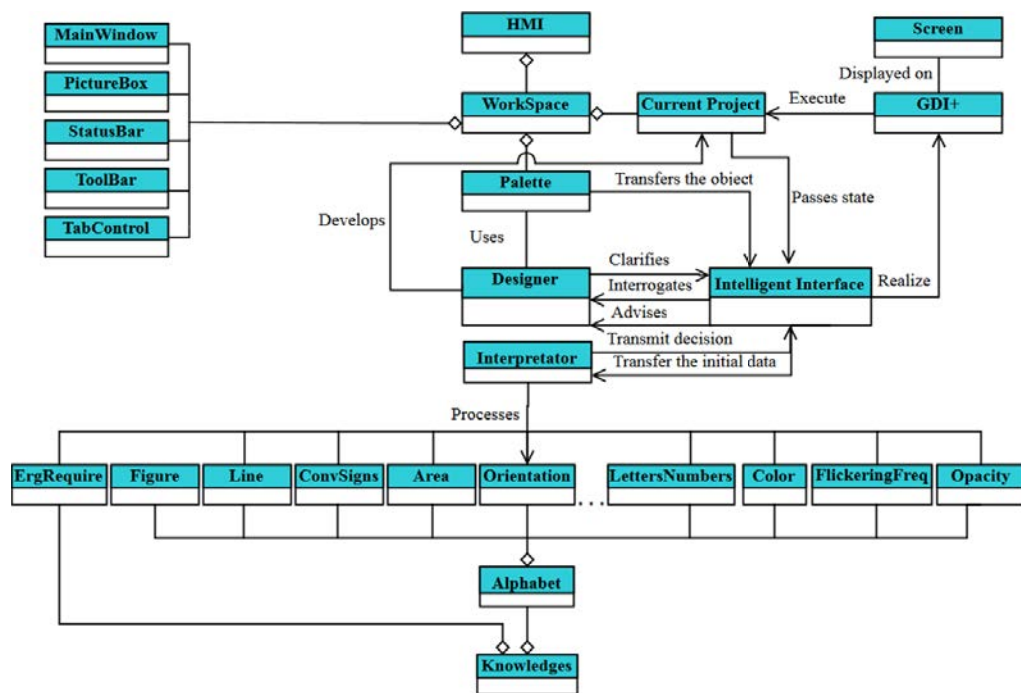


Рисунок 2.1 – Концептуальна модель предметної області «Проектування ЛМІ АСУТП»

Елементи, з якими працює проектувальник у робочому просторі, представлені різними програмними класами. Клас «WorkSpace» є контейнером для наступних класів: MainWindow (головне вікно середовища розробки), PictureBox (елемент для відображення графічних об'єктів), StatusBar (панель стану), ToolBar (панель інструментів) та TabControl (панель вкладок для керування сторінками).

Інтелектуальна підтримка проектувальника здійснюється за допомогою експертної системи ES. Типова ES включає інтерпретатор (вирішувач), робочу пам'ять (база даних для вихідних та проміжних даних), базу знань, компонент для придбання знань, пояснювальний компонент та діалоговий компонент. До концептуальної моделі (рис. 2.1) додано відповідні класи для відображення цих елементів: «Interpreter» для вирішувача; «Knowledge» для бази знань (KB) та «IntelligentInterface» для інших компонентів.

Інтелектуальна підтримка проектувальника реалізується через взаємодію з підсистемою «IntelligentInterface», яка генерує поради на основі поточного стану проєкту, вибраних елементів інтерфейсу з «Palette» та діалогу з проектувальником. Ці дані передаються «Interpreter» як передумови, які

описують поточне завдання. Інтерпретатор, використовуючи вихідні дані та знання з KB, формує послідовність правил та пропозицій для вирішення проектної задачі TS_i .

База знань KB експертної системи складається з багатьох локальних баз. Перша KB, «ErgRequire», містить такі знання з ергономічного забезпечення проектування ЛМІ: психофізіологічні характеристики оператора; правила розроблення інформаційних моделей; правила компоновання кодових знаків; правила та рекомендації щодо проектування шкал приладів, графіків.

Друга KB, «Alphabet», представляє знання про способи кодування (алфавіти) інформації про об'єкти контролю та управління (ОКУ) технологічного процесу. Якісні та кількісні характеристики ОКУ кодуються в інформаційних моделях різними способами (алфавітами, кодами). Відомі 17 способів кодування [15]. Види алфавіту, наведені на рис. 2.1, а також склад такого важливого елементу концептуальної моделі як «Palette» (рис. 2.2) більш детально розглянуто в [13].

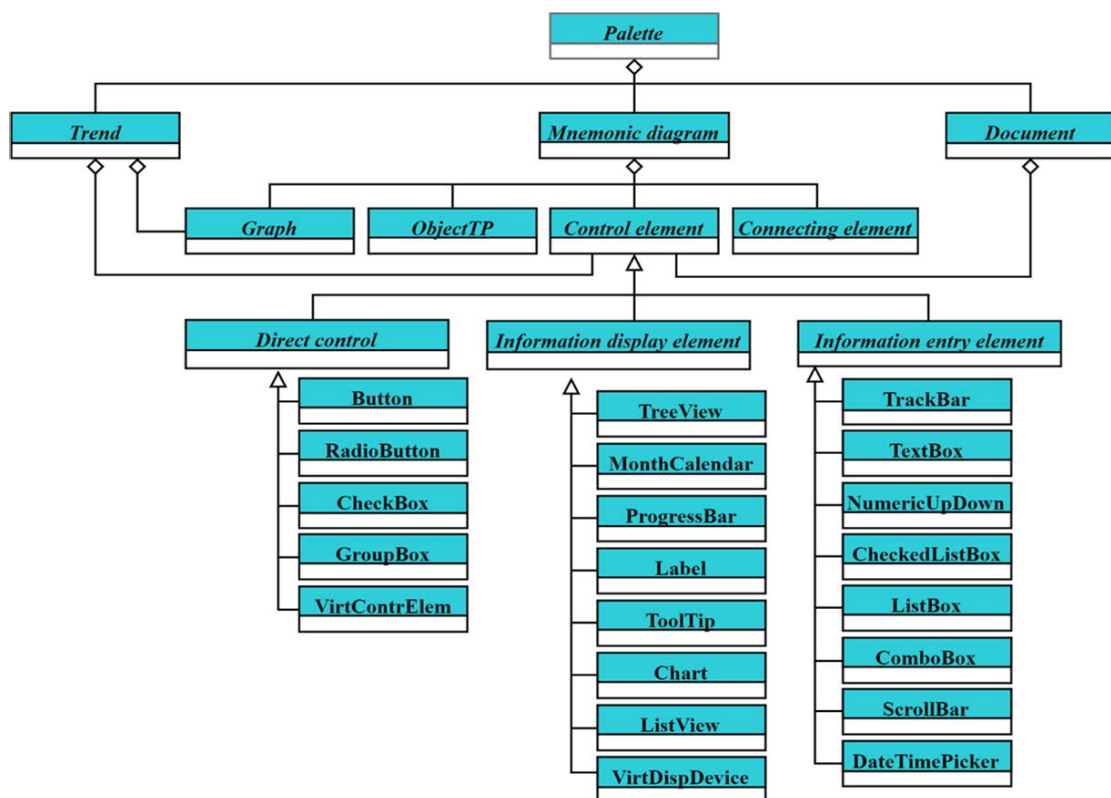


Рисунок 2.2 – Модель предметної області «Palette»

Для досягнення мети «Ергономічне проєктування ЛМІ», проєктувальник повинен вирішити множину завдань, номенклатура яких обумовлена відомими методиками проєктування інформаційних моделей та ЛМІ [7], [16], [17].

ЛМІ АСУТП складаються з графічних елементів Windows Forms. Ці елементи можна поділити на три основні категорії: двомірна (2-D) векторна графіка; рисунки; друкарська розмітка.

Інтерфейс керованих класів, пов'язаних із відображенням на екрані в .NET Framework 4.6 та 4.7 складається з наступних просторів імен: System.Drawing; System.Drawing.Drawing2D; System.Drawing.Imaging; System.Drawing.Text; System.Drawing.Configuration та System.Drawing.Design.

Батьківський простір імен System.Drawing містить типи, що підтримують базові графічні функції GDI+. Основою інтерфейсу GDI є клас Graphics. Він безпосередньо виконує малювання прямих та кривих ліній, геометричних фігур, виведення рисунків та тексту.

Було проведено аналіз функціонального призначення класів простору імен System.Drawing, пов'язаних з ними методів класу Graphics, а також видів алфавіту, на можливість їх застосування для кодування графічних елементів Windows Forms.

Введемо чотири множини:

- A_1 – інформація, що відображається на екрані;
- A_2 – простір імен System.Drawing;
- A_3 – методи класу Graphics;
- A_4 – види алфавіту.

Для того щоб визначити знання, необхідні підсистемі «IntelligentInterface» для реалізації функцій відображення та кодування елементів ЛМІ, введемо тетрарне відношення R_1 "Для відображення та кодування a_1 використовується a_2 , a_3 , a_4 " на декартовому добутку $A_1 \times A_2 \times A_3 \times A_4$.

Відношенню R_1 можна поставити у відповідність 4-місний предикат. Кортеж належить відношенню тоді і лише тоді, коли предикат цього відношення $P(a_1, a_2, a_3, a_4) = 1$.

Усі елементи відношення R_1 є однотипними кортежами. Однотипність кортежів дозволяє вважати їх аналогами рядків (i) у простій таблиці, тобто в такій таблиці, в якій всі рядки складаються з однакового числа чарунок і у відповідних чарунках розміщуються однакові типи даних. Приклад представлення відношення R_1 у табличній формі наведено в таблиці 2.1.

Таблиця 2.1 – Приклад представлення відношення R_1

Відображувані дані	Класи простору імен System.Drawing	Методи класу Graphics	Вид алфавіту								
			Геометричні фігури	Умовні знаки	Площа фігур	Орієнтація у просторі	Штрихування	Буквено-цифровий	Колір, яскравість	Частота миготіння	Прозорість
2-D векторна графіка	Brush	FillClosed Curve							1		
		FillEllipse					1		1		
		FillPolygon					1		1		
		FillRectangle					1		1		
	Pen	DrawArc	1			1			1		
		DrawClosed Curve	1	1	1	1			1		
		DrawEllipse	1		1	1			1		
		DrawLine				1			1		
		DrawPolygon	1		1	1			1		
Растрові зображення	Image	DrawImage	1	1	1	1	1	1	1	1	1
	Icon	DrawIcon		1	1				1		
Символи та рядки	StringFormat	DrawString						1			

Кількість стовпців таблиці 2.1 $j=4$ обумовлено кількістю множин декартового добутку. Для зручності подання інформації (скорочення кількості рядків) таблиця 2.1 має такі особливості:

– 4-й стовпець представлений у вигляді 15 елементів множини A_4 (у прикладі таблиці 2.1 представлено дев'ять алфавітів);

– в одному рядку об'єднані кортежі із загальними елементами зі стовпців $j=1 \div 3$ та різними елементами множини A_4 ;

– для елементів таблиці x_{ij} виконується умова

$$x_{ij} = \begin{cases} 1, & \text{if } R_1 = \text{true}, \\ 0, & \text{if } R_1 = \text{false} \end{cases}$$

Такі окремі коди як букви, знаки пунктуації та цифри у таблиці 2.1 представлені одним алфавітом «Буквено-цифровий». У зв'язку з тим, що комп'ютерна реалізація кодів «Колір» і «Яскравість» пов'язана з однією і тією ж моделлю RGB, в таблиці 2.1 вони представлені в одному стовпці.

Після формалізації та внесення знань, наведених у таблиці 2.1 в KB, експертна система матиме знання про технологію відображення та вибору кодів для графічних елементів Windows Forms. Наприклад, під час вирішення завдання відображення растрових зображень, підсистема «IntelligentInterface», запропонує використовувати клас Image I/АБО клас Icon простору System.Drawing і відповідні методи класу Graphics. При виборі проєктувальником класу Image йому будуть рекомендовані, як можливі, всі види кодів. При виборі класу Icon, проєктувальнику будуть рекомендовані такі види кодів: «Умовні знаки» I/АБО «Площа фігур» I/АБО «Позиція» I/АБО «Колір» I/АБО «Яскравість».

Щоб сукупність розроблених моделей, технологій та знань разом створили нову інформаційну технологію проєктування ЛМІ АСУТП, на наступних етапах дослідження, треба формалізувати знань, що мають бути представлені в базі KB за допомогою багаторівневої онтології.

2.2 Розробка ієрархічної структури онтології

За основу розробленої ієрархії були взята «Методика функціонального аналізу енергоблоків атомних станцій» [18], що описує структуровану модель яка розподілена на рівні:

- рівень цілей;
- рівень функцій;

- рівень задач;
- рівень дій;
- рівень операцій.

Де цілі – це кінцевий результат всього процесу. Функція – це деяка робота, що виконується технологічним обладнанням та/або персоналом та націлена на досягнення цілі. Задача – це важка алгоритмічна діяльність, що виконується персоналом разом з автоматикою. Дія – це деякий закінчений акт, що виконується оператором або автоматикою та яка веде до певного результату. А операція – це найпростіша подія техпроцесу на які розбивається дія.

Деградація функції приводить до виникнення технологічної події.

Технологічна подія – поняття, що описує подію, що пов'язана з деградацією функції.

Також передбачається наявність у техпроцесі технологічної події – це деяка подія, значуща з точки зору задач діагностики та керування, що виконуються.

Для того щоб технологічна подія була виявлена, кожна така подія повинна викликати деякі зміни, наприклад - миготіння на табло, змінення показників приладів, загоряння сигнального індикатора.

Для ліквідації таких подій оператору потрібно виконувати деякі керуючі дії, що передбачені відповідними інструкціями.

Згідно з [18], технологічна подія – це певна подія, значуща з точки зору виконуваних завдань діагностики і управління. Типовою технологічною подією є порушення технологічної умови (наприклад, технологічний параметр вийшов за межі).

Відповідно, при появі технологічної події, у оператора, що керує техпроцесом, виникають задачі, що визначають цілі, яка мають бути досягнуті для ліквідації помилки.

Також, кожна технологічна подія має причину, що викликала аварійну ситуацію і є реалізацією аварійних ситуацій під час техпроцесу. Відповідно

технологічна подія ініціалізує задачу, вирішення якої призводить до усунення технологічної події.

У свою чергу, для вирішення такої задачі, мають бути виконані певні дії, що також розподіляються на операції (аналогічно для техпроцесу без аварійних ситуацій).

2.3 Ерготехнічна система та її структура

Система «людина-машина», або ергатична система, це система, складовим елементом якої є людина-оператор (або група людей-операторів).

У загальному випадку ергатична система – це складна ієрархічна система керування, в якій людина може брати участь на будь-якому рівні. Ергатичними системами є, наприклад, керування засобом транспорту, екскаватором, диспетчерська служба шахт, збагачувальних фабрик тощо [15].

Ерготехнічна система (ЕТС) – різновид ергатичної системи, що включає до свого складу в якості обов'язкового компонента технічні пристрої.

Класифікація ергатичних систем може бути проведена за низкою ознак. За основною цільовою функцією вони діляться на: контрольні, управління, пошукові та навчальні ергатичні системи.

За типом інформаційної моделі ЕТС поділяються на:

- ЕТС з диференціальною інформаційною моделлю;
- ЕТС з інтегральною інформаційною моделлю.

Властивості ЕТС визначаються властивостями основних її ланок, а саме, оператора і «машини». Оператор або машина можуть перевершувати один одного при виконанні певних функцій.

Оператор перевершує «машину» в наступних функціях [19]:

- виявлення слабких світлових і звукових сигналів;
- сприйняття, інтерпретація та організації сигнальних образів різних модальностей;

- здійснення гнучких операцій управління;
- зберігання великої кількості інформації в плинні тривалого часу і її використанні в потрібний момент;
- генерація індуктивних висновків;
- зміна показників в результаті навчання;
- формування понять і вироблення методів;
- організація та об'єднання показань входів, різних за модальністю, по параметрах.

«Машина» перевершує людину-оператора у наступному [19]:

- швидкість відповіді на сигнал;
- здатність застосовувати плавно і точно керуючі дії великої сили;
- виконання повторних стереотипних дій і завдань;
- зберігання інформації в стислій формі і повним звільненням від непотрібної інформації;
- швидкістю розрахунків;
- здатність виконувати одночасно декілька різних функцій.

Згідно [14], об'єкти у ЕТС поділяються на рис. 2.3:

- об'єкти організаційного забезпечення ЕТС, тобто: персонал та організаційні елементи ЕТС (E1);
- об'єкти трудового процесу, а саме: знаряддя праці, предмет праці та продукт праці (E2);
- об'єкти забезпечення праці (E3).

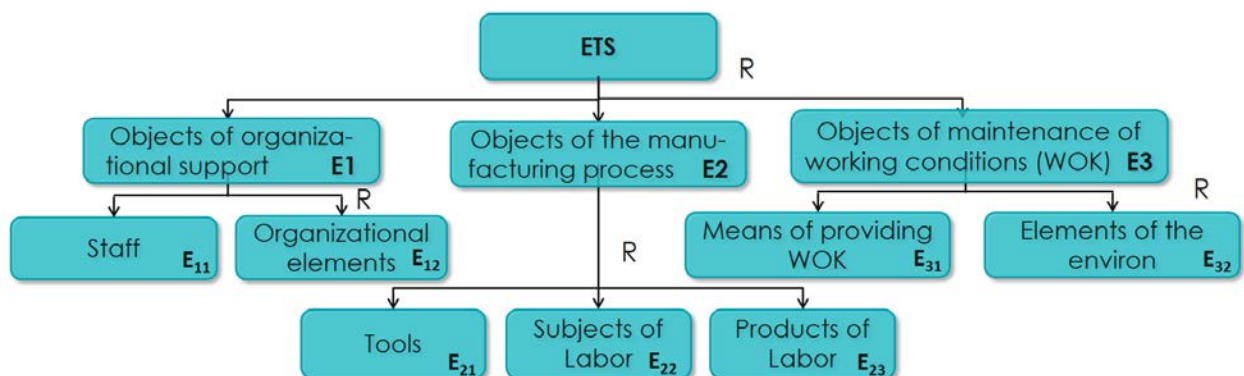


Рисунок 2.3 – Ієрархія об'єктів ЕТС

Згідно з рис. 2.3, об'єкти у ЕТС зв'язані відношенням R «Частка- ціле».

Визначимо деякі терміни. Персонал – це група людей або сукупність осіб, що здійснюють трудові функції на основі певних домовленостей. Персонал кожного конкретного техпроцесу має конкретні цілі, задачі та виконує свою роль у техпроцесі.

Організаційні елементи – це і є цілі, задачі відповідного процесу. Саме цілі та задачі визначають, що має бути досягнуто після виконання техпроцесу, та що саме має бути зроблено протягом самого техпроцесу.

Знаряддя праці – це деякі інструменти за допомогою яких оператор на предмет своєї праці й перетворює його. Знаряддями праці можуть бути будь-які предмети, що призначаються для роботи, наприклад: гайковий ключ, станок, пила, комп'ютер тощо.

Предмет праці – це об'єкт або суб'єкт на який спрямована праця і засоби праці. Предметами праці можуть бути заготовки, деталі, вихідна інформація.

Також необхідно зазначити ресурси, що необхідні для функціонування техпросесу, наприклад: електрична енергія, метал, запчастини, паливо тощо. Персонал також можна вважати ресурсом, що можливо окремо виділяти у так званий «людський ресурс».

Продукт праці – це результат, праці, що можуть являти собою як і якісь конкретні фізичні об'єкти, так і змінення станів якихось об'єктів, якісь конкретні дії або нову інформацію. До продуктів праці також можуть відноситися побічні або браковані об'єкти, що є результатом праці.

2.4 Висновки до другого розділу

У даному розділі розроблено моделі, що обумовлюють архітектуру експертної системи, за допомогою якої буде здійснюватися інтелектуальна підтримка процесу ергономічного проєктування ЛМІ АСУТП, а також ієрархічну структуру онтології для формалізації необхідних для цього знань.

3 РОЗРОБКА БАГАТОРІВНЕВОЇ ОНТОЛОГІЇ

3.1 Розробка ієрархії класів та зв'язків метаонтології

Згідно з проведеним у п. 1.2 аналізом онтологій, за основу була взята метаонтологія DUL, що представляє собою метаонтологію, яка містить у собі велику кількість понять, термінів, зв'язків тощо: починаючи від поняття об'єкту, закінчуючи техпроцесом, та зв'язками у ньому [20].

На рис. 3.1 наведена часткова ієрархія класів розробленої метаонтології.

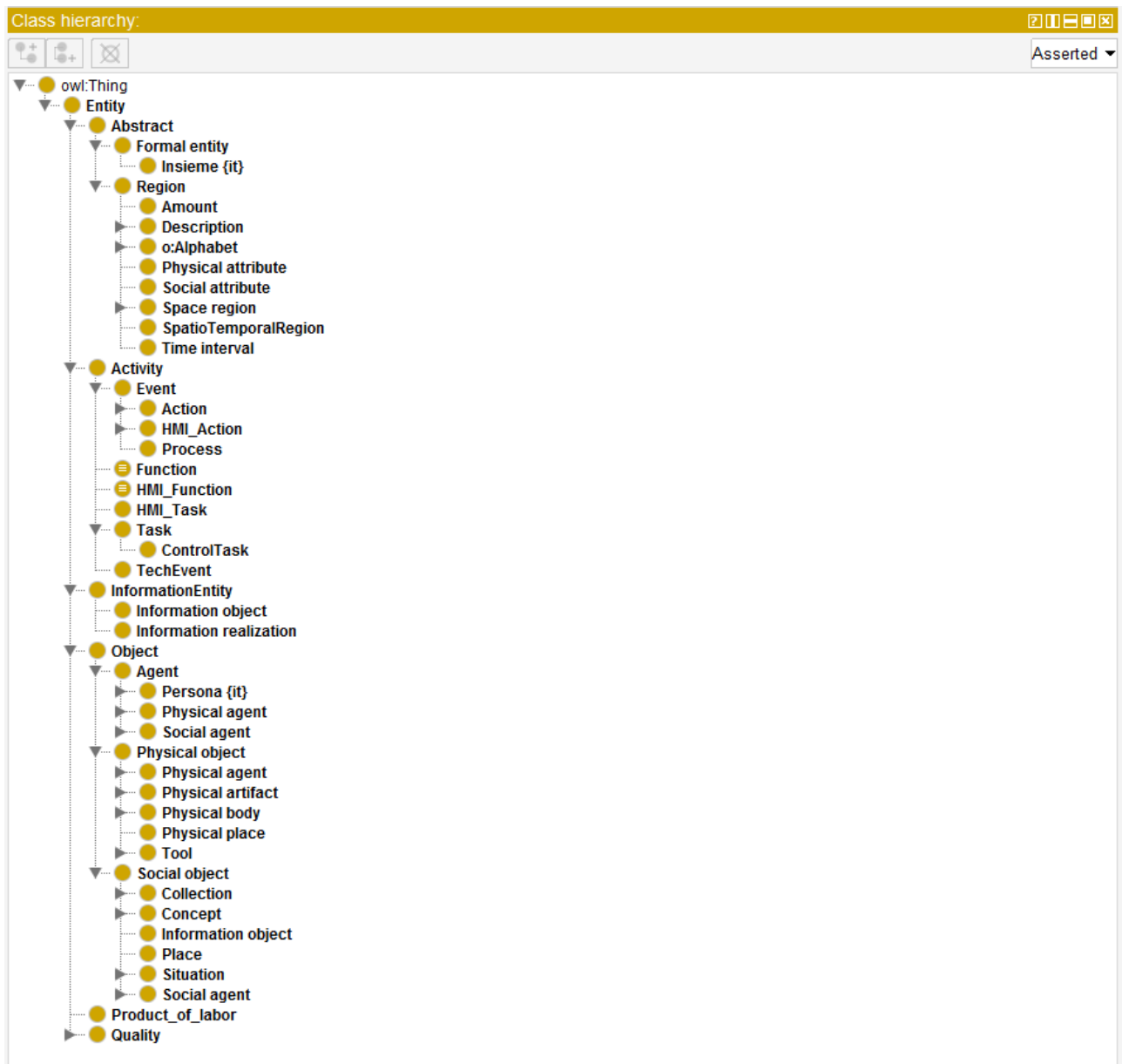


Рисунок 3.1 – Часткова ієрархія класів розробленої метаонтології

Цілям дослідження задовольняють лише класи DUL:Goal, DUL:Task та DUL:Action, що вже є у DUL.

DUL:Goal – це клас, що і є рівнем цілей у DUL та буде вершиною ієрархії. Goal є підкласом Description та має зв'язки з іншими класами

Зв'язки або залежності – це зазвичай арибут, значенням якого є інший об'єкт. DUL має багато зв'язків, що слугують для створення складеної ієрархії в онтології від зв'язку “isSubclass”, що являє відношення дочірнього класу до батьківського, до відношень, що визначають конкретні фізичні арибути об'єктів (їх місцезнаходження у просторі і часі; визначення, що класифікує об'єкт або чим об'єкт реалізований та інші). Зв'язки розробленої метаонтології наведені на рис. 3.2. За допомогою зв'язків підкласи з'єднуються з іншими гілками онтології, утворюючи семантичну мережу понять та відношень.

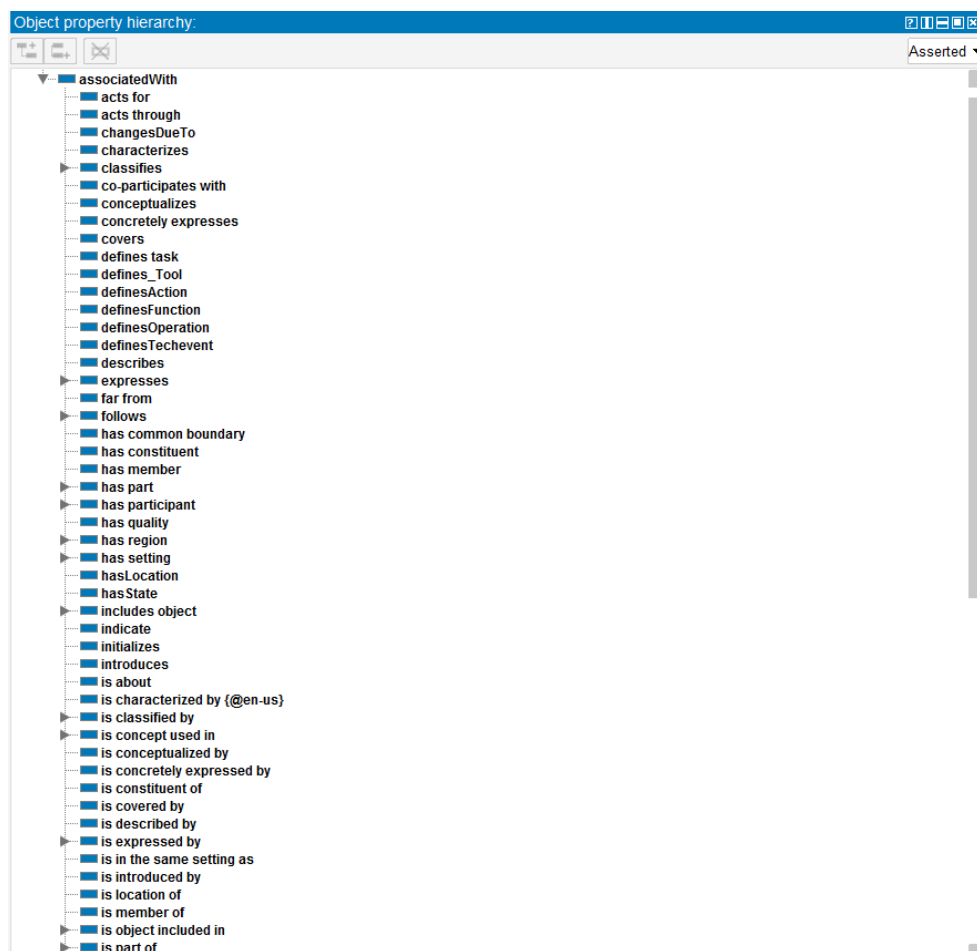


Рисунок 3.2 – Зв'язки у розробленій метаонтології

Клас Event у DUL є базовим класом для будь яких подій, це може бути будь-який фізичний, соціальний чи психічний процес, подія чи стан. Більш теоретично, Event можна класифікувати по-різному, можливо, на основі "аспекту" (наприклад, стаціонарного, безперервного, виконання, досягнення тощо), "агентності" (наприклад, навмисної, природної тощо) або "типових учасників" (наприклад, людські, фізичні, абстрактні, харчові тощо).

DUL:Event є підкласом DUL:Entity та також має зв'язки з іншими класами. Клас DUL:Event не може бути абстрактним, бо обов'язково повинен мати об'єкти свого класу. Також клас DUL:Event має два дочірні класи: DUL:Action та DUL:Process. DUL:Event може мати типи, що визначені класом DUL:Event type (рис. 3.3), що є деякою концепцією, яка класифікує подію. DUL:Event type описує, як подію слід інтерпретувати, виконувати, очікувати, бачити тощо.

Клас Task, що є у DUL являє собою DUL:EventType, який класифікує дію, яку потрібно виконати. Наприклад, досягнення пункту призначення – це DUL:Task, яку можна виконати, виконуючи певні DUL:Action. Наприклад, водіння автомобіля, придбання квитка на поїзд тощо. Action для виконання завдання також можуть бути організовані відповідно до DUL:Plan, який не є однаковим із тим, що визначає DUL:Task. Клас DUL:Task є підкласом DUL:EventType та також має зв'язки з іншими класами.

Клас DUL:Action – це подія з принаймні одним DUL:Agent, який є учасником, і який виконує DUL:Task, яке зазвичай є визначеним у DUL:Plan/ DUL:Workflow. А клас DUL:Process – це заповнювач подій, які враховуються в процесі їх розвитку, або в будь-якому випадку не суворо залежать від агентів, завдань та планів.

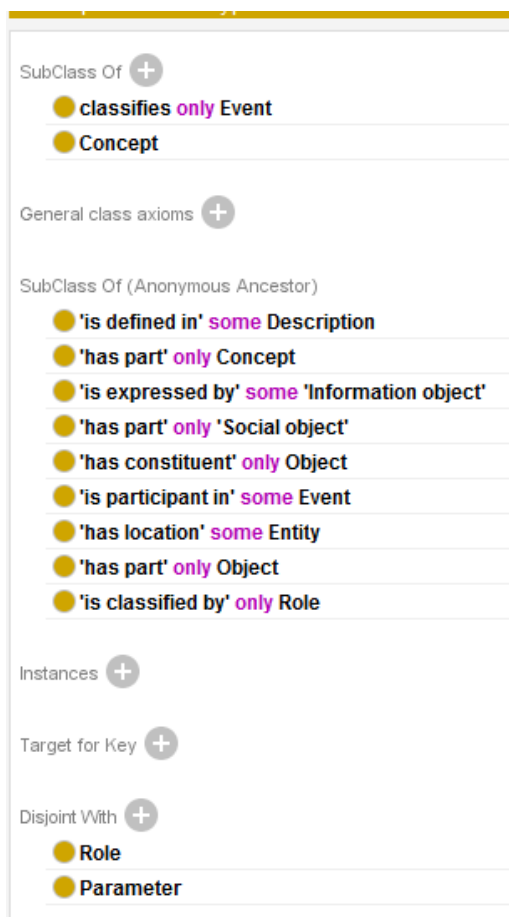


Рисунок 3.3 – Зв’язки класу DUL:EventType

3.2 Розробка онтології «Технологічний процес»

Для розробки онтології «Технологічний процес» була перероблена архітектура метаонтології DUL та змінені деякі взаємозв’язки та класи (рис. 3.4).

Основними класами, що реалізують техпроцес є:

- **Product_of_labor** – клас, що являє собою продукт праці: фізичний об’єкт, інформація, подія и т.п, це і є ключовим класом, що найбільш часто є результатом якоїсь Function, DUL:Task, Action, Operation тощо;
- **DUL:Workflow** – основний клас для визначення та додавання техпроцесу до онтології. Саме клас Workflow і визначає таку сутність, як техпроцес і є описом всіх складових, як об’єктів, суб’єктів, так і подій, що відбуваються або можуть відбутися, протягом техпроцесу.

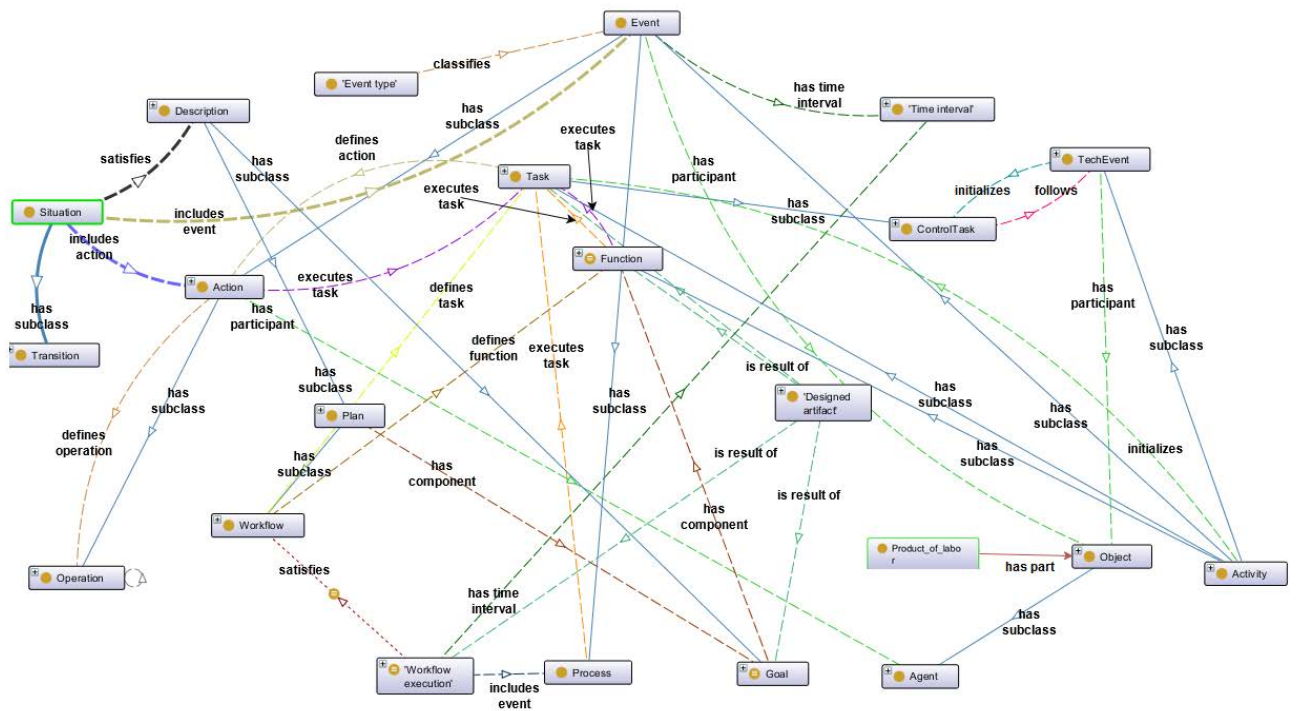


Рисунок 3.4 – Ієрархія основних класів техпроцесу у розробленій онтології

Структурну модель техпроцесу було реалізовано за допомогою наступних класів:

- DUL:Goal – клас, що реалізує перший рівень ієрархії – рівень цілей;
- проміжним класом між Entity та Event є клас Activity, що є базовим класом для подій таких як: Event, Function, Task, TechEvent (рис. 3.4). Він був створений, через неможливість деяких класів базової версії DUL мати об'єкти свого класу, що у свою черго викликало конфлікти у зв'язках та запитах онтології;
- Function – реалізує другий рівень ієрархії. Функція, що підтримує необхідні технологічні умови для виконання цілей. Клас Function – це клас, що описує деяку роботу, що виконується технологічним обладнанням та/або персоналом, має ресурси та стан, і націлена на досягнення цілі (рис. 3.4).

Також для кожної функції був доданий State, що є концептом, що визначає безліч чітких значень змінних параметрів об'єкту. Тобто, кожна сутність має деяку кількість станів, що характеризуються значеннями параметрів (температура, розмір, положення у просторі та часі і т. ін)

Підкласами State є класи, що визначають стани функції (FunctionState), режим (Mode), стан об'єкту (ObjectState) та стан приладів (ToolState).

Стани функції:

- Active – стан, що описує функцію, коли вона активна;
- InCondition – у стані, що задовольняє потреби техпроцесу до функції у даний момент;
- NotReady – стан, що визначає функцію, яка справна, але на даний момент не готова до виконання;
- Passive – стан у якому функція не виконується;
- Ready – стан, що описує функцію, яка готова до виконання;
- Wrong – стан, коли функція не задовольняє потребам техпроцесу на даний момент.

Стан об'єкту визначає множину його характеристик та узагальнює їх у 1 з декількох можливих загальних станів.

Стани об'єктів підрозділяються на:

- Abnormal – стан об'єкту, що описує аномальний стан об'єкту, який не є критичним або небезпечним, але і не є оптимальним;
- Dangerous – небезпечний стану об'єкту, що описує стан, при якому об'єкт є небезпечним для техпроцесу, навколишнього середовища або для себе;
- Emergency – критичний стан об'єкту, що описує об'єкт, який знаходиться у аварійному стані, що має бути найбільш швидко усуненим;
- Not_Working – стан, що описує об'єкт, що не є працездатним;
- Woking – стан об'єкту, який є нормальним та робочим у даний момент.

ToolState же у свою чергу описує стан інструментів, що є учасниками техпроцесу, якихось передвизначених станів немає через велику кількість інструментів, що можуть брати участь у техпроцесі.

Зазначивши State потрібно також звернутися до поняття Parameter, що є це узагальненою назва певної фізичної, геометричної чи іншої властивості пристрою

(процесу). Це можуть бути, наприклад, розмір, швидкість, електрична напруга тощо.

Для техпроцесу було вирішено розділити параметри на Quantitive та Qualitive, де перші – це параметри, за допомогою яких можливо проводити кількісні виміри та дослідження, наприклад: обороти, напруга, вага тощо. Qualitive – це такі параметри, вимірювання яких дозволяє проводити якісні виміри та дослідження, наприклад: якість виробу, звук повідомлення, якість керування тощо. Відповідно Quantitive параметри мають Unit of measure (одиниці виміру), що можуть бути виражені фізичними величинами (кг, км/г, дБ, Паскалі, м³). А Qualitive параметри не можуть бути виражені конкретними абсолютними фізичними величинами, а можуть бути оцінені лише відносними.

DUL:Task – третій рівень ієрархії – задачі. Це складна алгоритмічна діяльність, що виконується персоналом спільно з автоматикою [18], яку необхідно виконати для задовільнення функцій. Task у розробленій онтології тепер є дочірнім класом Activity, а не EventType, що необхідно для більш гнучкого використання класу, що не буде обмежений рамками класу EventType. Також клас Task тепер є базовим класом для класу ControlTask, що реалізує Task при аварійних ситуаціях (рис. 3.5).

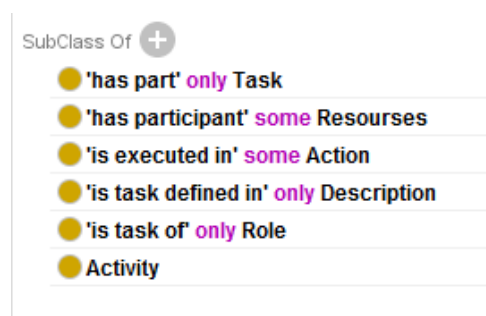


Рисунок 3.5 – Зв'язки класу Task у розробленій онтології

DUL:Action – четвертий рівень – деякі дії, що приймаються для досягнення результату з вирішення певної задачі.

Наступним класом є клас Action, що являє собою клас Action (рис. 3.6) з початкової версії DUL, але він тепер отримав дочірній клас Operation. Також, у

розробленій онтології Action також має ресурси, на відмінність від початкової версії DUL.

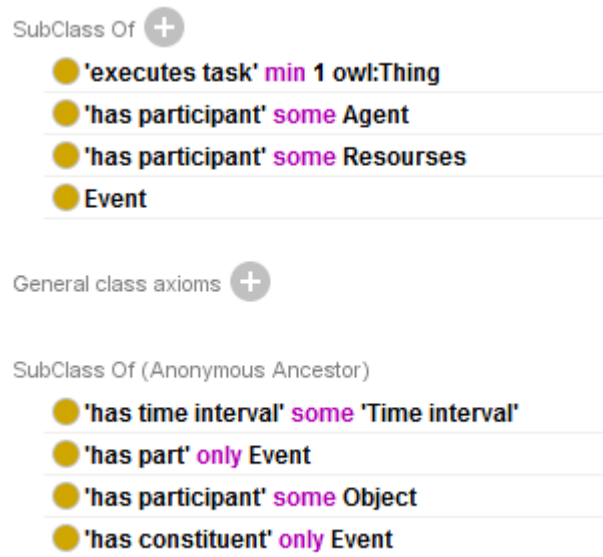


Рисунок 3.6 – Зв'язки класу Action

Клас Operation – це п'ятий рівень, операції на які поділяється кожна дія.

Також необхідно зазначити, що Function, Task, Action та Operation можуть бути описані зв'язками, що визначають їх послідовність, тобто, деякі Function можуть виконуватися паралельно одна одній (зв'язок isParallel) або передувати, або відбуватися одна після одної. Саме завдяки цим зв'язкам і будується ієрархія виконання техпроцесу.

Також був введений клас Tool, що реалізує інструменти, які необхідні для виконання техпроцесу (рис. 3.7), такі як, наприклад: бункер, молоток, гайковий ключ, інтерфейс тощо. Предмет, який є Tool може бути одночасно також і Resource типу Exploited, що використовується для потреб техпроцесу.

У розробленій онтології Tool має два дочірні класи HMI та Screen, що будуть описані пізніше.

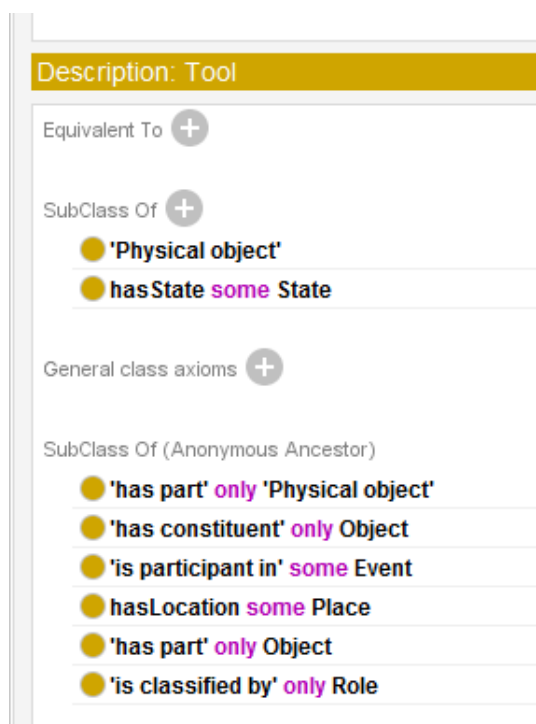


Рисунок 3.7 - Зв'язки класу Tool

3.3 Розробка онтології «Ерготехнічна система»

У метаонтології DUL не представлені класи, що могли б описувати роль людини, у взаємодії з машиною. Найбільш наближеним поняттям до поняття «персонал» є клас Group, що представляє собою групу людей, що не відповідає потребам техпроцесу. Також є клас DUL:Organization, що являє собою велику організацію без розподілення її на якихось конкретних представників, а тому не є відповідними нашим цілям.

Основою для «соціальних» класів у DUL – є клас DUL:Social agent, що є класом, який описує будь-який Object, який існує лише в межах якоїсь DUL:Event зв'язку, в якій бере участь принаймні один DUL:PhysicalObject.

Іншими словами – це всі об'єкти, які були створені в процесі соціальної комунікації, а саме: DUL:InformationObject (заради спілкування), DUL:SocialAgent, DUL:Place (для включення нових індивідів), DUL:Description, DUL:Concept (для контекстуалізації або інтерпретації існуючих сутностей) або DUL:Collection (для збору існуючих об'єктів).

Будучи залежними від спілкування, всі «соціальні» об'єкти повинні бути виражені якимось інформаційним об'єктом (інформаційні об'єкти самовиражаються). Клас DUL:Personification – це DUL:SocialObject з агентивними рисами, але статус якого є результатом культурної трансформації. Наступним класом є клас DUL:Collective agent, що має два дочірніх класи. Цей клас – це SocialAgent, який діє за допомогою DUL:Agents, які є (і діють як) члени DUL:Collective. DUL:Collective agent може мати DUL:Roles, які також є ролями цих агентів.

Наприклад, у соціології «групова дія» – це ситуація, коли певна кількість людей (внаслідок чого стають членами колективу) у певній галузі поводить себе злагоджено для досягнення (часто спільної) мети. DUL:Agent у такій ситуації не одиночний, а DUL:Collective agent (DUL:Group). Це можна узагальнити до поняття соціального руху, який передбачає, що великою громадою або навіть усім DUL:Community є агенти.

Також існує клас DUL:Social Person, що є DUL:Social agent, який потребує існування конкретного DUL:NaturalPerson, щоб діяти (але термін служби DUL:NaturalPerson повинен лише перекривати той, що є DUL:SocialPerson)

Через те, що ETC не була реалізована у DUL, як така були додані класи та зв'язки, що її реалізують (рис. 3.8). Зеленим кольором зазначені класи, що були додані для формального представлення ETC у онтології.

Через відсутність класу, що описує персонал техпроцесу до метаонтології DUL було додано клас Staff, зв'язки якого представлені на рис. 3.9.

Персонал Staff реалізує сутність персоналу, а класи Designer і Operator – це вже конкретні діючі особи техпроцесу. Operator – оперує автоматикою та забезпечує виконання техпроцесу, а Designer – забезпечує проектування та розробку НМІ для конкретного техпроцесу за допомогою SCADA. Ці класи також було зв'язано з уже доданими до онтології, такими як: Tool, Exploited, Function та Resources.

Деградація функції приводить до виникнення технологічної події TechEvent, що ініціює виконання ControlTask.

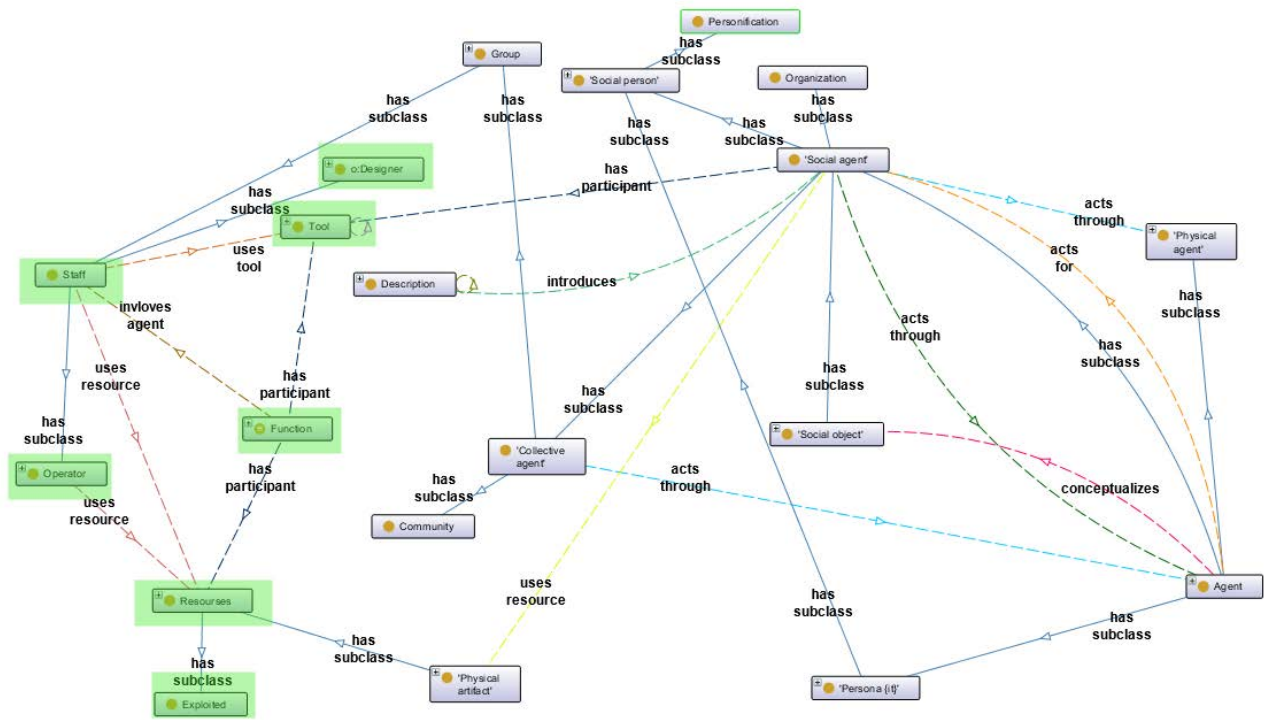


Рисунок 3.8 – Граф класів, що описують ЕТС у розробленій онтології

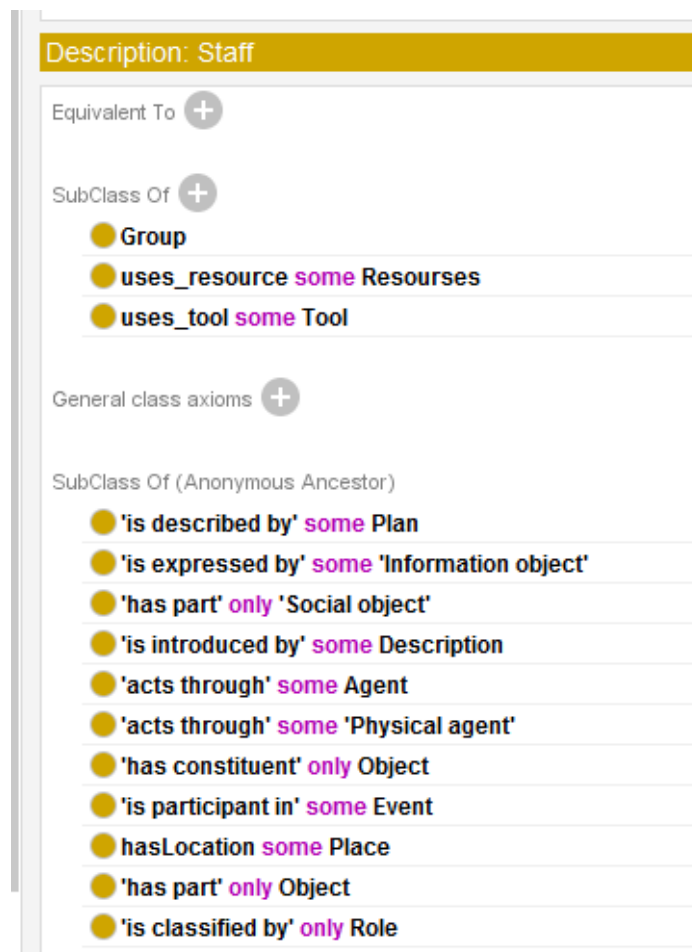


Рисунок 3.9 - Зв'язки класу Staff

TechEvent – клас, що реалізує технологічну подію, що пов’язана з деградацією функції. Відповідно, при появі TechEvent генерує ControlTask, що визначає ціль, яка має бути досягнута для ліквідації помилки. Також, кожен TechEvent має причину, що викликала аварійну ситуацію і є реалізацією аварійних ситуацій під час техпроцесу.

На рис. 3.10 зображений граф, що показує ієрархію класів, необхідних для реалізації техпроцесу та ЕТС. Зеленим - позначені класи, що необхідні для реалізації техпроцесу. Червоним - позначені класи, що мають відношення до ЕТС. Жовті класи неможливо конкретно віднести до якоїсь предметної області, бо вони мають відношення до багатьох класів онтології. Синій блок – це блок основних класів, що необхідні для реалізації аварійної ситуації.

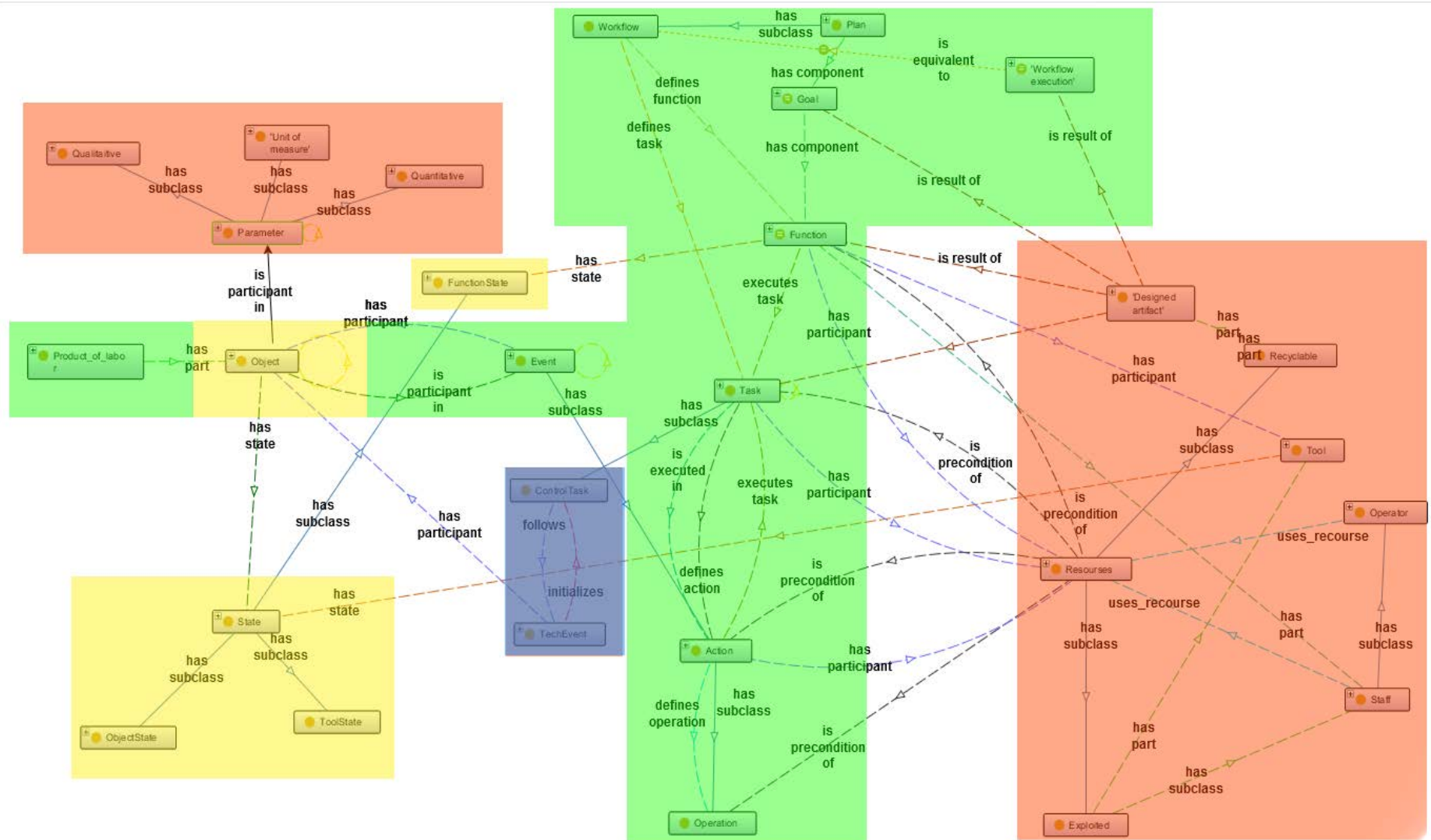


Рисунок 3.10 - Класи, що необхідні для реалізації онтології техпроцесу та ЕТС

3.4 Розробка онтології «Людино-машинний інтерфейс»

Початково у DUL відсутнє поняття ЛМІ (HMI), а також усі поняття, що зв'язані з ним, наприклад: Designer, Screen, Palette і т.д (рис. 2.1, 2.2). Навіть базового класу, що міг би стати батьківським для HMI, у DUL не існує. Тому було вирішено зробити HMI дочірнім класом, від класу Tool, що був описаний раніше. Аналогічно для класів, що представляють собою конкретні об'єкти HMI, не було знайдено класів, що могли би задовольняти потребам системи, що розроблюється, а тому архітектури HMI та техпроцесу розробки HMI будувалася майже з нуля. Класом, що найбільше відповідав вимогам HMI був DUL:Info.

При створенні техпроцесу проектування ЛМІ за основу було взято концептуальну модель [13]. Для введення в онтологію, що розроблюється, відповідних модулів, було приєднано аналогічні класи до онтології DUL. Часткову структуру онтології з доданими класами зображено на рис. 3.11. Блакитними блоками позначено класи Tool, що було приєднано (зв'язок subClass) до класу DUL:PhysicalObject та Staff до класу DUL:Group ще у процесі створення онтології техпроцесу. До класу Tool було приєднано HMI, до якого у свою чергу приєднано клас ElementHMI, а до Staff приєднано клас Designer.

Зеленими блоками позначено класи, що були додані у процесі створення ієрархії для реалізації техпроцесу проектування ЛМІ. Таким чином, до класу HMI було додано клас Workspace, до якого в свою чергу було додано класи Form та Pallette. Клас Form має вкладені класи, які визначають об'єкти якими дизайнер Designer заповнює Workspace. До Workspace приєднано клас Pallette.

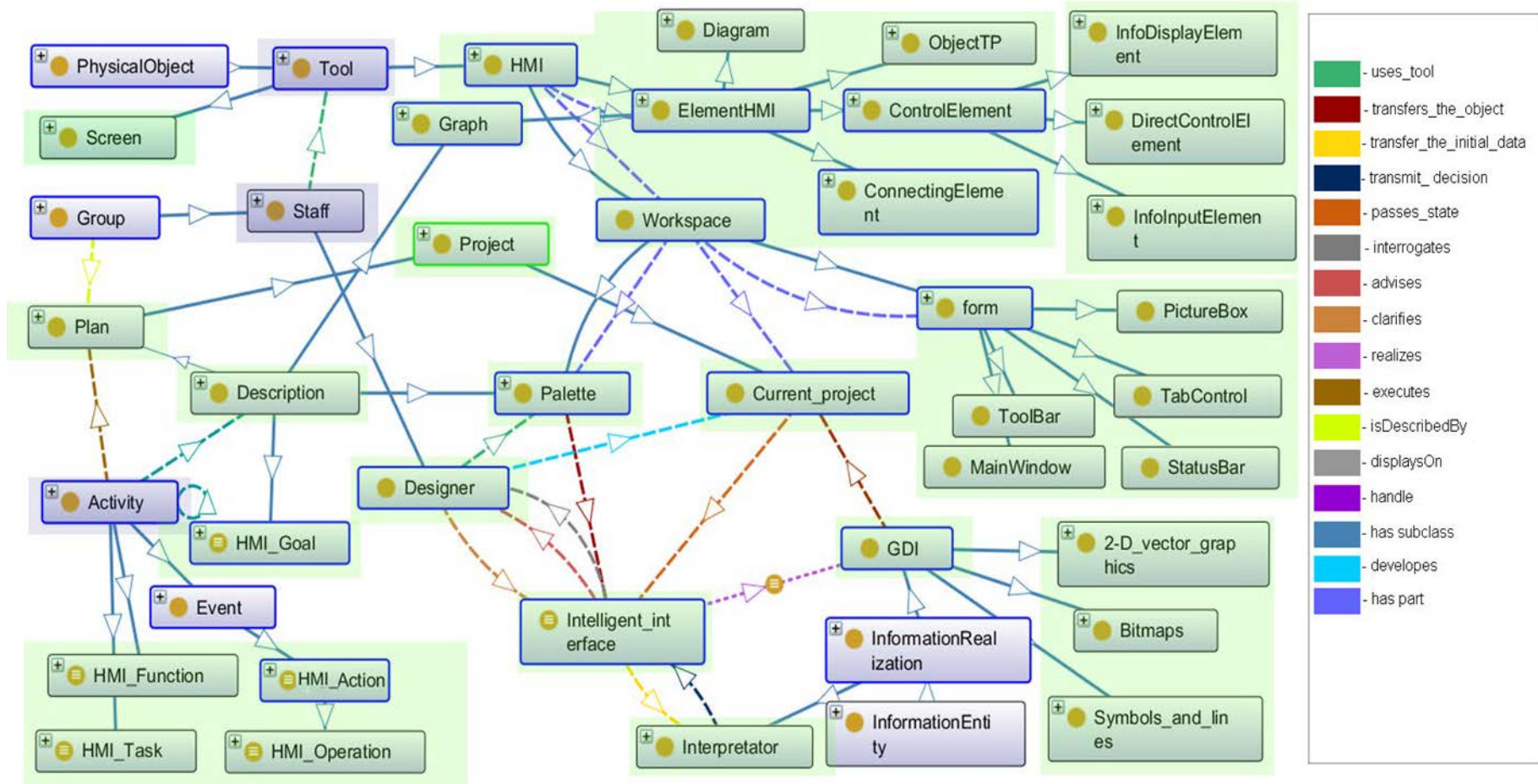


Рисунок 3.11 – Класи, що стосуються проектування ЛМІ

Зображені на рис. 3.11 класи Document, Trend, Mnemoscheme приєднано до класу DUL:Diagnosis, що визначається як опис ситуацій у системі, які класи Document, Trend та Mnemoscheme і будуть описувати.

Клас GDI та Interpretator приєднано до DUL:InformationRealization, виходячи з того, що GDI є програмними засобами створення користувацького інтерфейсу, а Interpretator є програмним засобом, що реалізує механізм виводу.

Для проектування інтерфейсу також існує техпроцес, що включає п'ять вищезазначених рівнів, та має ціль HMI_Goal, яку долучено до класу DUL:Description, з урахуванням того, що вона є еквівалентим DUL:Goal.

Блакитним блоком виділено клас Activity, що було попередньо створено на етапі розробки онтології техпроцесу, як клас, що визначає діяльність при виконанні техпроцесу. До нього приєднано HMI_Function та HMI_Task. До класу DUL:Event додано HMI_Action, а до HMI_Action HMI_Operation.

3.5 Вирівнювання онтології

Вирівнювання (або узгодження) онтології – це процес визначення відповідності між поняттями в онтологіях. Набір відповідностей також називають вирівнюванням [21].

Потреба у вирівнюванні онтологій виникає через необхідність інтеграції неоднорідних баз знань та даних, розроблених незалежно одна від одної таким чином, що кожна має власний словниковий запас даних. Інструменти вирівнювання онтології знаходять класи даних, семантично еквівалентні, наприклад, «машина» та «автомобіль». Класи не обов'язково логічно ідентичні.

У конкретному випадку онтології, що стосується предметних областей «Технологічний процес» та «Людино-машинний інтерфейс», мова йде не лише про вирівнювання онтології DUL за іншими онтологіями, що мають необхідні класи, зв'язки або структури, що задовольняють потребам, але й про розширення

онтології класами та зв'язками, що не належать до якоїсь конкретної онтології, а визначені як поняття у предметній області. Це також можна назвати багатомовним вирівнюванням.

3.5.1 Вирівнювання онтології «Технологічний процес» з DUL

Базова онтологія DUL має гарний фундамент щодо вирівнювання її класів з необхідними поняттями з предметної області «Технологічний процес», а також чаткової модифікації ієрархії у DUL з переосмисленням деяких класів.

Структура розробленої онтології хоч і базувалася на основі структури онтології DUL, проте як було показано у попередніх пунктах, зазнала змін.

По-перше, було переосмислено такі класи, як Task і введено Action, HMI_Action, Operation, HMI_Operation, HMI_Function, HMI_Task, що зараз є дочірніми класами класу Activity, який є базовим класом у розробленій онтології та описує усі типи активностей, що можуть відбуватися.

Клас Event тепер не є базовим класом для активностей, через то, що він мав обмеження, яке не дозволяло створювати події, що володіли б якимись якостями, наприклад, у DUL, клас Event не міг мати зв'язків з класами Object, Abstract та Quality, що робило неможливим роботу з ним в умовах техпроцесу.

Також, клас Event type тепер на має своїм дочірнім класом клас Task. Тепер Task – це підклас Activity, через те, що входить у ієрархію техпроцесу розглянуту у п. 2.2.

По-друге, додано класи, що визначають об'єкти, процеси, ситуації, умови, що мають місце у техпроцесах. Класи, що були введені наведено у табл. 3.1.

Таблиця 3.1 – Введені до онтології класи, що мають відношення до техпроцесу

Назва класу	Опис класу
1	2
Activity	Введений через неможливість класу Event, що є базовим для DUL забезпечити роботу з об'єктами інших класів, тому що клас Event був базовим для усіх класів, які представляють активності.
Goal	Перший рівень ієрархії [18], що описує техпроцес та являє собою рівень цілей, де ціль – це сутність, що описує кінцевий результат функціонування техпроцесу.
Function	Це клас, що описує деяку роботу, що виконується технологічним обладнанням та/або персоналом, має ресурси та стан, і націлена на досягнення цілі. Цей клас був введений для стандартизації опису техпроцесу і є другим рівнем у ієрархії [18], що описує техпроцес.
Operation	Клас, що реалізує найнижчий, п'ятий рівень ієрархії техпроцесу [18], та являє собою найпростішу активність, що більше не може бути розділена на більш прості активності.
TechEvent	TechEvent – клас, що реалізує певну подію, значущу з точки зору виконуваних завдань діагностики і управління.
ControlTask	ControlTask є аналогом класу Task, але при виникненні аварійної ситуації, клас TechEvent ініціалізує одну чи декілька ControlTask, що мають бути досягнені для усунення аварійної ситуації.
Qualitive	Клас, що реалізує якісні параметри, тобто параметри, вимірювання яких дозволяє проводити якісні виміри та дослідження, наприклад: якість виробу, звук повідомлення, якість керування тощо.
Quantitive	Клас, що реалізує параметри, за допомогою яких можливо проводити кількісні виміри та дослідження, наприклад: кількість обертів, напруга, вага тощо.
Resource	Клас, що використовується для опису ресурсів, необхідних для потреб техпроцесу, що являють собою різноманітні засоби, що можуть виконувати роль ресурсів.

Продовження таблиці 3.1

1	2
Exploited	Клас, що реалізує один із видів ресурсів - ресурси, що використовуються, але не перероблюються, протягом техпроцесу. Вони є лише засобами за допомогою яких виконується техпроцес, наприклад станок, оператор, програмне забезпечення, креслення.
Recyclable	Клас, що реалізує вид ресурсів, що перетворюються протягом техпроцесу, наприклад: електроенергія, метал, суха суміш для виробки газобетону тощо. Такі ресурси протягом техпроцесу переробляються у щось інше, наприклад: з листу металу виготовляють трубу.
State	Клас, що є концептом, який визначає безліч чітких значень змінних параметрів об'єкту. Тобто, кожна сутність має деяку кількість станів, що характеризуються значеннями параметрів (температура, розмір, положення у просторі та часі і т.д)
FunctionState	Клас, що конкретно реалізує клас State для класу Function для стандартизації станів функції. Це потрібно для реалізації можливості відстежувати конкретний стан функції у конкретний момент часу. Вони розподіляються на п'ять підкласів, зазначених нижче.
InCondition	Клас стану, що задовольняє потреби техпроцесу до функції у даний момент.
NotReady	Клас стану, що визначає функцію, яка справна, але на даний момент не готова до виконання.
Passive	Клас стану у якому функція не виконується.
Ready	Клас стану, що описує функцію, яка готова до виконання.
Wrong	Клас стану, коли функція не задовольняє потребам техпроцесу на даний момент
ObjectState	ObjectState є аналогом FunctionState, але для об'єкту. Клас був створений задля можливості відстежувати конкретний стан об'єкту у конкретний момент часу протягом техпроцесу.

Продовження таблиці 3.1

1	2
Abnormal	Клас стану об'єкту, що описує аномальний стан об'єкту, який не є критичним або небезпечним, але і не є нормальним.
Dangerous	Клас, що реалізує небезпечний стан об'єкту (об'єкт є небезпечним для техпроцесу, навколишнього середовища або для себе)
Emergency	Клас, що описує критичний стан об'єкту, що описує об'єкт, який знаходиться у аварійному стані, що має бути усуненим.
Not_working	Клас, що описує стан, який описує об'єкт, що не є працездатним.
Working	Клас, що описує стан об'єкту, який є нормальним та робочим у даний момент.
ToolState	Клас ToolState описує стан інструментів, що є учасниками техпроцесу. Якихось передвизначених станів немає через велику кількість інструментів, що можуть брати участь у техпроцесі.
Product_of_labor	Клас, що являє собою продукт праці: фізичний об'єкт, інформація, подія і т.іню Він є ключовим класом, що є результатом якоїсь Function, Task, Action, Operation тощо. Цей клас був введений задля можливості задавання чогось як результат виконання якоїсь активності техпроцесу.
Tool	Клас, що описує знаряддя праці, що частину засобів виробництва, за допомогою яких відбувається впливає на об'єкти, предмети праці. Знаряддя праці головна частина коштів виробництва. Включає інструменти, машини, прилади, двигуни і т.п., з допомогою яких в процесі виробництва обробляються предмети праці, виготовляється продукція.

Рисунок 3.12 зображує класи, що введені чи модифіковані у DUL для отримання розробленої онтології. Помаранчевим кольором зазначені класи, що вже були у DUL та не зазнали змін протягом розробки. Блакитним кольором позначені класи, що були змінені протягом розробки, а зеленим кольором

зазначені класи, що були додані до онтології. Жовтим – класи, що було перенесено до інших класів, та тим самим переосмислені.

До класу DUL:Entity, що є основним класом для усіх інших класів DUL (його не зазначено на графі, через велику кількість зв'язків з іншими класами, що ускладнювало б розуміння графу) було додано клас Activity, що включив в себе Event, який містить аналогічні класи з DUL – Action та Process та новий NMI_Action. Activity був створений задля узагальнення усіх видів активностей в одному базовому класі, що дозволяв би під'єднати усі інші класи, що описують події. Узагальнення було необхідно для отримання можливості роботи з усіма Activity, минаючи обмеження класів, що були задані DUL, наприклад неможливість класів мати зв'язки з іншими класами.

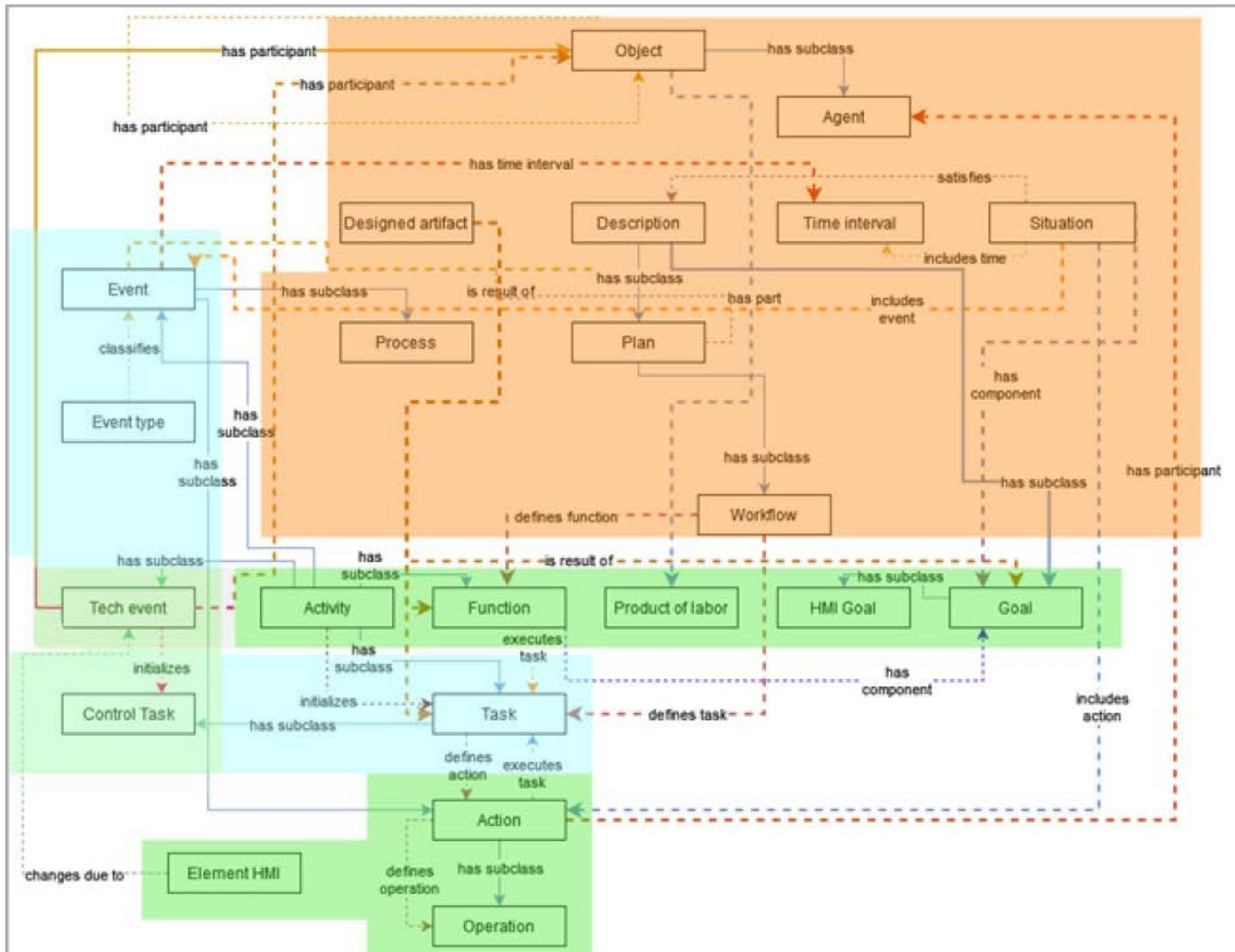


Рисунок 3.12 – Зображення графу онтології з позначеними класами, що були додані та змінені

До Action був доданий клас на який він розділяється на нижчому рівні – Operation, а до HMI_Action – HMI_Operation, відповідно до ієрархії ТП [18].

До Event також додано підклас TechEvent, що є класом для реалізації технологічної події, а саме можливих аварійних ситуацій в техпроцесі [18].

До Activity було внесено Task, що змінив власне призначення з дії (Action), що має бути виконана, на дію, що буде виконуватись. Батьківський клас Task був в онтології DUL Event type - став Activity.

Так як Activity є базовим класом, то усі інші активності є його підкласами та також можуть мати інші зв'язки з ним, наприклад: Activity має зв'язок initializes з класом Task.

Інші ж класи, що визначають активності (Action, Task, Event тощо) змінилися відповідно з додаванням класу Activity та є його дочірніми класами. Іншою значною зміною до класів, що визначають активності також є структура техпроцесу [18], що була описана раніше.

Такі зв'язки класу Action, як has participant з класом Agent та includes action з класом Situation є зв'язкам, що були у DUL та не зазнали змін.

Клас Function є дочірнім до класу Activity та також був приєднаний до класу Designed artifact зв'язком is result of, що визначає, що результатом виконання певної Function може бути деякий Designed artifact. Також Function є класом, що зв'язаний з класом Workflow зв'язком defines function, який описує, що Workflow визначає цю конкретну Function.

Аналогічний зв'язок також мають класи Goal та Task. Також клас Workflow має зв'язок defines Task з класом Task, яким він визначає конкретну задачу, що має відношення до технологічного процесу Workflow.

Клас TechEvent, що визначає технологічну подію, окрім зв'язків з класом Activity також отримав зв'язок has participant з класом Object, через те, що у будь-якому TechEvent є учасником деякий об'єкт.

Ще одним класом, що був створений є клас Product_of_labor, що описує продукт праці та може бути не лише якимось об'єктом, але і станом об'єкту, явищем, тощо. Тому батьківським класом для нього був обраний клас Entity (не зазначений на графі, бо він є найбільш базовим класом для усіх інших класів у розробленій онтології та DUL) та має зв'язок has part з класом Object.

3.5.2 Вирівнювання онтології «Ерготехнічна система» з DUL

Говорячи про вирівнювання DUL з поняттями, що мають відношення до ETC, було виконано додавання багатьох класів, які не були представлені у DUL.

По-перше, було переосмислено клас Group і введено класи: Staff, Operator, Designer, інші класи, що стосуються розробки ETC та НМІ були об'єднані класами Staff та Tool.

Детально класи ETC описані у таблиці 3.2.

Таблиця 3.2 – Розроблені класи, що мають відношення до ETC

Назва класу	Опис класу
1	2
Operator	Клас, що реалізує конкретну особу, що відноється до групи професій з управління роботою обладнання різного виду і призначення.
Staff	Клас, що реалізує особовий склад або працівників установи, підприємства, що складають групу за професійними або іншими ознаками. Виконує роль персоналу, як сукупності багатьох окремих сутностей (фахівців різного роду діяльності) та об'єднує їх у одну сутність.
Product_of_labor	Клас, що являє собою продукт праці: фізичний об'єкт, інформація, подія и т.п. Він є ключовим класом. Найбільш часто є результатом якоїсь Function, DUL:Task, Action, Operation тощо.
Designer	Проектувальник –фахівець, який створює зовнішній вигляд сайтів, застосунків так, щоб вони були зручні і інтуїтивно зрозумілі.
Tool	Клас, що описує знаряддя праці - засобів виробництва, за допомогою яких відбувається вплив на об'єкти (предмети праці). Включає інструменти, машини, прилади, двигуни і т.п., за допомогою яких в процесі виробництва виготовляється продукт праці.
Resource	Клас, що можна використати для виробництва продуктів праці та для задоволення потреб техпроцесу.

Рисунок 3.13 зображує класи, що введені чи модифіковані у DUL при розробці онтології. Зеленими блоками було позначено додані класи, блакитними – класи, що було переосмислено, а помаранчевими блоками позначені класи, що мають відношення до розробки ЛМІ.

Клас Group був змінений через те, що у DUL він мав надлишковий зв'язок, що дозволяв об'єктам класу Group бути створеними лише за умови, що цей об'єкт класу Group був визначений певним Plan. Також, клас Group мав зв'язок із класом Event, а через те, що Event також був змінений, то і зв'язок Group з Event зазнав зміни.

Аналогічно до класу Group був змінений і клас Community, через те, що мав аналогічні проблеми.

До класу Group було додано клас Staff, через необхідність мати сутність, що визначає персонал, який включив у себе Operator та Designer, а у DUL необхідні класи були відсутні. Ці класи також було зв'язано з уже доданими до онтології, такими як: Tool, Exploited, Function та Resources, що реалізують інструменти та ресурси (ресурси, що перероблюються та ресурси, що використовуються).

Клас Staff, що визначає певний персонал став підкласом класу Group, а не Collective agent. Це обумовлено тим, що клас Staff базово являє собою групу людей, а клас Collective agent оперує більш філософськими та менш конкретними поняттями, а тому не є найбільш відповідним для того, щоб бути батьківським класом для класу Staff, через те, що є недостатньо конкретним на відміну від класу Group.

Було додано клас *Resource*, що визначає деякі ресурси, які будуть використовуватися протягом техпроцесу, який не був наявним у DUL та найбільш відповідним класом для приєднання класу *Resource* до онтології був клас *Physical artifact*, що дозволяє говорити про фізичні “артефакти” в дуже загальному сенсі, тобто включаючи об’єкти, що переробляються, об’єкти з навмисною функціональною зміною, природні об’єкти, яким надається певна функція, навіть якщо вони не модифіковані чи структурно спроектовані.

Клас *Physical artifact* був класом, що найбільш наближений до поняття *Resource*, а тому саме він був обраним як батьківський клас.

Через необхідність використання певних інструментів був створений клас *Tool*, що був приєднаний до класу *Social agent* зв’язком *uses_tool*, через те, що об’єкти класу *Social agent* та його дочірніх класів будуть використовувати деякі *Tool* протягом техпроцесу.

Зв’язок *uses_tool* є дочірнім від зв’язку *uses*, якого не було в DUL, та був створений власноруч задля можливості опису використання об’єктів іншими сутностями системи. У початковій версії DUL був зв’язок *uses concept*, що не підходив для загальних цілей, бо описував зв’язок між *Description* та *Concept*, що є гарним рішенням для вузьконаправленої задачі, але не може бути застосованим до більш широкого кола ситуацій. Тому було створено зв’язок *uses*, що є батьківським для інших зв’язків *uses*, які вже конкретизують, що саме буде використовуватися та чим. Відповідно з цим, зв’язком *uses_tool* клас *Tool* був приєднаний до *Social agent*.

Велика кількість класів, що була додана, стосується проектування ЛМІ, та була описана вище.

3.6 Висновки до третього розділу

У третьому розділі було розглянуто основні рішення щодо побудови багаторівневої онтології на базі метаонтології DUL. Онтологія формалізує знання щодо трьох предметних областей «Технологічний процес», «Ерготехнічна система» та «ЛМІ», необхідних для реалізації інтелектуальної підтримки процесу ергономічного проєктування людино-машинних інтерфейсів АСУТП. Розглянуто також питання вирівнювання онтологій. Онтологія побудована за допомогою редактору онтологій Protege.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РОЗРОБЛЕНОЇ ОНТОЛОГІЇ

4.1 Приклад онтології технологічного процесу виробництва газобетону

Онтологічна система була перевірена на прикладі техпроцесу виробництва газобетону. В розроблену у розділі 3 онтологію було введено знання про наступні п'ять рівнів, що описують ТП.

Рівень «Ціль»: отримання газобетонової суміші у форму.

Рівень «Функції»:

- пуск: вхід в систему, налаштування запуску, налаштування параметрів, вибір рецепту;
- дозування сухих компонентів: набір цементу в дозатор сухих, набір ВКВ у дозатор; вивантаження компонентів з дозатора в змішувач. Ці компоненти дозуються послідовно;
- дозування шламу: набір прямого шламу в дозатор шламу; вивантаження шламу з дозатора в змішувач. Набір води в дозатор шламу і вивантаження води з дозатора в змішувач. Ці компоненти дозуються і вивантажуються послідовно;
- дозування суспензії: набір суспензії в дозатор суспензії; вивантаження суспензії з дозатора суспензії в змішувач;
- при пуску циклу починається паралельний набір цементу, суспензії і шламу;
- коли вони набрані (цементу, суспензії і шламу) запускається процес вивантаження і вивантажується шлам. Після цього набирається в дозатор шламу і відразу вивантажується в змішувач вода.

Як тільки набирається ВКВ, вивантажуються сухі компоненти з дозатора сухого. Після закінчення часу перемішування (задається оператором)

вивантажується суспензія. Після закінчення часу остаточного перемішування (задається оператором) вивантажується вся суміш у форми.

Для задання часу, що необхідний для виконання Function/Task/Action/Operation тощо, були взяті приблизні значення часу необхідні для виконання цих Function/Task/Action/Operation.

Хто забезпечує виконання функції а): оператор; пульт управління (операційна панель); видатковий бункер цементу (з датчиками верхнього та нижнього рівня), пневмообвалювачі; шнек цементу з Ел.двигуни; частотний перетворювач (ПЧ) керуючий швидкістю обертання двигунів цементу і ВКВ; заслінка подачі цементу в дозатор (з датчиками положення «відкрито» - «закрита»); видатковий бункер ВКВ (з датчиками верхнього та нижнього рівня), пневмообвалювачі ВКВ; шнек ВКВ з Ел.двигуни; заслінка подачі ВКВ в дозатор дозатор (з датчиками положення «відкрито» - «закрита»); дозатор з тензодатчиками, пневмообвалювачі дозатора; заслінка вивантаження (з датчиками положення «відкрито» - «закрита»).

Хто забезпечує виконання функції б): оператор; видаткова ємність прямого шламу 1 (з датчиками верхнього та нижнього рівня); заслінка подачі шламу 1 в дозатор (з датчиками положення «відкрито» - «закрита»); видаткова ємність прямого шламу 2 (з датчиками верхнього та нижнього рівня); заслінка подачі шламу 2 в дозатор (з датчиками положення «відкрито» - «закрита»); заслінка подачі шламу в дозатор (з датчиками положення «відкрито» - «закрита»); заслінка подачі води в дозатор (з датчиками положення «відкрито» - «закрита»); дозатор з тензодатчиками; заслінка вивантаження (з датчиками положення «відкрито» - «закрита»)

Хто забезпечує виконання функції в): оператор; пульт управління (операційна панель); видаткова ємність суспензії (з датчиками верхнього та нижнього рівня), пневмообвалювачі; шнек цементу з Ел.двигуни; заслінка подачі суспензії в дозатор (з датчиками положення «відкрито» - «закрита»); дозатор з

тензодатчиками, пневмообвалювачі дозатора; заслінка вивантаження (з датчиками положення «відкрито» - «закрита»).

Хто забезпечує виконання функції г): оператор; змішувач з електродвигуном (блок контакторів показує, що подано живлення, тепловий захист - при досягненні певної температури спрацює захист - відключиться живлення двигуна. За цим сигналом - виводиться повідомлення і забарвлюється червоним ум. Позначення двигуна); вихідна заслінка (з датчиками положення «відкрито» - «закрита»).

Рівні «Задачі» та «Дії» оператора.

Нижче наведені задачі та дії оператора, де пункти з нумерацією. першого рівня – це Task, а з нумерацією другого рівня – це Action оператора.

Task 1 Вхід до системи.

Action 1.1 Авторизація оператора.

Task 2. Встановлення параметрів налаштування.

Action 2.1 Налаштування дозатора сухих.

Task 3. Налаштування запуску.

Action 3.1 Налаштування пульта керування.

Task 4. Запуск процесу.

Action 4.1 Перевірка і запуск.

Task 5. Встановлення рецепту.

Action 5.1 Вибір рецепту.

Task 6. Набір сухих компонентів у дозатор.

Action 6.1 Набір цементу в дозатор сухих.

Action 6.2 Набір ВКВ в дозатор сухих.

Task 7. Набір шламу у дозатор.

Action 7.1 Набір прямого шламу в дозатор з басейну 1.

Action 7.2 Набір прямого шламу в дозатор з басейну 2.

Task 8. Набір суспензії у дозатор.

Action 8.1 Набір суспензій в дозатор суспензій.

Task 9. Вивантаження і змішування шламу і сухих.

Action 9.1 Вивантаження шламу у змішувач.

Action 9.2 Вивантаження сухих компонентів у змішувач.

Action 9.3 Вивантаження води у змішувач.

Action 9.4 Набір води у дозатор шламу.

Action 9.5 Вивантаження води у змішувач.

Task 10. Вивантаження і змішування суспензії.

Action 10.1 Вивантаження суспензії.

Action 10.2 Остаточне змішування.

Task 11. Вивантаження суміші у форму.

Для реалізації ієрархії техпроцесу у системі були створені об'єкти, що реалізують техпроцес створення газобетону та їх ієрархію. Наприклад, створено об'єкти класу Goal «Підготовка_до_робочого_режиму», «Отримання_газобетонної_суміші_в_форму» та їх зв'язки: *definesFunction*, що визначає функцію, яка має бути виконана для досягнення цієї Goal та *is_precondition_of*, що визначає, що досягнення Goal «Підготовка_до_робочого_режиму» є передумовою для того щоб перейти до Goal «Отримання_газобетонної_суміші_в_форму».

Також створено об'єкти класу Function та їх зв'язки. Наприклад, Function «2. Дозування_суспензії» має зв'язок *HasResult*, що визначає, що результатом успішного виконання функції є заповнення резервуару суспензії. Зв'язок *is_function_defined_in* визначає, що ця функція є визначеною у Goal «Отримання_газобетонної_суміші_в_форму». Зв'язок *includes object* визначає, який об'єкт приймає участь у виконанні Function.

Зв'язок *directly precedes* визначає, що ця функція виконується безпосередньо перед «4. Вивантаження та змішування». Аналогічно до Goal зв'язок *has Result* визначає результат Function. Зв'язки *isParallelTo* визначають,

що ця Function відбувається паралельно з «2. Дозування_суспензії» та «Дозування_сухих_компонентів».

Кожна з функцій визначає одну чи декілька Task, що мають бути вирішені, та може мати багато зв'язків з іншими Function, Task, Result тощо.

У випадку техпроцеса виробництва газобетону у розробленій системі функції будуть виконуватися у наступному порядку: «Підготовка_до_пуску» -> «Пуск» -> «Дозування_сухих_компонентів» -> «Дозування_суспензії» -> «Дозування_шлама» -> «Вивантаження та змішування». Причому функції «Дозування_сухих_компонентів», «Дозування_суспензії» та «Дозування_шлама» будуть відбуватися паралельно.

Кожна з Task також має велику кількість зв'язків, що визначають умови, передумови, об'єкти, що приймають участь тощо. Наприклад Task «Набір_сухих_компонентів_у_дозатор». Для неї зв'язки *includes object* визначають об'єкти, що потрібні для виконання функції. Зв'язки *definesAction* описують, які Action мають бути виконані для виконання Task.

Далі, кожна з Task розбивається на деяку кількість Action, а кожна Action на рівень «Operation». Тому, приклад розвернення ієрархії до кроку Operation буде приведений для одного Action, інші ж Action, Task та Function розгортаються до Operation аналогічним чином.

У якості прикладу розглянемо Action «Набір_суспензії_в_дозатор_суспензії», що є Action у Task «Набір_суспензії_в_дозатор».

Action «Набір_суспензії_в_дозатор», у свою чергу визначає три Operation, що потрібні для виконання цього Action:

- «Відкривається_заслінка_бункера_підготовки_суспензії»;
- «Затримка_на_час_вивантаження_суспензії_з_бункера_суспензії»;
- «Закривається_заслінка_подачі_суспензії_в_дозатор_суспензії».

Operation «Відкривається_заслінка_бункера_підготовки_суспензії_для_подачі_суспензії_в_дозатор» передуює Operation «Затримка_на_час

_вивантаження_суспензії_з_бункера_підготовки_суспензії», що означає, що одразу після успішного виконання Operation «Затримка_на_час_вивантаження_суспензії_з_бункера_підготовки_суспензії» почнеться виконання Operation «Закривається заслінка подачі суспензії_в_дозатор».

4.2 Приклад реалізації аварійної ситуації

Протягом техпроцесу можуть виникати технологічні події, що потребують усунення оператором:

- аварія електричного приводу (попадання чужорідного тіла у шнек, згоряння двигуна);
- перевірити чи не відчинились або не зачинились заслінки за відведений час відсутності пневмообвалення;
- перевищення часу дозування.

Причини аварії 3:

- попадання чужорідного тіла до електродвигуна;
- перевищення тертя у підшипниках;
- перетворювач частоти (ПЧ) видає сигнал про аварію.

Параметр: Процент від номінальної частоти 50 Гц, яка подається на двигун дорівнює 0.

Для реалізації цих аварій були введені наступні об'єкти.

TechEvent «Перевищено_час_дозування_(набір_або_вивантаження)_компонента» ініціалізує ControlTask «Усунення_помилки_перевищення_часу_дозування» та має дві причини (*hasReason*).

ControlTask «Усунення_помилки_перевищення_часу_дозування» визначає дії, що мають бути виконані для вирішення аварійної ситуації.

Також, ControlTask «Усунення_помилки_перевищення_часу_дозування» визначає учасників, що задіяні у ControlTask:

- *has participant* «Кнопка_ «Аварія»;
- *has participant* «Кнопка_ «Пуск»;
- *has participant* «Кнопка_F14»;
- *has participant* «Кнопка_«Деблокування».

Дія ControlTask починається одразу після TechEvent «Перевищено_час_дозування_(набір_або_вивантаження)_компонента».

Кожна з дій у свою чергу визначає Operation, що мають бути виконані для того щоб була виконана Action, Наприклад, Action «Аналіз_ходу_набору_і_визначення_причин_помилки_набору» визначає дві Operation, що мають бути виконані.

Процес ліквідації аварійної ситуації конкретніше буде описаний нижче.

Operation, у свою чергу, є найпростішими сутностями, техпроцесу і, по суті, виконання техпроцесу – це виконання визначеної кількості Operation у правильній послідовності, що мають привести до досягнення цілі.

Нижче описано більш структурований алгоритм вирішення аварійної ситуації при її виникненні.

Повідомлення при аварії 3: «Не спрацював контактор КМ№1 або №2».

Нумерація першого рівня – це Action, а нумерація другого рівня є Operation.

Усунення технологічних подій оператором:

а) виявлення стану «несправна» для функції 3:

1) перевірка миготіння кнопки «Аварія». У разі миготіння, перехід до б) 1), інакше «Кінець»;

2) зчитування повідомлення про помилку №13 «Помилка набору ваги дозатора сухих!» у полі аварійних повідомлень-помилки;

б) перехід на потрібний екран для діагностування:

1) натискання кнопки F14 на операторній панелі, щоб повернутися до головного меню;

2) затримка на час відкриття форми «Стартовий екран дозування»;

3) вибір дозатора сухих за допомогою натискання відповідної сенсорної кнопки;

4) затримка на час відкриття форми «Дозатор сухих»;

в) контрольне повторення набору компонентів у дозатор з детальним моніторингом:

1) зчитування з фіксацією в пам'яті/папері показань поточної ваги компонентів;

2) натискання кнопки «ДЕБЛОКУВАННЯ»;

3) натискання кнопки «ПУСК» для повтору набору компонентів;

г) аналіз ходу набору та визначення причин помилки набору (якщо вона з'явиться):

1) перевірка роботи (візуально по мнемосхемі форми «Дозатор сухих») в автоматичному режимі заслінок на виході дозатора сухих, шнекового живильника або появи інших помилок в полі аварійних повідомлень. Якщо заслінка відкрилася, шнек спрацював і немає інших помилок, то перехід на е) 13);

г) визначення і усунення причини 2 (не відчинилися заслінки):

1) перевірка тиску стисненого повітря. Якщо воно в нормі, то перехід на крок 3). Інакше зателефонувати в компресорну, дочекатися поки тиск не збільшиться, перехід на в) 3);

2) переключити тумблер установки режиму управління на пульті в положення «РУЧ»;

3) відкрити заслінку на вивантаження за допомогою натискання кнопки F6 на операторній панелі;

- 4) затримка на час відкриття заслінки;
- 5) якщо заслінка відкрилася, то перехід на е) 1), інакше натиснути кнопку F8 на операторній панелі для закриття заслінки;
- 6) виклик механіків для огляду/ремонту механізму. Після усунення поломки перейти на г) 5);
- д) визначення і усунення причини 1:
 - 1) включення приводу шнека, натискаючи та утримуючи кнопку F2 на операторній панелі;
 - 2) перевірка роботи шнека (візуально по мнемосхемі форми «Дозатор сухих»). Якщо шнек працює перейти на е) 3), інакше натиснути кнопки F4 і F8 на операторній панелі, для зупинки шнека і закриття заслінок відповідно, перейти до д) 2);
- е) визначення і усунення причин зависання або відсутності матеріалу:
 - 1) зчитування показань поточної ваги компонентів. Якщо вага не зростає, то е) 4), інакше крок е) 10);
 - 2) включення пневмообвалювачів натисканням сенсорної кнопки «Обрушення» на екранній формі «Дозатор сухих» (робоча операція P13);
 - 3) зчитування показань поточної ваги компонентів. Якщо вага не зростає, то е) 6), інакше крок е) 10);
 - 4) натискання кнопок F4 і F8 на операторній панелі для зупинки шнека і закриття заслінок;
 - 5) дзвінок обслуговуючому персоналу для візуальної перевірки наявності матеріалу в бункері. Очікування відповіді. Якщо матеріалу немає, то е) 9);
 - 6) ліквідація обслуговуючим персоналом зависання матеріалу в бункері сухих механічним способом, перехід на г) 5);
 - 7) заповнення обслуговуючим персоналом бункера матеріалом. Після підтвердження наявності матеріалу перехід на г) 5);

- 8) натискання кнопок F4 і F8 на операторній панелі для зупинки шнека і закриття заслінок;
- 9) переведення функції у звичайне положення з моніторингом відновлення динаміки набору;
- 10) переключити тумблер установки режиму управління на пульті в положення «АВТ»;
- 11) натискання кнопки «ДЕБЛОКУВАННЯ» на пульті для скидання аварійного повідомлення;
- 12) натискання кнопки «ПУСК» на пульті для повтору набору компонентів
- 13) перевірка динаміки зміни ваги компонента в дозаторі за показаннями у відповідному полі мнемосхеми форми «Дозатор сухих».

4.3 Реалізація технології проєктування ЛМІ

Таблиця 4.1 показує, як реалізована технологія проєктування ЛМІ з використанням ієрархії Function-Task-Action-Operation на прикладі функції “Кодування інформації у зоні”. Кожна з функцій може бути розкрита багатьма способами через наявність продукційних правил, що описують як необхідно проєктувати інтерфейс та що використовувати.

Правила слугують для визначення елементів, кольорів, шрифтів та інших властивостей елементів проєктування інтерфейсу, при його проєктуванні.

В залежності від функції, що виконується, є критерії, що задаються користувачем чи системою. За допомогою логічного виводу проєктувальник отримує рекомендації від системи як бажано закодувати об’єкт.

Таблиця 4.1 – Ієрархія виконання функції «Кодування інформації у зоні» та задачі «Кодування інформації у зоні «Мнемосхема».

Function	Task	Action	Operation
1	2	3	4
26 Кодування інформації в зоні	261 Кодування інформації в зоні «Мнемосхема»	2611 Вибір об'єкта для кодування в зоні «Мнемосхема»	26111 Вибір типу об'єкта для кодування в зоні «Мнемосхема»
		2612 Визначення властивостей і характеристик для кодування об'єкта Об'єкт ТП в зоні «Мнемосхема»	26121 Визначення необхідності дублювання зорової модальності слуховий модальністю
			26122 Визначення специфіки виконуваних завдань.
			26123 Вибір виду кодування Об'єкта ТП на підставі завдань
			26124 Отримання правил і рекомендацій
			26125 Кодування властивостей і характеристик об'єкта відповідно до правил
		2613 Визначення властивостей і характеристик для кодування об'єкта елемент Керування в зоні «Мнемосхема»	26131 Вибір типу елементу керування
			26132 Вибір примірника елемента управління
			26132. Визначення специфіки виконуваних завдань.

Продовження таблиці 4.1

1		2	3	4
				26133 Вибір виду кодування елемента управління на підстав завдань
				26134 Отримання правил і рекомендацій
				26135 Кодування властивостей і характеристик елемента управління відповідно до правил
		262 Кодування інформації в зоні «Тренд»	2621 Вибір об'єкта для кодування в зоні «Тренд»	
			2622 Визначення властивостей і характеристик для кодування об'єкта в зоні «Тренд»	
		263 Кодування інформації в зоні «Документ»	2631 Вибір об'єкта для кодування в зоні «Документ»	
			2632 Визначення властивостей і характеристик для кодування об'єкта в зоні «Документ»	

Розробка ЛМІ має одну NMI_Goal, яка визначає функції, що мають бути виконані для її досягнення.

Для досягнення NMI_Goal мають бути виконані функції, що вона визначає:

- «Визначення_кількості_екранів»;
- «Визначення_кількості_зон_на_екрані»;
- «Визначення_кількості_інформації_в_зоні2»;
- «Виділення_зони»;
- «Вибір_розміщення_зони»;
- «Кодування_інформації_в_зоні».

Відповідно, після виконання визначених HMI_Function, HMI_Goal буде досягнута.

Кожна з цих функцій визначає Task, що зазначені у таблиці 4.1. Для більш докладного пояснення буде описана HMI_Function – «Кодування_інформації_в_зоні».

HMI_Function «Кодування_інформації_в_зоні» визначає Task, що мають бути виконані для виконання HMI_Function, і вони є ідентичними лише за відмінністю зони для якої буде проводитися кодування інформації.

Також задля коректної роботи системи логічного виводу HMI_Function має свою властивість даних *hasName* «Кодування_інформації_в_зоні» задля можливості пошуку цього об'єкту системою (нумерація об'єктів не є послідовністю виконання, а необхідна задля більш легкої орієнтації у системі, через неможливість групування об'єктів за якимись властивостями у редакторі онтологій Protege).

Задля виконання HMI_Task «Кодування_інформації_в_зоні»_«Мнемосхема» мають бути виконані три HMI_Action.

Аналогічно до розглянутої вище HMI_Task, інші HMI_Task також мають свої Hmi_Action, що мають бути виконані. Розглянемо більш детально HMI_Task «Кодування_інформації_в_зоні_«Мнемосхема»». Відповідно до ієрархії техпроцесу, кожна з HMI_Task визначає HMI_Action, а HMI_Action, у свою чергу, визначає HMI_Operation.

Як зазначено вище, HMI_Task «Кодування_інформації_в_зоні»_«Мнемосхема», визначає три HMI_Action:

- «Визначення_властивостей_і_характеристик_для_кодування_об'єкта_Об'єктТП_в_зоні_«Мнемосхема»;
- «Визначення_властивостей_і_характеристик_для_кодування_об'єкта_ЕлементУправління_в_зоні_«Мнемосхема»;
- «Вибір_об'єкта_для_кодування_в_зоні_«Мнемосхема».

4.4 Приклад реалізація кодування інформації у зоні

Як і у попередніх випадках, визначмо необхідні для досягнення цілі рівні ієрархії.

Ціль: Розробка ЛМІ.

Для досягнення цілі розробки ЛМІ визначено перелік виконуваних дій - функцій, що задовольняють ціль, а також задіюють ресурси.

Функції:

- визначення кількості екранів;
- визначення кількості зон на екрані;
- визначенні кількості інформації в зоні;
- виділення зони;
- вибір розміщення зони;
- кодування інформації в зоні.

Для введення вищезгаданих цілі та функцій використані сутності онтології HMI_Goal та HMI_Function.

Функція №6 кодування інформації в зоні визначає задачі кодування інформації у зонах «Мнемосхема», «Документ» та «Тренд», що у онтології

представленні об'єктами сутності HMI_Task та пов'язані з HMI_Function зв'язком *definesFunction*.

Задача кодування у зоні «Мнемосхема» декомпозується на такі дії, як: вибір об'єкта для кодування в зоні «Мнемосхема»; визначення властивостей та характеристик для кодування об'єкта ТП у зоні «Мнемосхема», а також визначення властивостей та характеристик кодування елементу керування в зоні «Мнемосхема». Дії у онтології представлені сутністю HMI_Action та пов'язані з HMI_Task.

Задачі кодування у зоні «Тренд» та «Документ» поділяються на дії: вибір об'єкта ТП для кодування у зоні «Тренд»/«Документ» та визначення властивостей та характеристик для кодування об'єкта ТП у зоні «Тренд»/«Документ». Виконання задачі вибору об'єкту для кодування в зоні «Мнемосхема» визначає операція вибору типу об'єкту для кодування в цій зоні.

Задача визначення властивостей та характеристик для кодування об'єкта ТП у зоні «Мнемосхема» декомпозується на операції: визначення необхідності дублювання зорової модальності слуховою модальністю та специфіки виконуваних завдань; застосування правил 500-503, а також операції: вибору виду кодування об'єкта ТП на підставі завдань; виконання правила 504, що стосується рекомендації дублювати всі вибранні види кодування (окрім кольору) кольором, та операції кодування властивостей та характеристик об'єкта відповідно до правил.

Задача визначення властивостей та характеристик кодування елементу керування в зоні «Мнемосхема» визначає операції: вибір типу елемента керування; визначення специфіки завдань, що виконуються; вибір виду кодування елемента управління на підставі завдань; отримання правил та рекомендацій, а також кодування властивостей та характеристик елемента управління відповідно до правил.

Задачі визначення властивостей та характеристик для кодування об'єкту ТП та елементів керування паралельні, що свідчить про перехід на ту гілку, об'єкт для якої було обрано на задачі вибору об'єкта для кодування в зоні «Мнемосхема».

Операції керуються правилами, що визначенні у онтології сутністю Rules. Правила, що доцільно застосовувати на певних операціях наведено у таблиці 4.2.

Розглянемо приклад кодування об'єкту у зоні за вищезазначеними кроками.

На кроці 2.6 за правилом П1001 необхідно обрати об'єкт для кодування у зоні мнемосхеми. У якості прикладу обрано тип елементу керування Button. Проєктувальник може отримати інформацію щодо властивостей елементу Button за допомогою запиту до бази знань системи (рис. 4.1).

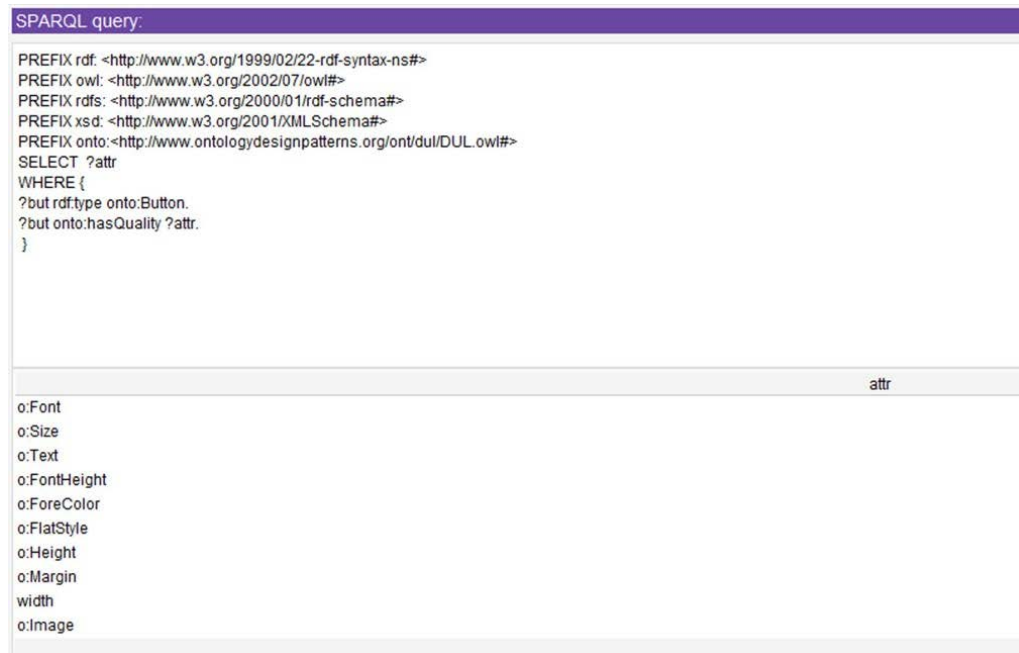


Рисунок 4.1 – Приклад запиту до бази знань системи

Як видно з рис. 4.1 у якості відповіді на запит, проєктувальник отримує список властивостей елементу Button, таких як Font, Size, Text тощо.

Далі за правилом П1002 отримано рекомендацію застосування малонасичених кольорів для фону зони мнемосхеми.

Далі, визначається специфіка задач, що стоїть перед проєктувальником відносно об'єкту, що буде закодовано. Для Button це завдання упізнавання та розрізнення. У цьому випадку правило П1004 рекомендує кодувати Button кольором, текстом, розміром і/або зображенням, а П1003 тільки кольором.

За правилом П1005 якщо кодування відбувається текстом та розміром, то необхідно визначити шрифт (Font), висоту шрифту (FontHeight) та його колір (ForeColor). За цим же правилом рекомендований розмір знаку 15,3 мм, рекомендовані кольори коду знаку та фону представлені наступним списком у порядку убування:

- синій на білому;
- чорний на жовтому;
- зелений на білому;
- зелений на червоному;
- червоний на жовтому;
- помаранчевий на чорному;
- чорний на пурпуровому;
- помаранчевий на білому;
- червоний на зеленому.

Висота Button за правилом не менше ніж мінімальний розмір знаку, а ширина не менше вмісту кнопки.

Операції керуються правилами, що визначенні у онтології сутністю Rules. Правила, що доцільно застосовувати на певних операціях наведено у таблиці 4.2.

Таблиця 4.2 – Приклади правил

Function	Task	Action	Operation	Rules
1	2	3	4	5
26 Кодування інформації в зоні				П1001
	261 Кодування інформації в зоні «Мнемосхема»	2611 Вибір об'єкта для кодування в зоні «Мнемосхема»	26111 Вибір типу об'єкта для кодування в зоні «Мнемосхема»	П1002
		2612 Визначення властивостей і характеристик для кодування об'єкта Об'єктТП в зоні «Мнемосхема»	26121 Визначення необхідності дублювання зорової модальності слуховою модальністю	П500-П503
			26122 Визначення специфіки виконуваних завдань	
			26123 Вибір виду кодування Об'єктаТП на підстав завдань	П504
			26124 Отримання правил і рекомендацій	
			26125 Кодування властивостей і характеристик об'єкта відповідно до правил	

Продовження табл. 4.2

1	2	3	4	5
			26125 Кодування властивостей і характеристик об'єкта відповідно до правил	
		2613 Визначення властивостей і характеристик для кодування об'єкта елементКерування в зоні «Мнемосхема»	26131 Вибір типу елементу керування	
			26132 Визначення специфіки виконуваних завдань	
			26133 Вибір виду кодування елемента управління на підстав завдань	П1003-П1011
			26134 Отримання правил і рекомендацій	
			26135 Кодування властивостей і характеристик елемента управління відповідно до правил	
	262 Кодування інформації в зоні «Тренд»	2621 Вибір об'єкта для кодування в зоні «Тренд»		

Продовження таблиці 4.2

1	2	3	4	5
	263 Кодування інформації в зоні «Документ»	2631 Вибір об'єкта для кодування в зоні «Документ»		
		2632 Визначення властивостей і характеристик для кодування об'єкта в зоні «Документ»		

Далі, так як виконані умови для правила П1006 (обрано об'єкт Button), то необхідно визначити шрифт (Font). Вимоги щодо шрифту будуть наступні:

- відношення ширини цифри до висоти від 1:3 до 3:4 та відношення ширини літери та знака пунктуації до висоти від 3:5 до 1:1;
- оптимальна ширина знаків 0,6-0,7 висоти;
- інтервал між літерами 0,125-0,5 їхньої ширини;
- та якщо значення освітленості від 0 до 100 лк, то букви та цифри повинні бути високі та вузькі.

Для Button також пропонується правило П1007, що також визначає вимоги для шрифту (Font) - літери та цифри мають бути високі та вузькі та утворені прямими лініями. Для визначення вимог для висоти шрифту пропонується правило П1008, необхідно знати показник освітлення. Для величини освітлення менше за 20лк, висота літер та цифр не менше 52` (22,7мм). Якщо величина освітлення більш ніж 20лк та менше 1000лк, то висота літер та цифр не менше 35` (15,3мм).

Далі, якщо для кодування Button було обрано зображення та розмір, пропонується правило П1009 у якому визначені вимоги до зображення, де оптимальний контраст 85%, найменший розмір деталі 6 (2,6 мм), а розрахунок висоти та ширини Button має бути з урахуванням найменшого розміру деталі.

Якщо ж для кодування Button для задачі розрізнення було обрано колір та розмір за правилом П1010 запропоновано один з кольорів: червоний, жовтий, зелений, синій, коричневий або пурпуровий.

Для задачі розпізнавання та кодування кольором та розміром запропоновано правило П1011, де рекомендовано наступні кольори: фіолетовий, блакитний, зелено-блакитний, блакитно-зелений, зелений, жовто-зелений, жовто-зелений, помаранчеве-червоний, червоний або білий.

Рекомендації із зазначених правил надходять до проєктувальника, і з їх застосуванням елемент керування Button кодується у зоні мнемосхеми.

Розглянемо кодування об'єкту керування Button за допомогою онтології.

Спочатку виконується запит за визначення задач, на які поділяється функція кодування інформації в зоні.

Код запиту:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX
onto:<http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#>
SELECT ?function ?task
WHERE {
  ?function onto:hasName ?name.
  ?function onto:definesTask ?task

  FILTER regex(?name, "Кодування_інформації", "i") }
```

Далі, отримується перелік усіх правил, що можуть бути застосовані при виконанні задачі «Кодування інформації _в_ зоні Мнемосхема».

Код запиту:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX
onto:<http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#>
SELECT ?task ?rule
WHERE {
  ?task rdf:type onto:HMI_Task.
  ?task onto:definesRule ?rule.
  ?task onto:hasName ?name.
  FILTER regex(?name, "Мнемосхема", "i")
}
```

Проектувальник може зробити запит до системи, щодо дій по проектуванню при кодуванні інформації у зоні мнемосхеми.

Код запиту:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX
onto:<http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#>
SELECT ?action ?operation
WHERE {
  ?task rdf:type onto:HMI_Task.
  ?task onto:definesAction ?action.
  ?task onto:hasName ?name.
  ?action onto:definesOperation ?operation
  FILTER regex(?name, "Мнемосхема", "i")
}
```

Далі, використовуючи правила, що реалізуються логічним виводом, проєктувальником будуть отримані рекомендації, щодо реалізації кодування об'єкту у зоні.

Правило П1001 надає рекомендацію про необхідність вибору об'єкту для кодування.

Для кодування обрано елемент керування Button у зоні «Мнемосхема», тому наступним кроком буде виконання правила П1002, що стосується фону зони.

Для елементу керування Button визначено ряд правил, виконання яких надає рекомендації проєктувальнику. При виконанні П1003 надається рекомендація щодо кодування Button фоном.

Згідно з правилом П1004 система надає рекомендацію для кодування Button при визначенні для неї завдання «упізнання» певними видами алфавіту.

П1005 надає рекомендації щодо кодування текстом та розміром. Проєктувальник може зберегти рекомендації, які він отримав від системи, у форматі .CSV та відкривати їх у текстовому редакторі.

4.5 Висновки до четвертого розділу

У даному розділі було розглянуто практичного застосування розробленої інформаційної технології ергономічного проєктування ЛМІ SCADA-систем на прикладі розробки інтерфейса оператора лінії виробництва газобетону. Для цього, багаторівнева метаонтологія, була наповнена предметними знаннями для п'яти рівнів «Цілі-Функції-Задачі-Дії_Операції». Продемонстровано відповідні стратегії, системи правил та запитів до бази знань інтелектуальної системи підтримки розробника ЛМІ.

ВИСНОВКИ

В результаті виконання дипломної кваліфікаційної роботи магістра було виконано наступні задачі:

- проведено аналіз проблем ергономічного проєктування ЛМІ АСУТП;
- проведено аналіз сучасних метаонтологій, та виявлена у якості базової метаонтологія DOLCE + DNS UltraLite для подальшого використання;
- розроблено концептуальну модель предметної області "Проектування ЛМІ АСУТП";
- розроблено предметні онтології «Технологічний процес». «ЛМІ» та «Ерготехнічна система» на базі метаонтології DUL;
- проведено вирівнювання розроблених онтологій з метаонтологією DUL;
- розроблено систему запитів до онтології з використанням мови запитів SPARQL;
- адекватність розроблених моделей, перевірено на прикладі проєктування ЛМІ для АСУТП виробництва газобетону.

Наукова новизна результатів дослідження полягає у тому, що вперше запропоновано інформаційну технологію інтелектуальної підтримки процесу ергономічного проєктування ЛМІ, що розробляються за допомогою SCADA-систем. Проведено аналіз структури ЛМІ та виявлено властивості його типових елементів, які можуть бути представлені у вигляді відомих категорій кодування інформаційних моделей. Отримані результати є теоретичною основою розробки експертної системи, що здійснює інтелектуальну підтримку проєктувальника ЛМІ.

Вперше запропоновано багаторівневу онтологічну модель представлення знань, необхідних для інтелектуальної підтримки процесу ергономічного проєктування ЛМІ. Використання метаонтології DUL забезпечило можливість інтеграції онтологій нижчого рівня – предметних

онтологій. Модель предметної області «Технологічні процеси» на відміну від відомої онтології PROV-O була розширена знаннями «Ерготехнічна система», що дозволило описати взаємозв'язки між концептами «ергатичний елемент», «знаряддя праці», «предмет праці», «продукт праці» та «ресурси», необхідні для формального опису технологічних процесів.

Запропоновано систему запитів до онтологічної моделі, що забезпечує користувача (проектувальника ЛМІ) інформацією, релевантною до поточного кроку проектування.

Експериментальні дослідження підтвердили коректність запропонованих методів та ІТ. Достовірність отриманих практичних результатів підтверджена експериментальними дослідженнями створеного програмного забезпечення. Отримані результати дозволили зробити висновок, що поставлені завдання виконані та мета досягнута.

Практична цінність роботи полягає в тому, що на підставі проведених досліджень розроблено програмне та інформаційне забезпечення, що забезпечують підвищення ергономічності ЛМІ SCADA-систем.

Наукові та практичні результати роботи доповідалися на II Міжнародній науково-практичній конференції з комп'ютерного моделювання та інтелектуальних систем (CMIS-2019 – Запоріжжя) та II Всеукраїнській науково-практичній конференції «Перспективні напрямки сучасної електроніки, інформаційних і комп'ютерних систем (MEICS-2017 - Дніпро).

Основні наукові результати роботи опубліковані у матеріалах міжнародної конференції (індексується у Scopus), статті у фаховому науковому виданні України та тезах доповіді Всеукраїнській науково-практичній конференції.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Nkosi, Mfundo & Gupta. Causes and Impact of Human Error in Maintenance of Mechanical Systems / Nkosi, Mfundo & Gupta, P. M. Kapil & Mashinini // MATEC Web of Conferences. – 2020. - №312. – P 27-58. /05001.10.1051/matecconf/202031205001
2. Human error [Electronic resource]. – Access mode: <https://oshwiki.osha.europa.eu/en/themes/human-error>
3. Human Machine Interface (HMIs) in Industrial Automation [Electronic resource]. – Access mode: <https://www.automationreadypanels.com/plc-hmi/human-machine-interface-hmi-in-industrial-automation/>
4. Usability through graphics possibilities [Electronic resource]. – Access mode: <https://www.copadata.com/en/product/zenon-software-platform-for-industrial-automation-energy-automation/visualization-control/usability-through-graphics-possibilities/>
5. SCADA Wincc Text List & Graphic List [Electronic resource]. – Access mode: <https://www.youtube.com/watch?app=desktop&v=DBOGbEx2Ooo>
6. Semy, S. K. Toward the use of an upper ontology for U.S. government and U.S. military domains: An evaluation / S. K. Semy, M. K. Pulvermacher, L. J. Obrst // Technical Report MTR 04B0000063. - The MITRE Corporation. - 2004.
7. Сердюк, С.М. Ергономічні питання побудови людино-машинних систем: навчальний посібник / С.М. Сердюк. – Запоріжжя: ЗНТУ, 2014. - 334 с.
8. Choi, N. A survey on ontology mapping. SIGMOD Record / N. Choi, I-Y Song, H. Han, - 2006. - Vol. 35, №. 3. - P. 34-41.
9. Karabulut, E. Ontologies in digital twins: A systematic literature review. / E. Karabulut, S. Pileggi // Future Generation Computer Systems. - 2024. - Vol. 153. - P. 442-456. DOI: 10.1016/j.future.2023.12.013
10. Otero-Cerdeira, L. Ontology matching: A literature review / L. Otero-Cerdeira, F. Rodríguez-Martínez, A. Gómez-Rodríguez // Expert Systems with Applications. - 2015. - Vol. 42. - P. 947-971.

11. Keet, C.M. The use of foundational ontologies in ontology development: an empirical assessment / C.M. Keet // *Proceedings of Extended Semantic Web Conference*, Springer. - 2011. № 32. – P. 234 – 241.

12. Bernabé, C.H. The use of foundational ontologies in biomedical research / C.H. Bernabé, N. Queralt-Rosinach, V.E. Silva Souza et al. // *Journal of Biomedical Semantics*. - 2023. - Vol.14 (21). – P. 132-148. DOI: [https:// doi.org/10.1186/s13326-023-00300-z](https://doi.org/10.1186/s13326-023-00300-z)

13. Kaminska, Zh. Intellectual support of Control System HMI designers / Zh. Kaminska // *Computer Modeling and Intelligent Systems: Proceedings*. – 2019. – №2353. – P. 277-291.

14. Камінська Ж.К. Аналіз проблем автоматизації процесу ергономічного проектування інформаційних моделей технологічних процесів / Ж.К. Камінська // *Вісник Житомирського державного технологічного університету. Технічні науки. Випуск 1, Житомир. №52*. - 2010. - С.103-108

15. Lavrov, E. Human factors in automated systems. Methodology for the computer system “International Center for the Support of Ergonomic Design and Research”/ E. Lavrov, S. Serdiuk, E. F. Ritter, O. Siryk, A. Presnall, Y. Chybiriak, S. Zarina //In 5th IEEE International Conference on Advanced Information and Communication Technologies (AICT). – Lviv, Ukraine, 1 – 5. DOI:10.1109/AICT61584.2023.10452692

16. Upton, C. Extending ecological interface design principles: a manufacturing case study. / C. Upton, G. Doherty // *International Journal of Human Computer Studies*. - 2008. - №66 (4). – P. 271-286.

17. Anokhin, A. Application of ecological interface design in NPP operator support system. // *Proceedings of the International Symposium on Future I&C for Nuclear Power Plants: ISOFIG 2017*, Gyeongju, Korea, November 26–30, 2017. – P. 251-261.

18. Khadzhibaeva, N. The concept of ergatic systems, their classification and properties. // *Eurasian Scientific Journal*. [Electronic resource]. - №.6, - 2017. - P–153-160. Access mode: <https://journalpro.kz/articles/ponyatie->

ergasticheskikh-sistem-ikh-klassifikatsiya-i-svoystva/

19. Ontology design patterns [Electronic resource]. – Access mode: http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite

20. Osman, I., Yahia, B., Diallo, G. Ontology Integration: Approaches and Challenging Issues, Information Fusion / I. Osman, B. Yahia, G. Diallo. - 2021. - Vol. 71, – P. 38-63. DOI: <https://doi.org/10.1016/j.inffus.2021.01.007>.

ДОДАТОК А
Копії слайдів презентації



Національний університет «Запорізька Політехніка»
Кафедра програмних засобів

ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ ЕРГОНОМІЧНОГО ПРОЄКТУВАННЯ ЛЮДИНО-МАШИННИХ ІНТЕРФЕЙСІВ АСУТП

РОЗРОБИЛА: ст. групи КНТ-123м

Жанна Камінська

КЕРІВНИК: доцент, к.т.н.

Сергій Сердюк

Рисунок А.1 – Слайд №1

Об'єкт, предмет, мета дослідження

- **Об'єкт дослідження** – процес проєктування людино-машинних інтерфейсів (ЛМІ) SCADA-систем
- **Предмет дослідження** – методи проєктування ЛМІ SCADA-систем.
- **Мета роботи** – підвищення ергономічності ЛМІ SCADA-систем на основі нової інформаційної технології проєктування.



НУ «ЗП», Кафедра програмних засобів

2

Рисунок А.2 – Слайд №2

Завдання дослідження

- проаналізувати проблеми ергономічного проєктування ЛМІ АСУТП;
- проаналізувати сучасні онтології, та їх роль в інтеграції знань для людино-орієнтованого проєктування інтерфейсів SCADA;
- розробити концептуальну модель предметної області "Проектування ЛМІ АСУТП";
- розробити предметні онтології «Технологічний процес», «Людино-машинний інтерфейс» та «Ерготехнічна система» на базі метаонтології DUL;
- провести вирівнювання розроблених онтологій;
- розробити систему запитів до онтологій;
- перевірити адекватність розроблених моделей на прикладі проєктування ЛМІ для АСУТП виробництва газобетону.

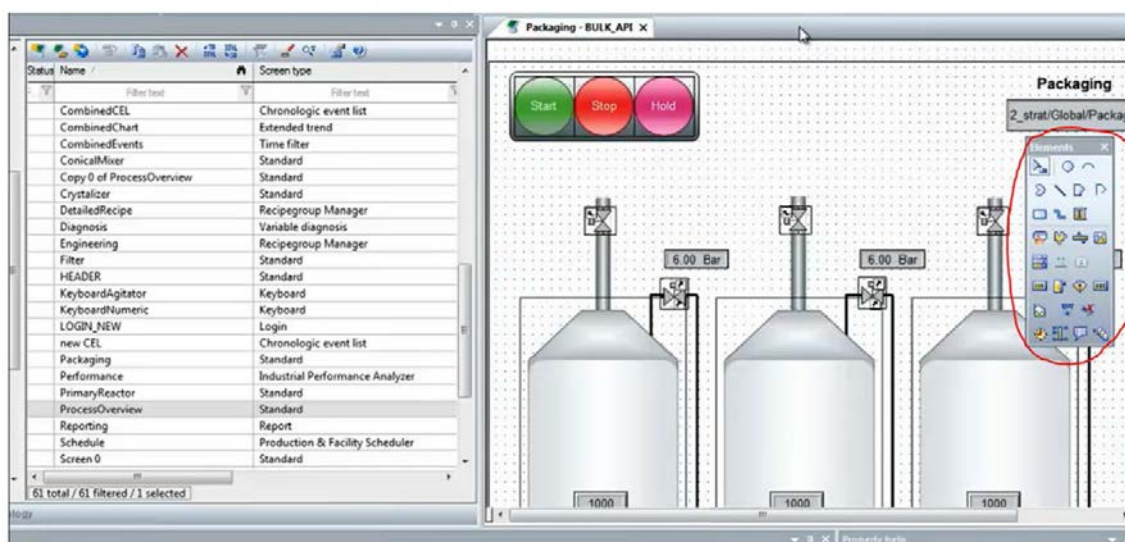


НУ "ЗП", Кафедра програмних засобів

3

Рисунок А.3 – Слайд №3

Приклад бібліотеки елементів zenon SCADA



<https://www.copadata.com/en/product/zenon-software-platform-for-industrial-automation-energy-automation/visualization-control/usability-through-graphics-possibilities/>



НУ "ЗП", Кафедра програмних засобів

4

Рисунок А.4 – Слайд №4

Порівняльний аналіз онтологій

Онтологія	Технологічні процеси	Проектування інтерфейсів SCADA	Ерготехнічні системи
DOLCE+DnS Ontology (DUL)	Підтримує моделювання процесів через структурні події та агентську взаємодію.	Описує відносини агента, артефакту та цілей, що може бути корисним для опису взаємодій людини з інтерфейсами.	Моделює складні людино-машинні системи через ролі, агенти, артефакти та події.
SUMO (Suggested Upper Merged Ontology)	Має потужну підтримку моделювання фізичних і ТП, з акцентом на формальні системи і причини зв'язку.	Може частково підтримувати проектування SCADA через моделювання агентів та їх взаємодію із системами.	Підтримує моделювання як фізичних систем, так і агентських взаємодій, включаючи людино-машинні системи.
PROV-O (Provenance Ontology)	Спеціалізується на відстеженні вихідних даних і процесів, може бути корисним для опису ТП.	Не призначена для проектування інтерфейсів SCADA, основний акцент на управлінні даними.	Підтримує опис взаємодій між системами та людьми через дані та події.
SSN (Semantic Sensor Network Ontology)	Орієнтована на опис сенсорних мереж та даних від сенсорів, що може бути корисним для ТП, пов'язаних з контролем та вимірюваннями.	Може бути корисною в контексті інтеграції даних від сенсорів до інтерфейсів SCADA.	Описує сенсорні системи та взаємодії з користувачами через дані та сенсори.
Upper Cys	Здатна описувати високорівневі ТП, проте потребує специфічних уточнень більш вузької області.	Підтримує високорівневе моделювання взаємодій людини із системами, але потребує налаштування для SCADA.	Підтримує моделювання людино-машинних систем через агентні структури та ролі.
OPMW (Ontology for Provenance and Workflow Management)	Орієнтована на моделювання процесів управління та робочих потоків, але без глибокої технологічної специфікації.	Слабо пристосована для опису інтерфейсів користувача, більше фокусується на процесах.	Можна використовувати для відстеження взаємодій між системою та користувачами через керування робочими процесами.

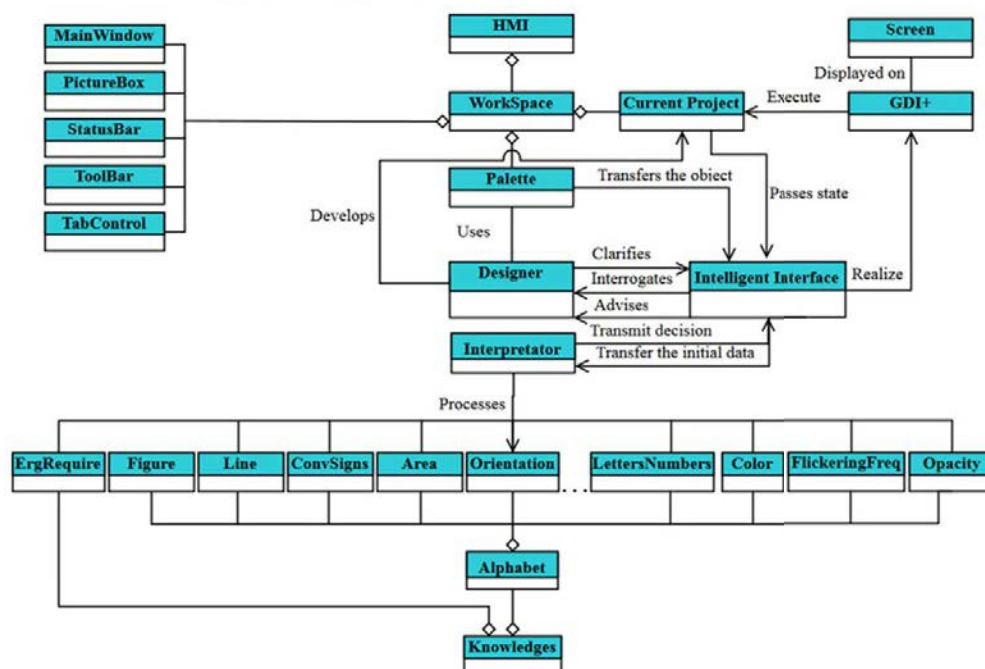


НУ "ЗП", Кафедра програмних засобів

5

Рисунок А.5 – Слайд №5

Концептуальна модель предметної області "Проектування ЛМІ АСУТП"

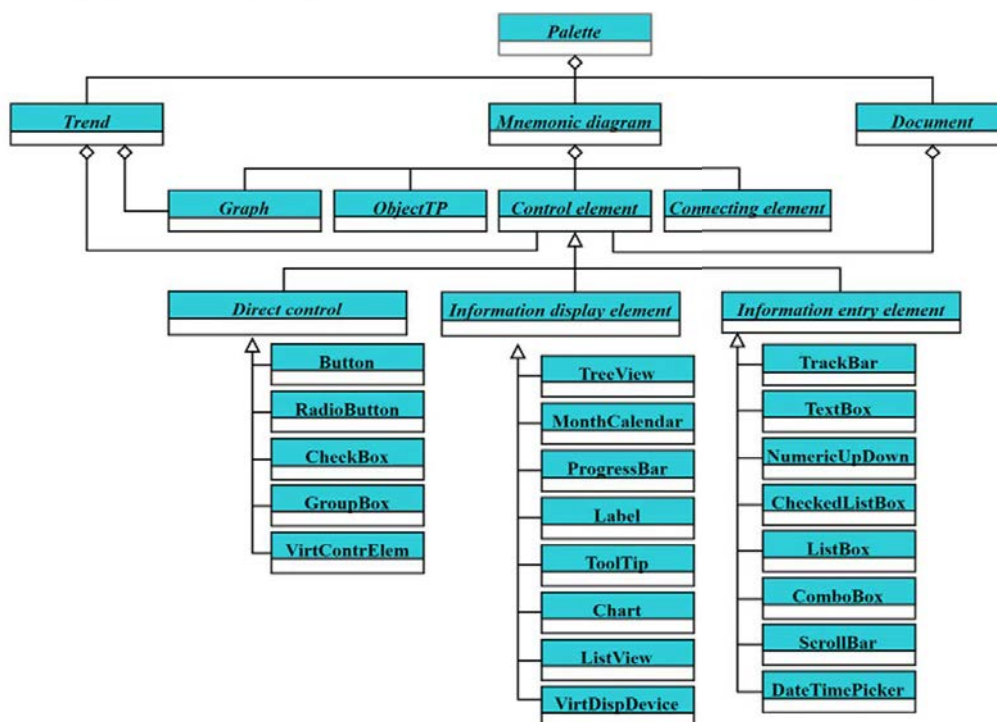


НУ "ЗП", Кафедра програмних засобів

6

Рисунок А.6 – Слайд №6

Модель предметної області «Палітра»



НУ "ЗП", Кафедра програмних засобів

7

Рисунок А.7 – Слайд №7

Приклад формального представлення знань про кодування елементів ЛМІ

Відображувані інформація	Класи простору імен System.Drawing	Методи класу Graphics	Вид алфавіту									
			Геометричні фігури	Умовні знаки	Площа фігур	Орієнтація у просторі	Штрихування	Буквено-цифровий	Колір, яскравість	Частота миготіння	Прозорість	
2-D векторна графіка	Brush	FillClosed Curve							1			
		FillEllipse					1		1			
		FillPolygon					1		1			
		FillRectangle					1		1			
	Pen	DrawArc	1			1			1			
		DrawClosed Curve	1	1	1	1			1			
		DrawEllipse	1		1	1			1			
		DrawLine				1			1			
Растрові зображення	Image	DrawImage	1	1	1	1	1	1	1	1	1	
		DrawIcon		1	1				1			
Символи та рядки	StringFormat	DrawString						1				



НУ "ЗП", Кафедра програмних засобів

8

Рисунок А.8 – Слайд №8

Розробка ієрархічної структури онтології

- рівень цілей;
- рівень функцій;
- рівень задач;
- рівень дій;
- рівень операцій.

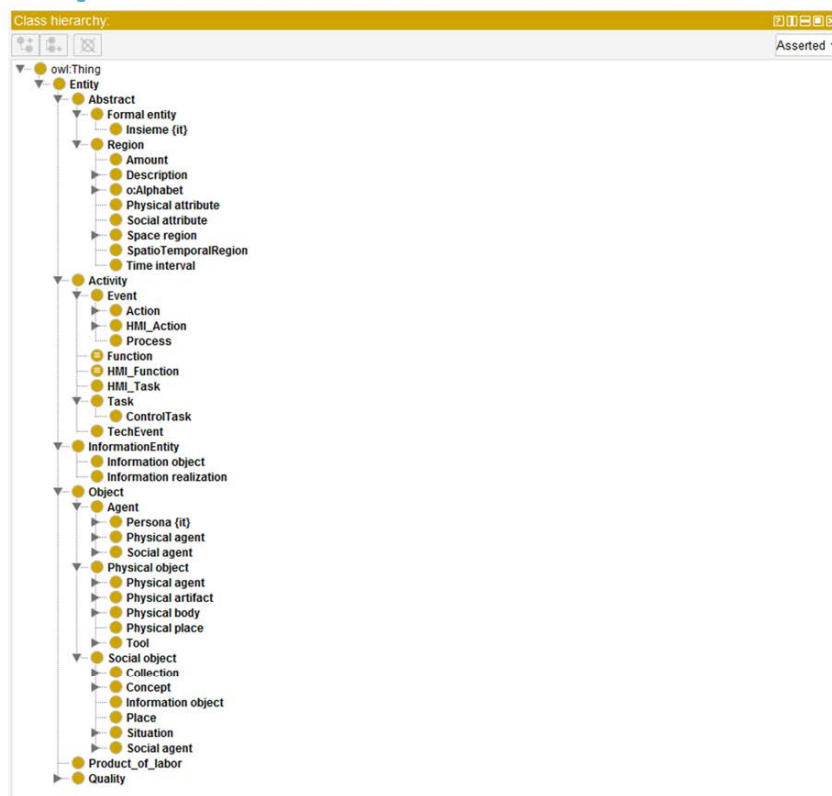


НУ “ЗП”, Кафедра програмних засобів

9

Рисунок А.9 – Слайд №9

Розробка класів метаонтології

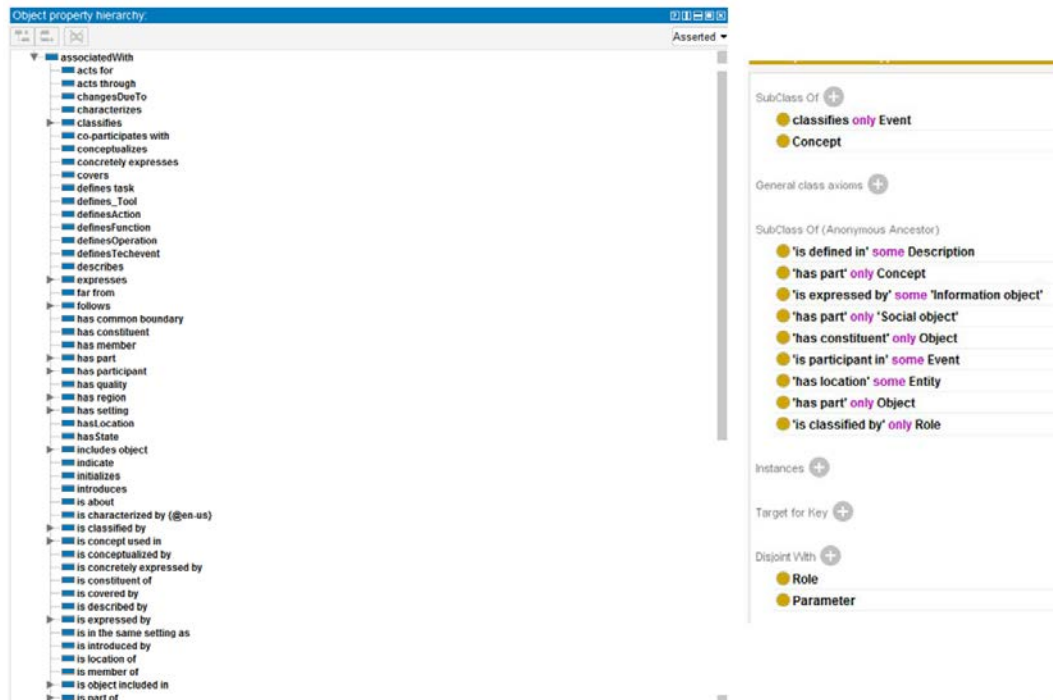


НУ “ЗП”, Кафедра програмних засобів

10

Рисунок А.10 – Слайд №10

Розробка зв'язків метаонтології

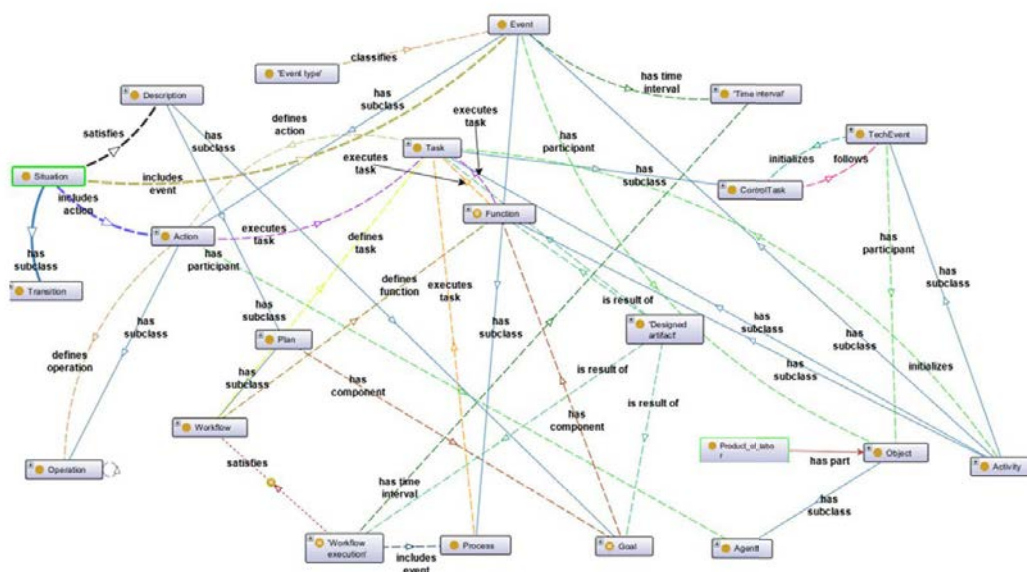


НУ "ЗП", Кафедра програмних засобів

11

Рисунок А.11 – Слайд №11

Розробка онтології «Технологічний процес»



НУ "ЗП", Кафедра програмних засобів

12

Рисунок А.12 – Слайд №12

Розробка онтології «Ерготехнічна система»

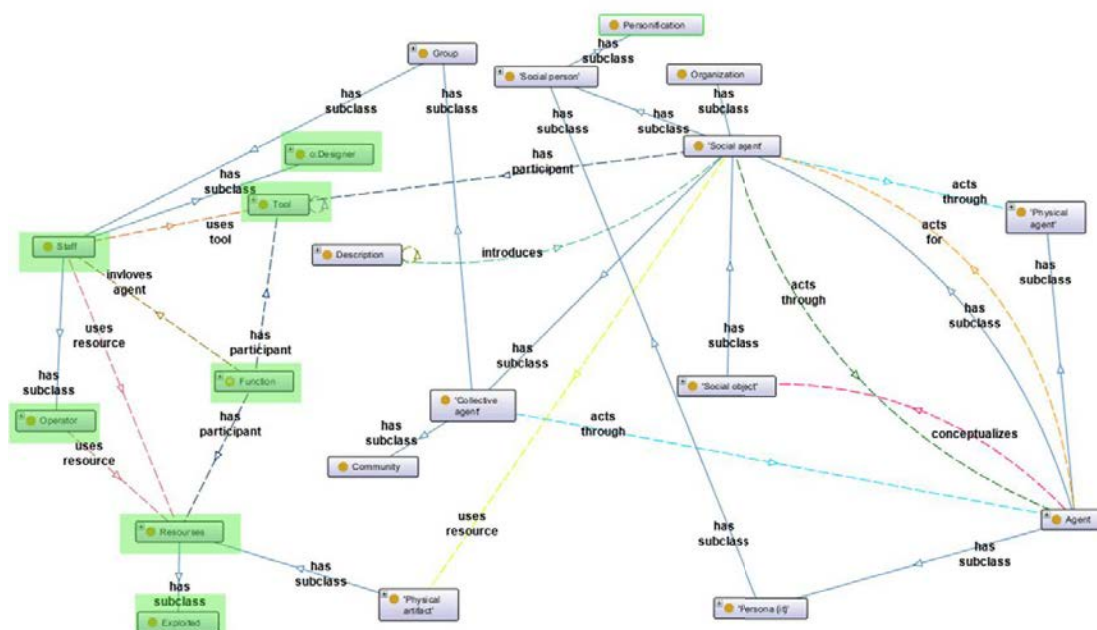


Рисунок А.13 – Слайд №13

Розробка онтології «ЛМІ»

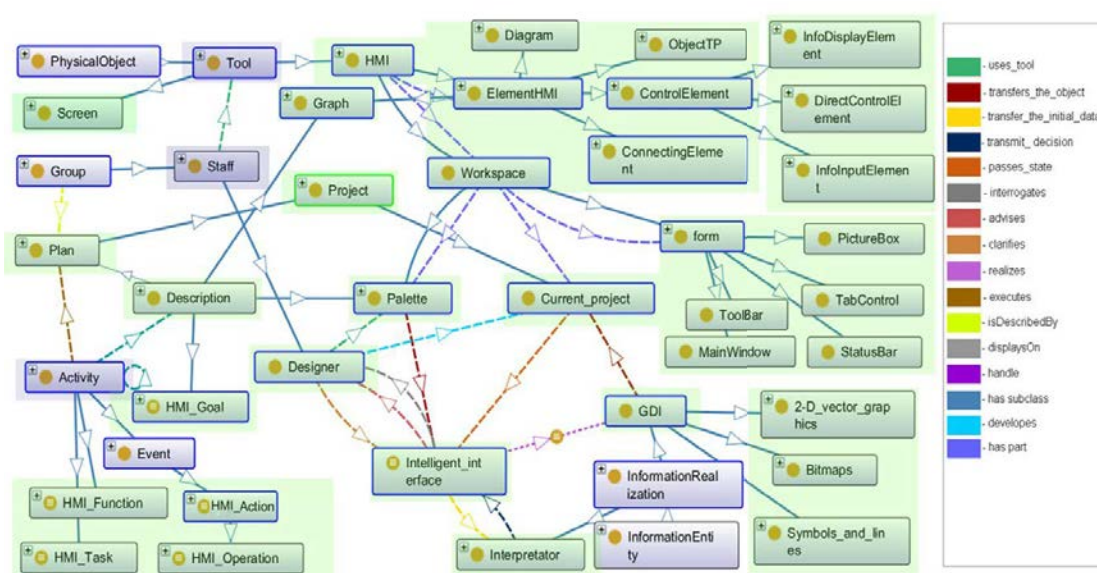
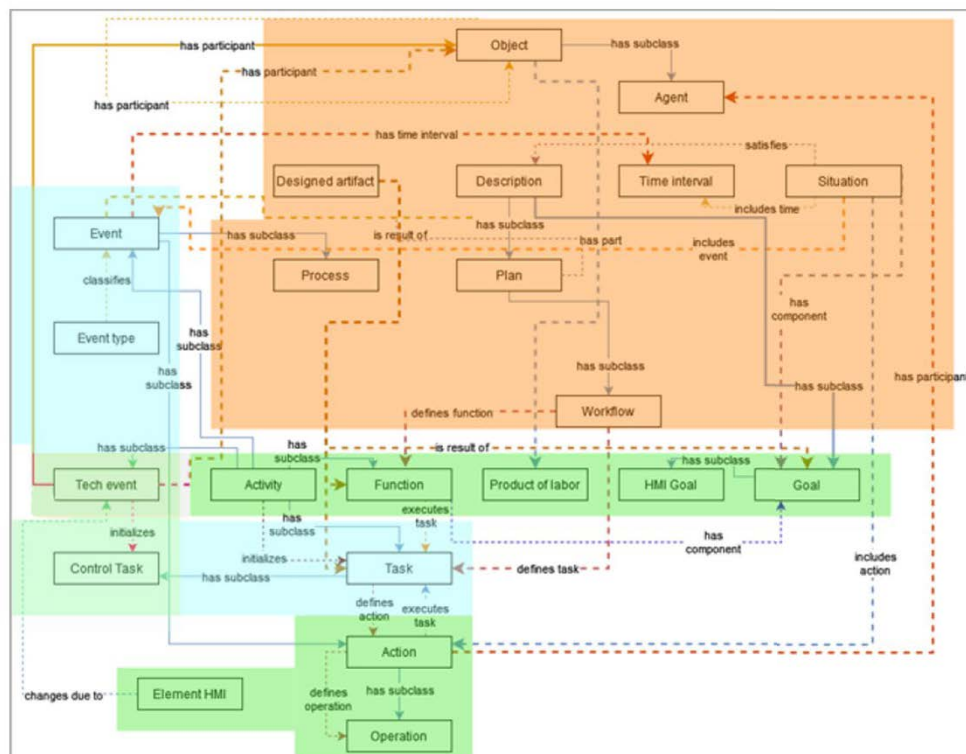


Рисунок А.14 – Слайд №14

Приклад вирівнювання онтологій



НУ "ЗП", Кафедра програмних засобів

15

Рисунок А.15 – Слайд №15

Приклад виконання функції та задачі

Function	Task	Action	Operation
26 Кодування інформації в зоні	261 Кодування інформації в зоні «Мнемосхема»	2611 Вибір об'єкта для кодування в зоні «Мнемосхема»	26111 Вибір типу об'єкта для кодування в зоні «Мнемосхема»
		2612 Визначення властивостей і характеристик для кодування об'єкта Об'єктТІ в зоні «Мнемосхема»	26121 Визначення необхідності дублювання зорової модальності слуховий модальністю
			26122 Визначення специфіки виконуваних завдань.
			26123 Вибір виду кодування Об'єктаТІ на підстав завдань
			26124 Отримання правил і рекомендацій
			26125 Кодування властивостей і характеристик об'єкта відповідно до правил
		2613 Визначення властивостей і характеристик для кодування об'єкта елементКерування в зоні «Мнемосхема»	26131 Вибір типу елементу керування
			26132 Вибір примірника елемента управління

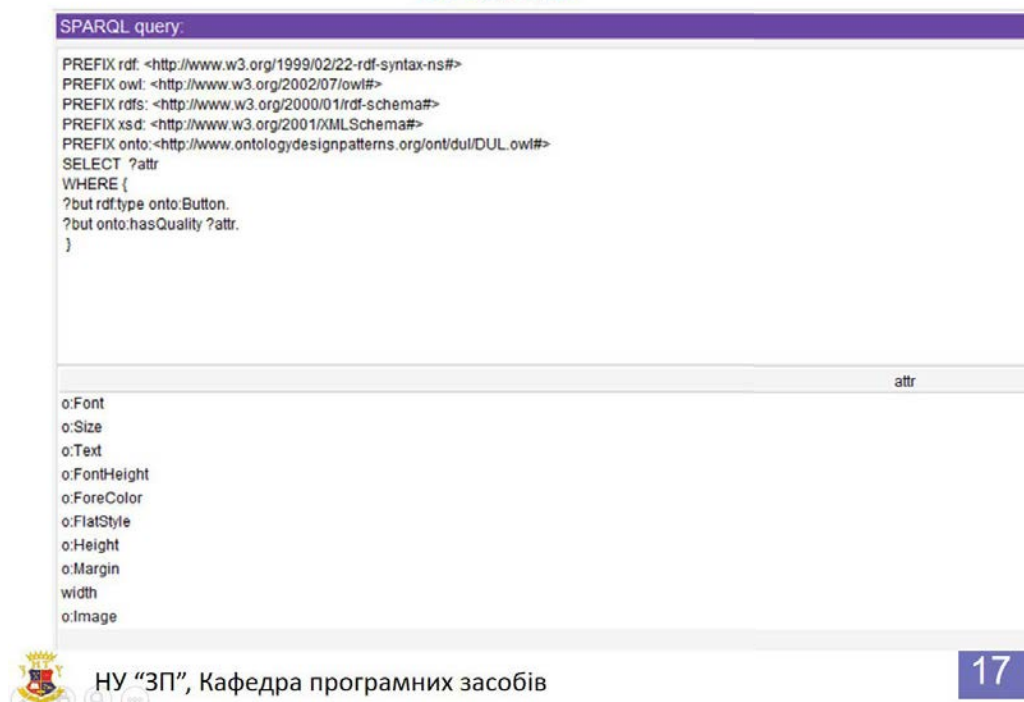


НУ "ЗП", Кафедра програмних засобів

16

Рисунок А.16 – Слайд №16

Приклад SPARQL запиту до бази знань



SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX onto: <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#>
SELECT ?attr
WHERE {
  ?but rdf:type onto:Button.
  ?but onto:hasQuality ?attr.
}
```

attr

o:Font
o:Size
o:Text
o:FontHeight
o:ForeColor
o:FlatStyle
o:Height
o:Margin
width
o:Image

НУ "ЗП", Кафедра програмних засобів

17

Рисунок А.17 – Слайд №17

ВИСНОВКИ

- проведено аналіз проблем ергономічного проєктування ЛМІ АСУТП;
- проведено аналіз сучасних метаонтологій, та виявлена у якості базової метаонтологія DOLCE + DNS UltraLite для подальшого використання;
- розроблено концептуальну модель предметної області "Проєктування ЛМІ АСУТП";
- розроблено предметні онтології «Технологічний процес». «ЛМІ» та «Ерготехнічна система» на базі метаонтології DUL;
- проведено вирівнювання розроблених онтологій з метаонтологією DUL;
- розроблено систему запитів до онтології з використанням мови запитів SPARQL;
- адекватність розроблених моделей, перевірено на прикладі проєктування ЛМІ для АСУТП виробництва газобетону.

Рисунок А.18 – Слайд №18

Новизна, апробація та публікації

Наукова новизна результатів дослідження полягає у тому, що вперше запропоновано інформаційну технологію інтелектуальної підтримки процесу ергономічного проектування ЛМІ, що розробляються за допомогою SCADA-систем. Проведено аналіз структури ЛМІ та виявлено властивості його типових елементів, які можуть бути представлені у вигляді відомих категорій кодування інформаційних моделей. Отримані результати є теоретичною основою розробки експертної системи, що здійснює інтелектуальну підтримку проєктувальника ЛМІ

Наукові та практичні результати роботи доповідалися на II Міжнародній науково-практичній конференції з комп'ютерного моделювання та інтелектуальних систем (CMIS-2019 – Запоріжжя) та II Всеукраїнській науково-практичній конференції «Перспективні напрямки сучасної електроніки, інформаційних і комп'ютерних систем (MEICS-2017 - Дніпро).

Основні наукові результати роботи опубліковані у матеріалах міжнародної конференції (індексується у Scopus), статті у фаховому науковому виданні України та тезах доповіді Всеукраїнській науково-практичній конференції.



НУ “ЗП”, Кафедра програмних засобів

19

Рисунок А.19 – Слайд №19